ITUPublications
Recommendations

International Telecommunication Union
Standardization Sector

Recommendation

ITU-T G.709/Y.1331 (2020) Amd.3 (03/2024)

SERIES G: Transmission systems and media, digital systems and networks

Digital terminal equipments – General

SERIES Y: Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities

Internet protocol aspects – Transport

Interfaces for the optical transport network
Amendment 3

ITU-T G-SERIES RECOMMENDATIONS

Transmission systems and media, digital systems and networks

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100-G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200-G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300-G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400-G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450-G.499
TRANSMISSION MEDIA AND OPTICAL SYSTEMS CHARACTERISTICS	G.600-G.699
DIGITAL TERMINAL EQUIPMENTS	G.700-G.799
General	**G.700-G.709**
Coding of voice and audio signals	G.710-G.729
Principal characteristics of primary multiplex equipment	G.730-G.739
Principal characteristics of second order multiplex equipment	G.740-G.749
Principal characteristics of higher order multiplex equipment	G.750-G.759
Principal characteristics of transcoder and digital multiplication equipment	G.760-G.769
Operations, administration and maintenance features of transmission equipment	G.770-G.779
Principal characteristics of multiplexing equipment for the synchronous digital hierarchy	G.780-G.789
Other terminal equipment	G.790-G.799
DIGITAL NETWORKS	G.800-G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900-G.999
MULTIMEDIA QUALITY OF SERVICE AND PERFORMANCE – GENERIC AND USER-RELATED ASPECTS	G.1000-G.1999
TRANSMISSION MEDIA CHARACTERISTICS	G.6000-G.6999
DATA OVER TRANSPORT – GENERIC ASPECTS	G.7000-G.7999
PACKET OVER TRANSPORT ASPECTS	G.8000-G.8999
ACCESS NETWORKS	G.9000-G.9999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T G.709/Y.1331

Interfaces for the optical transport network

Amendment 3

Summary

Recommendation ITU-T G.709/Y.1331 defines the requirements for the optical transport network (OTN) interface signals of the optical transport network, in terms of:

– OTN hierarchy

– functionality of the overhead in support of multi-wavelength optical networks

– frame structures

– bit rates

– formats for mapping client signals.

Edition 6.5 (Amendment 3) of this Recommendation adds mapping of 800GBASE-R clients, clarifications to the description of GMP, and a new fine-grained path layer and tributary slot structure.

History *

Edition	Recommendation	Approval	Study Group	Unique ID
1.0	ITU-T G.709/Y.1331	2001-02-09	15	11.1002/1000/5350
1.1	ITU-T G.709/Y.1331 (2001) Amd. 1	2001-11-29	15	11.1002/1000/5629
2.0	ITU-T G.709/Y.1331	2003-03-16	15	11.1002/1000/6265
2.1	ITU-T G.709/Y.1331 (2003) Amd. 1	2003-12-14	15	11.1002/1000/7060
2.2	ITU-T G.709/Y.1331 (2003) Cor. 1	2006-12-14	15	11.1002/1000/8982
2.3	ITU-T G.709/Y.1331 (2003) Amd. 2	2007-11-22	15	11.1002/1000/9155
2.4	ITU-T G.709/Y.1331 (2003) Cor. 2	2009-01-13	15	11.1002/1000/9646
2.5	ITU-T G.709/Y.1331 (2003) Amd. 3	2009-04-22	15	11.1002/1000/9671
3.0	ITU-T G.709/Y.1331	2009-12-22	15	11.1002/1000/10398
3.1	ITU-T G.709/Y.1331 (2009) Cor. 1	2010-07-29	15	11.1002/1000/10875
3.2	ITU-T G.709/Y.1331 (2009) Amd. 1	2010-07-29	15	11.1002/1000/10874
3.3	ITU-T G.709/Y.1331 (2009) Amd. 2	2011-04-13	15	11.1002/1000/11115
4.0	ITU-T G.709/Y.1331	2012-02-13	15	11.1002/1000/11485
4.1	ITU-T G.709/Y.1331 (2012) Cor. 1	2012-10-29	15	11.1002/1000/11776
4.2	ITU-T G.709/Y.1331 (2012) Amd. 1	2012-10-29	15	11.1002/1000/11774
4.3	ITU-T G.709/Y.1331 (2012) Amd. 2	2013-10-22	15	11.1002/1000/11982
4.4	ITU-T G.709/Y.1331 (2012) Amd. 3	2014-12-05	15	11.1002/1000/12363
4.5	ITU-T G.709/Y.1331 (2012) Cor. 2	2015-01-13	15	11.1002/1000/12365
4.6	ITU-T G.709/Y.1331 (2012) Amd. 4	2015-01-13	15	11.1002/1000/12364

* To access the Recommendation, type the URL https://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID.

5.0	ITU-T G.709/Y.1331	2016-06-22	15	11.1002/1000/12789
5.1	ITU-T G.709/Y.1331 (2016) Amd. 1	2016-11-13	15	11.1002/1000/13080
5.2	ITU-T G.709/Y.1331 (2016) Cor. 1	2017-08-13	15	11.1002/1000/13298
5.3	ITU-T G.709/Y.1331 (2016) Amd. 2	2018-06-06	15	11.1002/1000/13520
5.4	ITU-T G.709/Y.1331 (2016) Amd. 3	2019-03-22	15	11.1002/1000/13742
5.5	ITU-T G.709/Y.1331 (2016) Cor. 2	2019-11-06	15	11.1002/1000/13995
6.0	ITU-T G.709/Y.1331	2020-06-06	15	11.1002/1000/14199
6.1	ITU-T G.709/Y.1331 (2020) Amd. 1	2020-12-22	15	11.1002/1000/14518
6.2	ITU-T G.709/Y.1331 (2020) Cor. 1	2021-05-29	15	11.1002/1000/14619
6.3	ITU-T G.709/Y.1331 (2020) Amd. 2	2022-02-13	15	11.1002/1000/14894
6.4	ITU-T G.709/Y.1331 (2020) Cor. 2	2022-11-13	15	11.1002/1000/15134
6.5	ITU-T G.709/Y.1331 (2020) Amd. 3	2024-03-08	15	11.1002/1000/15818

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents/software copyrights, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the appropriate ITU-T databases available via the ITU-T website at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Page

Recommendation ITU-T G.709/Y.1331

Interfaces for the optical transport network

Amendment 3

Editorial note: This is a complete-text publication. Modifications introduced by this corrigendum are shown in revision marks relative to Recommendation ITU-T G.709/Y.1331 (2020) plus its Amendment 1, Corrigendum 1, Erratum 1, Amendment 2, Corrigendum 2 and Erratum 2.

1 Scope

The optical transport network (OTN) supports the operation and management aspects of optical networks of various architectures, e.g., point-to-point, ring and mesh architectures.

This Recommendation defines the interfaces of the OTN to be used within and between subnetworks of the optical network, in terms of:

– OTN hierarchy;

– functionality of the overhead in support of multi-wavelength optical networks;

– frame structures;

– bit rates;

– formats for mapping client signals.

Functions and process flows associated with the defined interfaces are specified in [ITU-T G.798].

The interfaces defined in this Recommendation can be applied at user-to-network interfaces (UNI) and network node interfaces (NNI) of the OTN.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T G.652] Recommendation ITU-T G.652 (2016), *Characteristics of a single-mode optical fibre and cable*.

[ITU-T G.693] Recommendation ITU-T G.693 (2009), *Optical interfaces for intra-office systems*.

[ITU-T G.695] Recommendation ITU-T G.695 (2018), *Optical interfaces for coarse wavelength division multiplexing applications*.

[ITU-T G.698.1] Recommendation ITU-T G.698.1 (2009), *Multichannel DWDM applications with single-channel optical interfaces*.

[ITU-T G.698.2] Recommendation ITU-T G.698.2 (2018), *Amplified multichannel dense wavelength division multiplexing applications with single channel optical interfaces*.

[ITU-T G.698.3] Recommendation ITU-T G.698.3 (2012), *Multichannel seeded DWDM applications with single-channel optical interfaces*.

[ITU-T G.707] Recommendation ITU-T G.707/Y.1322 (2007), *Network node interface for the synchronous digital hierarchy (SDH)*.

[ITU-T G.709.1] Recommendation ITU-T G.709.1~~/Y.1331.1~~ (~~2018~~2024), *Flexible OTN ~~short-reach interfaces~~common elements*.

~~[ITU-T G.709.3] Recommendation ITU-T G.709.3/Y.1331.3 (2020), *Flexible OTN long-reach interfaces.*~~

[ITU-T G.709.4] Recommendation ITU-T G.709.4/Y.1334.4 (2020), *OTU25 and OTU50 short-reach interfaces*.

[ITU-T G.780] Recommendation ITU-T G.780/Y.1351 (2010), *Terms and definitions for synchronous digital hierarchy (SDH) networks*.

[ITU-T G.798] Recommendation ITU-T G.798 (2017), *Characteristics of optical transport network hierarchy equipment functional blocks*.

[ITU-T G.805] Recommendation ITU-T G.805 (2000), *Generic functional architecture of transport networks*.

[ITU-T G.806] Recommendation ITU-T G.806 (2012), *Characteristics of transport equipment – Description methodology and generic functionality*.

[ITU-T G.807] Recommendation ITU-T G.807 (2020), *Generic functional architecture of the optical media network.*

[ITU-T G.870] Recommendation ITU-T G.870/Y.1352 (2016), *Terms and definitions for optical transport networks (OTN)*.

[ITU-T G.872] Recommendation ITU-T G.872 (~~2019~~2024), *Architecture of optical transport networks*.

[ITU-T G.873.1] Recommendation ITU-T G.873.1 (2014), *Optical Transport Network (OTN): Linear protection*.2017.

[ITU-T G.873.2] Recommendation ITU-T G.873.2 (2015), ODUk shared ring protection (SRP).

[ITU-T G.959.1] Recommendation ITU-T G.959.1 (~~2018~~2024), *Optical transport network physical layer interfaces*.

[ITU-T G.984.6] Recommendation ITU-T G.984.6 (2008), *Gigabit-capable passive optical networks (GPON): Reach extension*.

[ITU-T G.987.4] Recommendation ITU-T G.987.4 (2012), 10 *Gigabit-capable passive optical networks (XG-PON): Reach extension*.

[ITU-T G.7041] Recommendation ITU-T G.7041/Y.1303 (2016), *Generic framing procedure (GFP)*.

[ITU-T G.7044] Recommendation ITU-T G.7044/Y.1347 (2011), *Hitless Adjustment of ODUflex(GFP)*.

[ITU-T G.7712] Recommendation ITU-T G.7712/Y.1703 (2019), *Architecture and specification of data communication network.*

[ITU-T G.7714.1] Recommendation ITU-T G.7714.1/Y.1705.1 (2010), *Protocol for automatic discovery in SDH and OTN networks*.

[ITU-T G.8251] Recommendation ITU-T G.8251 (2022), *The control of jitter and wander within the optical transport network (OTN).*

[ITU-T G.8260] Recommendation ITU-T G.8260 (2020), *Definitions and terminology for synchronization in packet networks*.

[ITU-T G.8262] Recommendation ITU-T G.8262/Y.1362 (2018), *Timing characteristics of a synchronous equipment slave clock.*

[ITU-T G.8264] Recommendation ITU-T G.8264 (2017), *Distribution of timing information through packet networks*.

[ITU-T M.1400] Recommendation ITU-T M.1400 (2001), *Designations for interconnections among operators' networks*.

[ITU-T M.3100] Recommendation ITU-T M.3100 (2004), *Generic network information model*.

[ITU-T O.150] Recommendation ITU-T O.150 (1996), *General requirements for instrumentation for performance measurements on digital transmission equipment*.

[ITU-T T.50] Recommendation ITU-T T.50 (1992), *International Reference Alphabet (IRA) (Formely International Alphabet No. 5 or IA5) – Information technology – 7-bit coded character set for information interchange*.

[IEEE 802.3] IEEE Std. 802.3~~:2018~~-2022, *Standard for Ethernet*.

~~[IEEE 802.3cd] IEEE Std 802.3cd-2018, *Standard for Ethernet Amendment 3: Media Access Control Parameters for 50 Gb/s and Physical Layers and Management Parameters for 50 Gb/s, 100 Gb/s, and 200 Gb/s Operation.*~~

[IEEE 802.3df] IEEE Std 802.3df-2024, *Standard for Ethernet Amendment 9: Media Access Control Parameters for 800 Gb/s and Physical Layers and Management Parameters for 4000 Gb/s and 800 Gb/s Operation.*

[OIF FlexE IA] OIF, Flex *Ethernet 2.0 Implementation Agreement* (2018).

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 Terms defined in [ITU-T G.780]:

– BIP-X

– network node interface

3.1.2 Terms defined in [ITU-T G.805]:

– adapted information (AI)

– characteristic information (CI)

– network

– subnetwork

3.1.4 Terms defined in [ITU-T G.872]:

– optical multiplex section (OMS)

– optical transmission section (OTS)

3.1.5 Terms defined in [ITU-T G.959.1]:

– optical tributary signal (OTSi).

3.1.6 Terms defined in [OIF FlexE IA]:

– FlexE Client

3.1.7 Terms defined in [ITU-T G.807]:

– optical signal maintenance entity (OSME)

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 ODUk.ts: The ODUk.ts is an increment of bandwidth which when multiplied by a number of tributary slots gives the recommended size of an ODUflex(GFP,n,k) optimized to occupy a given number of tributary slots of an OPUk.

3.2.2 OTU0 low latency: The OTU0 low latency (OTU0LL) is the information structure used for transport of an ODU0 over a multi-vendor optical network interface in one administrative domain at the edge of the optical transport network.

3.2.3 optical tributary signal assembly (OTSiA): The OTSiG together with the non-associated overhead (OTSiG-O).

3.2.4 optical tributary signal group (OTSiG): The set of OTSi signals that supports an OTU.

3.2.5 optical tributary signal overhead (OTSiG-O): The non-associated overhead for an OTSiG.

3.2.6 connection monitoring end-point (CMEP): Connection monitoring end-points represent end-points of trails and correspond as such with the trail termination functions. Connection monitoring overhead (CMOH) is inserted and extracted at the CMEPs.

For the ODU and OTU the CMEPs are categorized in three classes:

– OTU section CMEP (OS_CMEP), which represents the end-points of the OTUk/OTUCn trail. The SM overhead field contains the related CMOH.

– ODU tandem connection CMEP (TC_CMEP), which represents the end-points of ODUkT/ODUCnT trails. The TCM1.6 overhead fields contain the related CMOH.

– ODU path CMEP (P_CMEP), which represents the end-points of the ODUkP/ODUCnP trail. The PM overhead field contains the related CMOH.

3.2.7 non-associated overhead: Supervisory information transported in an optical supervisory signal, overhead communication channel or other means.

3.2.8 optical data unit (ODU): The ODU is an information structure consisting of the information payload (OPU) and ODU related overhead, either as ODUk or ODUCn.

3.2.9 optical payload unit (OPU): The OPU is the information structure used to adapt client information for transport over the OTN, either as an OPUk or an OPUCn. It comprises client information together with any overhead needed to perform rate adaptation between the client signal rate and the OPU payload rate, and other OPU overheads supporting the client signal transport. The index k or Cn (see clause 5) is used to identify the approximate bit rate and different versions.

3.2.10 optical transport unit (OTU): The OTU is the information structure used for transport of an ODU over an OTSiA or OCh, either as an OTUk or an OTUCn.

For OCh and OTN point-to-point interface cases, the OTU overhead is used as overhead for the transport over the optical layer, and the OTU must terminate where the OCh and OTN point-to-point interface is terminated. For an OTSiA, the OTU overhead may or may not be terminated where the OTSiA is terminated.

Three versions of the OTUk are defined: completely standardized OTUk (OTUk), functionally standardized OTUk (OTUkV) and an OTUk with completely standardized overhead and functionally standardized FEC (OTUk-v). The completely standardized OTUk is used on OTN IrDIs and may be used on OTN IaDIs. The other two are used on OTN IaDIs.

3.2.11 OCh: The OCh is an information structure consisting of the information payload (OTSiG) with a certain bandwidth and non-associated overhead (OCh-O) for management of the OCh.

3.2.12 optical transport network: An optical transport network (OTN) is composed of a set of optical network elements connected by optical fibres, that provide functionality to encapsulate, transport, multiplex, route, manage, supervise and provide survivability of client signals. The information of the client signals is processed in the digital domain and carried across the media, according to the requirements given in [ITU-T G.872].

3.2.13 optical transport network node interface (ONNI): The interface at an optical transport network node which is used to interconnect with another optical transport network node.

3.2.14 ODUflex(GFP): ODUflex for GFP-F mapped client signals with user configured bit rate.

3.2.15 ODUflex(GFP,n,k): ODUflex for GFP-F mapped client signals with user configured bit rate of n × ODUk.ts in which n = number of 1.25G tributary slots and k = ODUk.ts representative.

ODUflex(GFP,n,k) is a subset of ODUflex(GFP).

3.2.16 ODUflex(IMP): ODUflex for IMP mapped client signals with user configured bit rate.

3.2.17 ODUflex(IMP,s): ODUflex for IMP mapped client signals with bit rate of s × 239/238 × 5 156 250 kbit/s ± 100 ppm in which s = 2, 8, n × 5 with n ≥ 1.

ODUflex(IMP,s) is a subset of ODUflex(IMP).

3.2.18 fgODUflex(p): fgODUflex for mapping sub1G client signals with user configured bit rate of p × 1 244 160 kbit/s / 119.525 ± 20 ppm in which p = 1 to 119.

3.2.19 ODUflex(fgTS,n): ODUflex for multiplexing one or more fgODUflex signals with bit rate of n × 1 244 160 kbit/s ± 20 ppm in which n = 3 to 7.

3.2.20 ODUk(fgTS): ODUk for multiplexing one or more fgODUflex signals in which k = 0, 1 and 2.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

16FS	16 columns with Fixed Stuff
3R	Reamplification, Reshaping and Retiming
AI	Adapted Information
AIS	Alarm Indication Signal
AM	Alignment Marker
AMP	Asynchronous Mapping Procedure
API	Access Point Identifier
APS	Automatic Protection Switching
ASI	Asynchronous Serial Interface for DVB
BDI	Backward Defect Indication
BDI-O	Backward Defect Indication Overhead
BDI-P	Backward Defect Indication Payload
BEI	Backward Error Indication
BGMP	Bit-synchronous Generic Mapping Procedure

BI	Backward Indication
BIAE	Backward Incoming Alignment Error
BIP	Bit Interleaved Parity
BMP	Bit-synchronous Mapping Procedure
BWR	Bandwidth Resize
CAUI	(Chip to) 100 Gb/s Attachment Unit Interface
CB	Control Block
CBR	Constant Bit Rate
CDI	Client Degrade Indication
CFS	Client Frame Start
CI	Characteristic Information
CM	Connection Monitoring
C_m	number of m-bit Client data entities
CMEP	Connection Monitoring End Point
CMGPON_D	Continuous Mode GPON Downstream
CMGPON_U2	Continuous Mode GPON Upstream 2
CMOH	Connection Monitoring Overhead
CMXGPON_D	Continuous Mode XGPON Downstream
CMXGPON_U2	Continuous Mode XGPON Upstream 2
Cn	number of n-bit client data entities
CnD	difference between Cn and (m/n x Cm)
CPRI	Common Public Radio Interface
CRC	Cyclic Redundancy Check
CS	Client Specific
CSF	Client Signal Fail
CTRL	Control word sent from source to sink
DA	Difference Accumulation (phase)
DAi	Difference Accumulation overhead of index i in an fgODUflex frame
DAPI	Destination Access Point Identifier
DDR	Double Data Rate
DI	Decrement Indicator
DMp	Delay Measurement of ODUk path
DMti	Delay Measurement of TCMi
DNU	Do Not Use
DVB	Digital Video Broadcast
EDC	Error Detection Code
EXC	Electrical cross Connect

EOS	End Of Sequence
ESCON	Enterprise Systems Connection
eSSM	enhanced Synchronization Status Message
EXP	Experimental
ExTI	Expected Trace Identifier
FA	Frame Alignment
FAS	Frame Alignment Signal
FC	Fibre Channel
FC	Flag Continuation
FDI	Forward Defect Indication
FDI-O	Forward Defect Indication Overhead
FDI-P	Forward Defect Indication Payload
FEC	Forward Error Correction
fg	Fine Grain
FlexO	Flexible Optical transport network
~~FlexO-x~~	~~FlexO interface information structure of order x~~
FlexO-x-<~~fec~~int>	FlexO interface signal of order x with a FEC and modulation scheme
~~FlexO-x-<fec>-m~~	~~Group of m × FlexO-x-<fec> interfaces~~
GCC	General Communication Channel
GID	Group Identification
GMP	Generic Mapping Procedure
GPON	Gigabit-capable Passive Optical Networks
HRN	Half Row Numbering
IAE	Incoming Alignment Error
IB	InfiniBand
II	Increment Indicator
IMP	Idle Mapping Procedure
IP	Internet Protocol
JC	Justification Control
JOH	Justification Overhead
LCR	Link Connection Resizing
LD	Local Degrade
LF	Local Fault
LLM	Logical Lane Marker
LSB	Least Significant Bit
MFAS	MultiFrame Alignment Signal
MFI	Multiframe Indicator

MPLS	Multi Protocol Label Switching
MPLS-TP	Multi Protocol Label Switching – Transport Profile
MS	Maintenance Signal
MSB	Most Significant Bit
MSI	Multiplex Structure Identifier
NJO	Negative Justification Opportunity
NNI	Network Node Interface
NOS	Not_Operational Sequence
OADM	Optical Add/Drop Multiplexer
OCC	Overhead Communication Channel
OCI	Open Connection Indication
OCN	Overhead Communication Network
ODTUG	Optical Data Tributary Unit Group
ODTUGk/Cn	Optical Data Tributary Unit Group-k/Cn
ODTUjk	Optical Data Tributary Unit j into k
ODTUk/Cn.ts	Optical Data Tributary Unit k/Cn with ts tributary slots
ODU	Optical Data Unit
ODUk/Cn	Optical Data Unit-k/Cn
ODUk/Cn.ts	Optical Data Unit k/Cn fitting in ts tributary slots
ODUk/CnP	Optical Data Unit-k/Cn Path monitoring level
ODUk/CnT	Optical Data Unit-k/Cn Tandem connection monitoring level
OH	Overhead
OMFI	OPU Multi-Frame Identifier
OMS	Optical Multiplex Section
OMS-O	Optical Multiplex Section Overhead
OMS-P	Optical Multiplex Section Payload
ONNI	Optical Network Node Interface
OPS	Optical Physical Section
OPSM	Optical Physical Section Multilane
OPU	Optical Payload Unit
OPUk/Cn	Optical Payload Unit-k/Cn
OSC	Optical Supervisory Channel
OSMC	OTN synchronization messaging channel
OSME	Optical Signal Maintenance Entity
OTL	Optical Transport Lane
OTLk.n	group of n Optical Transport Lanes that carry one OTUk
OTLC.n	group of n Optical Transport Lanes that carry one OTUC of an OTUCn

OTN	Optical Transport Network
OTS	Optical Transmission Section
OTS-O	Optical Transmission Section Overhead
OTS-P	Optical Transmission Section Payload
OTSi	Optical Tributary Signal
OTSiA	Optical Tributary Signal Assembly
OTSiG	Optical Tributary Signal Group
OTSiG-O	Optical Tributary Signal Group – Overhead
OTU	Optical Transport Unit
OTU0LL	Completely standardized Optical Transport Unit-0 Low Latency
OTU4-SC	Optical Transport Unit 4 with ~~Stair Case~~staircase FEC
OTUCn-M	Optical Transport Unit-Cn with n OxUC overhead instances and M 5G tributary slots
OTUk/Cn/25/50	completely standardized Optical Transport Unit-k/Cn/25/50
OTUkV	functionally standardized Optical Transport Unit-k
OTUk-v	Optical Transport Unit-k with vendor specific OTU FEC
OXC	Optical cross Connect
PCC	Protection Communication Channel
P-CMEP	Path-Connection Monitoring End Point
PCS	Physical Coding Sublayer
PJO	Positive Justification Opportunity
PKT	Packet
PLD	Payload
PM	Path Monitoring
PMA	Physical Medium Attachment sublayer
PMI	Payload Missing Indication
PMOH	Path Monitoring Overhead
PN	Pseudo-random Number
POS	Position field
ppm	parts per million
PRBS	Pseudo Random Binary Sequence
PSI	Payload Structure Identifier
PT	Payload Type
QDR	Quad Data Rate
RCOH	Resizing Control Overhead
RD	Remote Degrade
RES	Reserved for future international standardization

RF	Remote Fault
RS	Reed-Solomon
RS-Ack	Re-sequence Acknowledge
SAPI	Source Access Point Identifier
SBCON	Single-Byte command code sets Connection
SDI	Serial Digital Interface
SDR	Single Data Rate
Sk	Sink
SM	Section Monitoring
SMOH	Section Monitoring Overhead
SNC	Subnetwork Connection
SNC/I	Subnetwork Connection protection with Inherent monitoring
SNC/N	Subnetwork Connection protection with Non-intrusive monitoring
SNC/S	Subnetwork Connection protection with Sublayer monitoring
So	Source
SSM	Synchronization Status Message
TC	Tandem Connection
TC-CMEP	Tandem Connection-Connection Monitoring End Point
TCM	Tandem Connection Monitoring
TCMOH	Tandem Connection Monitoring Overhead
TS	Tributary Slot
TSI	Transmitter Structure Identifier
TSMxOH	Tributary Slot Multiplexing Overhead
TSOH	Tributary Slot Overhead
TTI	Trail Trace Identifier
TTT	Timing Transparent Transcoding
TxTI	Transmitted Trace Identifier
UNI	User-to-Network Interface
XC	Cross Connect
XGPON	10 Gigabit-capable Passive Optical Networks

5 Conventions

This Recommendation uses the following conventions:

– k **or Cn**: The index "k" is used to represent a supported bit rate and the different versions of OPUk, OPUCn, ODUk, ODUCn, OTUk and OTUCn. Examples for k are "0" for an approximate bit rate of 1.25 Gbit/s, "1" for an approximate bit rate of 2.5 Gbit/s, "2" for an approximate bit rate of 10 Gbit/s, "3" for an approximate bit rate of 40 Gbit/s, and "4" for an approximate bit rate of 100 Gbit/s. When the "Cn" index is used the approximate bit rate is n×100 Gbit/s. For example for Cn = C4 the approximate bit rate is 400 Gbit/s.

– x: The index **x** gives the approximate bit rate for a CBR signal. It is used in the form "unit value, unit, [fractional unit value]". The currently defined unit value is "G" for Gbit/s. Examples for x are "40G" for 40 Gbit/s and "2G5" for 2.5 Gbit/s.

– -P: The suffix -P is used to identify the payload element of the signal.

– -O: The suffix -O is used to identify the non-associated overhead element of the signal.

– ODUj: When two different ODUk information structures are involved, ODUj is used to denote the lower rate ODUk information structure (e.g., for ODUk multiplexing).

The functional architecture of the optical transport network as specified in [ITU-T G.872] is used to derive the ONNI. The ONNI is specified in terms of the adapted and characteristic information present in each layer as described in [ITU-T G.805].

Transmission order: The order of transmission of information in all the diagrams in this Recommendation is first from left to right and then from top to bottom. Within each byte the most significant bit is transmitted first. The most significant bit (bit 1) is illustrated at the left in all the diagrams.

Mapping order: The serial bit stream of a constant bit rate signal is inserted into the OPU payload so that the bits will be transmitted on the OPU/ODU in the same order that they were received at the input of the AMP, BMP or GMP mapper function. If m bits b_a, b_b, b_c up to b_m are client signal bits of which b_a is the bit that is received first and b_m is the bit that is received last, then b_a will be mapped into bit 1 of a first OPU byte and b_m will be mapped into bit 8 of an n^{th} OPU byte (with n = m/8).

Value of reserved bit(s): The value of an overhead bit, which is reserved or reserved for future international standardization shall be set to "0".

Value of non-sourced bit(s): Unless stated otherwise, any non-sourced bits shall be set to "0".

OTUk (k=0,1,2,3,4), OTUCn, OTU25, OTU50, ODUk (k=0,1,2,2e,3,4,flex,25,50), ODUCn, OPUk (k=0,1,2,2e,3,4,flex,25,50) and OPUCn overhead assignment: The assignment of an overhead in the optical transport/data/payload unit signal to each part is defined in Figure 5-1. OTUk, OTU25, OTU50, ODUk, OPUk contain one instance of OTU, ODU, OPU overhead. OTUCn, ODUCn and OPUCn contain n instances of OTU, ODU, OPU overhead, numbered 1 to n.

Interleaved versions of the OTU, ODU and OPU overhead may be present on OTUCn interfaces. This interleaving is interface specific and specified for OTN interfaces with standardized application codes in the interface specific Recommendations (ITU-T G.709.x series). Within the other clauses of this Recommendation an OTUCn, ODUCn and OPUCn are presented in an interface independent manner, by means of n OTUC, ODUC and OPUC instances that are marked #1 to #n.

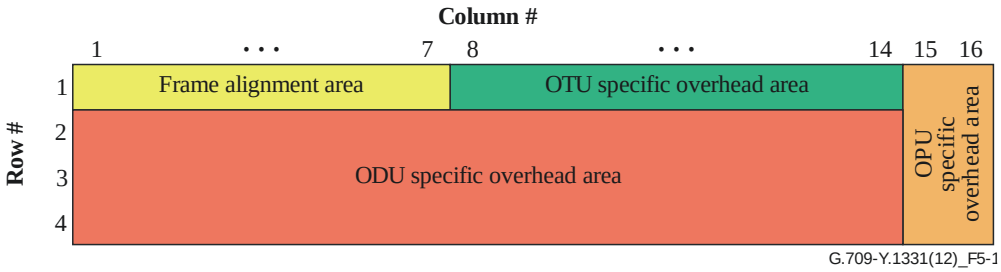


Figure 5-1 – OTU, ODU and OPU overhead

6 Optical transport network interface structure

The optical transport network as specified in [ITU-T G.872] provides for two OTN interface types: point-to-point interfaces and optical networking interfaces. Both OTN interface types consist of one or more optical tributary signals.

The OTN point-to-point interfaces are defined with 3R processing at each end of the interface (terminating the optical tributary signals) and do not provide for optical layer overhead.

The OTN optical networking interfaces are defined with optical layer overhead and may not provide for 3R processing at each end of the interface, which allow optical tributary signals to be forwarded beyond the interface.

These two types of OTN interfaces consist of digital and management information defined in this Recommendation as carried by a physical interface, the specifications of which are outside the scope of this Recommendation.

Each OTN interface may be associated with one or more application identifiers. Each application identifier represents either a standardized application code or a vendor specific identifier. An interface with one or more application codes may be used to interconnect (OTN) equipment from different vendors. An interface with one or more vendor specific identifiers is to be used to interconnect (OTN) equipment from the same vendor. When considering interconnection over an interface between user and provider administrations or provider to provider administrations, 3R termination shall be used on either side of the interface, unless mutually agreed otherwise.

6.1 Basic signal structure

The basic structure is shown in Figures 6-1 and 6-2 and consists of a digital and an optical structure.

6.1.1 OTN digital structure

The OTN digital structure (see Figure 6-1) is based on three classes of optical transport units (OTU); the OTUk, OTU25 and OTU50 and OTUCn.

– The OTUk signal consists of a 4080 column by 4 row frame, which includes 256 columns allocated to contain a forward error correction code (FEC) and is operated at various bit rates that are represented by the value of k.

– The OTU25 and OTU50 signals consist of a 3824 column by 4 row frame, which do not include a FEC area and are operated at 25 Gbit/s and 50 Gbit/s bit rates that are represented by the values 25 and 50. FEC for the OTU25 and OTU50 signals are interface specific and not included in the OTU25 and OTU50 definition.

– The OTUCn signal consists of n interleaved 3824 columns by 4 row frames, which do not include a FEC area and is operated at n times a basic rate that is represented by OTUC. FEC for the OTUCn signal is interface specific and not included in the OTUCn definition.

The OTUCn signal may be carried ~~via a FlexO-x-<fec>-m interface group with m = ⌈n/x⌉ FlexO-x-<fec> interfaces~~over FlexO; refer to [ITU-T G.709.1]~~ and [ITU-T G.709.3]~~.

The OTU25 and OTU50 signals are carried via OTU25-RS and OTU50-RS interfaces; refer to [ITU-T G.709.4].

The OTU contains an optical data unit (ODU), which contains an optical payload unit (OPU). The OTUk, OTU25, OTU50, OTUCn and ODUCn are digital section layers. The ODUk (and ODUj) is a path layer.

The OTUk (k ∈ {0,1,2,3,4,4-SC}) is fully standardized. There are two functionally standardized alternatives: OTUk-v and OTUkV. In all cases, these provide supervision and condition the signal for transport between 3R regeneration points in the OTN.

The OTUCn, OTU25 and OTU50 provide supervision for transport between 3R regeneration points in the OTN. Note that the OTUCn, OTU25 and OTU50 do not condition the signal for transport between 3R regeneration points because encapsulation and forward error correction are not included in the OTUCn, OTU25 and OTU50 definition.

The ODU provides:

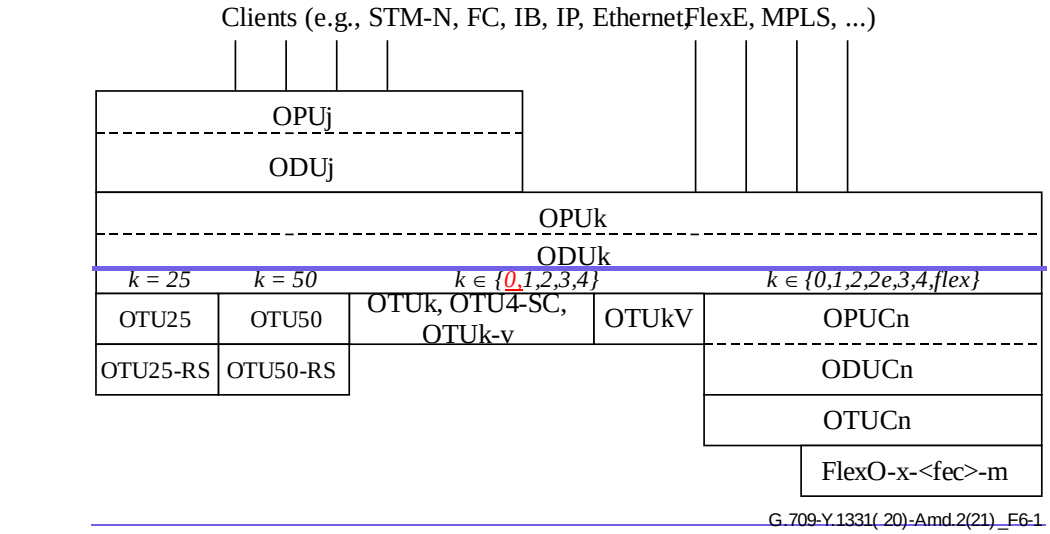
– Tandem connection monitoring (ODUjT, ODUkT, ODUCnT);

– end-to-end section supervision (ODUjP, ODUkP, ODUCnP);

– adaptation of client signals via the OPUk and OPUj (j ∈ {0, 1, 2, 2e, 3, 4, flex}, k ∈ {0, 1, 2, 2e, 3, 4, flex});

– adaptation of client ODUj signals via the OPUk (j ∈ {0, 1, 2, 2e, 3, 4, flex}, k ∈ {1, 2, 3, 4, 25, 50}, the bit rate of the ODUj is less than the OPUk payload bit rate);

– adaptation of client ODUk signals via the OPUCn (k ∈ {0, 1, 2, 2e, 3, 4, flex}).



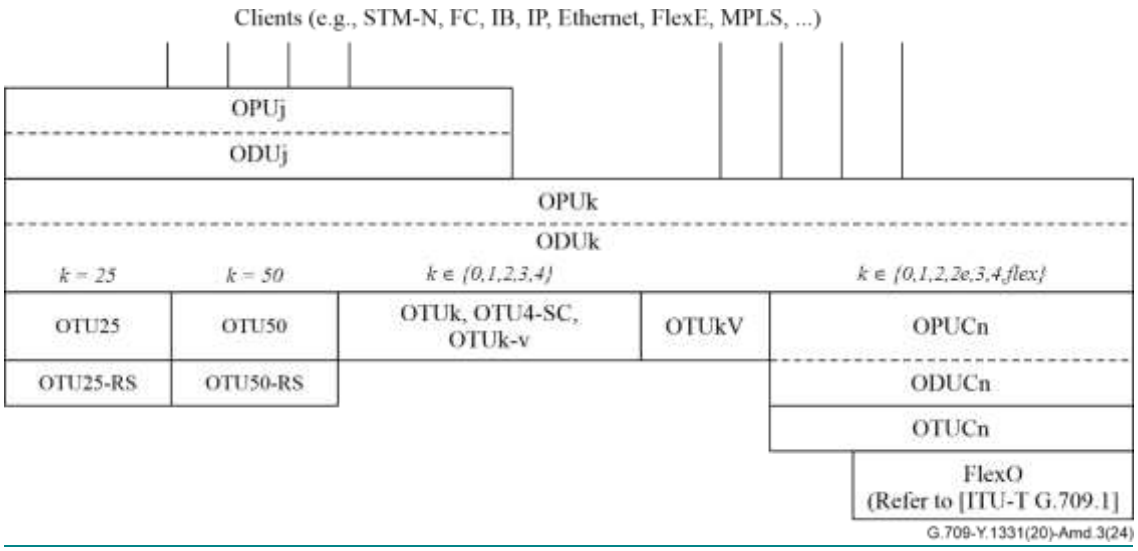


Figure 6-1 – Digital structure of the OTN interfaces

6.1.2 OTN optical structure

The OTN optical structure (see Figure 6-2) is based on the presence of one or more optical tributary signals combined with three classes of optical overhead deployment:

– Absence of optical overhead in the case of OTN point-to-point interfaces.

– Optical overhead carried in the same fibre as the one or more optical tributary signals in the case of OTN optical networking interfaces, type I.

– Optical overhead carried via the overhead communication network (OCN) outside the fibre with one or more optical tributary signals in the case of OTN optical networking interfaces, type II.

The optical overhead is divided into three digital layers that coincide with the maintenance entities defined in [ITU-T G.872], as shown in Figure 6-2:

– Overhead to monitor the optical path maintenance entity (OCh-O and OTSiG-O), which extends between two OTSiG modulators and demodulators.

– Overhead to monitor the optical multiplex section maintenance entity, which extends between two points of optical multiplexing (OMS-O).

– Overhead to monitor the optical transmission section maintenance entity, which extends between two optical amplifiers, or between an optical multiplexer (without an amplifier) and the next optical amplifier (OTS-O).

The OTN optical networking interfaces may support optical layer overhead (represented by the entities described above). Such overhead may be transported within the optical supervisory channel (OSC), the overhead communication channel (OCC) that is provided by the overhead communication network (OCN) (refer to [ITU-T G.7712]) or an alternative communication channel. Interfaces that support OCh-O and/or OTSiG-O support switching in the optical layer of the OCh and/or OTSiA signals which carry one optical transport unit or ~~FlexO-x-<fec>-m-group~~FlexO-x-<int> signal between 3R regeneration points. Interfaces that support OTS-O and OMS-O also support deployment of in-line optical amplifiers between optical layer switching points.

The OTN point-to-point interfaces do not support optical layer overhead and are designed to transport the optical transport unit or ~~FlexO-x-<fec>-m-group~~FlexO-x-<int> signals over a single optical span with 3R regeneration points at each end. For such cases there is no optical path layer present in the interface stack.

OTN point-to-point interfaces carry one or more OTUk or OTUCn or ~~FlexO-x-<fec> of an OTUCn/FlexO-x-<fec>-m-group~~FlexO-x-<int>, which are monitored by means of OTUk or OTUCn or ~~FlexO-x-<fec>-m-group~~FlexO overhead.

OTN optical networking interfaces carry one or more optical tributary signals, each transporting an OTUk or an OTUCn or one ~~FlexO-x-<fec> out of an OTUCn/FlexO-x-<fec>-m-group~~FlexO-x-<int> or a non-OTN client signal.

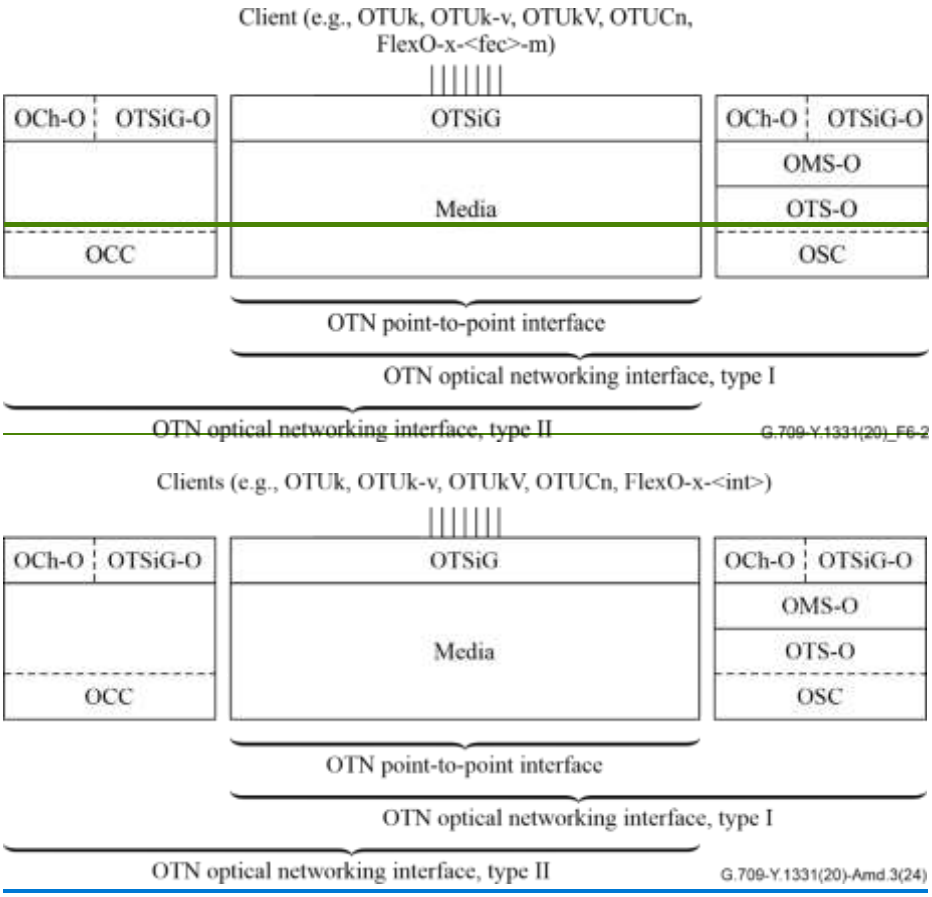


Figure 6-2 – Optical structure of the OTN interfaces

6.2 Information structure for OTN interfaces

The information structure for OTN interfaces is represented by information containment relationships and flows. The principal information containment relationships of the digital structure of the OTUk and OTUCn are described in Figures 6-3 and 6-4.

For supervision purposes in OTN optical networking interfaces, the OTU signal is terminated whenever the OCh signal is terminated, and the OTU signal may be terminated when the OTSiA signal is terminated. For supervision purposes in OTN point-to-point interfaces, the OTU signals are terminated whenever the interface signal is terminated.

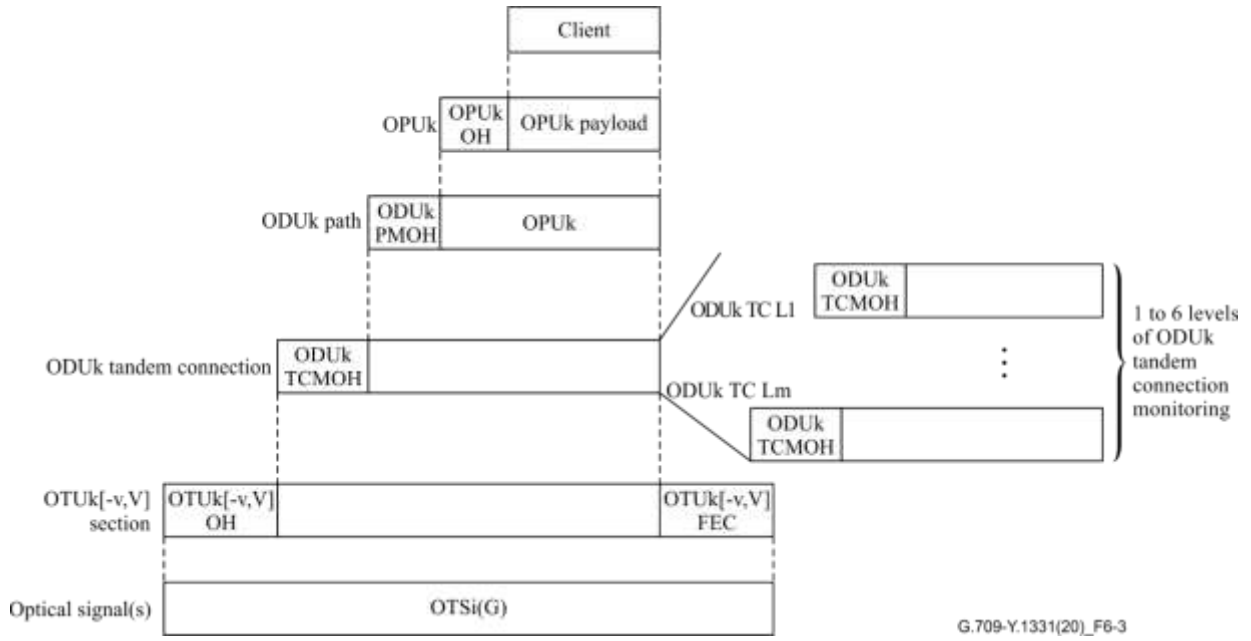


Figure 6-3 – Principal information containment relationship of digital structure of OTUk

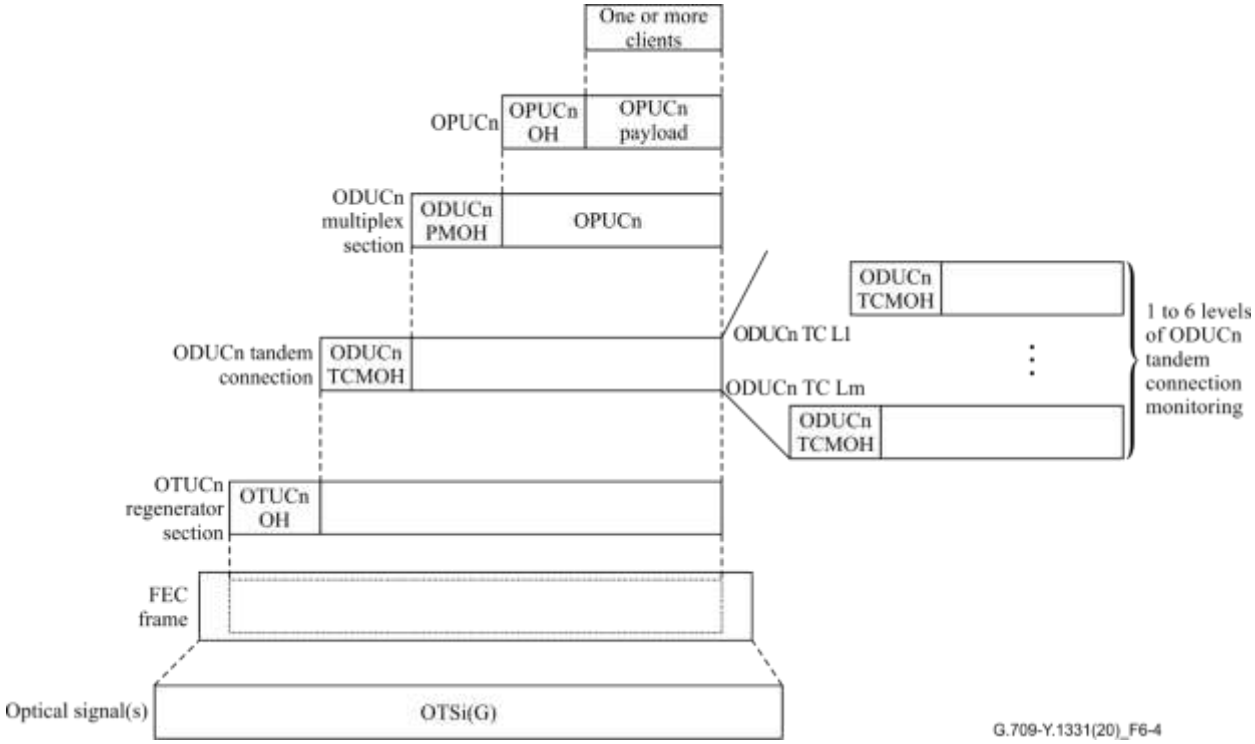


Figure 6-4 – Principal information containment relationship of digital structure of OTUCn

7 Multiplexing/mapping principles and bit rates

Figures 7-1A and 7-1B show the relationship between various information structure elements and illustrates the multiplexing structure and mappings for the OTU. In the multi-domain OTN any combination of the ODU multiplexing layers may be present at a given OTN NNI. The interconnection of and visibility of ODU multiplexing layers within an equipment or domain is outside the scope of this Recommendation. Figures 7-1A and 7-1B show that a (non-OTN) client signal is mapped into an OPU. This OPU signal is mapped into the associated ODU. This ODU signal is either mapped into the associated OTU[V] signal, or into an ODTU. This ODTU signal is multiplexed into an ODTU Group (ODTUG). The ODTUG signal is mapped into an OPU. This OPU signal is mapped into the associated ODU, etc.

The OPUk (k=0,1,2,2e,3,4,flex,25,50) are the same information structures, but with different client signals. The OPUCn has a different information structure than the OPUk; the OPUCn information structure consists of n times the information structure of the OPU while the OPUk contain a single instance of the OPU information structure.

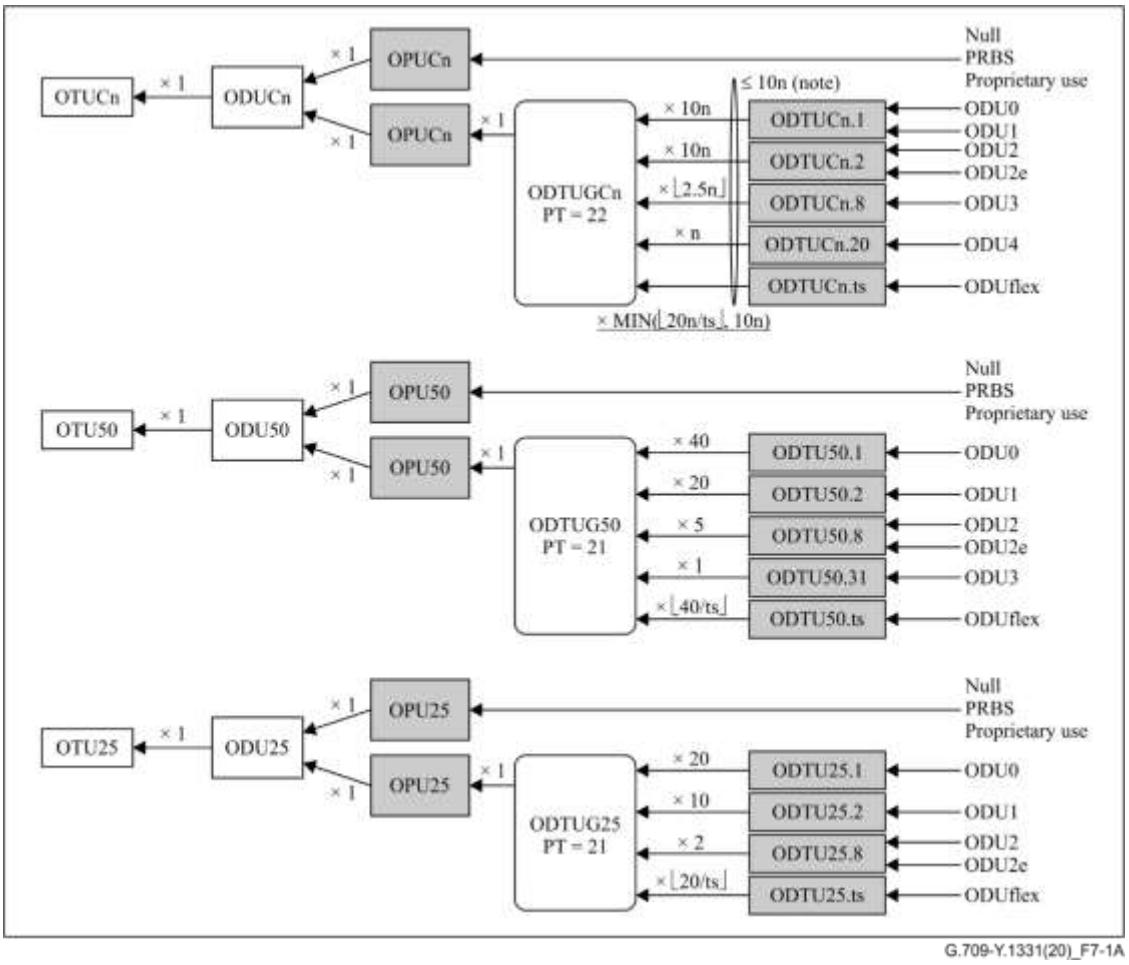
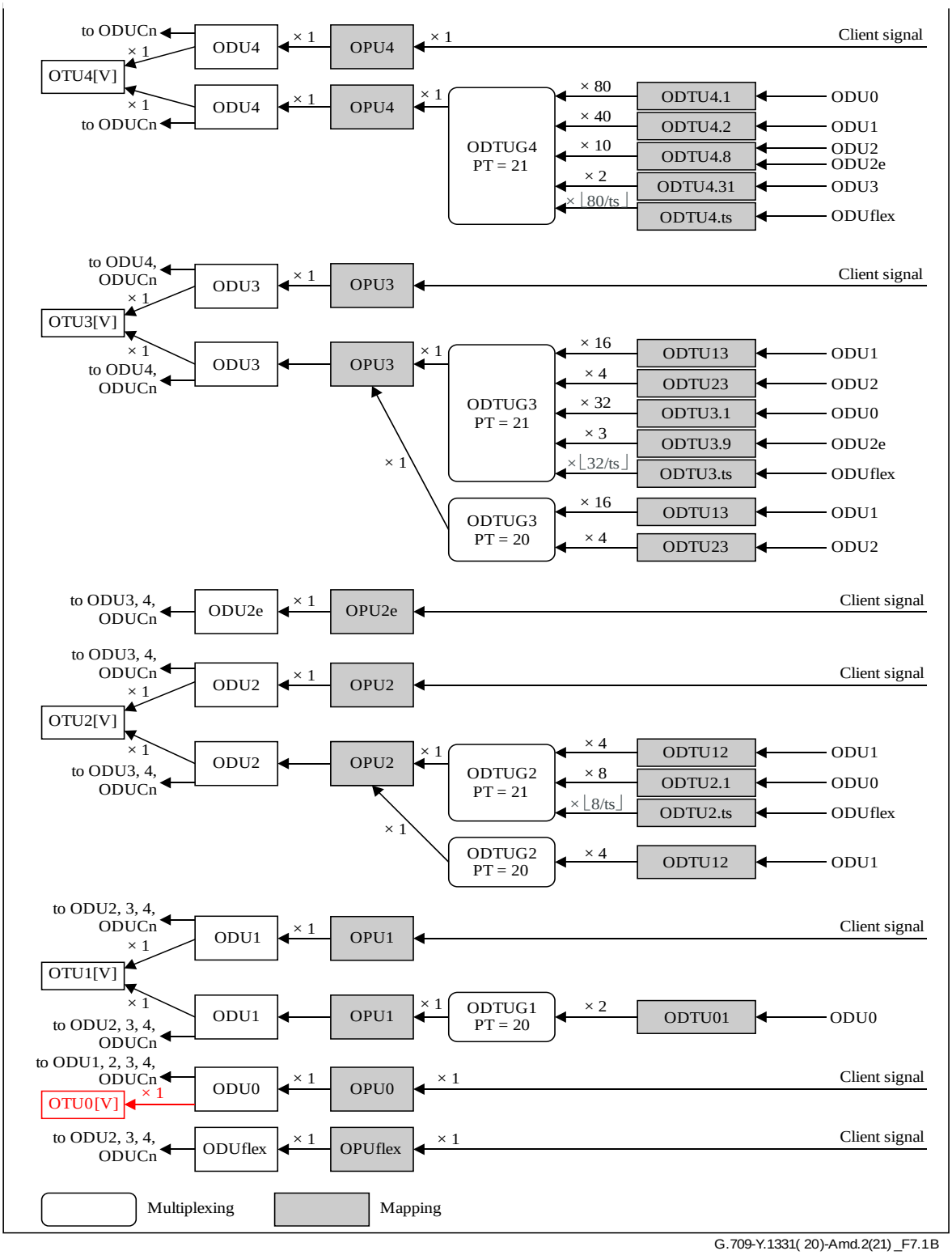


Figure 7-1A – OTN multiplexing and mapping structures



G.709-Y.1331(20)-Amd.2(21) _F7.1B

NOTE – Implementations should support the multiplexing of up to 10n ODUk (k=0,1,2,2e,3,4,flex) signals into an OPUCn.

Figure 7-1B – OTN multiplexing and mapping structures

7.1 Mapping

The client signal or an optical data tributary unit group (ODTUG) is mapped into the OPU. The OPU is mapped into an ODU and the ODU is mapped into an OTU. The OTU is mapped into an OTSiG. The OTUk may also be mapped into an OTLk.n and an OTLk.n is then mapped into an OTSiG.

7.2 Wavelength division multiplex

Wavelength division multiplexing is used to multiplex the OTSi of one or more OTSiG for transport over an OTN interface.

For the case of the OTN optical networking interfaces, type I, the OTSi of the OSC is multiplexed for transport over the interface using wavelength division multiplexing.

7.3 Bit rates and capacity

The bit rates and tolerance of the OTU signals are defined in Table 7-1.

The bit rates and tolerance of the ODU signals are defined in clause 12.2 and Table 7-2.

The bit rates and tolerance of the OPU payload are defined in Table 7-3.

The OTU, ODU and OPU frame periods are defined in Table 7-4.

The types and bit rates of the OTL signals are defined in Table 7-5.

The 2.5G and 1.25G tributary slot related OPUk multiframe periods and 5G tributary slot OPUCn multiframe periods are defined in Table 7-6.

The ODTU payload area bandwidths are defined in Table 7-7. The bandwidth depends on the OPU type and the mapping procedure (AMP or GMP). The AMP bandwidths include the bandwidth provided by the NJO overhead byte. GMP is defined without such NJO bytes.

The bit rates and tolerance of the ODUflex(GFP,n,k) are defined in Table 7-8.

The number of OPU tributary slots required by a client ODU are summarized in Table 7-9 and specified in clauses 19.6 and 20.5.

The bit rates and tolerance of the GCC signals are defined in Table 7-10.

Table 7-1 – OTU types and bit rates

OTU type	OTU nominal bit rate	OTU bit-rate tolerance
OTU0	255/239 × 1 244 160 kbit/s	±20 ppm (Note 5)
OTU1	255/238 × 2 488 320 kbit/s	
OTU2	255/237 × 9 953 280 kbit/s	
OTU3	255/236 × 39 813 120 kbit/s	
OTU4	255/227 × 99 532 800 kbit/s	
OTUCn	n × 239/226 × 99 532 800 kbit/s	
OTU25	61677/58112 × 24 883 200 kbit/s	
OTU50	61677/58112 × 49 766 400 kbit/s	

NOTE 1 – The nominal OTU rates are approximately: 1 327 451.046 kbit/s (OTU0), 2 666 057.143 kbit/s (OTU1), 10 709 225.316 kbit/s (OTU2), 43 018 413.559 kbit/s (OTU3), 111 809 973.568 kbit/s (OTU4), 26 409 711.013 kbit/s (OTU25), 52 819 422.026 kbit/s (OTU50) and n × 105 258 138.053 kbit/s (OTUCn).

NOTE 2 –OTU2e and OTUflex are not specified in this Recommendation. ODU2e signals are to be transported over ODU3, ODU4, ODU25, ODU50 and ODUCn signals and ODUflex signals are transported over ODU2, ODU3, ODU4, ODU25, ODU50 and ODUCn signals.

NOTE 3 – The OTUk (k=0,1,2,3,4) signal bit rates include the FEC overhead area. The OTUCn signal bit rates do not include a FEC overhead area.

NOTE 4 – The OTU25 [OTU50] bit rate can be based on the OTU25-RS [OTU50-RS] bit rate as follows: 514/544 × 41118/41120 × OTU25-RS [OTU50-RS] bit rate = 514/544 × 41118/41120 × 255/227 × 24 883 200 [49 766 400] kbit/s.

NOTE 5 – Refer to clause 12.2.1.1 for the bit rate tolerance of synchronous OTUk (k=0,1,2,3,4), OTU25/50 and OTUCn.

Table 7-2 – ODU types and bit rates

ODU type	ODU nominal bit rate	ODU bit-rate tolerance
ODU0	1 244 160 kbit/s	±20 ppm (Note 4)
ODU1	239/238 × 2 488 320 kbit/s	
ODU2	239/237 × 9 953 280 kbit/s	
ODU3	239/236 × 39 813 120 kbit/s	
ODU4	239/227 × 99 532 800 kbit/s	
ODU25	61677/58112 × 24 883 200 kbit/s	
ODU50	61677/58112 × 49 766 400 kbit/s	
ODUCn	n × 239/226 × 99 532 800 kbit/s	
ODU2e	239/237 × 10 312 500 kbit/s	±100 ppm
ODUflex for CBR sub 800G client signals	239/238 × client signal bit rate	±100 ppm (Notes 2, 3)
ODUflex for 800GBASE-R CBR client signals	239/238 × client signal bit rate	±50 ppm
ODUflex for GFP-F mapped client signals (ODUflex(GFP))	239/238 × OPUflex(GFP) payload signal rate	±100 ppm
ODUflex for GFP-F mapped client signals based on ODUk.ts and n 1.25G tributary slots (ODUflex(GFP,n,k))	Configured bit rate (see Table 7-8)	
ODUflex for IMP mapped client signals (ODUflex(IMP))	239/238 × OPUflex(IMP) payload signal rate	
ODUflex for IMP mapped FlexE Client signals (ODUflex(IMP,s))	s × 239/238 × 5 156 250 kbit/s s = 2, 8, n × 5 with n ≥ 1 (refer to clause 12.2.6)	
ODUflex for FlexE-aware client signals	103 125 000 × 240/238 × n/20 kbit/s ($n = n_1 + n_2 + .. + n_p$)	

NOTE 1 – The nominal ODU rates are approximately: 2 498 775.126 kbit/s (ODU1), 10 037 273.924 kbit/s (ODU2), 40 319 218.983 kbit/s (ODU3), 104 794 445.815 kbit/s (ODU4), 26 409 711.013 kbit/s (ODU25), 52 819 422.026 kbit/s (ODU50), 10 399 525.316 kbit/s (ODU2e), n × 105 258 138.053 kbit/s (ODUCn).

NOTE 2 – The bit-rate tolerance for ODUflex(CBR sub 800G) signals is specified as ±100 ppm. This value may be larger than the tolerance for the client signal itself (e.g., ±20 ppm). For such case, the tolerance is determined by the ODUflex(CBR sub 800G) maintenance signals, which have a tolerance of ±100 ppm.

NOTE 3 – For ODUflex(CBR sub 800G) signals with nominal bit rates close to the maximum ODTUk.ts payload bit rate and client rate tolerances less than ±100 ppm (e.g., ±10 ppm), the ODUflex(CBR sub 800G) maintenance signal bit rates may exceed the ODTUk.ts payload bit rate. For such cases either an additional tributary slot may be used (i.e., ODTUk.(ts+1)), or the nominal bit rate of the ODUflex(CBR sub 800G) signal may be artificially reduced to a value of 100 ppm below the maximum ODUflex(CBR sub 800G) signal bit rate.

NOTE 4 – Refer to clause 12.2.1.1 for the bit rate tolerance of synchronous ODUk (k=1,2,3,4), ODU25/50 and ODUCn.

Table 7-3 – OPU types and bit rates

OPU type	OPU payload nominal bit rate	OPU payload bit-rate tolerance
OPU0	238/239 × 1 244 160 kbit/s	±20 ppm (Note 2)
OPU1	2 488 320 kbit/s	
OPU2	238/237 × 9 953 280 kbit/s	
OPU3	238/236 × 39 813 120 kbit/s	
OPU4	238/227 × 99 532 800 kbit/s	
OPU25	7339563/6944384 × 24 883 200 kbit/s	
OPU50	7339563/6944384 × 49 766 400 kbit/s	
OPUCn	n × 238/226 × 99 532 800 kbit/s	
OPU2e	238/237 × 10 312 500 kbit/s	±100 ppm
OPUflex for sub-800G CBR client signals	client signal bit rate	client signal bit-rate tolerance, with a maximum of ±100 ppm
OPUflex for 800GBASE-R CBR client signals	238/237 × 800 000 000 kbit/s	client signal bit-rate tolerance, with a maximum of ±50 ppm
OPUflex for GFP-F mapped client signals (OPUflex(GFP)) (Note 3)	Configured bit rate (refer to clause 12.2.5)	±100 ppm
OPUflex for GFP-F mapped client signals based on ODUk.ts and n 1.25G tributary slots (ODUflex(GFP,n,k) (Note 3)	238/239 × ODUflex(GFP,n,k) signal rate	
OPUflex for IMP mapped client signals (OPUflex(IMP))	Configured bit rate (refer to clause 12.2.6)	
OPUflex for IMP mapped FlexE Client signals OPUflex(IMP,s))	s × 5 156 250 kbit/s s = 2, 8, n × 5 with n ≥ 1 (refer to clause 12.2.6)	
OPUflex for FlexE-aware client signals	103 125 000 × 240/239 × n/20 kbit/s ($n = n_1 + n_2 + .. + n_p$)	

NOTE 1 – The nominal OPU payload rates are approximately: 1 238 954.310 kbit/s (OPU0 Payload), 2 488 320.000 kbit/s (OPU1 Payload), 9 995 276.962 kbit/s (OPU2 Payload), 40 150 519.322 kbit/s (OPU3 Payload), 104 355 975.330 (OPU4 Payload), 26 299 210.130 kbit/s (OPU25 Payload), 52 598 420.261 kbit/s (OPU50 Payload), 10 356 012.658 kbit/s (OPU2e Payload), n × 104 817 727.434 kbit/s (OPUCn Payload).

NOTE 2 – Refer to clause 12.2.1.1 for the bit rate tolerance of synchronous OxUk (k=1,2,3,4), OxU25/50 and OxUCn.

NOTE 3 – Transport of packet clients with GFP mapping is specified up to 100 Gbit/s.

Table 7-4 – OTU/ODU/OPU frame periods

OTU/ODU/OPU type	Period (Note)
OTU0/ODU0/OPU0	98.354 μs
OTU1/ODU1/OPU1	48.971 μs
OTU2/ODU2/OPU2	12.191 μs
OTU3/ODU3/OPU3	3.035 μs
OTU4/ODU4/OPU4	1.168 μs
OTU25/ODU25/OPU25	4.633 μs
OTU50/ODU50/OPU50	2.317 μs
ODU2e/OPU2e	11.767 μs
OTUCn/ODUCn/OPUCn	1.163 μs
ODUflex/OPUflex	CBR client signals: 121856/client_signal_bit_rate
	GFP-F mapped client signals: 122368/ODUflex_bit_rate
	IMP mapped client signals: 122368/ODUflex_bit_rate
	FlexE-aware client signals: 122368/ODUflex_bit_rate
NOTE – The period is an approximated value, rounded to 3 decimal places.	

Table 7-5 – OTL types and bit rates

OTL type	OTL nominal bit rate	OTL bit-rate tolerance
OTL3.4	4 lanes of 255/236 × 9 953 280 kbit/s	±20 ppm
OTL4.4	4 lanes of 255/227 × 24 883 200 kbit/s	
NOTE – The nominal OTL rates are approximately: 10 754 603.390 kbit/s (OTL3.4) and 27 952 493.392 kbit/s (OTL4.4).		

Table 7-6 – OPUk multiframe periods for 2.5G and 1.25G tributary slots and ODUCn multiframe periods for 5G tributary slots

OPU type	1.25G tributary slot multiframe period (Note)	2.5G tributary slot multiframe period (Note)	5G tributary slot multiframe period (Note)
OPU1	97.942 μs	–	–
OPU2	97.531 μs	48.765 μs	–
OPU3	97.119 μs	48.560 μs	–
OPU4	93.416 μs	–	–
OPU25	92.669 μs	–	–
OPU50	92.669 μs	–	–
OPUCn	–	–	23.251 μs
NOTE – The period is an approximated value, rounded to 3 decimal places.			

Table 7-7 – ODTU payload bandwidth (kbit/s)

ODTU type	ODTU payload nominal bandwidth		ODTU payload bit-rate tolerance
ODTU01	(1904 + 1/8)/3824 × ODU1 bit rate		±20 ppm
ODTU12	(952 + 1/16)/3824 × ODU2 bit rate		
ODTU13	(238 + 1/64)/3824 × ODU3 bit rate		
ODTU23	(952 + 4/64)/3824 × ODU3 bit rate		
ODTU2.ts	ts × 476/3824 × ODU2 bit rate		
ODTU3.ts	ts × 119/3824 × ODU3 bit rate		
ODTU4.ts	ts × 47.5/3824 × ODU4 bit rate		
ODTU25.ts	ts × 190.4/3824 × ODU25 bit rate		
ODTU50.ts	ts × 95.2/3824 × ODU50 bit rate		
ODTUCn.ts	ts × 190.4/3824 × ODU*Cn* bit rate/n		
	Minimum	**Nominal**	**Maximum**
ODTU01	1 244 216.796	1 244 241.681	1 244 266.566
ODTU12	2 498 933.311	2 498 983.291	2 499 033.271
ODTU13	2 509 522.012	2 509 572.203	2 509 622.395
ODTU23	10 038 088.048	10 038 288.814	10 038 489.579
ODTU2.ts	ts × 1 249 384.632	ts × 1 249 409.620	ts × 1 249 434.608
ODTU3.ts	ts × 1 254 678.635	ts × 1 254 703.729	ts × 1 254 728.823
ODTU4.ts	ts × 1 301 683.217	ts × 1 301 709.251	ts × 1 301 735.285
ODTU25.ts, ODTU50.ts	ts × 1 314 934.207	ts × 1 314 960.507	ts × 1 314 986.806
ODTUCn.ts	ts × 5 240 781.554	ts × 5 240 886.372	ts × 5 240 991.189
NOTE – The bandwidth is an approximated value, rounded to 3 decimal places.			

Table 7-8 – Recommended ODUflex (GFP,n,k) bit rates and tolerance

ODU type	Nominal bit-rate	Tolerance
ODU2.ts (Note 1)	1 249 177.230 kbit/s	
ODU3.ts (Note 1)	1 254 470.354 kbit/s	
ODU4.ts (Note 1)	1 301 467.133 kbit/s	
ODUflex(GFP,n,2) of n 1.25G tributary slots, 1 ≤ n ≤ 8	n × ODU2.ts	±100 ppm
ODUflex(GFP,n,3) of n 1.25G tributary slots, 9 ≤ n ≤ 32 (Note 3)	n × ODU3.ts	±100 ppm
ODUflex(GFP,n,4) of n 1.25G tributary slots, 33 ≤ n ≤ 80 (Note 2)	n × ODU4.ts	±100 ppm
NOTE 1 – The values of ODUk.ts are chosen to permit a variety of methods to be used to generate an ODUflex(GFP) clock. See Appendix XI for the derivation of these values and example ODUflex(GFP) clock generation methods. NOTE 2 – Transport of packet clients via an ODUflex(GFP) or ODUflex(GFP,n.k) is specified up to 100 Gbit/s.		

Table 7-8 – Recommended ODUflex (GFP,n,k) bit rates and tolerance

NOTE 3 – The value of n may be larger than 8 for the case where an ODU25u or ODU50u is used as server. Refer to Annex L.

Table 7-9 – Number of tributary slots required for ODUj into OPUk and for ODUk into OPUCn

	# 5G TS	# 2.5G tributary slots		# 1.25G tributary slots					
	OPUCn	OPU2	OPU3	OPU1	OPU2	OPU3	OPU4	OPU25	OPU50
ODU0	1	–	–	1	1	1	1	1	1
ODU1	1	1	1	–	2	2	2	2	2
ODU2	2	–	4	–	–	8	8	8	8
ODU2e	2	–	–	–	–	9	8	8	8
ODU3	8	–	–	–	–	–	31	–	31
ODU4	20	–	–	–	–	–	–	–	–
ODUflex(CBR)									
– ODUflex(IB SDR)	1	–	–	–	3	3	2	2	2
– ODUflex(IB DDR)	1	–	–	–	5	5	4	4	4
– ODUflex(IB QDR)	2	–	–	–	–	9	8	8	8
– ODUflex(FC-400)	1	–	–	–	4	4	4	4	4
– ODUflex(FC-800)	2	–	–	–	7	7	7	7	7
– ODUflex(FC-1600)	3	–	–	–	–	12	11	11	11
– ODUflex(FC-3200)	6	–	–	–	–	23	22	–	22
– ODUflex(3G SDI) (2 970 000)	1	–	–	–	3	3	3	3	3
– ODUflex(3G SDI) (2 970 000/1.001)	1	–	–	–	3	3	3	3	3
– ODUflex(25GBASEase-R)	5	–	–	–	–	21	20	20	20
– ODUflex(50GBASEase-R)	10	–	–	–	–	–	40	–	40
– ODUflex(200GBASEase-R)	40	–	–	–	–	–	–	–	–
– ODUflex(400GBASEase-R)	80	–	–	–	–	–	–	–	–
– ODUflex(800GBASE-R)	154 (Note 2)	–	–	–	–	–	–	–	–
ODUflex(GFP)	Note	–	–	–	Note 1	Note 1	Note 1	Note 1	Note 1
– ODUflex(GFP,n,k)	Note	–	–	–	n	n	n	n	n
ODUflex(IMP)	Note	–	–	–	Note 1	Note 1	Note 1	Note 1	Note 1
– ODUflex(IMP,s)	Note	–	–	–	–	Note 1	Note 1	Note 1	Note 1
ODUflex(FlexE)	Note	–	–	–	–	Note 1	Note 1	Note 1	Note 1

NOTE 1 – Refer to equations 19-1a, 19-1b, 20-1a and 20-1b in clauses 19.6 and 20.5.

NOTE 2 – In some applications, 160 5G tributary slots are used to carry an ODUflex(800GBASE-R)

Table 7-10 – GCC types and bit rates (kbit/s)

OTU/ODU type	Nominal bit rate GCC0	Nominal bit rate GCCi (i = 1,2)	Nominal bit rate GCC1+GCC2	Bit-rate tolerance
OTU0/ODU0	162.687	162.687	325.356	±20 ppm
OTU1/ODU1	326.723	326.723	653.445	
OTU2/ODU2	1 312.405	1 312.405	2 624.810	
OTU3/ODU3	5 271.864	5 271.864	10 543.729	
OTU4/ODU4	13 702.203	13 702.203	27 404.405	
OTUCn/ODUCn	n × 13 762.832	n × 13 762.832	n × 27 525.664	
OTU25/ODU25	3 453.153	3 453.153	6 906.305	
OTU50/ODU50	6 906.305	6 906.305	13 812.610	
ODU2e	N/A	1 359.771	2 719.541	±100 ppm
ODUflex for CBR client signals	N/A	ODUflex(CBR) / 7648	ODUflex(CBR) / 3824	
ODUflex(GFP)	N/A	ODUflex(GFP) / 7648	ODUflex(GFP) / 3824	
ODUflex(GFP,n,k)	N/A	ODUflex(GFP,n,k) / 7648	ODUflex(GFP,n,k) / 3824	
ODUflex(IMP)	N/A	ODUflex(IMP) / 7648	ODUflex(IMP) / 3824	
ODUflex(IMP,s)	N/A	ODUflex(IMP,s) / 7648	ODUflex(IMP,s) / 3824	
ODUflex for FlexE-aware client signals	N/A	ODUflex(FlexE) / 7648	ODUflex(FlexE) / 3824	
NOTE – The bandwidth is an approximated value, rounded to 3 decimal places.				

7.4 ODUk time-division multiplex

Figure 7-1 shows the relationship between various time-division multiplexing elements that are defined below and illustrates possible multiplexing structures. Table 7-11 provides an overview of valid tributary slot types and mapping procedure configuration options.

Up to 2 ODU0 signals are multiplexed into an ODTUG1 (PT=20) using time-division multiplexing. The ODTUG1 (PT=20) is mapped into the OPU1.

Up to 4 ODU1 signals are multiplexed into an ODTUG2 (PT=20) using time-division multiplexing. The ODTUG2 (PT=20) is mapped into the OPU2.

A mixture of p ($p \leq 4$) ODU2 and q ($q \leq 16$) ODU1 signals can be multiplexed into an ODTUG3 (PT=20) using time-division multiplexing. The ODTUG3 (PT=20) is mapped into the OPU3.

A mixture of p ($p \leq 8$) ODU0, q ($q \leq 4$) ODU1, r ($r \leq 8$) ODUflex signals can be multiplexed into an ODTUG2 (PT=21) using time-division multiplexing. The ODTUG2 (PT=21) is mapped into the OPU2.

A mixture of p ($p \leq 32$) ODU0, q ($q \leq 16$) ODU1, r ($r \leq 4$) ODU2, s ($s \leq 3$) ODU2e and t ($t \leq 32$) ODUflex signals can be multiplexed into an ODTUG3 (PT=21) using time-division multiplexing. The ODTUG3 (PT=21) is mapped into the OPU3.

A mixture of p ($p \leq 80$) ODU0, q ($q \leq 40$) ODU1, r ($r \leq 10$) ODU2, s ($s \leq 10$) ODU2e, t ($t \leq 2$) ODU3 and u ($u \leq 80$) ODUflex signals can be multiplexed into an ODTUG4 (PT=21) using time-division multiplexing. The ODTUG4 (PT=21) is mapped into the OPU4.

NOTE 1 – The ODTUGk is a logical construct and is not defined further. ODTUjk and ODTUk.ts signals are directly time-division multiplexed into the tributary slots of an OPUk.

A mixture of p (p ≤ 20) ODU0, q (q ≤ 10) ODU1, r (r ≤ 2) ODU2, s (s ≤ 2) ODU2e and u (u ≤ 20) ODUflex signals can be multiplexed into an ODTUG25 (PT=21) using time-division multiplexing. The ODTUG25 (PT=21) is mapped into the OPU25.

A mixture of p (p ≤ 40) ODU0, q (q ≤ 20) ODU1, r (r ≤ 5) ODU2, s (s ≤ 5) ODU2e, t (t ≤ 1) ODU3 and u (u ≤ 40) ODUflex signals can be multiplexed into an ODTUG50 (PT=21) using time-division multiplexing. The ODTUG50 (PT=21) is mapped into the OPU50.

A mixture of p (p ≤ 10n) ODU0, q (q ≤10n) ODU1, r (r ≤ 10n) ODU2, s (s ≤ 10n) ODU2e, t (t ≤int(10n/4)) ODU3, u (u ≤ n) ODU4 and v (v ≤ 10n) ODUflex signals can be multiplexed into an ODTUGCn (PT=22) using time-division multiplexing. The ODTUGCn (PT=22) is mapped into the OPUCn.

NOTE 2 – The ODTUGk and ODTUGCn are logical constructs and are not defined further. ODTUjk and ODTUk.ts signals are directly time-division multiplexed into the tributary slots of an OPUk. ODTUCn.ts signals are directly time-division multiplexed into the tributary slots of an OPUCn.

NOTE 3 – Implementations should support the multiplexing of up to 10n ODUk (k=0,1,2,2e,3,4,flex) signals into an OPUCn. Support for the multiplexing of up to 20n ODUk (k=0,1,flex) signals into the OPUCn is not required.

Table 7-11 – Overview of ODUj into OPUk and ODUk into OPUCn mapping types

	5G tributary slots	**2.5G tributary slots**		**1.25G tributary slots**					
	OPUCn (PT=22)	**OPU2 (PT=20)**	**OPU3 (PT=20)**	**OPU1 (PT=20)**	**OPU2 (PT=21)**	**OPU3 (PT=21)**	**OPU4 (PT=21)**	**OPU25 (PT=21)**	**OPU50 (PT=21)**
ODU0	GMP – Note	–	–	AMP	GMP	GMP	GMP	GMP	GMP
ODU1	GMP – Note	AMP	AMP	–	AMP	AMP	GMP	GMP	GMP
ODU2	GMP	–	AMP	–	–	AMP	GMP	GMP	GMP
ODU2e	GMP	–	–	–	–	GMP	GMP	GMP	GMP
ODU3	GMP	–	–	–	–	–	GMP	–	GMP
ODU4	GMP	–	–	–	–	–	–	–	–
ODUflex	GMP	–	–	–	GMP	GMP	GMP	GMP	GMP
NOTE – Mapping ODU0 and ODU1 into a 5G tributary slot of the OPUCn does not fully occupy the tributary slot's bandwidth.									

Figures 7-2, 7-3 and 7-4 show how various signals are multiplexed using the ODTUG1/2/3 (PT=20) multiplexing elements. Figure 7-2 presents the multiplexing of four ODU1 signals into the OPU2 signal via the ODTUG2 (PT=20). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 1 into 2 (ODTU12) using the AMP justification overhead (JOH). The four ODTU12 signals are time-division multiplexed into the optical data tributary unit group 2 (ODTUG2) with payload type 20, after which this signal is mapped into the OPU2.

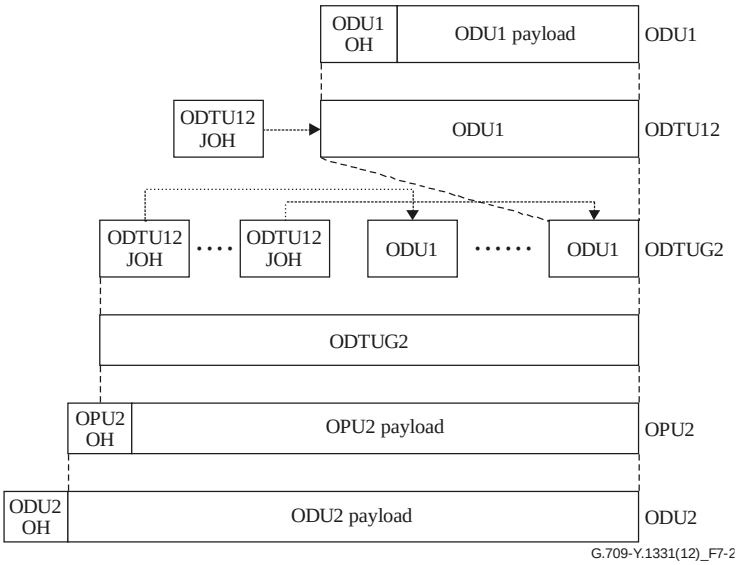


Figure 7-2 – ODU1 into ODU2 multiplexing method via ODTUG2 (PT=20)

Figure 7-3 presents the multiplexing of up to 16 ODU1 signals and/or up to 4 ODU2 signals into the OPU3 signal via the ODTUG3 (PT=20). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 1 into 3 (ODTU13) using the AMP justification overhead (JOH). An ODU2 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 2 into 3 (ODTU23) using the AMP justification overhead (JOH). "x" ODTU23 ($0 \leq x \leq 4$) signals and "16-4x" ODTU13 signals are time-division multiplexed into the optical data tributary unit group 3 (ODTUG3) with payload type 20, after which this signal is mapped into the OPU3.

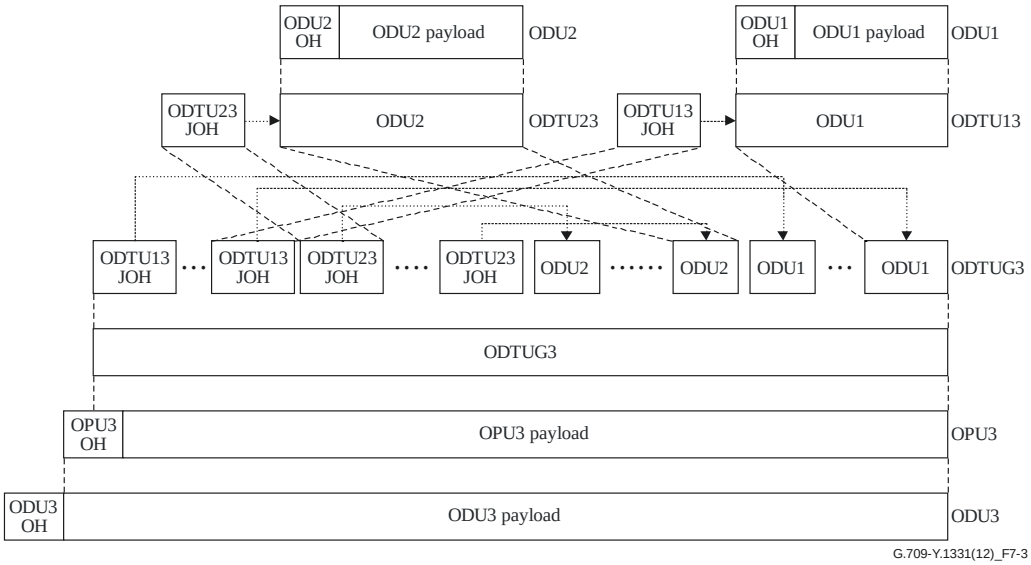


Figure 7-3 – ODU1 and ODU2 into ODU3 multiplexing method via ODTUG3 (PT=20)

Figure 7-4 presents the multiplexing of two ODU0 signals into the OPU1 signal via the ODTUG1 (PT=20). An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 0 into 1 (ODTU01) using the AMP justification overhead (JOH). The two ODTU01 signals are time-division multiplexed into the optical data tributary unit group 1 (ODTUG1) with payload type 20, after which this signal is mapped into the OPU1.

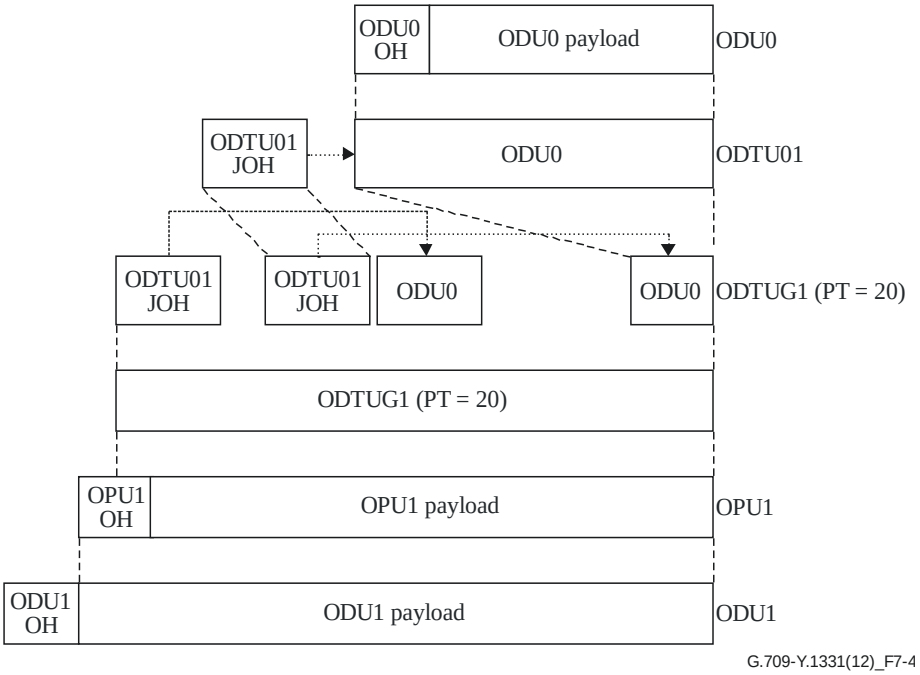


Figure 7-4 – ODU0 into ODU1 multiplexing method via ODTUG1 (PT=20)

Figures 7-5, 7-6 and 7-7 show how various signals are multiplexed using the ODTUG2/3/4 (PT=21) multiplexing elements.

Figure 7-5 presents the multiplexing of up to eight ODU0 signals, and/or up to four ODU1 signals and/or up to eight ODUflex signals into the OPU2 signal via the ODTUG2 (PT=21). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 1 into 2 (ODTU12) using the AMP justification overhead (JOH). An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 2.1 (ODTU2.1) using the GMP justification overhead. An ODUflex signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 2.ts (ODTU2.ts) using the GMP justification overhead. Up to eight ODTU2.1 signals, up to four ODTU12 signals and up to eight ODTU2.ts signals are time-division multiplexed into the optical data tributary unit group 2 (ODTUG2) with payload type 21, after which this signal is mapped into the OPU2.

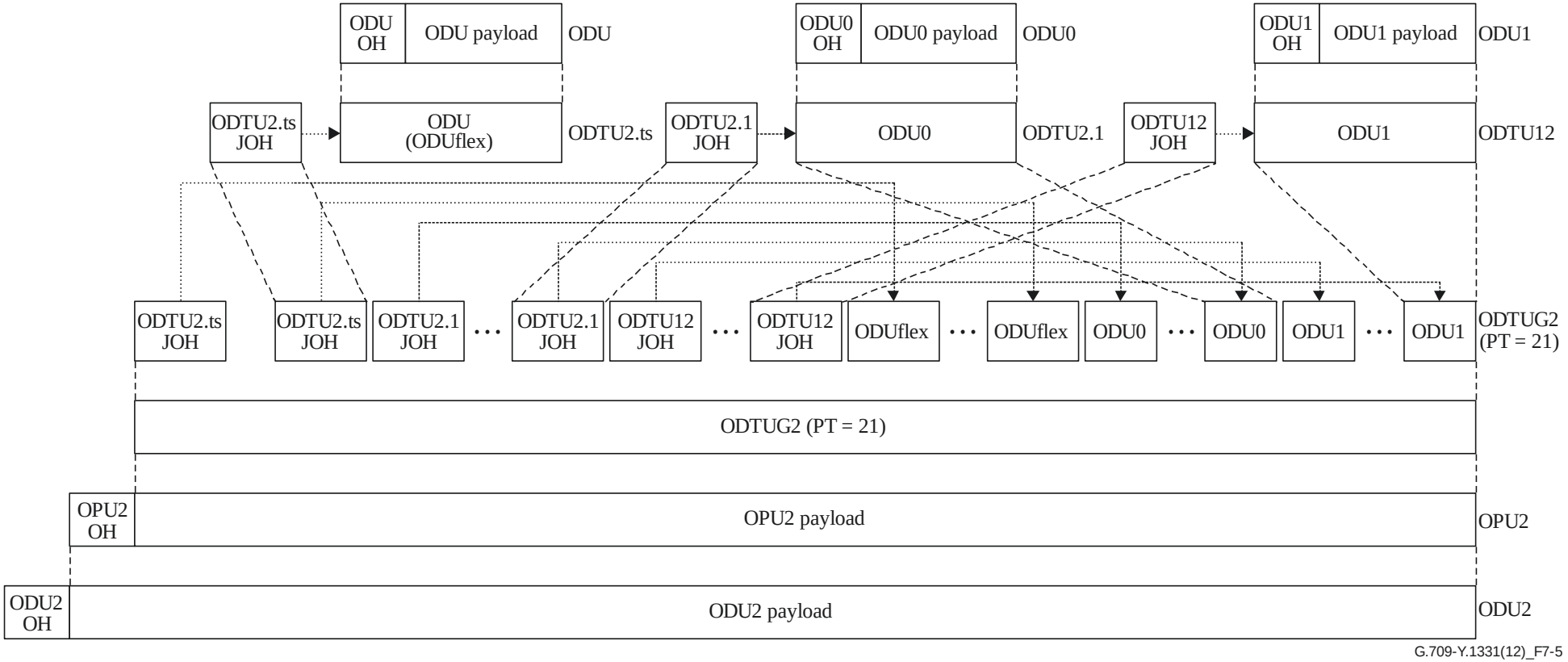


Figure 7-5 – ODU0, ODU1 and ODUflex into ODU2 multiplexing method via ODTUG2 (PT=21)

Figure 7-6 presents the multiplexing of up to thirty-two ODU0 signals and/or up to sixteen ODU1 signals and/or up to four ODU2 signals and/or up to three ODU2e signals and/or up to thirty-two ODUflex signals into the OPU3 signal via the ODTUG3 (PT=21). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 1 into 3 (ODTU13) using the AMP justification overhead (JOH). An ODU2 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 2 into 3 (ODTU23) using the AMP justification overhead. An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 3.1 (ODTU3.1) using the GMP justification overhead. An ODU2e signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 3.9 (ODTU3.9) using the GMP justification overhead. An ODUflex signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 3.ts (ODTU3.ts) using the GMP justification overhead. Up to thirty-two ODTU3.1 signals, up to sixteen ODTU13 signals, up to four ODTU23 signals, up to three ODTU3.9 and up to thirty-two ODTU3.ts signals are time-division multiplexed into the optical data tributary unit group 3 (ODTUG3) with payload type 21, after which this signal is mapped into the OPU3.

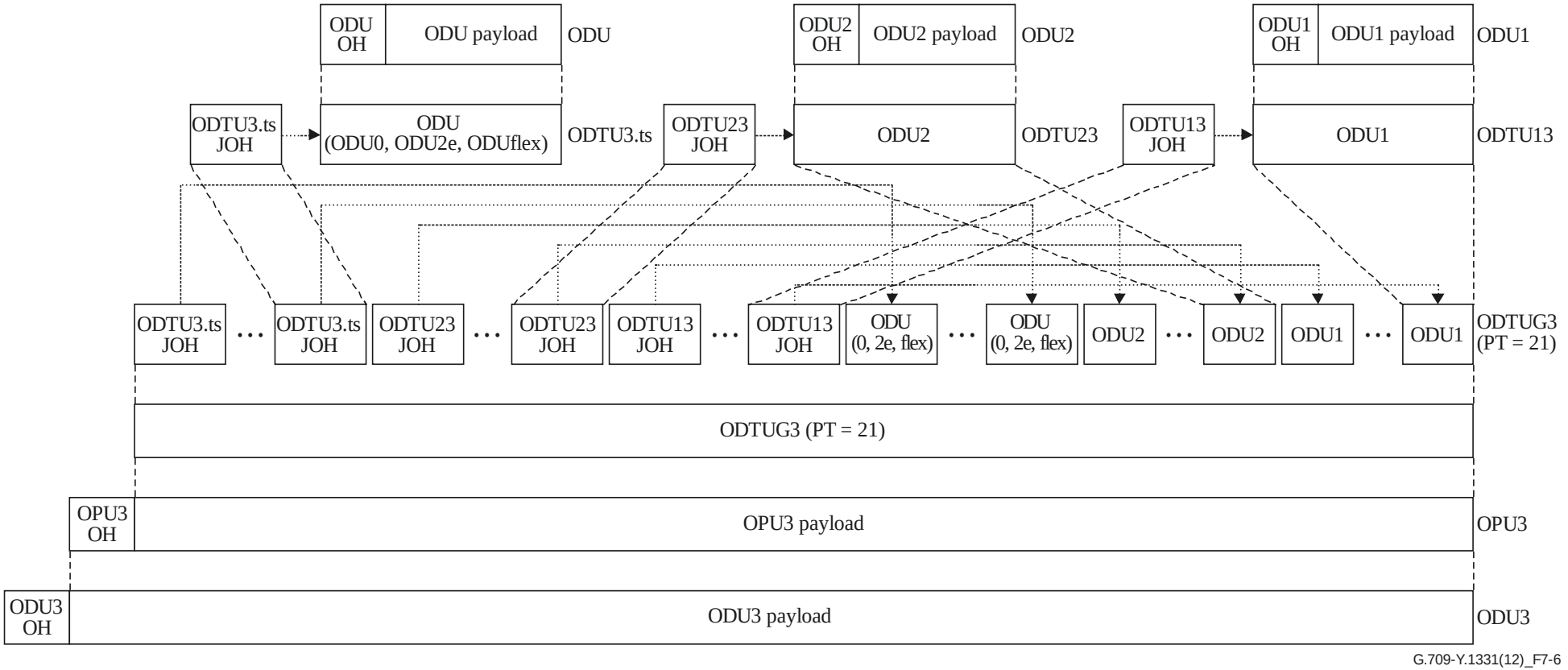


Figure 7-6 – ODU0, ODU1, ODU2, ODU2e and ODUflex into ODU3 multiplexing method via ODTUG3 (PT=21)

Figure 7-7 presents the multiplexing of up to eighty ODU0 signals and/or up to forty ODU1 signals and/or up to ten ODU2 signals and/or up to ten ODU2e signals and/or up to two ODU3 signals and/or up to eighty ODUflex signals into the OPU4 signal via the ODTUG4 (PT=21). An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.1 (ODTU4.1) using the GMP justification overhead (JOH). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.2 (ODTU4.2) using the GMP justification overhead. An ODU2 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.8 (ODTU4.8) using the GMP justification overhead (JOH). An ODU2e signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.8 (ODTU4.8) using the GMP justification overhead. An ODU3 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.31 (ODTU4.31) using the GMP justification overhead. An ODUflex signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 4.ts (ODTU4.ts) using the GMP justification overhead (JOH). Up to eighty ODTU4.1 signals, up to forty ODTU4.2 signals, up to ten ODTU4.8 signals, up to two ODTU4.31 and up to eighty ODTU4.ts signals are time-division multiplexed into the optical data tributary unit group 4 (ODTUG4) with payload type 21, after which this signal is mapped into the OPU4.

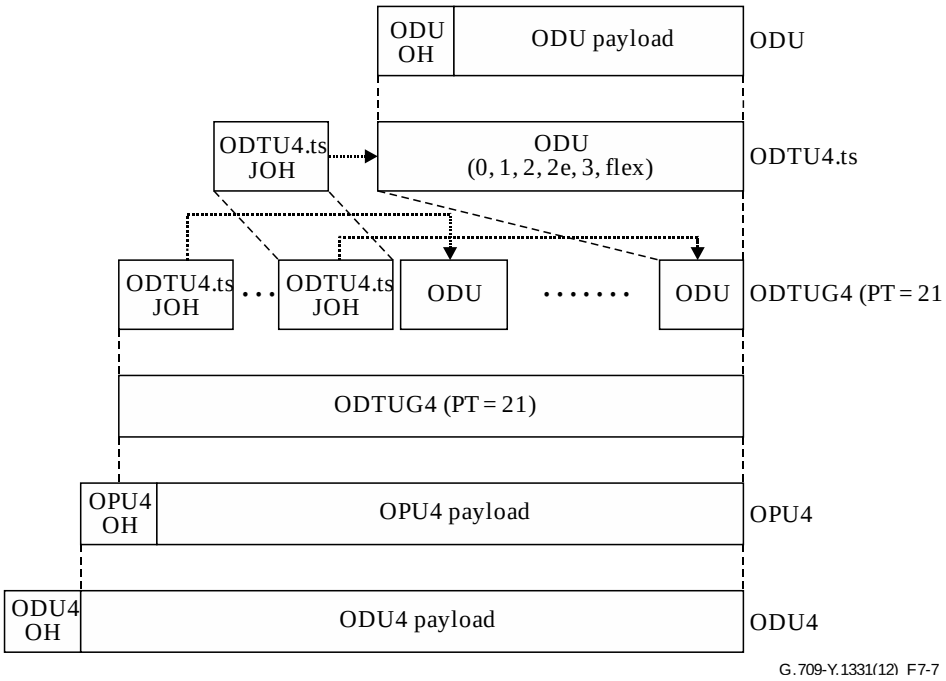


Figure 7-7 – ODU0, ODU1, ODU2, ODU2e, ODU3 and ODUflex into ODU4 multiplexing method via ODTUG4 (PT=21)

Figure 7-8 presents the multiplexing of up to 10n ODU0 signals and/or up to 10n ODU1 signals and/or up to 10n ODU2 signals and/or up to 10n ODU2e signals and/or up to $\lfloor 2.5n \rfloor$ ODU3 signals and/or up to n ODU4 signals and/or up to 10n ODUflex signals into the OPUCn signal via the ODTUGCn (PT=22). An ODUk signal is extended with frame alignment overhead and asynchronously mapped into the Optical Data Tributary Unit Cn.ts (ODTUCn.ts) (<k,ts> = <0,1>, <1,1>, <2,2>, <2e,2>, <3,8>, <4,20>, <flex,ts>) using the GMP justification overhead (JOH). Up to 10n ODTUCn.1 signals, up to 10n ODTUCn.2 signals, up to $\lfloor 2.5n \rfloor$ ODTUCn.8 signals, up to n ODTUCn.10 signals and up to 10n ODTUCn.ts signals are time-division multiplexed into the Optical

Data Tributary Unit Group Cn (ODTUGCn) with Payload Type 22, after which this signal is mapped into the OPUCn.

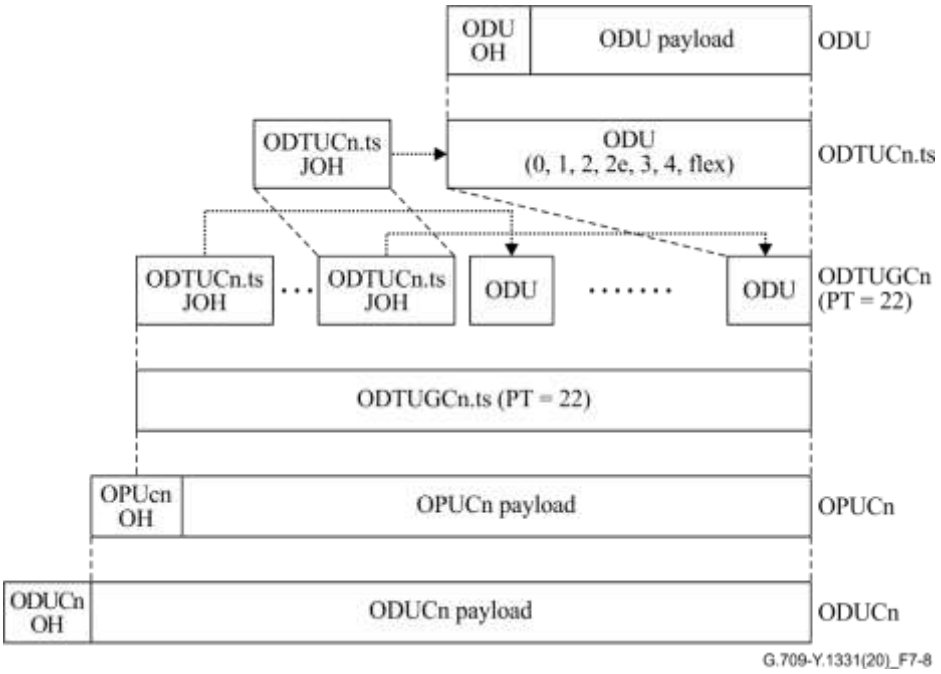


Figure 7-8 – ODU0, ODU1, ODU2, ODU2e, ODU3, ODU4 and ODUflex into ODUCn multiplexing method via ODTUGCn (PT=22)

Figure 7-9 presents the multiplexing of up to twenty ODU0 signals and/or up to ten ODU1 signals and/or up to two ODU2 signals and/or up to two ODU2e signals and/or up to twenty ODUflex signals into the OPU25 signal via the ODTUG25 (PT=21). An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 25.1 (ODTU25.1) using the GMP justification overhead (JOH). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 25.2 (ODTU25.2) using the GMP justification overhead. An ODU2 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 25.8 (ODTU25.8) using the GMP justification overhead (JOH). An ODU2e signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 25.8 (ODTU25.8) using the GMP justification overhead. An ODUflex signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 25.ts (ODTU25.ts) using the GMP justification overhead (JOH). Up to twenty ODTU25.1 signals, up to ten ODTU25.2 signals, up to two ODTU25.8 signals and up to twenty ODTU25.ts signals are time-division multiplexed into the optical data tributary unit group 25 (ODTUG25) with payload type 21, after which this signal is mapped into the OPU25.

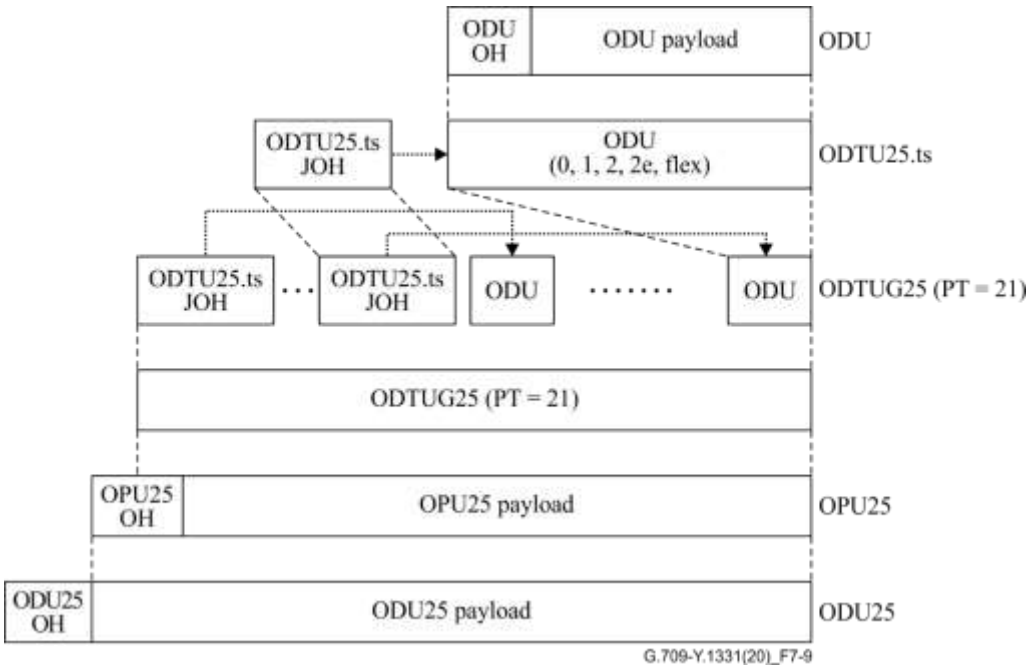


Figure 7-9 – ODU0, ODU1, ODU2, ODU2e and ODUflex into ODU25 multiplexing method via ODTUG25 (PT=21)

Figure 7-10 presents the multiplexing of up to forty ODU0 signals and/or up to twenty ODU1 signals and/or up to five ODU2 signals and/or up to five ODU2e signals and/or up to one ODU3 signals and/or up to forty ODUflex signals into the OPU50 signal via the ODTUG50 (PT=21). An ODU0 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.1 (ODTU50.1) using the GMP justification overhead (JOH). An ODU1 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.2 (ODTU50.2) using the GMP justification overhead. An ODU2 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.8 (ODTU50.8) using the GMP justification overhead (JOH). An ODU2e signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.8 (ODTU50.8) using the GMP justification overhead. An ODU3 signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.31 (ODTU50.31) using the GMP justification overhead. An ODUflex signal is extended with a frame alignment overhead and asynchronously mapped into the optical data tributary unit 50.ts (ODTU50.ts) using the GMP justification overhead (JOH). Up to forty ODTU50.1 signals, up to twenty ODTU50.2 signals, up to five ODTU50.8 signals, up to one ODTU50.31 and up to forty ODTU50.ts signals are time-division multiplexed into the optical data tributary unit group 50 (ODTUG50) with payload type 21, after which this signal is mapped into the OPU50.

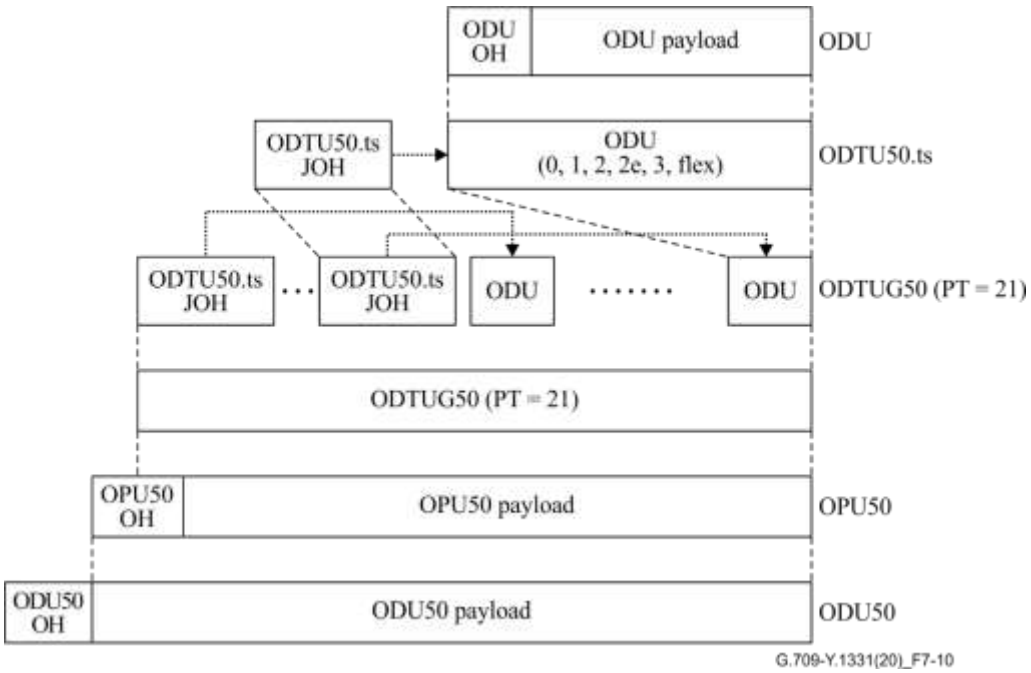


Figure 7-10 – ODU0, ODU1, ODU2, ODU2e, ODU3 and ODUflex into ODU50 multiplexing method via ODTUG50 (PT=21)

Details of the multiplexing method and mappings are given in clause 19 for OPUk and clause 20 for OPUCn.

Some examples illustrating the multiplexing of 2 ODU0 signals into an ODU1 and of 4 ODU1 signals into an ODU2 are presented in Appendix III.

7.5 Interconnection of Ethernet UNI and FlexE Group UNI in two administrative domains in the OTN

Ethernet PCS encoded client signals and Ethernet MAC client signals can be applied to the OTN via an Ethernet UNI and a FlexE group UNI.

The mapping method specified in clause 17 for the Ethernet PCS encoded client signals when applied via an Ethernet UNI are BMP, TTT+GMP and GMP, and via a FlexE group UNI is IMP.

The mapping method specified in clause 17 for the Ethernet MAC client signals with a bit rate up to and including 100 Gbit/s when applied via an Ethernet UNI is GFP-F and optionally can be IMP, and via a FlexE group UNI is IMP. The mapping method specified in clause 17 for both Ethernet MAC client and FlexE Client signals with a bit rate including and beyond 100 Gbit/s is IMP.

For the case of Ethernet PCS encoded client services between an Ethernet UNI in one administrative domain and a FlexE group UNI in another administrative domain, the rule for interconnecting an Ethernet PCS encoded client signal will be to use BMP, TTT+GMP or GMP on the interface between the two administrative domains.

For the case of Ethernet MAC client services with a bit rate up to and including 100 Gbit/s between an Ethernet UNI that applies the GFP-F mapping method in one administrative domain and a FlexE group UNI or an Ethernet UNI that applies the IMP mapping method in another administrative domain, the rule for interconnecting an Ethernet MAC client signal with a bit rate up to 100 Gbit/s will be to use GFP-F on the interface between the two administrative domains.

NOTE – For the case of Ethernet MAC client services with a bit rate including and beyond 100 Gbit/s between an Ethernet UNI in one administrative domain and a FlexE group UNI or an Ethernet UNI in another administrative domain, IMP mapping is used as the interconnect for the interface between the two administrative domains.

8 OTN Interfaces

8.1 Point-to-point interface, type I

The OTN point-to-point interface type I supports either one OTU ~~on one OTSiG~~ or one FlexO-x-<~~fec~~int> ~~(out of one OTUCn/FlexO-x-<fec>-m group)~~ on one OTSiG on a single optical span with 3R regeneration at each end. The OTSiG is composed of one or more OTSi.

Application codes of the optical interface signal carrying the OTU or FlexO-x-<~~fec~~int> are contained in [ITU-T G.959.1] and [ITU-T G.693] and [ITU-T G.695].

Vendor specific application identifiers of the interfaces carrying these OTUs and FlexO-x-<int> signals are outside the scope of this Recommendation.

8.2 Point-to-point interface, type II

The OTN point-to-point interface, type II supports n OTU and/or ~~OTUCn/FlexO-x-RS-<fec>-m~~FlexO-x-<int> signals on n OTSiG on a single optical span with 3R regeneration at each end. Each OTSiG is composed of one or more OTSi.

At least one of the OTU or FlexO-x-<int> signals is present during normal operation.

There is no predefined order in which the OTU and/or FlexO-x-<int> signals are taken into service.

NOTE – OTN point-to-point optical layer interface overhead is not defined. The interface will use the OTU SMOH or FlexO-x OH in this multi-channel interface for supervision and management. The interface connectivity (TIM) failure reports will be computed from the individual OTU or FlexO-x-<int> reports by means of failure correlation in fault management. Refer to the equipment Recommendations for further details.

Application codes of the optical interface signal carrying n OTU or FlexO-x-<~~fec~~int> signals are contained in [ITU-T G.959.1] and [ITU-T G.695].

Vendor specific application identifiers of the interfaces carrying these OTU~~s~~ and FlexO-x-<int> are outside the scope of this Recommendation.

8.3 Optical networking interface, type II

The OTN optical networking interfaces, type II carry one or more optical tributary signals, each transporting an OTUk, ~~or an~~ OTUCn, ~~or one~~ FlexO-x-<~~fec~~int>, ~~out of an OTUCn/FlexO-x-<fec>-m group~~ or a non-OTN client signal, and OCh-O or OTSiG-O carried by an OCC. ~~The OTUk, OTUCn or FlexO-x-<fec> is carried over an OTSi.~~

The OTN optical networking interfaces, type II may carry one optical tributary signal, transporting an OTUk, ~~or an~~ OTUCn, ~~or one~~ FlexO-x-<~~fec~~int>, ~~out of an OTUCn/FlexO-x-<fec>-m group~~ or a non-OTN client signal, and OCh-O or OTSiG-O carried by an OCC. The OTUk, OTUCn or FlexO-x-<~~fec~~int> is carried over an OTSi.

The OTN optical networking interfaces, type II may carry more than one optical tributary signals, each transporting an OTUCn, ~~or one~~ FlexO-x-<~~fec~~int>, ~~out of an OTUCn/FlexO-x-<fec>-m group~~ or a non-OTN client signal, and OCh-O or OTSiG-O carried by an OCC. The OTUCn or FlexO-x-<~~fec~~int> is carried over an OTSi.

Application codes of the optical tributary signal carrying the OTUk or FlexO-x-<~~fec~~int> are contained in [ITU-T G.698.1] and [ITU-T G.698.2].

Specifications of the OCC carrying the OCh-O or OTSiG-O are contained in [ITU-T G.7712].

Vendor specific application identifiers of the OTSi carrying these OTUks, ~~or~~ OTUCns, or FlexO-x-<~~fec~~int>s are outside the scope of this Recommendation.

8.4 Optical networking interface, type I

The OTN optical networking interfaces, type I carry one or more optical tributary signals, each transporting an OTUk, ~~or an~~ OTUCn, or ~~one~~ FlexO-x-<~~fec~~int> ~~out of an OTUCn/FlexO-x-<fec> m group~~ or a non-OTN client signal and OTS-O, OMS-O and OCh-O carried by an OSC and OTSiG-O carried by an OSC or partly carried by other means. 3R regeneration is not required at the interface.

Application codes of the optical tributary signal carrying the OTUk or FlexO-x-<~~fec~~int> are contained in [ITU-T G.698.1] and [ITU-T G.698.2].

Vendor specific application identifiers of the OTSi or OTSiG carrying these OTUs or FlexO-x-<~~fec~~int>s are outside the scope of this Recommendation.

9 Media Element

A description of the media element will be provided in [ITU-T G.872].

10 OCh and OTSiA

The OCh and OTSiA transport a digital client signal between 3R regeneration points. The OCh and OTSiA client signals defined in this Recommendation are the OTUk, OTUk-v, OTUkV and OTUCn signals.

10.1 OCh

The OCh structure is conceptually shown in Figure 10-1. The OCh contains two parts: an overhead part (OCh-O) and a payload part (OTSiG).

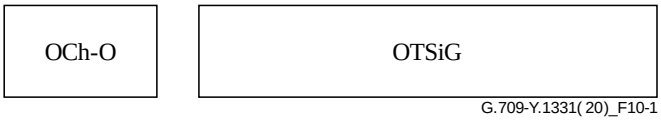


Figure 10-1 – OCh information structure

10.2 Optical tributary signal assembly (OTSiA)

The OTSiA structure is conceptually shown in Figure 10-2. The OTSiA contains two parts: a payload part (OTSiG) and an overhead part (OTSiG-O).

Figure 10-2 – OTSiA information structure

11 Optical transport unit (OTU)

The OTUk[V] conditions the ODUk for transport over an OCh network connection. The OTUk frame structure, including the OTUk FEC is completely standardized. The OTUkV is a frame structure, including the OTUkV FEC that is only functionally standardized (i.e., only the required functionality is specified); refer to Appendix II. Besides these two, there is an OTUkV in which the completely standardized OTUk frame structure is combined with a functionally standardized OTUkV FEC; refer to appendix II. This combination is identified as OTUk-v.

The OTUCn frame structure is defined without an OTUCn FEC area. The FEC associated with an OTUCn is interface dependent and specified as an element of each interface.

The OTU25 and OTU50 frame structure is defined without an OTU FEC area. The FEC associated with an OTU25 and OTU50 is interface dependent and specified as an element of each interface.

11.1 OTUk (k=0,1,2,3,4,4-SC) frame structure

The OTUk (k = 0,1,2,3,4,4-SC) frame structure is based on the ODUk frame structure and extends it with a forward error correction (FEC) as shown in Figure 11-1. 256 columns are added to the ODUk frame for the FEC and the reserved overhead bytes in row 1, columns 8 to 14 of the ODUk overhead are used for an OTUk specific overhead, resulting in an octet-based block frame structure with four rows and 4080 columns. The MSB in each octet is bit 1, the LSB is bit 8.

NOTE – This Recommendation does not specify an OTUk frame structure for k=2e or k=flex. See Annex G for the specification of OTU0LL.

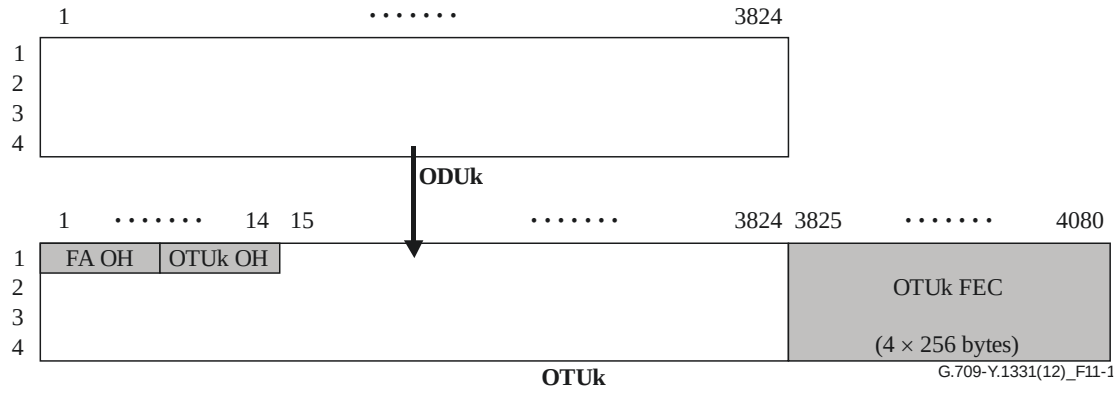


Figure 11-1 – OTUk frame structure

The bit rates of the OTUk signals are defined in Table 7-1.

The OTUk forward error correction (FEC) area contains an interface specific FEC type. Two FEC types are specified:

Reed-Solomon RS(255,239) FEC

The OTUk (k=0,1,2,3,4) FEC area contains the Reed-Solomon RS(255,239) FEC codes. Transmission of the OTUk FEC is mandatory for k=4 and optional for k=0,1,2,3. If no FEC is transmitted, fixed stuff bytes (all-0s pattern) are to be used.

The RS(255,239) FEC code shall be computed as specified in Annex A.

For interworking of equipment supporting FEC, with equipment not supporting FEC (inserting fixed stuff all-0s pattern in the OTUk (k=0,1,2,3) FEC area), the FEC supporting equipment shall support the capability to disable the FEC decoding process (ignore the content of the OTUk (k=0,1,2,3) FEC).

Staircase FEC

The OTU4 FEC area contains the ~~Stair Case~~staircase FEC codes as specified in [ITU-T G.709.2] and the resulting signal is referred to as OTU4-SC.

The transmission order of the bits in the OTUk frame is left to right, top to bottom, and MSB to LSB (see Figure 11-2).

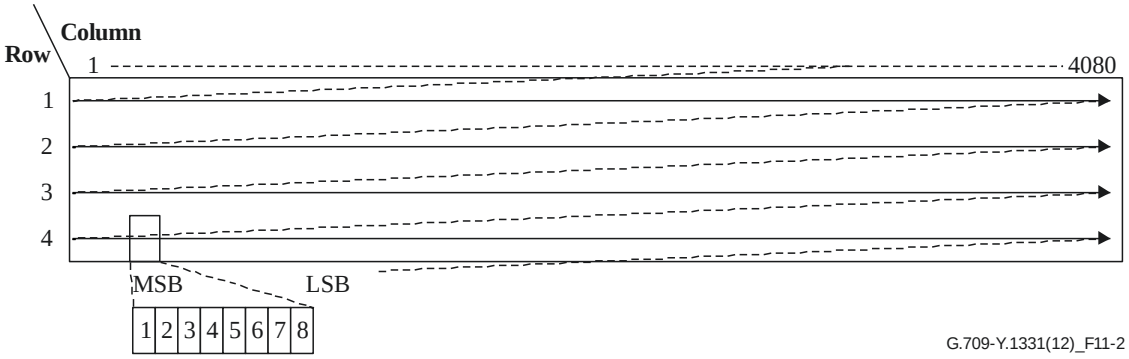


Figure 11-2 – Transmission order of the OTUk frame bits

11.2 Scrambling for OTUk (k=0,1,2,3,4)

The OTUk (k=0,1,2,3,4) signal must have sufficient bit timing content at the ONNI. A suitable bit pattern, which prevents a long sequence of "1"s or "0"s, is provided by using a scrambler.

The operation of the scrambler shall be functionally identical to that of a frame synchronous scrambler of sequence length 65535 operating at the OTUk rate.

The generating polynomial shall be $1 + x + x^3 + x^{12} + x^{16}$. Figure 11-3 shows a functional diagram of the frame synchronous scrambler.

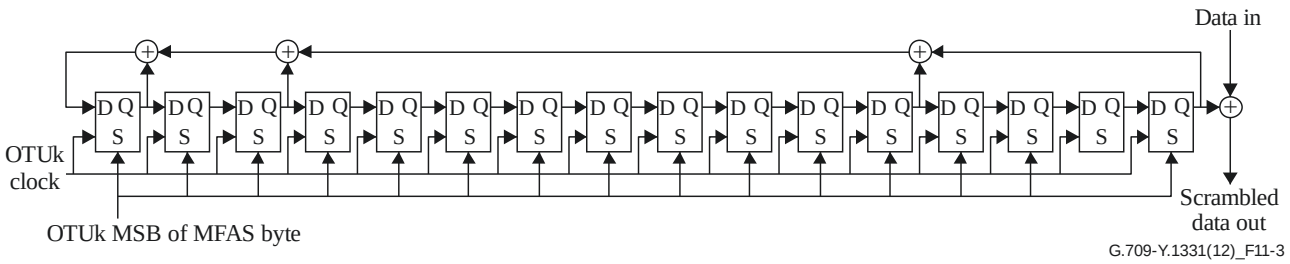


Figure 11-3 – Frame synchronous scrambler

The scrambler shall be reset to "FFFF" (HEX) on the most significant bit of the byte following the last framing byte in the OTUk frame, i.e., the MSB of the MFAS byte. This bit, and all subsequent bits to be scrambled shall be added modulo 2 to the output from the x^{16} position of the scrambler. The scrambler shall run continuously throughout the complete OTUk frame. The framing bytes (FAS) of the OTUk overhead shall not be scrambled.

Scrambling is performed after FEC computation and insertion into the OTUk signal.

11.3 OTUCn frame structure

The OTUCn frame structure (Figure 11-4) is based on the ODUCn frame structure and deploys the reserved overhead bytes in row 1, columns 8 to 14 of each ODU frame structure in the ODUCn overhead for an OTUCn specific overhead, resulting in an octet-based block frame structure with n × four rows and 3824 columns. The MSB in each octet is bit 1, the LSB is bit 8.

Interleaving of the n frame and multi-frame synchronous OTU frame structure instances within the OTUCn, forward error correction, encoding (e.g., scrambling), deskewing and transmission order of the OTUCn are interface specific and specified for inter-domain OTN interfaces with application codes in the interface specific Recommendations (ITU-T G.709.x series). For OTN interfaces with vendor specific application identifiers these specifications are vendor specific and out of scope of this Recommendation.

The bit rates of the OTUCn signals are defined in Table 7-1.

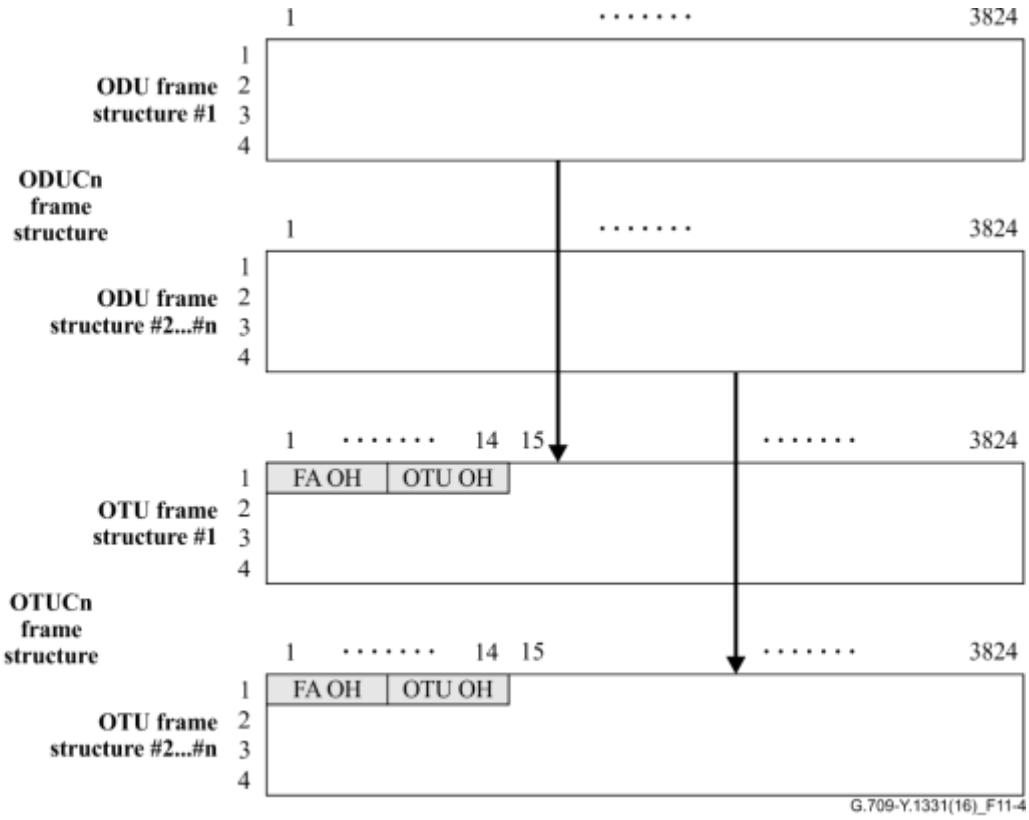


Figure 11-4 – OTUCn frame structure

11.4 OTU25 and OTU50 frame structure

The OTU25 and OTU50 frame structure is based on the ODUk (k=25, 50) frame structure and deploys the reserved overhead bytes in row 1, columns 8 to 14 of the ODUk overhead for an OTU specific overhead, resulting in an octet-based block frame structure with four rows and 3824 columns. The MSB in each octet is bit 1, the LSB is bit 8.

Forward error correction, encoding (e.g., scrambling), deskewing and transmission order of the OTU25 and OTU50 are specified for inter-domain OTN interfaces with application codes in the interface specific Recommendations [ITU-T G.709.4].

The bit rates of the OTU25 and OTU50 signals are defined in Table 7-1.

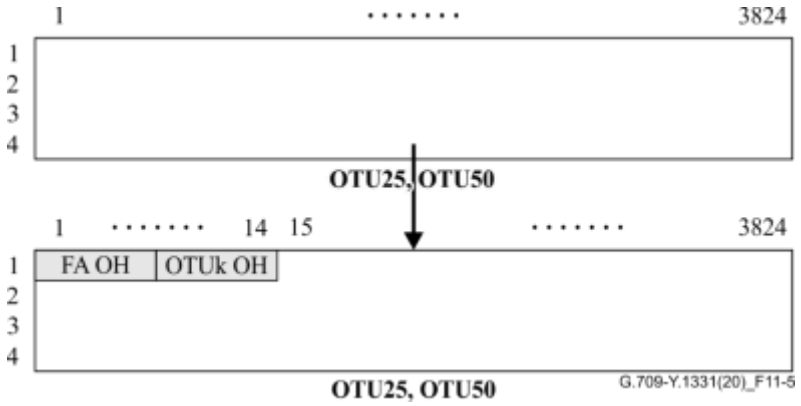


Figure 11-5 – OTU25 and OTU50 frame structure

12 Optical data unit (ODU)

12.1 ODU frame structure

The ODU frame structure is shown in Figure 12-1. It is organized in an octet-based block frame structure with four rows and 3824 columns.

The ODUk (k=0,1,2,2e,3,4,flex) frame structure contains one instance of the ODU frame structure. The ODUCn frame structure contains n frame and multi-frame synchronous instances of the ODU frame structures, numbered 1 to n (ODU #1 to ODU #n).

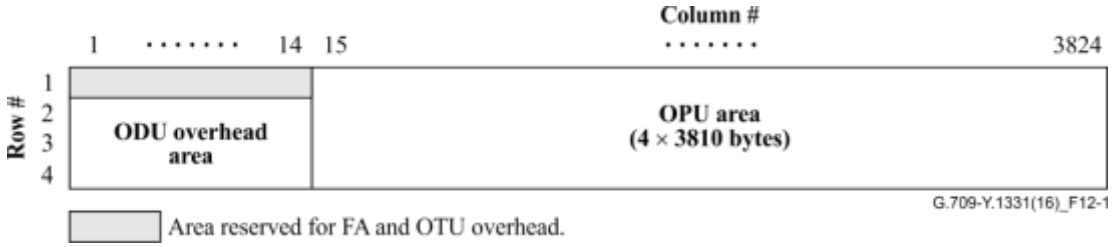


Figure 12-1 – ODU frame structure

The two main areas of the ODU frame are:

– ODU overhead area

– OPU area.

Columns 1 to 14 of the ODU are dedicated to ODU overhead area.

NOTE – Columns 1 to 14 of row 1 are reserved for a frame alignment and OTU specific overhead.

Columns 15 to 3824 of the ODU are dedicated to OPU area.

12.2 ODU bit rates and bit-rate tolerances

ODUk signals may be generated using either a local clock, or the recovered clock of the client signal. In the latter case the ODUk frequency and frequency tolerance are locked to the client signal's frequency and frequency tolerance. In the former case the ODUk frequency and frequency tolerance are locked to the local clock's frequency and frequency tolerance. The local clock frequency tolerance for the OTN is specified to be ±20 ppm.

ODUCn signals are generated using a local clock. The ODUCn frequency and frequency tolerance are locked to the local clock's frequency and frequency tolerance. The local clock frequency tolerance for the OTN is specified to be ±20 ppm.

ODU maintenance signals (ODU AIS, OCI, LCK) are generated using a local clock. In a number of cases this local clock may be the clock of a server ODU signal over which the ODU signal is transported between equipment or through equipment (in one or more of the tributary slots). For these cases, the nominal justification ratio should be deployed to comply with the ODU's bit-rate tolerance specification.

12.2.1 ODU0, ODU1, ODU2, ODU3, ODU4, ODU25, ODU50, ODUCn

The local clocks used to create the ODU0, ODU1, ODU2, ODU3, ODU4, ODU25, ODU50 and ODUCn signals are generated by clock crystals that are also used for the generation of SDH STM-N signals. The bit rates of these ODUk (k=0,1,2,3,4,25,50) and ODUCn signals are therefore related to the STM-N bit rates and the bit-rate tolerances are the bit-rate tolerances of the STM-N signals.

The ODU0 bit rate is 50% of the STM-16 bit rate.

The ODU1 bit rate is 239/238 times the STM-16 bit rate.

The ODU2 bit rate is 239/237 times 4 times the STM-16 bit rate.

The ODU3 bit rate is 239/236 times 16 times the STM-16 bit rate.

The ODU4 bit rate is 239/227 times 40 times the STM-16 bit rate.

The ODU25 bit rate is 41118/41120 times 514/544 times 255/227 times 10 times the STM-16 bit rate, which is equal to 61677/58112 times 10 times the STM-16 bit rate.

The ODU50 bit rate is 41118/41120 times 514/544 times 255/227 times 20 times the STM-16 bit rate, which is equal to 61677/58112 times 20 times the STM-16 bit rate.

The ODUCn bit rate is n times 239/226 times 40 times the STM-16 bit rate.

ODU1, ODU2 and ODU3 signals which carry an STM-N (N = 16, 64, 256) signal may also be generated using the timing of these client signals.

Refer to Table 7-2 for the nominal bit rates and bit-rate tolerances.

12.2.1.1 ODUk (k=0,1,2,3,4), ODU25/50 and ODUCn transporting frequency synchronization information

The local clocks used to create the frequency synchronous OPU0/ODU0/OTU0, OPU1/ODU1/OTU1, OPU2/ODU2/OTU2, OPU3/ODU3/OTU3, OPU4/ODU4/OTU4, OPU25/ODU25/OTU25, OPU50/ODU50/OTU50 and OPUCn/ODUCn/OTUCn/FlexO signals are generated by a synchronous equipment clock (SEC) function that is nominally locked to a primary reference clock (PRC). The bit rate tolerance of these frequency synchronous ODUk (and its OTUk) (k=0,1,2,3,4,4-SC), ODU25/50 (and their OTU25/50) and ODUCn (and its OTUCn and FlexO) signals are therefore related to the bit-rate tolerances of the SEC. Refer to [ITU-T G.8262].

During signal fail conditions, the frequency synchronous OTU and/or ODU signals are replaced by replacement signals (OTU-AIS and/or ODU-AIS) with a bit rate tolerance of ±20 ppm.

12.2.2 ODU2e

An ODU2e signal is generated using the timing of its client signal.

The ODU2e bit rate is 239/237 times the 10GBASE-R client bit rate.

Refer to Table 7-2 for the nominal bit rate and bit-rate tolerances.

12.2.3 ODUflex for CBR client signals

An ODUflex(CBR) signal is generated using the timing of its client signal.

The ODUflex bit rate is 239/238 times the adapted CBR client bit rate.

The client signal may have a bit-rate tolerance up to ±100 ppm.

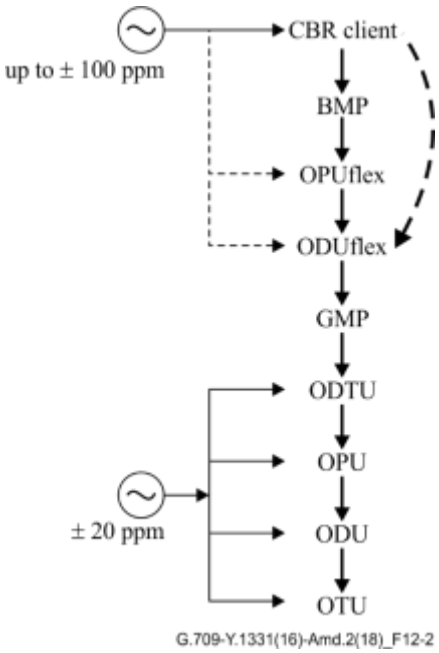


Figure 12-2 – ODUflex clock generation for CBR signals

12.2.4 ODUflex for PRBS and Null test signals

ODUflex(CBR) connections may be tested using a PRBS or NULL test signal as the client signal instead of the CBR client signal. For such a case, the ODUflex(PRBS) or ODUflex(NULL) signal should be generated with a frequency within the tolerance range of the ODUflex(CBR) signal.

If the CBR client clock is present such ODUflex(PRBS) or ODUflex(NULL) signal may be generated using the CBR client clock, otherwise the ODUflex(PRBS) or ODUflex(NULL) signal is generated using a local clock.

12.2.5 ODUflex for GFP-F mapped packet client signals

ODUflex(GFP) signals are generated using a local clock. This clock may be the local server ODUk (or OTUk) clock, the local ODUCn (or OTUCn) clock, or an equipment internal clock of the signal over which the ODUflex is carried through the equipment. Refer to Appendix XV for examples on ODUflex(GFP) clock generation methods.

Any bit rate is possible for an ODUflex(GFP) signal. The ODUflex(GFP) bit rate is specified as 239/238 × X Gbit/s ± 100 ppm, in which X represents the configured GFP-F encapsulated packet client nominal bidirectional bit rate.

For ODUflex hitless resizing applications or for the case that the ODUflex(GFP) should fill an integral number of 1.25G tributary slots of the smallest server ODUk path over which the ODUflex(GFP) may be carried, a set of bit-rates with values n × ODUk.ts that meet this criterion are specified in Table 7-8. The derivation of the specific ODUflex(GFP,n,k) values is provided in Appendix XI.

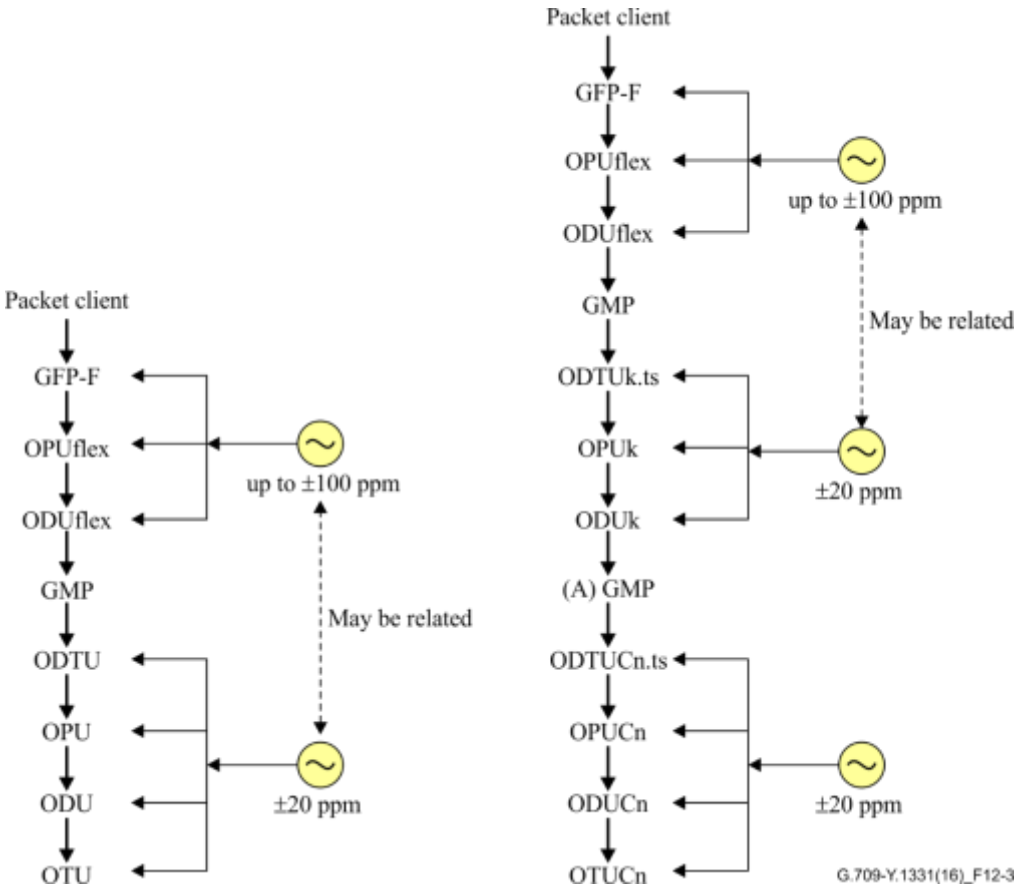


Figure 12-3 – ODUflex clock generation for GFP-F mapped packet client signals

12.2.6 ODUflex for IMP mapped client signals

ODUflex(IMP) signals are generated using a local clock or the timing of its client signal. The local clock may be the local server ODUk (or OTUk) clock, the local ODUCn (or OTUCn) clock, or an equipment internal clock of the signal over which the ODUflex is carried through the equipment. Refer to Appendix XV for examples on ODUflex(IMP) clock generation methods.

Any bit rate is possible for an ODUflex(IMP) signal. The ODUflex(IMP) bit rate is specified as 239/238 × X Gbit/s ± 100 ppm, in which X represents the configured 64B/66B encoded packet client nominal bidirectional bit rate.

For the case that the ODUflex(IMP) would carry a FlexE Client signal with a bit rate of s × 5,156,250.000 kbit/s ± 100 ppm and s = 2, 8 or n × 5 (n ≥ 1), the ODUflex(IMP,s) bit rate is s × 239/238 × 5,156,250.000 kbit/s ± 100 ppm.

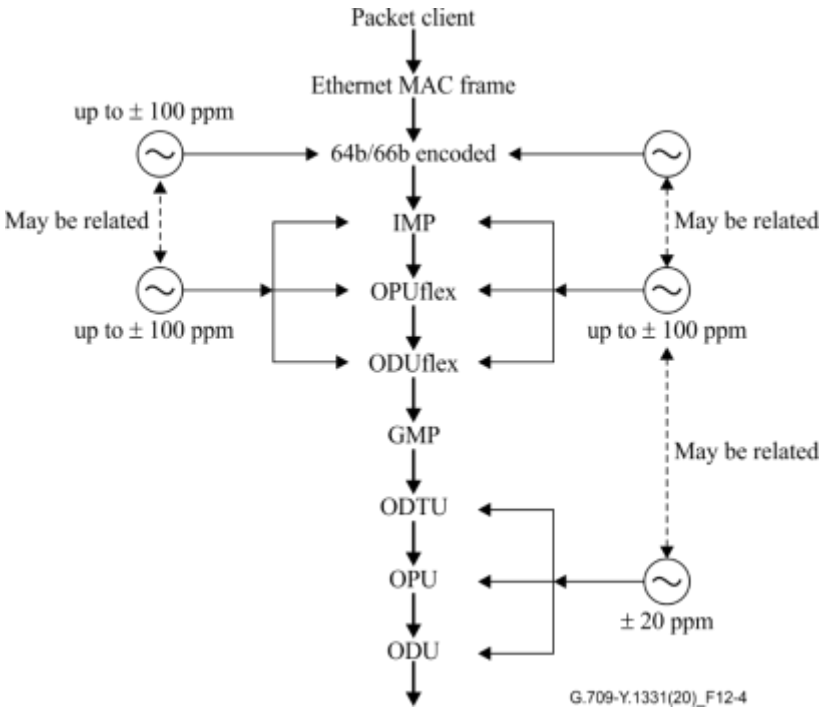


Figure 12-4 – ODUflex clock generation for IMP mapped packet client signals

13 Optical payload unit (OPU)

The OPU frame structure is shown in Figure 13-1. It is organized in an octet-based block frame structure with four rows and 3810 columns.

The OPUk (k=0,1,2,2e,3,4,flex,25,50) frame structure contains one instance of the OPU frame structure. The OPUCn frame structure contains n frame and multi-frame synchronous instances of the OPU frame structure, numbered 1 to n (OPU #1 to OPU #n).

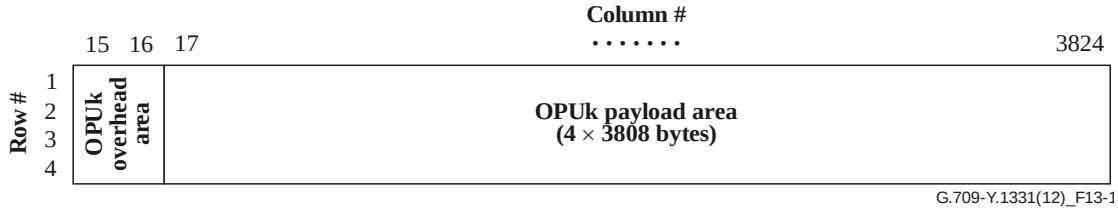


Figure 13-1 – OPU frame structure

The two main areas of the OPU frame are:

– OPU overhead area;

– OPU payload area.

Columns 15 to 16 of the OPU are dedicated to an OPU overhead area.

Columns 17 to 3824 of the OPU are dedicated to an OPU payload area.

NOTE – OPU column numbers are derived from the OPU columns in the ODU frame.

14 Overhead information carried over the OSC and OCC

14.1 OSC

The overhead information carried over the OSC consists of the OTS-O, OMS-O, OCh-O and OTSiG-O. The information content of this overhead is defined in clause 15. The format, structure and bit rate of this overhead is not defined in this Recommendation.

General management communications (COMMS)

Depending on an operator's logical management overlay network design, general management communications (COMMS) may also be transported within the OSC. Therefore, the OSC for some applications may also transport general management communications. General management communications may include signalling, voice/voiceband communications, software download, operator-specific communications, etc.

OTN synchronisation message channel (OSMC)

For synchronisation purposes, the OSC OSMC signal is defined as an OTN synchronisation message channel to transport SSM and PTP messages.

NOTE 1 – Support of OSC OSMC is optional.

NOTE 2 – Equipment designed prior to Edition 4.6 of this recommendation may not be able to support OSC OSMC.

14.2 OCC

The overhead information carried over the OCC consists of the OCh-O or OTSiG-O. The information content of this overhead is defined in clause 15. The format, structure and bit rate of this overhead is defined in [ITU-T G.7712].

14.2.1 OCh-O transport over OTN optical networking interface, type II

For an OTN optical networking interface, type II, the OCh-O is transferred over the overhead communication network (OCN) as described in [ITU-T G.7712].

NOTE – An OTN optical networking interface which transfers OCh-O over an OCN cannot provide fate-sharing of the OCh-O with the OTSiG across this interface.

The OCh FDI-P, FDI-O and OCI Overhead primitives are communicated over the OCN. The specification of the encapsulation, identification and transmission of this information is outside the scope of this Recommendation and specified in [ITU-T G.7712]. This information must be communicated with the peers such that the OCh-O primitives come into sync within one second in the absence of changes to OCh FDI-P, FDI-O or OCI. In the event of changes to any OCh-O primitive, the update must be sent within 10 ms of the change and with a mechanism to guarantee receipt in the event of packet loss.

14.2.2 OTSiG-O transport over OTN optical networking interface, type II

For an OTN optical networking interface, type II, the OTSiG-O is transferred over the overhead communication network (OCN) as described in [ITU-T G.7712].

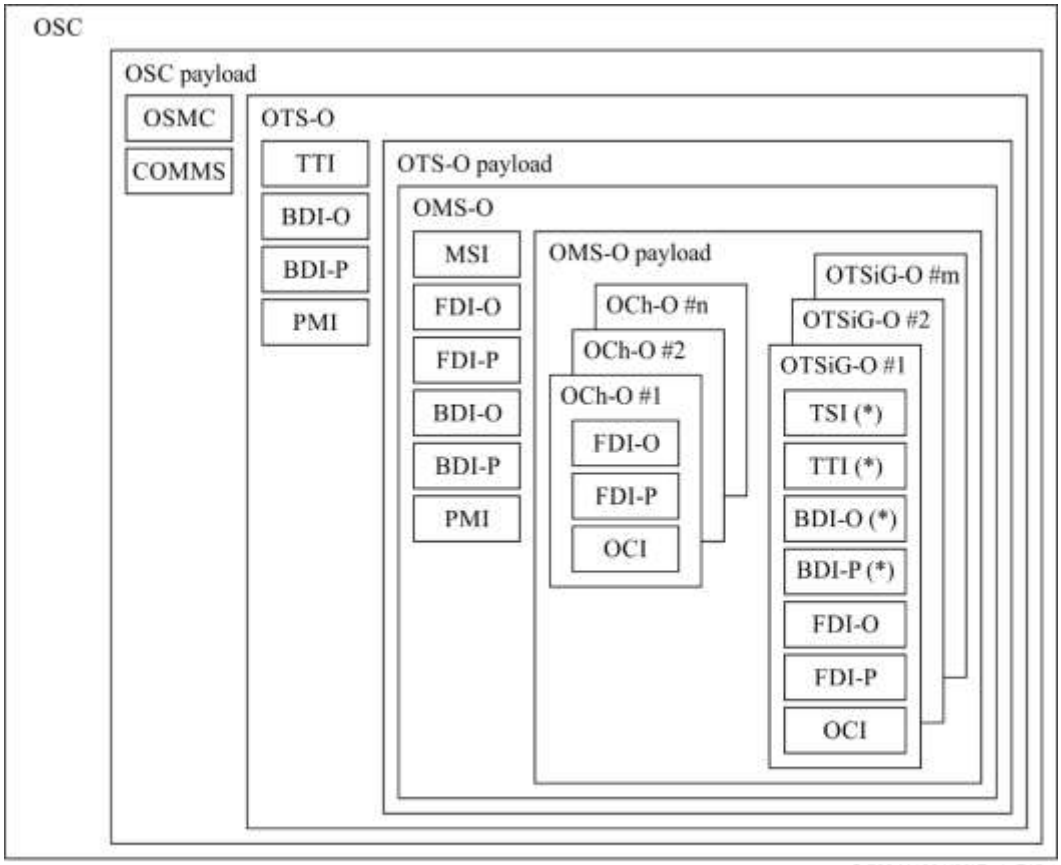
NOTE – An OTN optical networking interface which transfers OTSiG-O over an OCN cannot provide fate-sharing of the OTSiG-O with the OTSiG across this interface.

The OTSiG-O FDI-P, FDI-O, OCI, TTI, BDI-P, BDI-O and TSI Overhead primitives are communicated over the OCN. The specification of the encapsulation, identification and transmission of this information is outside the scope of this Recommendation and specified in [ITU-T G.7712]. This information must be communicated with the peers such that the OTSiG-O primitives come into sync within one second in the absence of changes to OTSiG FDI-P, FDI-O, OCI, TTI, BDI-P, BDI-O

or TSI. In the event of changes to any OTSiG-O primitive, the update must be sent within 10 ms of the change and with a mechanism to guarantee receipt in the event of packet loss.

15 Overhead description

An overview of OTS-O, OMS-O, OCh-O and OTSiG-O information carried within the OSC is presented in Figure 15-1. An overview of OCh-O information carried within the OCC is presented in Figure 15-2.



(*) This overhead may be carried over the OSC or over a communication channel modulated on one or more OTSi within the OTSiG.

Figure 15-1 – OTS-O, OMS-O, OCh-O and OTSiG-O information carried within the OSC

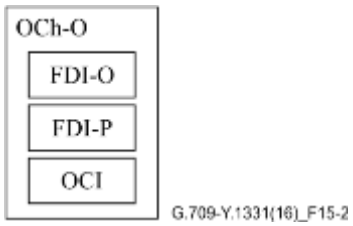


Figure 15-2 – OCh-O information carried within the OCC

An overview of OTUk, ODUk and OPUk overhead is presented in Figures 15-3A and 15-5. An overview of OTUCn, ODUCn and OPUCn overhead is presented in Figures 15-4 and 15-6. An overview of OTU25 and OTU50 overhead is presented in Figure 15-3B. It is the same as OTUk overhead.

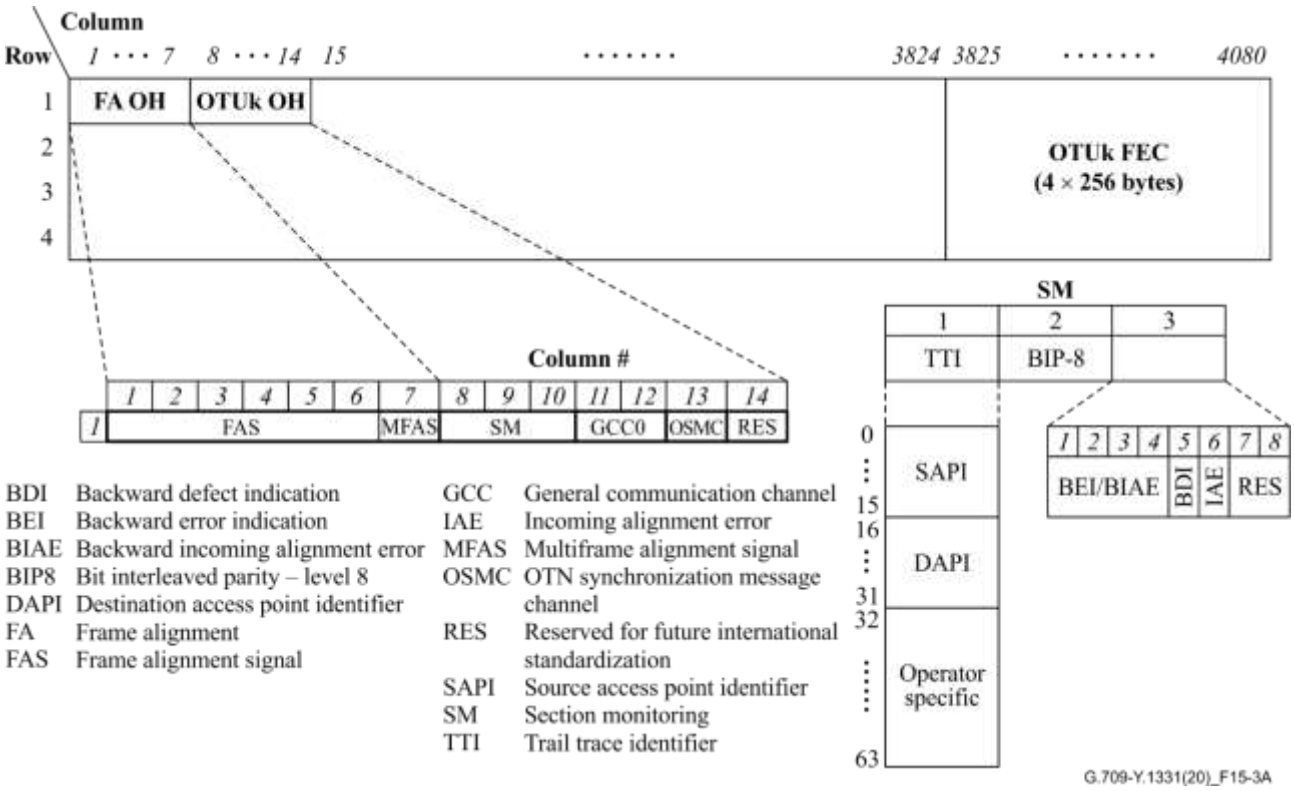


Figure 15-3A – OTUk frame structure, frame alignment and OTUk overhead

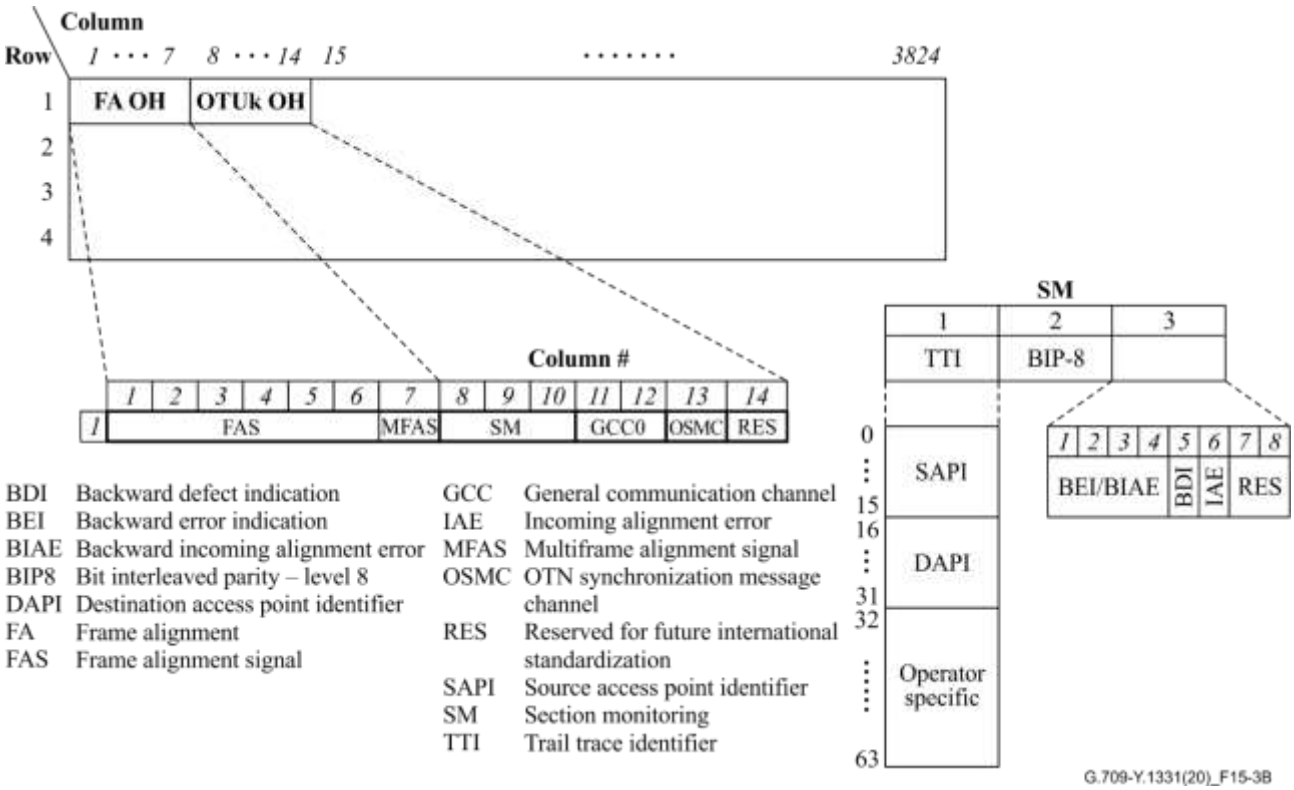


Figure 15-3B – OTU25 and OTU50 frame structure, frame alignment and OTU overhead

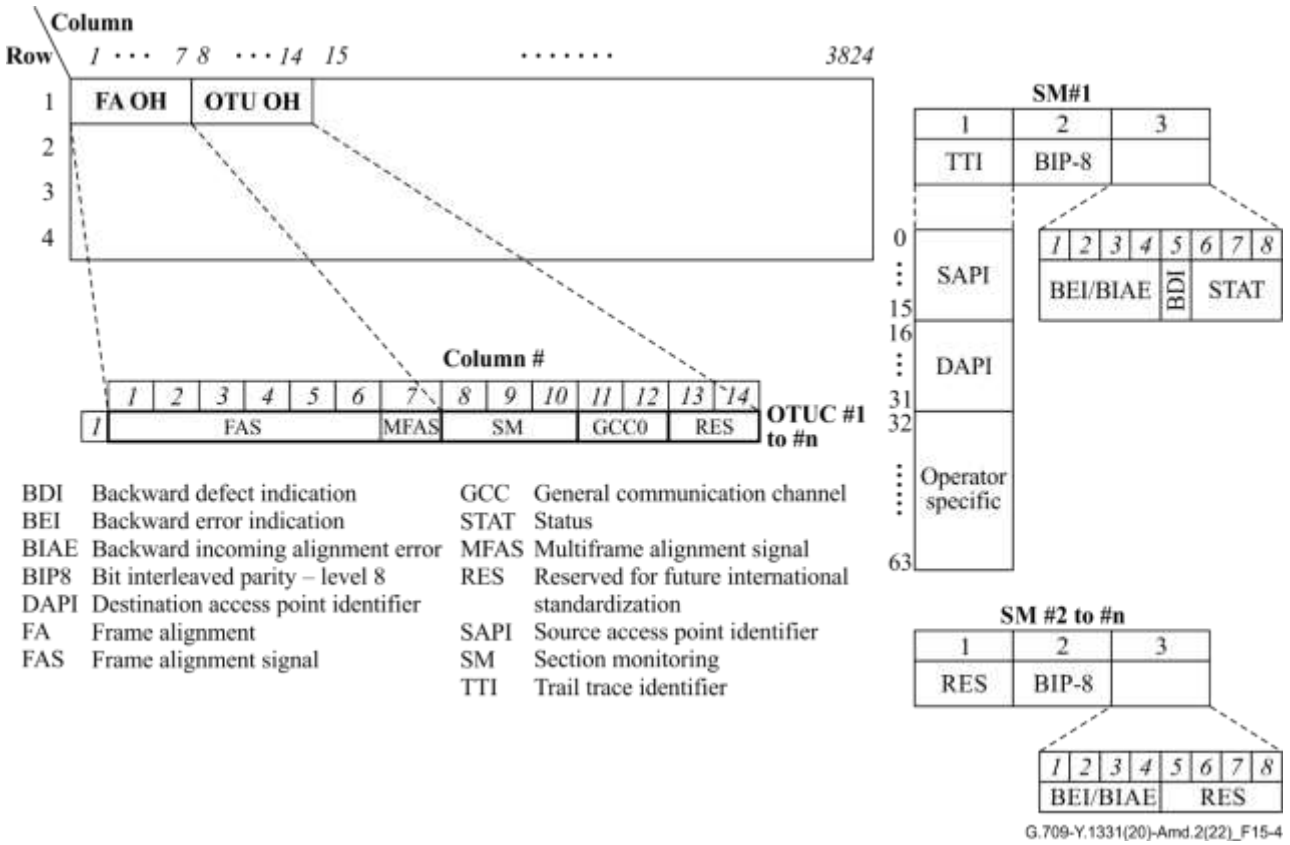
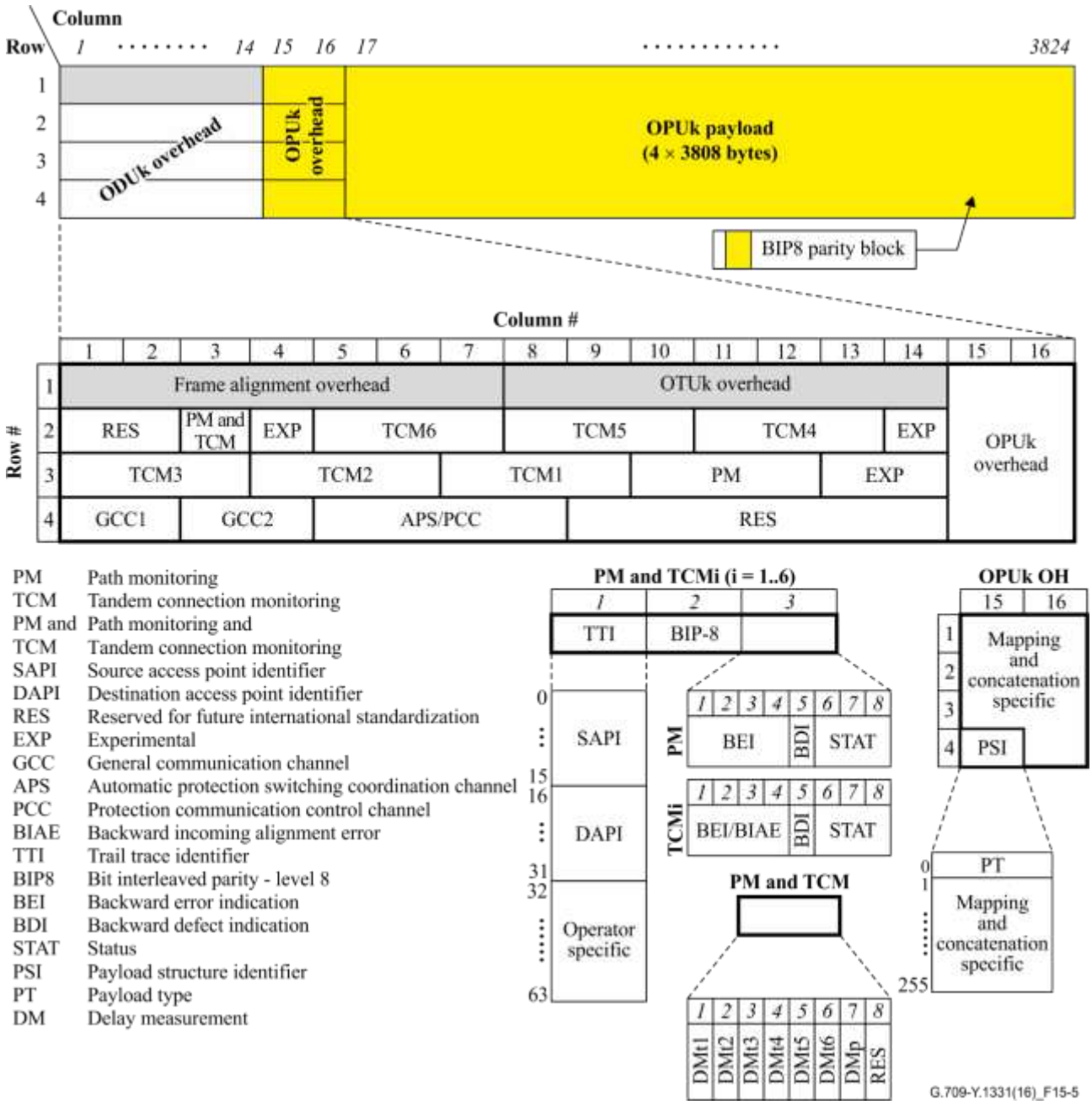


Figure 15-4 – OTUCn frame structure, frame alignment and OTUCn overhead



PM Path monitoring
TCM Tandem connection monitoring
PM and TCM Path monitoring and Tandem connection monitoring
SAPI Source access point identifier
DAPI Destination access point identifier
RES Reserved for future international standardization
EXP Experimental
GCC General communication channel
APS Automatic protection switching coordination channel
PCC Protection communication control channel
BIAE Backward incoming alignment error
TTI Trail trace identifier
BIP8 Bit interleaved parity - level 8
BEI Backward error indication
BDI Backward defect indication
STAT Status
PSI Payload structure identifier
PT Payload type
DM Delay measurement

Figure 15-5 – ODUk frame structure, ODUk and OPUk overhead

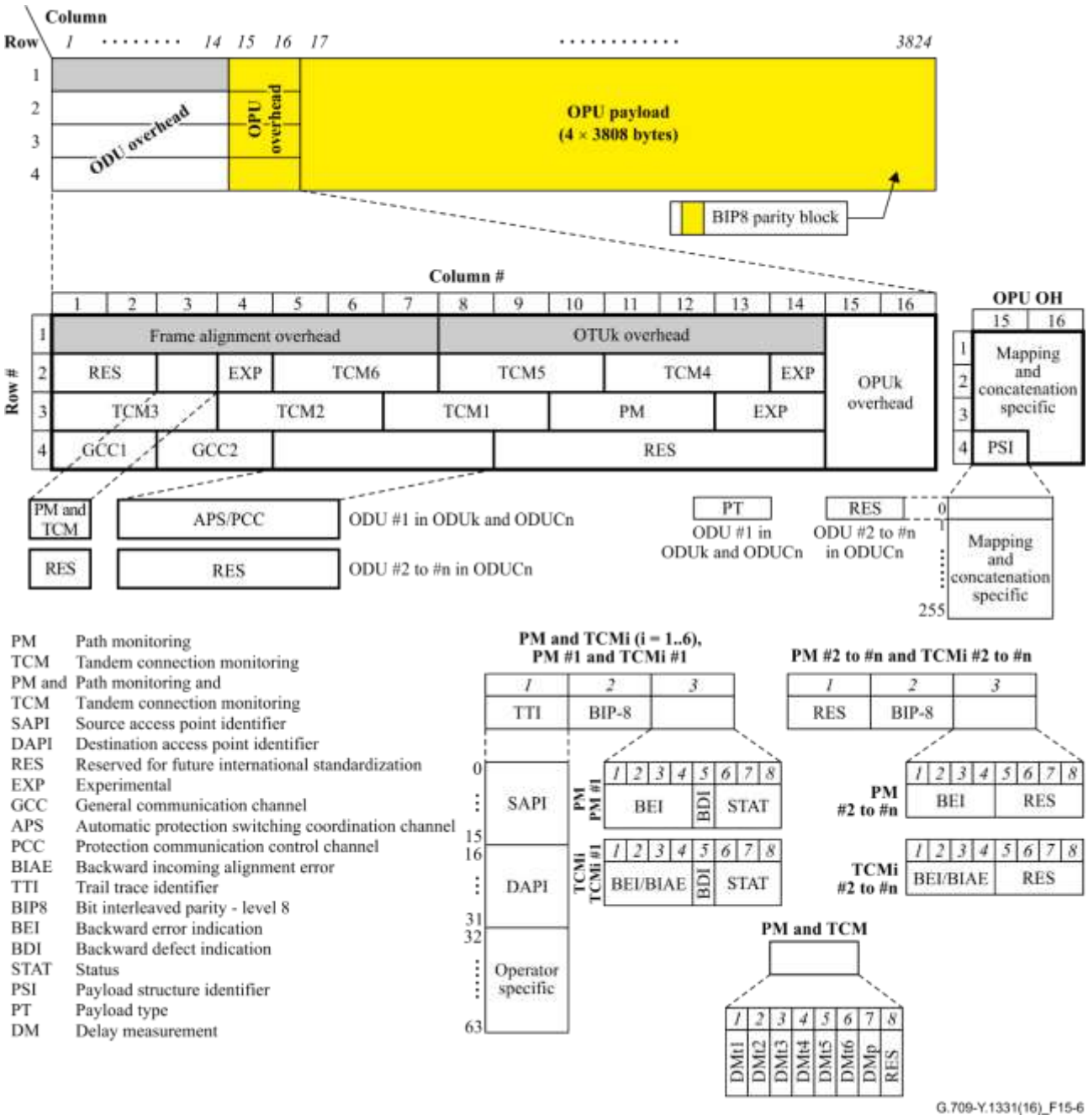


Figure 15-6 – ODUCn frame structure, ODUCn and OPUCn overhead

15.1 Types of overhead

15.1.1 Optical payload unit overhead (OPU OH)

OPU OH information is added to the OPU information payload to create an OPU. It includes information to support the adaptation of client signals. The OPU OH is terminated where the OPU is assembled and disassembled. The specific OH format and coding is defined in clause 15.9.

15.1.2 Optical data unit overhead (ODU OH)

ODU OH information is added to the ODU information payload to create an ODU. It includes information for maintenance and operational functions to support ODU connections. The ODU OH consists of portions dedicated to the end-to-end ODU path and to six levels of tandem connection monitoring. The ODU path OH is terminated where the ODU is assembled and disassembled. The TC OH is added and terminated at the source and sink of the corresponding tandem connections, respectively. The specific OH format and coding is defined in clauses 15.6 and 15.8.

15.1.3 Optical transport unit overhead (OTU OH)

OTU OH information is added to the OTU information payload to create an OTU. It includes information for maintenance and operational functions to support OTU connections. The OTU OH is terminated where the OTU signal is assembled and disassembled. The specific OH format and coding is defined in clauses 15.6 and 15.7.

The specific frame structure and coding for the non-standard OTUkV OH is outside the scope of this Recommendation. Only the required basic functionality that has to be supported is defined in clause 15.7.3.

15.1.4 OCh-O

OCh-O is the non-associated overhead information that accompanies an OTSiG. It includes information for maintenance functions to support fault management. The OCh-O is generated or modified at intermediate points along the OCh trail. All information elements of the OCh-O are terminated where the OTSiG signal is terminated.

The OCh-O information is defined in clause 15.5.

15.1.5 OMS-O

OMS-O is the non-associated overhead information that accompanies an OMS OSME. It includes information for maintenance and operational functions to support optical multiplex sections. The OMS-O is generated or modified at intermediate points along the OMS OSME and OMS-O trail and generated and terminated at the endpoints of the OMS OSME where the OMS-O trail is terminated.

OMS-O information elements other than OMS FDI-P and FDI-O are generated and terminated at the ends of the OMS-O trail. OMS FDI-P and FDI-O are generated at intermediate points along the OMS-O trail and terminated at the ends of the OMS-O trail.

The OMS-O information is defined in clause 15.4.

15.1.6 OTS-O

OTS-O is the non-associated overhead information that accompanies an OTS OSME. It includes information for maintenance and operational functions to support optical transmission sections. The OTS-O is terminated at the endpoints of the OTS OSME where the OTS-O trail is terminated.

The OTS-O information is defined in clause 15.3.

15.1.7 General management communications overhead (COMMS OH)

COMMS OH information provides general management communication between network elements. The specific frame structure and coding for the COMMS OH is outside the scope of this Recommendation.

15.1.8 OTSiG-O

OTSiG-O is the non-associated overhead information that accompanies an OTSiG. It includes information for maintenance functions to support fault management. The OTSiG-O TSI, TTI, BDI-P and BDI-O are generated where the OTSiA signal is assembled and the OCI, FDI-P and FDI-O are modified or generated at intermediate points along the OTSiA trail. All information elements of the OTSiG-O are terminated where the OTSiA is disassembled.

The OTSiG-O information is defined in clause 15.5.

NOTE – OSC signals designed prior to Edition 5 of this recommendation may only be able to transport a subset of the OTSiG-O (possibly excluding TSI, TTI, BDI-P and BDI-O) overhead.

15.2 Trail trace identifier and access point identifier definition

A trail trace identifier (TTI) is defined as a 64-byte string with the following structure (see Figure 15-7):

– TTI[0] contains the SAPI[0] character, which is fixed to all-0s.

– TTI[1] to TTI[15] contain the 15-character source access point identifier (SAPI[1] to SAPI[15]).

– TTI[16] contains the DAPI[0] character, which is fixed to all-0s.

– TTI[17] to TTI[31] contain the 15-character destination access point identifier (DAPI[1] to DAPI[15]).

– TTI[32] to TTI[63] are operator specific.

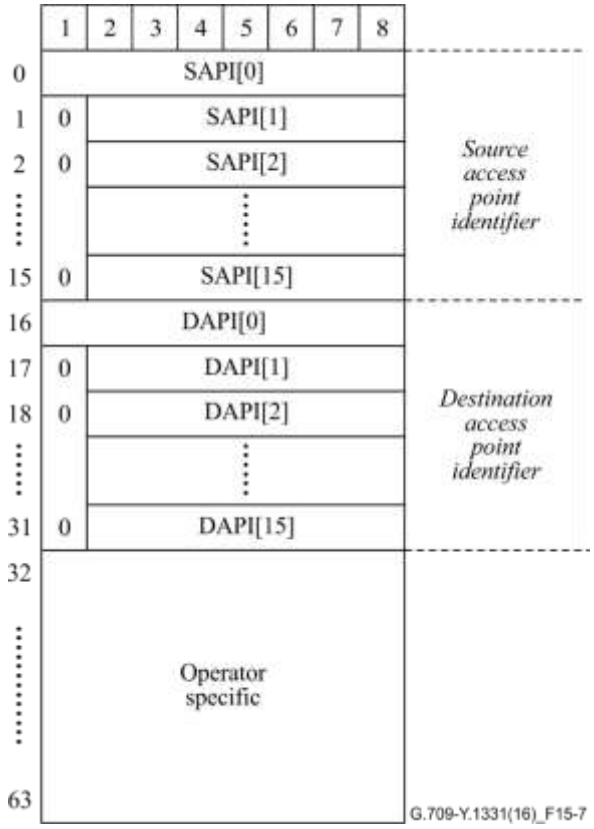


Figure 15-7 – TTI structure

The features of access point identifiers (APIs) are:

– Each access point identifier must be globally unique in its layer network.

– Where it may be expected that the access point may be required for path set-up across an inter-operator boundary, the access point identifier must be available to other network operators.

– The access point identifier should not change while the access point remains in existence.

– The access point identifier should be able to identify the country and network operator which is responsible for routing to and from the access point.

– The set of all access point identifiers belonging to a single administrative layer network should form a single access point identification scheme.

– The scheme of access point identifiers for each administrative layer network can be independent from the scheme in any other administrative layer network.

It is recommended that the ODUk, OTUk and OTS should each have the access point identification scheme based on a tree-like format to aid routing control search algorithms. The access point identifier should be globally unambiguous.

The access point identifier (SAPI, DAPI) shall consist of a three-character international segment and a twelve-character national segment (NS) (see Figure 15-8). These characters shall be coded according to [ITU-T T.50] (International Reference Alphabet – 7-bit coded character set for information exchange).

<table>
<tr><td colspan="3">IS character #</td><td colspan="12">NS character #</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td colspan="3">CC</td><td>ICC</td><td colspan="11">UAPC</td></tr>
<tr><td colspan="3">CC</td><td colspan="2">ICC</td><td colspan="10">UAPC</td></tr>
<tr><td colspan="3">CC</td><td colspan="3">ICC</td><td colspan="9">UAPC</td></tr>
<tr><td colspan="3">CC</td><td colspan="4">ICC</td><td colspan="8">UAPC</td></tr>
<tr><td colspan="3">CC</td><td colspan="5">ICC</td><td colspan="7">UAPC</td></tr>
<tr><td colspan="3">CC</td><td colspan="6">ICC</td><td colspan="6">UAPC</td></tr>
</table>

Figure 15-8 – Access point identifier structure

The international segment field provides a three-character ISO 3166 geographic/political country code (G/PCC). The country code shall be based on the three-character uppercase alphabetic ISO 3166 country code (e.g., USA, FRA).

The national segment field consists of two subfields: the ITU carrier code (ICC) followed by a unique access point code (UAPC).

The ITU carrier code is a code assigned to a network operator/service provider, maintained by the ITU-T Telecommunication Standardization Bureau (TSB) as per [ITU-T M.1400]. This code shall consist of 1-6 left-justified characters, alphabetic, or leading alphabetic with trailing numeric.

The unique access point code shall be a matter for the organization to which the country code and ITU carrier code have been assigned, provided that uniqueness is guaranteed. This code shall consist of 6-11 characters, with trailing NUL, completing the 12-character national segment.

15.3 OTS-O description

The following OTS-O information elements are defined:

– OTS-TTI

– OTS-BDI-P

– OTS-BDI-O

– OTS-PMI

15.3.1 OTS trail trace identifier (TTI)

The OTS-TTI is defined to transport a 64-byte TTI as specified in clause 15.2 for OTS connectivity monitoring.

15.3.2 OTS backward defect indication – Payload (BDI-P)

For OTS OSME and OTS-O trail monitoring, the OTS-BDI-P signal is defined to convey in the upstream direction the OTS OSME and OTS-O trail signal fail status detected in the sink point of the OTS OSME and OTS-O trail termination sink function.

15.3.3 OTS backward defect indication – Overhead (BDI-O)

For OTS-O trail monitoring, the OTS-BDI-O signal is defined to convey in the upstream direction the OTS-O signal fail status detected in the OTS-O trail termination sink function.

15.3.4 OTS payload missing indication (PMI)

The OTS PMI is a signal sent downstream as an indication that upstream at the source point of the OTS OSME no optical signal is present, in order to suppress the report of the consequential loss of signal condition at the sink point of the OTS OSME.

15.4 OMS-O description

The following OMS-O information elements are defined:

– OMS-FDI-P

– OMS-FDI-O

– OMS-BDI-P

– OMS-BDI-O

– OMS-PMI

– OMS-MSI

15.4.1 OMS forward defect indication – Payload (FDI-P)

For OMS OSME and OMS-O trail monitoring, the OMS-FDI-P signal is defined to convey in the downstream direction the signal status (normal or failed).

15.4.2 OMS forward defect indication – Overhead (FDI-O)

For OMS-O trail monitoring, the OMS-FDI-O signal is defined to convey in the downstream direction the OMS-O signal status (normal or failed).

15.4.3 OMS backward defect indication – Payload (BDI-P)

For OMS OSME and OMS-O trail monitoring, the OMS-BDI-P signal is defined to convey in the upstream direction the OMS-P signal fail status detected in the sink point of the OMS OSME and OMS-O trail termination sink function.

15.4.4 OMS backward defect indication – Overhead (BDI-O)

For OMS-O trail monitoring, the OMS-BDI-O signal is defined to convey in the upstream direction the OMS-O signal fail status detected in the OMS-O trail termination sink function.

15.4.5 OMS payload missing indication (PMI)

The OMS PMI is a signal sent downstream as an indication that upstream at the source point of the OMS OSME none of the frequency slots contain an optical tributary signal, in order to suppress the report of the consequential loss of signal condition at the sink point of the OMS OSME.

15.4.6 OMS multiplex structure identifier (MSI)

The OMS multiplex structure identifier (MSI) signal encodes the OTSiG multiplex structure and occupied frequency slots in the OMS OSME at the source end point. It is sent downstream to enable detection of OTSiG multiplex structure configuration mismatches between sink and source end points.

The OMS multiplex structure identifier (MSI) signal also encodes the media channel structure which is independent of the OTSiG in the OMS OSME at the source and sink end points. It is sent downstream to enable detection of media channel structure configuration mismatches between sink and source end points.

The OMS MSI is defined for flex grid OTN optical networking interfaces, type I. Fixed grid OTN optical networking interfaces may not support the OMS MSI. Flex grid OTN optical networking interfaces, type I designed prior to Edition 5.0 of this Recommendation may not support the OMS MSI.

15.5 OCh-O and OTSiG-O description

The following OCh-O and OTSiG-O information elements are defined:

– OCh-FDI-P and OTSiA-FDI-P

– OCh-FDI-O and OTSiA-FDI-O

– OCh-OCI and OTSiA-OCI

– OTSiA-BDI-P

– OTSiA-BDI-O

– OTSiA-TTI

– OTSiG-TSI

15.5.1 OCh and OTSiA forward defect indication – Payload (FDI-P)

For OCh and OTSiA trail monitoring, the OCh-FDI-P or OTSiA-FDI-P signal is defined to convey in the downstream direction the OTSiG signal status (normal or failed).

15.5.2 OCh and OTSiA forward defect indication – Overhead (FDI-O)

For OCh and OTSiA trail monitoring, the OCh-FDI-O or OTSiA-FDI-O signal is defined to convey in the downstream direction the OCh-O or OTSiG-O overhead signal status (normal or failed).

15.5.3 OCh and OTSiA open connection indication (OCI)

The OCh and OTSiA OCI is a signal sent downstream as an indication that upstream in a connection function the OCh (i.e., OTSiG and OCh-O) or OTSiA (i.e., OTSiG and OTSiG-O) matrix connection is opened as a result of a management command. The consequential detection of the OCh or OTSiA loss of signal condition at the OCh or OTSiA termination point can now be related to an open matrix.

15.5.4 Blank clause

This clause is intentionally left blank.

15.5.5 OTSiA backward defect indication – Payload (BDI-P)

For OTSiA path monitoring, the OTSiA-BDI-P signal is defined to convey in the upstream direction the OTSiG signal fail status detected in the OTSiG termination sink function.

15.5.6 OTSiA backward defect indication – Overhead (BDI-O)

For OTSiA path monitoring, the OTSiA-BDI-O signal is defined to convey in the upstream direction the OTSiG-O signal fail status detected in the OTSiG-O termination sink function.

15.5.7 OTSiA trail trace identifier (TTI)

The OTSiA-TTI is defined to transport a 64-byte TTI as specified in clause 15.2 for OTSiA path monitoring.

15.5.8 OTSiG transmitter structure identifier (TSI)

The OTSiG-TSI is defined to transport a TSI for monitoring consistent configuration of the OTSiG transmitter(s) at source end points and OTSiG receiver(s) at sink end points.

15.6 OTU/ODU frame alignment OH description

15.6.1 OTU/ODU frame alignment overhead location

The OTU/ODU frame alignment overhead location is shown in Figure 15-9. The OTU/ODU frame alignment overhead is applicable for both the OTU and ODU signals.

The OTUk/ODUk, OTU25/ODU25 and OTU50/ODU50 contain one instance of OTU/ODU frame alignment overhead. The OTUCn/ODUCn contains n instances of OTU/ODU frame alignment overhead, numbered 1 to n.

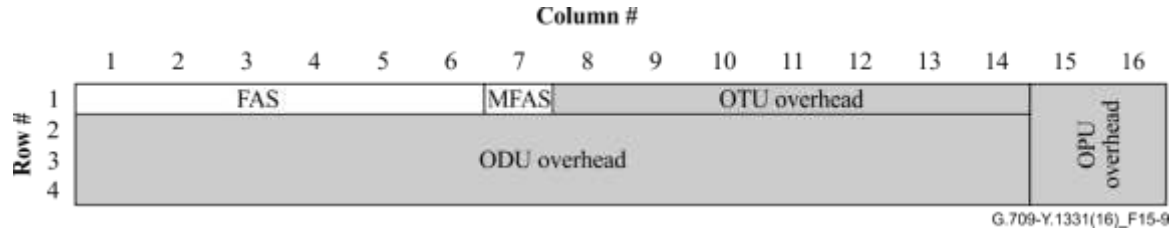


Figure 15-9 – OTU/ODU frame alignment overhead

15.6.2 OTU/ODU frame alignment overhead definition

15.6.2.1 Frame alignment signal (FAS)

A six byte OTU-FAS signal (see Figure 15-10) is defined in row 1, columns 1 to 6 of the OTU overhead. OA1 is "1111 0110". OA2 is "0010 1000".

The OTUk, OTU25 and OTU50 contain one instance of OTU multi-frame alignment overhead. The OTUCn contains n instances of OTU multi-frame alignment overhead, numbered 1 to n.

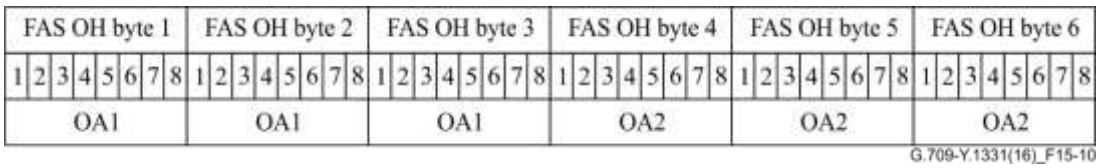


Figure 15-10 – Frame alignment signal overhead structure

15.6.2.2 Multiframe alignment signal (MFAS)

Some of the OTU and ODU overhead signals will span multiple OTU/ODU frames. Examples are the TTI and TCM-ACT overhead signals. These and other multiframe structured overhead signals require multiframe alignment processing to be performed, in addition to the OTU/ODU frame alignment.

A single multiframe alignment signal (MFAS) byte is defined in row 1, column 7 of the OTU/ODU overhead for this purpose (see Figure 15-11). The value of the MFAS byte will be incremented each OTU/ODU frame and provides as such a 256-frame multiframe.

The OTUk, OTU25 and OTU50 contain one instance of OTU multi-frame alignment overhead. The OTUCn contains n instances of OTU multi-frame alignment overhead, numbered 1 to n. All n MFAS bytes carry the same 256-frame sequence and in each frame all n MFAS bytes carry the same value.

Figure 15-11 – Multiframe alignment signal overhead

Individual OTU/ODU overhead signals may use this central multiframe to lock their 2-frame, 4-frame, 8-frame, 16-frame, 32-frame, etc., multiframes to the principal frame.

NOTE 1 – The 80-frame OPU4 multiframe cannot be supported. A dedicated 80-frame OPU4 multiframe indicator (OMFI) is used instead.

NOTE 2 – The 20-frame OPUCn and OPUk (k = 25) multiframe cannot be supported by MFAS. A dedicated 20-frame OPUCn and OPUk multiframe indicator (OMFI) is used instead.

NOTE 3 – The 40-frame OPUk (k = 50) multiframe cannot be supported by MFAS. A dedicated 40-frame OPUk multiframe indicator (OMFI) is used instead.

15.7 OTU OH description

15.7.1 OTU overhead location

The OTU overhead location is shown in Figures 15-12 and 15-13.

The OTUk contains one instance of OTU overhead. The OTUCn contains n instances of OTU overhead, numbered 1 to n (OTU OH #1 to OTU OH #n).

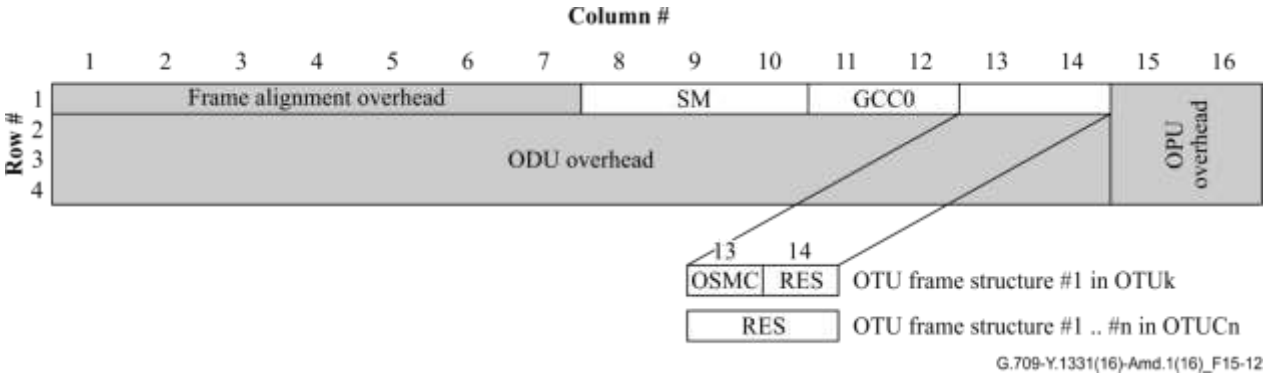


Figure 15-12 – OTU overhead

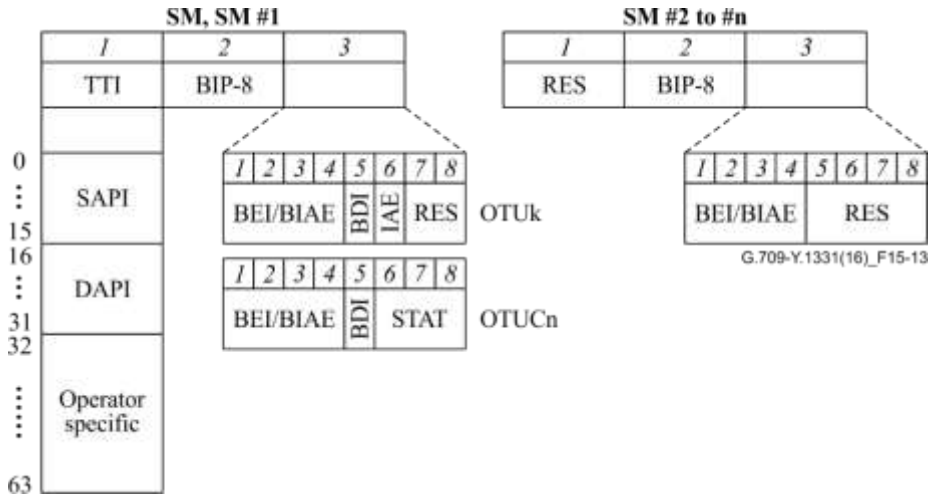


Figure 15-13 – OTU section monitoring overhead

15.7.2 OTU overhead definition

15.7.2.1 OTU section monitoring (SM) overhead

One field of OTU section monitoring (SM) overhead is defined in row 1, columns 8 to 10 to support section monitoring.

The OTUk, OTU25 and OTU50 contain one instance of OTU SM overhead. The OTUCn contains n instances of the OTU SM overhead, numbered 1 to n (SM #1 to SM #n).

The SM and SM #1 field contains the following subfields (see Figure 15-13):

– trail trace identifier (TTI);

– bit interleaved parity (BIP-8);

– backward defect indication (BDI);

– backward error indication and backward incoming alignment error (BEI/BIAE);

– incoming alignment error (IAE);

– status bits indicating the presence of an incoming alignment error or a maintenance signal (STAT);

– bits reserved for future international standardization (RES).

The SM #2 to #n fields contain the following subfields (see Figure 15-13):

– bit interleaved parity (BIP-8);

– backward error indication and backward incoming alignment error (BEI/BIAE);

– bits reserved for future international standardization (RES).

15.7.2.1.1 OTU SM trail trace identifier (TTI)

For section monitoring, a one-byte trail trace identifier (TTI) overhead is defined to transport the 64-byte TTI signal specified in clause 15.2 or a discovery message as specified in [ITU-T G.7714.1].

The OTUk, OTU25 and OTU50 and OTUCn contain one instance of OTU TTI overhead.

The 64-byte TTI signal shall be aligned with the OTU multiframe (see clause 15.6.2.2) and transmitted four times per multiframe. Byte 0 of the 64-byte TTI signal shall be present at OTU multiframe positions 0000 0000 (0x00), 0100 0000 (0x40), 1000 0000 (0x80) and 1100 0000 (0xC0).

15.7.2.1.2 OTU SM error detection code (BIP-8)

For section monitoring, a one-byte error detection code signal is defined in the OTU SM overhead. This byte provides a bit interleaved parity-8 (BIP-8) code.

NOTE – The notation *BIP-8* refers only to the number of BIP bits and not to the EDC usage (i.e., what quantities are counted). For definition of BIP-8 refer to BIP-X definition in [ITU-T G.707].

The OTU BIP-8 is computed over the bits in the OPU (columns 15 to 3824) area of OTU frame i, and inserted in the OTU BIP-8 overhead location in OTU frame i+2 (see Figure 15-14).

The OTUk, OTU25 and OTU50 contain one instance of OTU BIP-8 overhead. The OTUCn contains n instances of the OTU BIP-8 overhead, numbered 1 to n (BIP-8 #1 to BIP-8 #n).

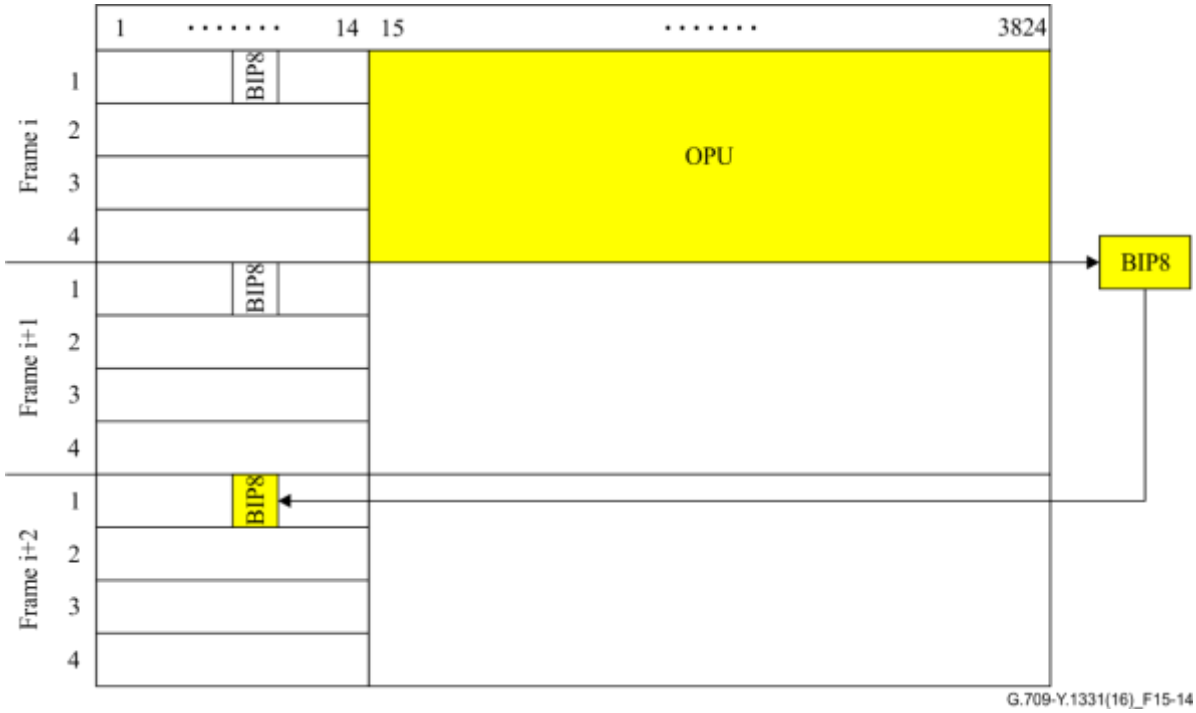


Figure 15-14 – OTU SM BIP-8 computation

15.7.2.1.3 OTU SM backward defect indication (BDI)

For section monitoring, a single-bit backward defect indication (BDI) signal is defined to convey the signal fail status detected in a section termination sink function in the upstream direction.

BDI is set to "1" to indicate an OTU backward defect indication; otherwise, it is set to "0".

The OTUk, OTU25 and OTU50 and OTUCn contain one instance of OTU BDI overhead.

15.7.2.1.4 OTU SM backward error indication and backward incoming alignment error (BEI/BIAE)

For section monitoring, a four-bit backward error indication (BEI) and backward incoming alignment error (BIAE) signal is defined. This signal is used to convey in the upstream direction the count of interleaved-bit blocks that have been detected in error by the corresponding OTU section monitoring sink using the BIP-8 code. It is also used to convey in the upstream direction an incoming alignment error (IAE) condition that is detected in the corresponding OTU section monitoring sink in the IAE overhead.

During an IAE condition the code "1011" is inserted into the BEI/BIAE field and the error count is ignored. Otherwise, the error count (0-8) is inserted into the BEI/BIAE field. The remaining six possible values represented by these four bits can only result from some unrelated condition and shall be interpreted as zero errors (see Table 15-1) and BIAE not active.

The OTUk, OTU25 and OTU50 contain one instance of OTU BEI/BIAE overhead. The OTUCn contains n instances of the OTU BEI/BIAE overhead, numbered 1 to n (BEI/BIAE #1 to BEI/BIAE #n).

NOTE – The BIAE indication of an OTUCn is transported n times (i.e., in OTUC SM # 1 to SM #n) and detected in OTUC SM #1 only.

Table 15-1 – OTU SM BEI/BIAE interpretation

OTU SM BEI/BIAE bits 1 2 3 4	**BIAE**	**BIP violations**
0 0 0 0	false	0
0 0 0 1	false	1
0 0 1 0	false	2
0 0 1 1	false	3
0 1 0 0	false	4
0 1 0 1	false	5
0 1 1 0	false	6
0 1 1 1	false	7
1 0 0 0	false	8
1 0 0 1, 1 0 1 0	false	0
1 0 1 1	true	0
1 1 0 0 to 1 1 1 1	false	0

15.7.2.1.5 OTUk, OTU25 and OTU50 SM incoming alignment error overhead (IAE)

A single-bit incoming alignment error (IAE) signal is defined to allow the S-CMEP ingress point to inform its peer S-CMEP egress point that an alignment error in the incoming signal has been detected.

IAE is set to "1" to indicate a frame alignment error, otherwise it is set to "0".

The S-CMEP egress point may use this information to suppress the counting of bit errors, which may occur as a result of a frame phase change of the OTUk, OTU25, OTU50 at the ingress of the section.

15.7.2.1.6 OTU SM reserved overhead (RES)

For section monitoring of the OTUk, OTU25 and OTU50, two bits in the SM overhead are reserved (RES) for future international standardization. They are set to "00".

For section monitoring of the OTUCn, 12 bits in the SM overhead in the OTU frame structures #2 to #n are reserved for future international standardization. The value of these bits is set to "0".

15.7.2.1.7 OTUCn SM status (STAT)

For section monitoring, three bits are defined as status bits (STAT). They indicate the presence of a maintenance signal or if there is an incoming alignment error at the source S-CMEP, (see Table 15-2).

Table 15-2 – OTUCn SM status interpretation

SM byte 3 bits 6 7 8	Status
0 0 0	Reserved for future international standardization
0 0 1	In use without IAE
0 1 0	In use with IAE
0 1 1	Reserved for future international standardization
1 0 0	Reserved for future international standardization (Note)
1 0 1	Reserved for future international standardization
1 1 0	Reserved for future international standardization
1 1 1	Maintenance signal: OTUCn-AIS
NOTE – This bit pattern may appear temporarily due to the presence of the FlexO SquelchText pattern. Refer to [ITU-T G.709.1].	

A S-CMEP ingress point sets these bits to either "001" to indicate to its peer S-CMEP egress point that there is no incoming alignment error (IAE), or to "010" to indicate that there is an incoming alignment error.

The S-CMEP egress point may use this information to suppress the counting of bit errors, which may occur as a result of a frame phase change of the ODUCn at the ingress of the section.

15.7.2.2 OTU general communication channel 0 (GCC0)

Two bytes are allocated in the OTU overhead to support a general communications channel or a discovery channel as specified in [ITU-T G.7714.1] between OTU termination points.

This general communication channel is a clear channel and any format specification is outside of the scope of this Recommendation. These bytes are located in row 1, columns 11 and 12 of the OTU overhead.

The OTUk, OTU25 and OTU50 contain one instance of OTU GCC0 overhead. The OTUCn contains n instances of the OTU GCC0 overhead, numbered 1 to n (GCC0 #1 to GCC0 #n).

The GCC0 #1 to #n overhead are combined to provide one communication channel as illustrated in Figure 15-15.

NOTE – For vendor specific interfaces it is an option not to combine the GCC0 #1 to #n and instead use only the first GCC0 (GCC0 #1) as communication channel. GCC0 #2 to #n are not used.

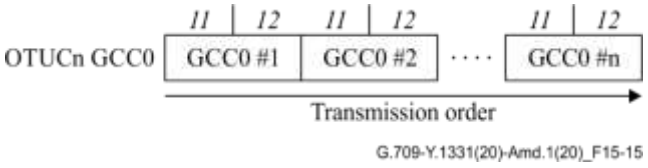


Figure 15-15 – OTUCn GCC0 transmission order

15.7.2.3 OTU reserved overhead (RES)

One byte of the OTU overhead in OTU frame structure #1 is reserved for future international standardization. This byte is located in row 1, column 14. This byte is set to all-0s.

Two bytes of the OTU overhead in OTU frame structures #2 to #n are reserved for future international standardization. These bytes are located in the OTU overhead in row 1, columns 13 and 14. These bytes are set to all-0s.

15.7.2.4 OTU OTN synchronisation message channel (OSMC)

For synchronisation purposes, one byte is defined in the OTU overhead as an OTN synchronisation message channel to transport SSM, eSSM and PTP messages within SOTU and MOTU interfaces. The OSMC bandwidth is listed in Table 15-3.

The OTUk, OTU25 and OTU50 contain one instance of OTU OSMC overhead.

NOTE 1 – Support of OTU OSMC in an OTN interface is optional.

NOTE 2 – Equipment designed prior to Edition 5.0 of this Recommendation may not be able to support OTU OSMC via their OTN point-to-point interfaces.

NOTE 3 – OTN interfaces with vendor specific application identifiers may support an OSMC function. The encapsulation of the messages and overhead location are then vendor specific.

NOTE 4 – Equipment designed prior to Edition 6.0 of this Recommendation may not be able to support OTU OSMC via their OTN optical networking interfaces.

Table 15-3 – OSMC bandwidth

OTUk	OSMC Bandwidth (kbit/s)
OTU0	81.339
OTU1	163.361
OTU2	656.203
OTU3	2,635.932
OTU4	6,851.101
OTU25	1,726.576
OTU50	3,453.153

The SSM, eSSM and PTP messages are encapsulated into GFP-F frames as specified into [ITU-T G.7041]. PTP event messages are timestamped and after encapsulation into GFP-F frames inserted into the OSMC as specified in clause 15.7.2.4.1. GFP-F encapsulated SSM and eSSM messages (and PTP non-event messages) are inserted into the OSMC at the earliest opportunity. GFP Idle frames may be inserted between successive GFP frames.

The mapping of generic framing procedure (GFP) frames is performed by aligning the byte structure of every GFP frame with the byte of the OSMC overhead field. Since the GFP frames are of variable length and longer than one byte, a frame crosses the OTUk (k=0,1,2,3,4), OTU25 or OTU50 frame boundary.

15.7.2.4.1 Generation of event message timestamps

15.7.2.4.1.1 OTN interface event message timestamp point

The OTN interface message timestamp point [ITU-T G.8260] for a PTP event message transported over the OSMC shall be the X-frame multiframe event preceding the beginning of the GFP frame in which the PTP event message is carried. See Figure 15-16. Since the GFP frames may be longer than X-4 bytes, a frame may cross the X-frame multiframe boundary. The X-frame multiframe contains frames numbered 0, 1, .., X-1.

15.7.2.4.1.2 Event timestamp generation

All PTP event messages are timestamped on egress and ingress OTN interfaces. The timestamp shall be the time at which the event message timestamp point passes the reference plane [ITU-T G.8260] marking the boundary between the PTP node (i.e., OTN node) and the network.

OTUk, OTU25 and OTU50

Event message timestamps are generated every X-frame multiframe period at the OTUk, OTU25 and OTU50 Access Point. X is 64 for k = 0,1 and 256 for k = 2,3,4 and OTU25 and OTU50.

The first byte of a GFP(PTP event message) frame is inserted into the OTUk, OTU25 or OTU50 OSMC between 4 and X-1 frames after the X-frame multiframe boundary.

NOTE 1 – The first byte of a GFP(PTP event message) frame is not inserted into the OTUk, OTU25 or OTU50 OSMC in frame 0…3 of the X-frame multiframe.

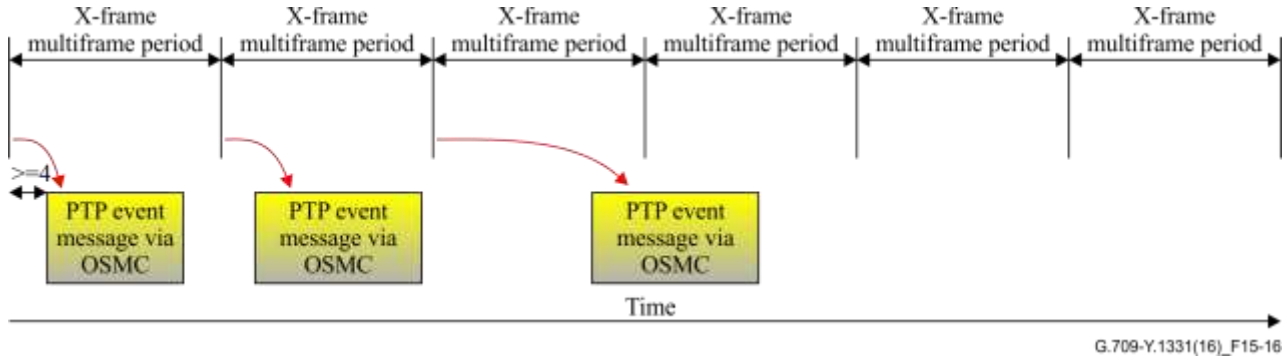


Figure 15-16 – Timing diagram example for OTUk

NOTE 2 – This time synchronization over OTN interface implementation does not generate event message timestamps using a point other than the message timestamp point [ITU-T G.8260].

In this time synchronization over OTN interface implementation, the timestamps are generated at a point removed from the reference plane. Furthermore, the time offset from the reference plane is likely to be different for inbound and outbound event messages. To meet the requirement of this subclause, the generated timestamps should be corrected for these offsets. Figure 19 in [b-IEEE 1588] illustrates these offsets. Based on this model, the appropriate corrections are as follows:

<egressTimestamp> = <egressMeasuredTimestamp> + egressLatency

<ingressTimestamp> = <ingressMeasuredTimestamp> ─ ingressLatency

where the actual timestamps <egressTimestamp> and <ingressTimestamp> measured at the reference plane are computed from the detected, i.e., measured, timestamps by their respective latencies. Failure to make these corrections results in a time offset between the slave and master clocks.

15.7.3 OTUkV overhead

The functionally standardized OTUkV frame should support, as a minimum capability, section monitoring functionality comparable to the OTUk section monitoring (see clause 15.7.2.1) with a trail trace identifier as specified in clause 15.2. Further specification of this overhead is outside the scope of this Recommendation.

15.8 ODU OH description

15.8.1 ODU OH location

The ODU overhead location is shown in Figures 15-17, 15-18 and 15-19.

The ODUk contains one instance of ODU overhead. The ODUCn contains n instances of ODU overhead, numbered 1 to n (ODU OH #1 to ODU OH #n).

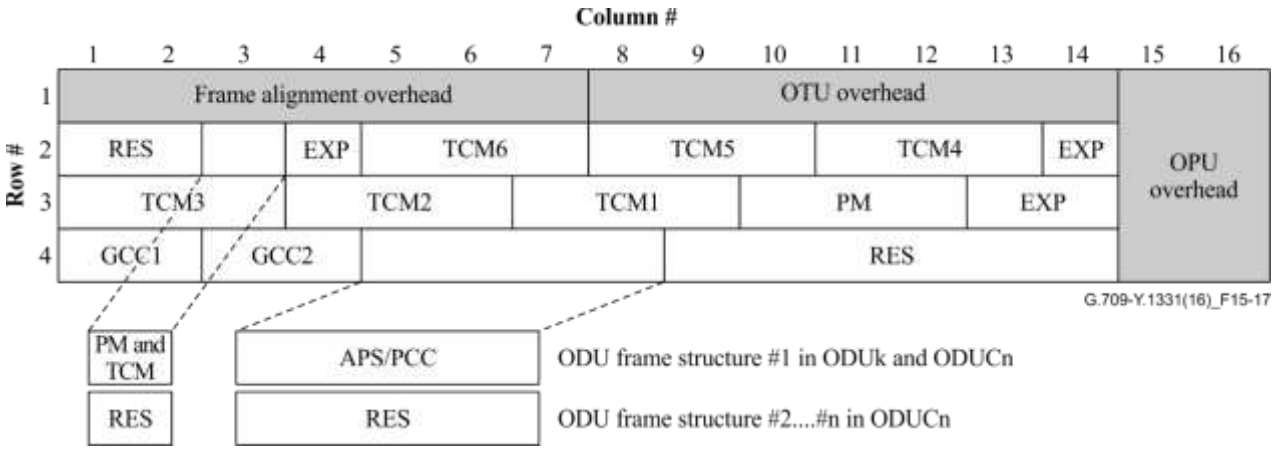


Figure 15-17 – ODU overhead

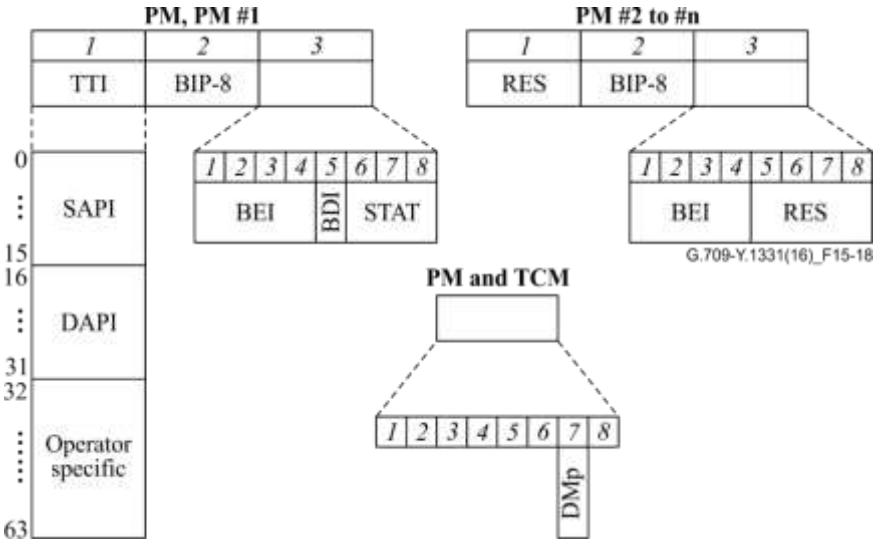


Figure 15-18 – ODU path monitoring overhead

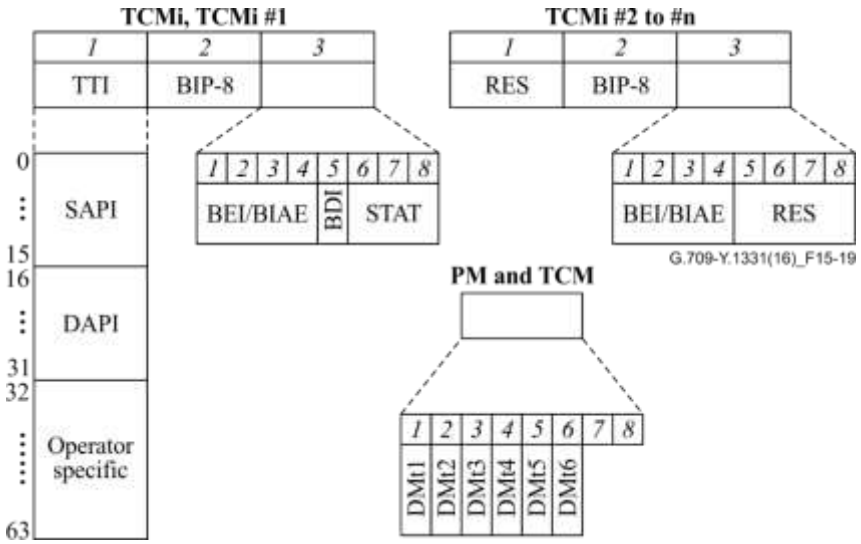


Figure 15-19 – ODU tandem connection monitoring #i overhead

15.8.2 ODU OH definition

15.8.2.1 ODU path monitoring (PM) overhead

One field of an ODU path monitoring overhead (PM) is defined in row 3, columns 10 to 12 to support path monitoring and one additional bit of path monitoring is defined in row 2, column 3, bit 7.

The ODUk contains one instance of ODU PM overhead. The ODUCn contains n instances of the ODU PM overhead, numbered 1 to n (PM #1 to PM #n).

The PM and PM #1 field contains the following subfields (see Figure 15-18):

– trail trace identifier (TTI);

– bit interleaved parity (BIP-8);

– backward defect indication (BDI);

– backward error indication (BEI);

– status bits indicating the presence of a maintenance signal (STAT).

The PM #2 to #n fields contain the following subfields (see Figure 15-18):

– bit interleaved parity (BIP-8)

– backward error indication (BEI)

– reserved (RES).

The PM&TCM field contains the following PM subfield (see Figure 15-18):

– path delay measurement (DMp).

For the case of ODUk, the content of the PM field, except the STAT subfield, will be undefined (pattern will be all-1s, 0110 0110 or 0101 0101 repeating) during the presence of a maintenance signal (e.g., ODU-AIS, ODU-OCI, ODU-LCK). The content of the PM&TCM field will be undefined (pattern will be all-1s, 0110 0110 or 0101 0101 repeating) during the presence of a maintenance signal. Refer to clause 16.5.

For the case of ODUCn, the content of the PM field, except the STAT subfield, will be undefined (pattern will be all-1s or 0101 0101 repeating) during the presence of a maintenance signal (e.g., ODUCn-AIS, ODUCn-LCK). Refer to clause 16.5.

15.8.2.1.1 ODU PM trail trace identifier (TTI)

For path monitoring, a one-byte trail trace identifier (TTI) overhead is defined to transport the 64-byte TTI signal specified in clause 15.2 or a discovery message as specified in [ITU-T G.7714.1].

The ODUk and ODUCn contain one instance of ODU PM TTI overhead.

The 64-byte TTI signal shall be aligned with the ODU multiframe (see clause 15.6.2.2) and transmitted four times per multiframe. Byte 0 of the 64-byte TTI signal shall be present at ODU multiframe positions 0000 0000 (0x00), 0100 0000 (0x40), 1000 0000 (0x80) and 1100 0000 (0xC0).

15.8.2.1.2 ODU PM error detection code (BIP-8)

For path monitoring, a one-byte error detection code signal is defined in the ODU PM overhead. This byte provides a bit interleaved parity-8 (BIP-8) code.

NOTE – The notation BIP-8 refers only to the number of BIP bits and not to the EDC usage (i.e., what quantities are counted). For definition of BIP-8, refer to the BIP-X definition in [ITU-T G.707].

Each ODU BIP-8 is computed over the bits in the OPU (columns 15 to 3824) area of ODU frame i, and inserted in the ODU PM BIP-8 overhead location in the ODU frame i+2 (see Figure 15-20).

The ODUk contains one instance of ODU PM BIP-8 overhead. The ODUCn contains n instances of the ODU PM BIP-8 overhead, numbered 1 to n (BIP-8 #1 to BIP-8 #n).

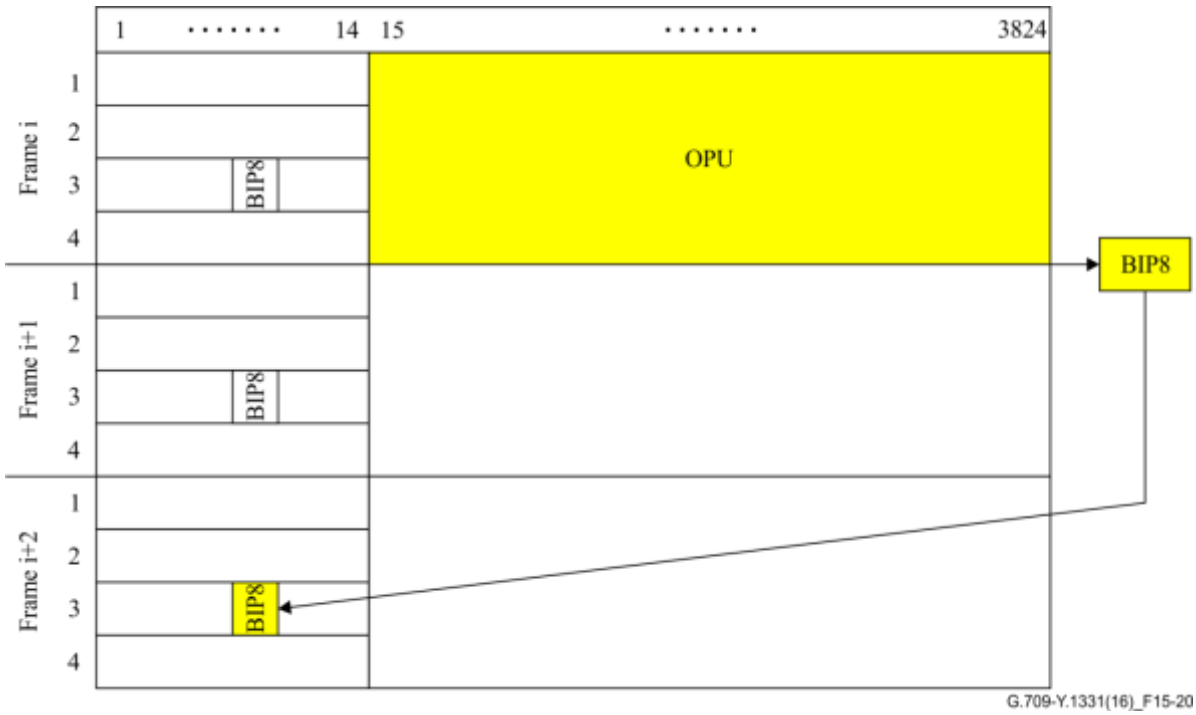


Figure 15-20 – ODU PM BIP-8 computation

15.8.2.1.3 ODU PM backward defect indication (BDI)

For path monitoring, a single-bit backward defect indication (BDI) signal is defined to convey the signal fail status detected in a path termination sink function in the upstream direction.

BDI is set to "1" to indicate an ODU backward defect indication, otherwise it is set to "0".

The ODUk and ODUCn contain one instance of ODU PM BDI overhead.

15.8.2.1.4 ODU PM backward error indication (BEI)

For path monitoring, a four-bit backward error indication (BEI) signal is defined to convey in the upstream direction the count of interleaved-bit blocks that have been detected in error by the corresponding ODU path monitoring sink using the BIP-8 code. This count has nine legal values, namely 0-8 errors. The remaining seven possible values represented by these four bits can only result from some unrelated condition and shall be interpreted as zero errors (see Table 15-2).

The ODUk contains one instance of ODU PM BEI overhead. The ODUCn contains n instances of the ODU PM BEI overhead, numbered 1 to n (BEI #1 to BEI #n).

Table 15-4 – ODU PM BEI interpretation

ODU PM BEI bits 1 2 3 4	BIP violations
0 0 0 0	0
0 0 0 1	1
0 0 1 0	2
0 0 1 1	3
0 1 0 0	4
0 1 0 1	5
0 1 1 0	6

Table 15-4 – ODU PM BEI interpretation

ODU PM BEI bits 1 2 3 4	BIP violations
0 1 1 1	7
1 0 0 0	8
1 0 0 1 to 1 1 1 1	0

15.8.2.1.5 ODU PM status (STAT)

For path monitoring, three bits are defined as status bits (STAT). They indicate the presence of a maintenance signal (see Table 15-3).

A P-CMEP sets these bits to "001".

The ODUk and ODUCn contain one instance of ODU PM STAT overhead.

Table 15-5 – ODU PM status interpretation

PM byte 3 bits 6 7 8	Status
0 0 0	Reserved for future international standardization
0 0 1	Normal path signal
0 1 0	Reserved for future international standardization
0 1 1	Reserved for future international standardization
1 0 0	Reserved for future international standardization (Note)
1 0 1	Maintenance signal: ODU-LCK
1 1 0	ODUk: Maintenance signal: ODU-OCI ODUCn: Reserved for future international standardization
1 1 1	Maintenance signal: ODU-AIS
NOTE – This bit pattern may appear temporarily due to the presence of the FlexO SquelchText pattern. Refer to [ITU-T G.709.1].	

15.8.2.1.6 ODU PM delay measurement (DMp)

For ODU path monitoring, a one-bit path delay measurement (DMp) signal is defined to convey the start of the delay measurement test.

The ODUk and ODUCn contain one instance of ODU PM DMp overhead.

The DMp signal consists of a constant value (0 or 1) that is inverted at the beginning of a two-way delay measurement test. The transition from 0→1 in the sequence …0000011111…, or the transition from 1→0 in the sequence …1111100000… represents the beginning of a path delay measurement. The new value of the DMp signal is maintained until the beginning of a new delay measurement test.

This DMp signal is inserted by the DMp originating P-CMEP and sent to the far-end P-CMEP. This far-end P-CMEP loops back the DMp signal towards the originating P-CMEP. The originating P-CMEP measures the number of frame periods between the moment the DMp signal value is inverted and the moment this inverted DMp signal value is received back from the far-end P-CMEP.

The receiver at the originating P-CMEP should apply a persistency check on the received DMp signal to be tolerant for bit errors emulating the transition of the delay measurement sequence. The additional frames that are used for such persistency checking should not be added to the delay frame count. The looping P-CMEP should loop back each received DMp bit within approximately 100 µs.

Refer to [ITU-T G.798] for the specific path delay measurement process specifications.

NOTE 1 – Path delay measurements can be performed on-demand, to provide the momentary two-way transfer delay status, and pro-active, to provide 15-minute and 24-hour two-way transfer delay performance management snapshots.

NOTE 2 – Equipment designed according to the 2008 or earlier versions of this Recommendation may not be capable of supporting this path delay monitoring. For such equipment, the DMp bit is a bit reserved for future international standardization and set to zero.

NOTE 3 – This process measures a round trip delay. The one way delay may not be half of the round trip delay in the case where the transmit and receive directions of the ODU network connection are of unequal lengths (e.g., in networks deploying unidirectional protection switching).

15.8.2.1.7 ODU PM reserved overhead (RES)

For path monitoring of the OTUCn, 12 bits in the PM overhead in the ODU OH #2 to #n are reserved for future international standardization. The value of these bits is set to "0".

The ODUk contains no ODU PM RES overhead. The ODUCn contains n-1 instances of the ODU PM RES overhead.

15.8.2.2 ODU tandem connection monitoring (TCM) overhead

Six fields of an ODU tandem connection monitoring (TCM) overhead are defined in row 2, columns 5 to 13 and row 3, columns 1 to 9 of the ODU overhead; and six additional bits of tandem connection monitoring are defined in row 2, column 3, bits 1 to 6.

TCM supports monitoring of ODUk connections for one or more of the following network applications (refer to [ITU-T G.805], [ITU-T G.872], [ITU-T G.873.2] and [ITU-T G.7714.1]):

– optical UNI-to-UNI tandem connection monitoring; monitoring the ODU connection through the public transport network (from public network ingress network termination to egress network termination);

– optical NNI-to-NNI tandem connection monitoring; monitoring the ODU connection through the network of a network operator (from operator network ingress network termination to egress network termination);

– sublayer monitoring for linear 1+1, 1:1 and 1:n ODUk subnetwork connection protection switching, to determine the signal fail and signal degrade conditions;

– sublayer monitoring for ODUk shared ring protection (SRP-1) protection switching as specified in [ITU-T G.873.2], to determine the signal fail and signal degrade conditions;

– sublayer monitoring for ODUk connection passing through two or more concatenated ODUk link connections (supported by back-to-back OTU trails), to provide a discovery message channel as specified in [ITU-T G.7714.1];

– monitoring an ODUk tandem connection for the purpose of detecting a signal fail or signal degrade condition in a switched ODUk connection, to initiate automatic restoration of the connection during fault and error conditions in the network;

– monitoring an ODUk tandem connection for, e.g., fault localization or verification of delivered quality of service.

TCM supports monitoring of segments of ODUCn connections which span multiple OTSiA subnetworks. The ODUCn TC-CMEP are located on OTUCn interface ports at the edge of OTSiA

subnetworks and/or on OTUCn interface ports in ODUCn terminating nodes (e.g., in ODUk cross connects).

The six TCM fields are numbered TCM1, TCM2, ..., TCM6.

The ODUk contains one instance of ODU TCM1 to TCM6 overhead. The ODUCn contains n instances of the ODU TCM1 to TCM6 overhead, numbered 1 to n (TCMi #1 to TCMi #n).

Each TCMi and TCMi #1 field contains the following subfields (see Figure 15-19):

– trail trace identifier (TTI);

– bit interleaved parity 8 (BIP-8);

– backward defect indication (BDI);

– backward error indication and backward incoming alignment error (BEI/BIAE);

– status bits indicating the presence of a TCM overhead, incoming alignment error, or a maintenance signal (STAT).

Each TCMi #2 to #n field contains the following subfields (see Figure 15-19):

– bit interleaved parity 8 (BIP-8);

– backward error indication and backward incoming alignment error (BEI/BIAE);

– reserved (RES).

The PMandTCM field contains the following TCM subfields (see Figure 15-19):

– tandem connection delay measurement (DMti, i=1 to 6).

For the case of ODUk, the content of the TCM fields, except the STAT subfield, will be undefined (pattern will be all-1s, 0110 0110 or 0101 0101 repeating) during the presence of a maintenance signal (e.g., ODUk-AIS, ODUk-OCI, ODUk-LCK). The content of the PM&TCM field will be undefined (pattern will be all-1s, 0110 0110 or 0101 0101 repeating) during the presence of a maintenance signal. Refer to clause 16.5.

For the case of ODUCn, the content of the TCM field groups, except the STAT subfield, will be undefined (pattern will be all-1s or 0101 0101 repeating) during the presence of a maintenance signal (e.g., ODUCn-AIS, ODUCn-LCK). The content of the PM and TCM field will be undefined (pattern will be all-1s or 0101 0101 repeating) during the presence of a maintenance signal. Refer to clause 6.5.

A TCM field and PM&TCM bit is assigned to a monitored connection as described in clause 15.8.2.2.6. The number of monitored connections along an ODU trail may vary between 0 and 6. These monitored connections may be nested, cascaded or both. Nesting and cascading are the default operational configurations. Overlapping is an additional configuration for testing purposes only. Overlapped monitored connections must be operated in a non-intrusive mode in which the maintenance signals ODU-AIS and ODU-LCK are not generated. For the case where one of the endpoints in an overlapping monitored connection is located inside an SNC protected domain while the other endpoint is located outside the protected domain, the SNC protection should be forced to working when the endpoint of the overlapping monitored connection is located on the working connection, and forced to protection when the endpoint is located on the protection connection.

Nesting and cascading configurations are shown in Figure 15-21. Monitored connections A1-A2/B1-B2/C1-C2 and A1-A2/B3-B4 are nested, while B1-B2/B3-B4 are cascaded. Overlapping is shown in Figure 15-22 (B1-B2 and C1-C2).

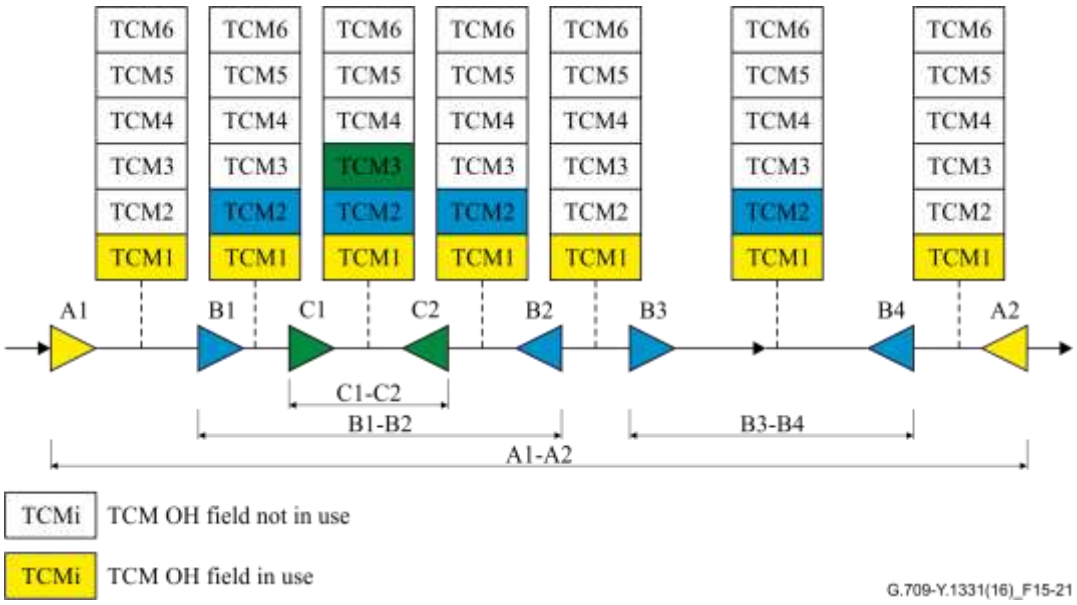


Figure 15-21 – Example of nested and cascaded ODU monitored connections

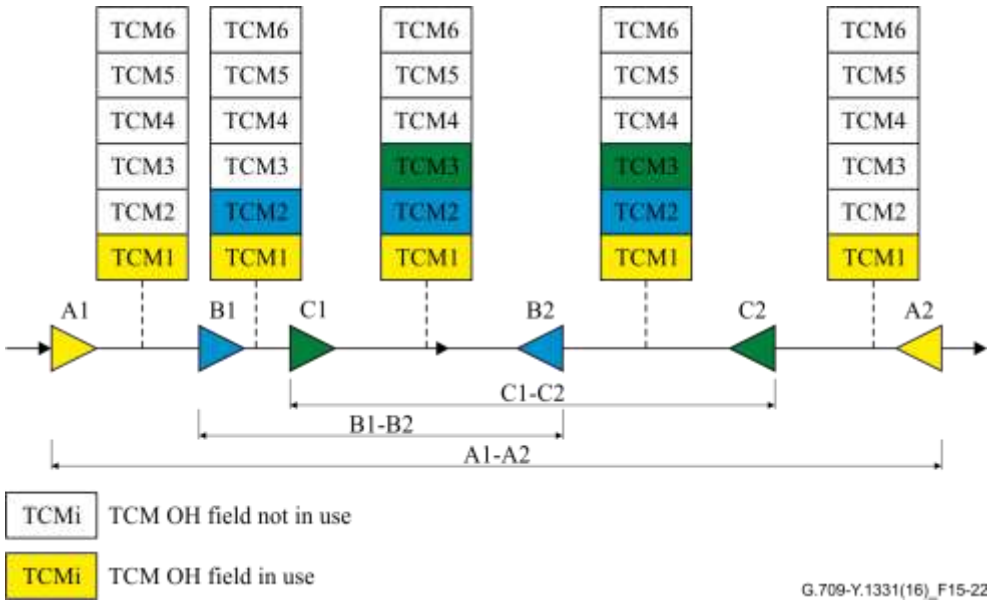


Figure 15-22 – Example of overlapping ODU monitored connections

15.8.2.2.1 ODU TCM trail trace identifier (TTI)

For each tandem connection monitoring field, one byte of overhead is allocated for the transport of the 64-byte trail trace identifier (TTI) specified in clause 15.2 or a discovery message as specified in [ITU-T G.7714.1] for TCM6.

The ODUk and ODUCn contain one instance of ODU TTI overhead.

The 64-byte TTI signal shall be aligned with the ODU multiframe (see clause 15.6.2.2) and transmitted four times per multiframe. Byte 0 of the 64-byte TTI signal shall be present at ODU multiframe positions 0000 0000 (0x00), 0100 0000 (0x40), 1000 0000 (0x80) and 1100 0000 (0xC0).

15.8.2.2.2 ODU TCM error detection code (BIP-8)

For each tandem connection monitoring field, a one-byte error detection code signal is defined in the ODU TCMi overhead. This byte provides a bit interleaved parity-8 (BIP-8) code.

NOTE – The notation *BIP-8* refers only to the number of BIP bits, and not to the EDC usage (i.e., what quantities are counted). For definition of BIP-8 refer to the BIP-X definition in [ITU-T G.707].

Each ODU TCM BIP-8 is computed over the bits in the OPU (columns 15 to 3824) area of ODU frame i, and inserted in the ODU TCM BIP-8 overhead location (associated with the tandem connection monitoring level) in ODU frame i+2 (see Figure 15-23).

The ODUk contains one instance of ODU TCMi BIP-8 overhead. The ODUCn contains n instances of the ODU TCMi BIP-8 overhead, numbered 1 to n (BIP-8 #1 to BIP-8 #n).

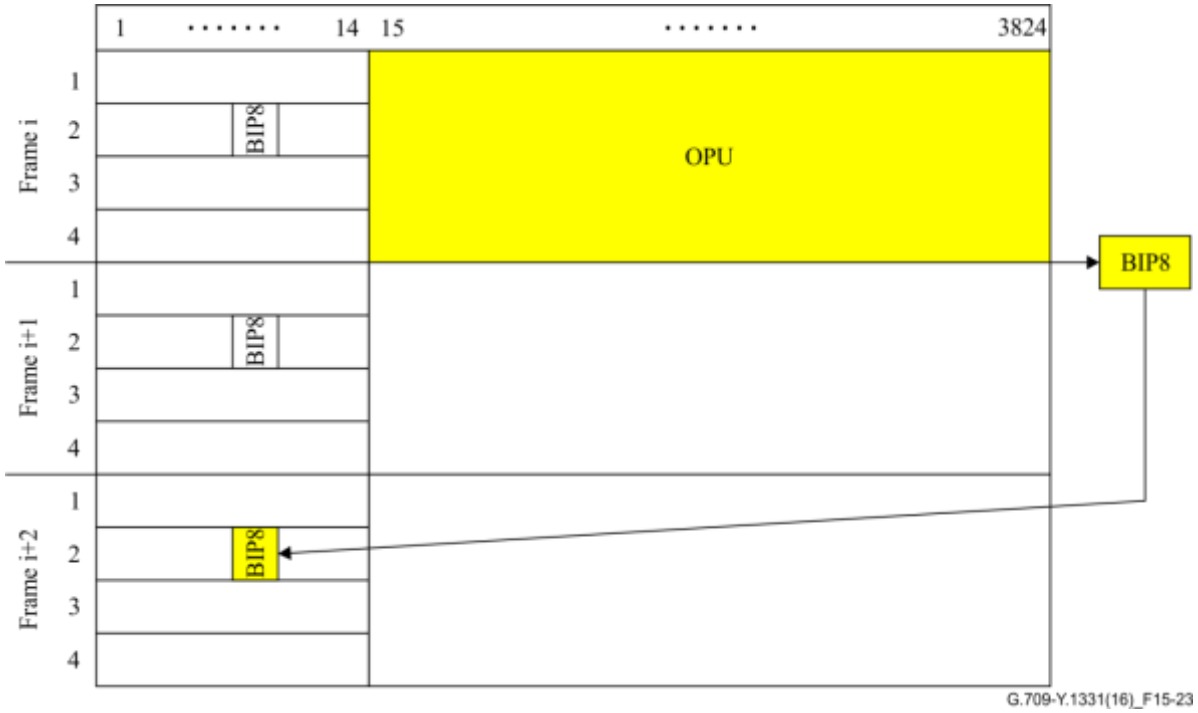


Figure 15-23 – ODU TCM BIP-8 computation

15.8.2.2.3 ODU TCM backward defect indication (BDI)

For each tandem connection monitoring field, a single-bit backward defect indication (BDI) signal is defined to convey the signal fail status detected in a tandem connection termination sink function in the upstream direction.

BDI is set to "1" to indicate an ODUk backward defect indication; otherwise, it is set to "0".

The ODUk and ODUCn contain one instance of ODU TCMi BDI overhead.

15.8.2.2.4 ODU TCM backward error indication (BEI) and backward incoming alignment error (BIAE)

For each tandem connection monitoring field, a 4-bit backward error indication (BEI) and backward incoming alignment error (BIAE) signal is defined. This signal is used to convey in the upstream direction the count of interleaved-bit blocks that have been detected as being in error by the corresponding ODU tandem connection monitoring sink using the BIP-8 code. It is also used to convey in the upstream direction an incoming alignment error (IAE) condition that is detected in the corresponding ODU tandem connection monitoring sink in the IAE overhead.

During an IAE condition the code "1011" is inserted into the BEI/BIAE field and the error count is ignored. Otherwise the error count (0-8) is inserted into the BEI/BIAE field. The remaining six possible values represented by these four bits can only result from some unrelated condition and shall be interpreted as zero errors (see Table 15-4) and BIAE not active.

The ODUk contains one instance of ODU TCM BEI/BIAE overhead. The ODUCn contains n instances of the ODU TCMi BEI/BIAE overhead, numbered 1 to n (BEI/BIAE #1 to BEI/BIAE #n).

Table 15-6 – ODU TCM BEI/BIAE interpretation

ODUk TCM BEI/BIAE bits 1 2 3 4	BIAE	BIP violations
0 0 0 0	false	0
0 0 0 1	false	1
0 0 1 0	false	2
0 0 1 1	false	3
0 1 0 0	false	4
0 1 0 1	false	5
0 1 1 0	false	6
0 1 1 1	false	7
1 0 0 0	false	8
1 0 0 1, 1 0 1 0	false	0
1 0 1 1	true	0
1 1 0 0 to 1 1 1 1	false	0

15.8.2.2.5 ODU TCM status (STAT)

For each tandem connection monitoring field, three bits are defined as status bits (STAT). They indicate the presence of a maintenance signal, if there is an incoming alignment error at the source TC-CMEP, or if there is no source TC-CMEP active (see Table 15-7).

The ODUk and ODUCn contain one instance of ODU TCM STAT overhead.

Table 15-7 – ODU TCM status interpretation

TCM byte 3 bits 6 7 8	Status
0 0 0	No source TC
0 0 1	In use without IAE
0 1 0	In use with IAE
0 1 1	Reserved for future international standardization
1 0 0	Reserved for future international standardization (Note)
1 0 1	Maintenance signal: ODU-LCK
1 1 0	ODUk: Maintenance signal: ODUk-OCI ODUCn: Reserved for future international standardization
1 1 1	Maintenance signal: ODU-AIS
NOTE – This bit pattern may appear temporarily due to the presence of the FlexO SquelchText pattern. Refer to [ITU-T G.709.1].	

A P-CMEP sets these bits to "000".

A TC-CMEP ingress point sets these bits to either "001" to indicate to its peer TC-CMEP egress point that there is no incoming alignment error (IAE), or to "010" to indicate that there is an incoming alignment error.

The TC-CMEP egress point may use this information to suppress the counting of bit errors, which may occur as a result of a frame phase change of the ODU at the ingress of the tandem connection.

15.8.2.2.6 TCM overhead field assignment

Each TC-CMEP will be inserting/extracting its TCM overhead from one of the 6 TCM_i overhead fields and one of the 6 DMt_i fields. The specific TCM_i/DMt_i overhead field is provisioned by the network operator, network management system or switching control plane.

At a domain interface, it is possible to provision the maximum number (0 to 6) of tandem connection levels which will be passed through the domain. The default is three. These tandem connections should use the lower TCM_i/DMt_i overhead fields $TCM_1/DMt_1...TCM_{MAX}/DMt_{MAX}$. Overhead in TCM/DMt fields beyond the maximum (TCM_{max+1}/DMt_{max+1} and above) may/will be overwritten in the domain.

The TCM6 overhead field is assigned to monitor an ODUk connection which is supported by two or more concatenated ODUk link connections (supported by back-to-back OTUk trails). [ITU-T G.7714.1] specifies a discovery application which uses the TCM6 TTI SAPI field as discovery message channel. [ITU-T G.873.2] specifies an ODUk SRP-1 protection application which uses the TCM6 field to monitor the status/performance of the ODUk connection between two adjacent ODUk SRP-1 nodes.

Example

For the case of an ODUk leased circuit, the user may have been assigned one level of TCM, the service provider one level of TCM and each network operator (having a contract with the service provider) four levels of TCM. For the case where a network operator subcontracts part of its ODUk connection to another network operator, these four levels are to be split; e.g., two levels for the subcontracting operator.

This would result in the following TCM OH allocation:

– User: TCM1/DMt1 overhead field between the two user subnetworks, and TCM1/DMt1..TCM6/DMt6 within its own subnetwork;

– Service provider (SP): TCM2/DMt2 overhead field between two UNIs;

– Network operators NO1, NO2, NO3 having contract with service provider: TCM3/DMt3, TCM4/DMt4, TCM5/DMt5, TCM6/DMt6. Note that NO2 (which is subcontracting) cannot use TCM5/DMt5 and TCM6/DMt6 in the connection through the domain of NO4;

– NO4 (having subcontract with NO2): TCM5/DMt5, TCM6/DMt6.

See Figure 15-24.

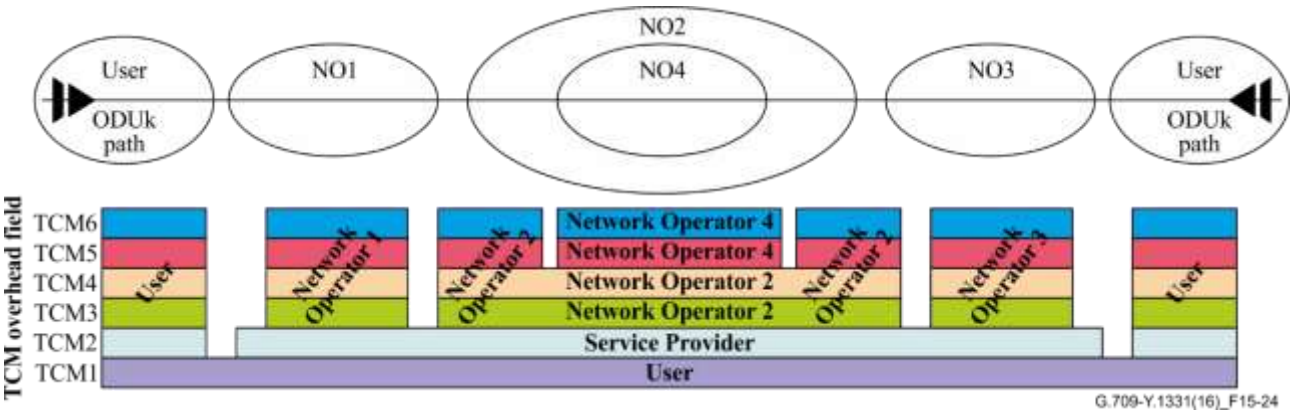


Figure 15-24 – Example of TCM overhead field assignment

15.8.2.2.7 Blank clause

This clause is intentionally left blank.

15.8.2.2.8 ODU TCM delay measurement (DMti, i=1 to 6)

For ODU tandem connection monitoring, a one-bit tandem connection delay measurement (DMti) signal is defined to convey the beginning of the delay measurement.

The ODUk and ODUCn contain six instances of ODU TCM DMti overhead.

The DMti signal consists of a constant value (0 or 1) that is inverted at the beginning of a two-way delay measurement test. The transition from 0→1 in the sequence …0000011111…, or the transition from 1→0 in the sequence …1111100000… represents the beginning of a tandem connection delay measurement. The new value of the DMti signal is maintained until the beginning of a new delay measurement test at the TCM level i.

This DMti signal is inserted by the DMti originating TC-CMEP and sent to the far-end TC-CMEP. This far-end TC-CMEP loops back the DMti signal towards the originating TC-CMEP. The originating TC-CMEP measures the number of frame periods between the moment the DMti signal value is inverted and the moment this inverted DMti signal value is received back from the far-end TC-CMEP. The receiver at the originating TC-CMEP should apply a persistency check on the received DMti signal to be tolerant for bit errors emulating the transition of the delay measurement sequence. The additional frames that are used for such persistency checking should not be added to the delay frame count. The looping TC-CMEP should loop back each received DMti bit within approximately 100 µs.

Refer to [ITU-T G.798] for the specific tandem connection delay measurement process specifications.

NOTE 1 – Tandem connection delay measurements can be performed on-demand, to provide the momentary two-way transfer delay status, and pro-active, to provide 15-minute and 24-hour two-way transfer delay performance management snapshots.

NOTE 2 – Equipment designed according to the 2008 or earlier versions of this Recommendation may not be capable of supporting this tandem connection delay monitoring. For such equipment, the DMti bit is a bit reserved for future international standardization.

NOTE 3 – This process measures a round trip delay. The one way delay may not be half of the round trip delay in the case where the transmit and receive directions of the ODUk tandem connection are of unequal lengths (e.g., in networks deploying unidirectional protection switching).

15.8.2.2.9 ODU TCM reserved overhead (RES)

For tandem connection monitoring, 12 bits in the TCMi overhead are reserved for future international standardization in TCMi OH #2 to #n. The value of these bits is set to "0".

The ODUk contains no ODU TCMi RES overhead. The ODUCn contains n-1 instances of the ODU TCMi RES overhead.

15.8.2.3 ODU general communication channels (GCC1, GCC2)

Two fields of two bytes are allocated in the ODU overhead to support two general communications channels or two discovery channels as specified in [ITU-T G.7714.1] between any two network elements with access to the ODU frame structure (i.e., at 3R regeneration points).

These general communication channels are clear channels and any format specification is outside of the scope of this Recommendation. The bytes for GCC1 are located in row 4, columns 1 and 2, and the bytes for GCC2 are located in row 4, columns 3 and 4 of the ODU overhead.

The ODUk contains one instance of ODU GCC1, GCC2 overhead. The ODUCn contains n instances of the ODU GCC1, GCC2 overhead, numbered 1 to n (GCC1 #1 to GCC1 #n, GCC2 #1 to GCC2 #n).

The GCC1 #1 to #n overhead are combined to provide one communication channel as illustrated in Figure 15-25.

The GCC2 #1 to #n overhead are combined to provide another communication channel as illustrated in Figure 15-25.

The GCC1 #1 to #n plus GCC2 #1 to #n overhead may be combined to provide one communication channel as illustrated in Figure 15-25.

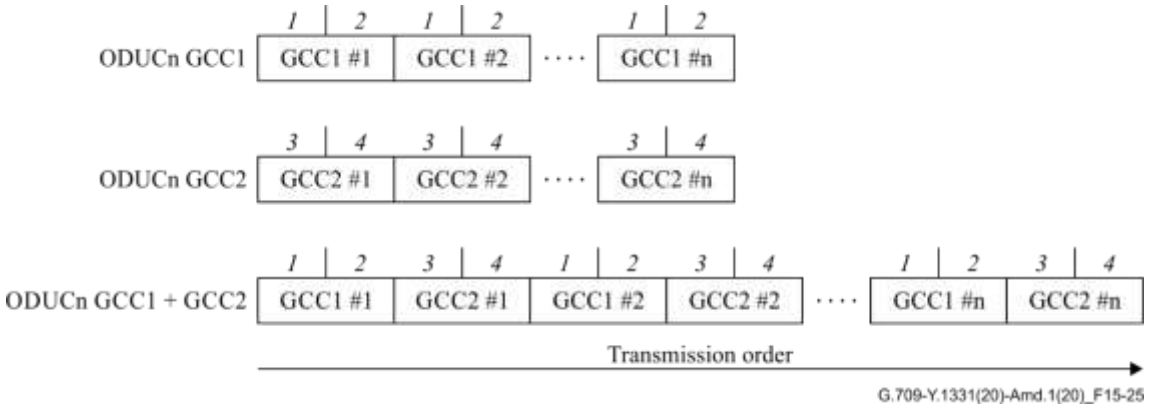


Figure 15-25 – ODUCn GCC1, GCC2 and GCC1+2 transmission order

15.8.2.4 ODU automatic protection switching and protection communication channel (APS/PCC)

A four-byte ODU-APS/PCC signal is defined in row 4, columns 5 to 8 of the ODU overhead. Up to eight levels of nested APS/PCC signals may be present in this field.

The ODUk and ODUCn contain one instance of ODU APS/PCC overhead.

For ODUk, the APS/PCC bytes in a given frame are assigned to a dedicated connection monitoring level depending on the value of MFAS as follows:

Table 15-8 – Multiframe to allow separate APS/PCC for each monitoring level

MFAS bits 6 7 8	APS/PCC channel applies to connection monitoring level	Protection scheme using the APS/PCC channel (Note 1)
0 0 0	ODUk Path	ODUk SNC/Ne, ODUj CL-SNCG/I, Client SNC/I, ODU SRP-p
0 0 1	ODUk TCM1	ODUk SNC/S, ODUk SNC/Ns
0 1 0	ODUk TCM2	ODUk SNC/S, ODUk SNC/Ns
0 1 1	ODUk TCM3	ODUk SNC/S, ODUk SNC/Ns
1 0 0	ODUk TCM4	ODUk SNC/S, ODUk SNC/Ns
1 0 1	ODUk TCM5	ODUk SNC/S, ODUk SNC/Ns
1 1 0	ODUk TCM6	ODUk SNC/S, ODUk SNC/Ns, ODU SRP-1
1 1 1	ODUk server layer trail (Note 2)	ODUk SNC/I
NOTE 1 – An APS channel may be used by more than one protection scheme and/or protection scheme instance. In case of nested protection schemes, care should be taken when an ODUk protection is to be set up in order not to interfere with the APS channel usage of another ODUk protection on the same connection monitoring level, e.g., protection can only be activated if that APS channel of the level is not already being used. NOTE 2 – Examples of ODUk server layer trails are an OTUk or a server ODUk (e.g., an ODU3 transporting an ODU1).		

For ODUCn, the APS/PCC signal is used to support coordination of the end points in linear (ODUk CL-SNCG/I) and ring (ODUk SRP) protection applications.

For linear protection schemes, the bit assignments for these bytes and the bit-oriented protocol are given in [ITU-T G.873.1]. Bit assignment and byte-oriented protocol for ring protection schemes are given in [ITU-T G.873.2].

15.8.2.5 Blank clause

This clause is intentionally left blank.

15.8.2.6 ODU experimental overhead (EXP)

Four bytes are allocated in the ODU overhead for experimental use. These bytes are located in row 2, columns 4 and 14 and row 3, columns 13 and 14 of the ODU overhead.

The ODUk contains one instance of ODU EXP overhead. The ODUCn contains n instances of the ODU EXP overhead, numbered 1 to n (EXP #1 to EXP #n).

The use of these bytes is not subject to standardization and outside the scope of this Recommendation.

An experimental overhead is provided in the ODU OH to allow a vendor and/or a network operator within their own (sub)network to support an application, which requires an additional ODU overhead.

There is no requirement to forward the EXP overhead beyond the (sub)network; i.e., the operational span of the EXP overhead is limited to the (sub)network with the vendor's equipment, or the network of the operator.

15.8.2.7 ODU reserved overhead (RES)

For the case of an ODUk, eight bytes and one bit are reserved in the ODU overhead for future international standardization. These bytes are located in row 2, columns 1 to 2 and row 4, columns 9 to 14 of the ODU overhead. The bit is located in row 2, column 3, bit 8 of the ODU overhead. These bytes are set to all-0s.

For the case of an ODUCn, eigth bytes and one bit in the ODU OH #1 and thirteen bytes in the ODU OH #2 to #n are reserved for future international standardization. These bytes are located in row 2, columns 1 to 2 and row 4, columns 9 to 14 of the ODU OH #1 and in row 2, columns 1 to 3 and row 4, columns 5 to 14 of the ODU OH #2 to #n. The bit is located in row 2, column 3, bit 8 of the ODU OH #1. These bytes and bit are set to all-0s.

15.9 OPU OH description

15.9.1 OPU OH location

The OPU overhead consists of: payload structure identifier (PSI) including the payload type (PT), the overhead associated with the mapping of client signals into the OPU payload and the overhead associated with the hitless adjustment of ODUflex client signals. The OPU PSI and PT overhead locations are shown in Figure 15-26.

The OPUk contains one instance of OPU overhead. The OPUCn contains n instances of OPU overhead, numbered 1 to n (OPU OH #1 to #n).

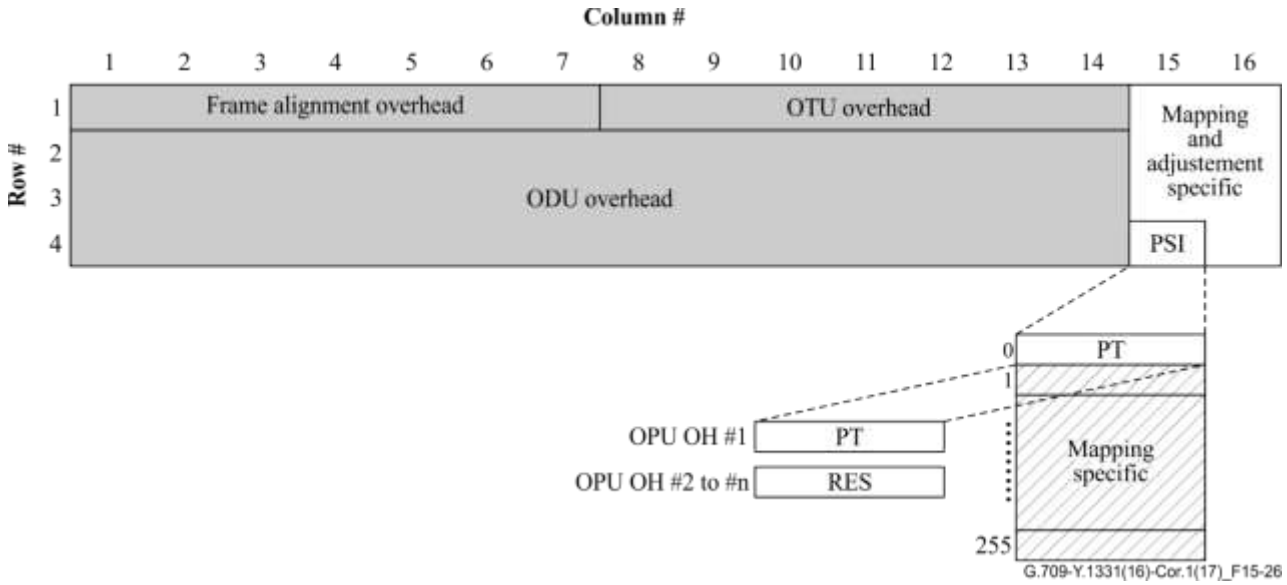


Figure 15-26 – OPU overhead

15.9.2 OPU OH definition

15.9.2.1 OPU payload structure identifier (PSI)

One byte is allocated in the OPU overhead to transport a 256-byte payload structure identifier (PSI) signal. The byte is located in row 4, column 15 of the OPU overhead.

The OPUk contains one instance of OPU PSI overhead. The OPUCn contains n instances of the OPU PSI overhead, numbered 1 to n (PSI #1 to PSI #n).

The 256-byte PSI signal is aligned with the ODU multiframe (i.e., PSI[0] is present at ODU multiframe position 0000 0000, PSI[1] at position 0000 0001, PSI[2] at position 0000 0010, etc.).

PSI[0] contains a one-byte payload type or is reserved for future international standardization. PSI[1] to PSI[255] are mapping specific, except for PT 0x01 (experimental mapping) and PTs 80-0x8F (for proprietary use).

15.9.2.1.1 OPU payload type (PT)

A one-byte payload type signal is defined in the PSI[0] byte of the payload structure identifier to indicate the composition of the OPU signal. The code points are defined in Table 15-9.

The OPUk and OPUCn contain one instance of OPU PT overhead.

Table 15-9 – Payload type code points

MSB 1 2 3 4	LSB 5 6 7 8	Hex code (Note 1)	Interpretation
0 0 0 0	0 0 0 1	01	Experimental mapping (Note 3)
0 0 0 0	0 0 1 0	02	Asynchronous CBR mapping, see clause 17.2
0 0 0 0	0 0 1 1	03	Bit-synchronous CBR mapping, see clause 17.2
0 0 0 0	0 1 0 0	04	Not available (Notes 2,7)
0 0 0 0	0 1 0 1	05	GFP mapping, see clause 17.4
0 0 0 0	0 1 1 0	06	Not available (Note 2)
0 0 0 0	0 1 1 1	07	PCS codeword transparent Ethernet mapping: 1000BASE-X into OPU0, see clauses 17.7.1 and 17.7.1.1 40GBASE-R into OPU3, see clauses 17.7.4 and 17.7.4.1 100GBASE-R into OPU4, see clauses 17.7.5 and 17.7.5.1
0 0 0 0	1 0 0 0	08	FC-1200 into OPU2e mapping, see clause 17.8.2
0 0 0 0	1 0 0 1	09	GFP mapping into extended OPU2 payload, see clause 17.4.1 (Note 5)
0 0 0 0	1 0 1 0	0A	STM-1 mapping into OPU0, see clause 17.7.1
0 0 0 0	1 0 1 1	0B	STM-4 mapping into OPU0, see clause 17.7.1
0 0 0 0	1 1 0 0	0C	FC-100 mapping into OPU0, see clause 17.7.1
0 0 0 0	1 1 0 1	0D	FC-200 mapping into OPU1, see clause 17.7.2
0 0 0 0	1 1 1 0	0E	FC-400 mapping into OPUflex, see clause 17.9
0 0 0 0	1 1 1 1	0F	FC-800 mapping into OPUflex, see clause 17.9
0 0 0 1	0 0 0 0	10	Bit stream with octet timing mapping, see clause 17.6.1
0 0 0 1	0 0 0 1	11	Bit stream without octet timing mapping, see clause 17.6.2
0 0 0 1	0 0 1 0	12	IB SDR mapping into OPUflex, see clause 17.9
0 0 0 1	0 0 1 1	13	IB DDR mapping into OPUflex, see clause 17.9
0 0 0 1	0 1 0 0	14	IB QDR mapping into OPUflex, see clause 17.9
0 0 0 1	0 1 0 1	15	SDI mapping into OPU0, see clause 17.7.1
0 0 0 1	0 1 1 0	16	(1.485/1.001) Gbit/s SDI mapping into OPU1, see clause 17.7.2
0 0 0 1	0 1 1 1	17	1.485 Gbit/s SDI mapping into OPU1, see clause 17.7.2
0 0 0 1	1 0 0 0	18	(2.970/1.001) Gbit/s SDI mapping into OPUflex, see clause 17.9
0 0 0 1	1 0 0 1	19	2.970 Gbit/s SDI mapping into OPUflex, see clause 17.9
0 0 0 1	1 0 1 0	1A	SBCON/ESCON mapping into OPU0, see clause 17.7.1
0 0 0 1	1 0 1 1	1B	DVB_ASI mapping into OPU0, see clause 17.7.1
0 0 0 1	1 1 0 0	1C	FC-1600 mapping into OPUflex, see clause 17.9
0 0 0 1	1 1 0 1	1D	Packet client mapping into OPUflex(IMP) and ODUflex(IMP,s), see clauses 17.10 and 17.11
0 0 0 1	1 1 1 0	1E	FlexE aware (partial rate) mapping into OPUflex, see clause 17.12

Table 15-9 – Payload type code points

MSB 1 2 3 4	LSB 5 6 7 8	Hex code (Note 1)	Interpretation
0 0 0 1	1 1 1 1	1F	FC-3200 mapping into OPUflex, see clause 17.9
0 0 1 0	0 0 0 0	20	ODU multiplex structure supporting ODTUjk only, see clause 19 (AMP only)
0 0 1 0	0 0 0 1	21	ODU multiplex structure supporting ODTUk.ts or ODTUk.ts and ODTUjk, see clause 19 (GMP capable) (Note 6)
0 0 1 0	0 0 1 0	22	ODU multiplex structure supporting ODTUCn.ts, see clause 20 (GMP capable)
0 0 1 0	0 0 1 1	23	ODU multiplex structure supporting fgODTU.fgts, see Annex N (fgGMP capable)
0 0 1 1	0 0 0 0	30	25GBASE-R mapping into OPUflex, see clause 17.13
0 0 1 1	0 0 0 1	31	200GBASE-R mapping into OPUflex, see clause 17.13
0 0 1 1	0 0 10	32	400GBASE-R mapping into OPUflex, see clause 17.13
0 0 1 1	0 0 1 1	33	50GBASE-R mapping into OPUflex, see clause 17.13
0 0 1 1	0 1 0 0	34	800GBASE-R mapping into OPUflex, see clause 17.14
0 1 0 1	0 1 0 1	55	Not available (Note 2)
0 1 1 0	0 1 1 0	66	Not available (Note 2)
1 0 0 0	x x x x	80-8F	Reserved codes for proprietary use (Note 4)
1 1 1 1	1 1 0 1	FD	NULL test signal mapping, see clause 17.5.1
1 1 1 1	1 1 1 0	FE	PRBS test signal mapping, see clause 17.5.2
1 1 1 1	1 1 1 1	FF	Not available (Note 2)

NOTE 1 – There are 197 spare codes left for future international standardization. Refer to Annex A of [ITU-T G.806] for the procedure to obtain one of these codes for a new payload type.

NOTE 2 – These values are excluded from the set of available code points. These bit patterns are present in ODUk maintenance signals or were used to represent client types that are no longer supported.

NOTE 3 – Value "01" is only to be used for experimental activities in cases where a mapping code is not defined in this table. Refer to Annex A of [ITU-T G.806] for more information on the use of this code.

NOTE 4 – These 16 code values will not be subject to further standardization. Refer to Annex A of [ITU-T G.806] for more information on the use of these codes.

NOTE 5 – Supplement 43 (2008) to the ITU-T G-series of Recommendations indicated that this mapping recommended using payload type 87.

NOTE 6 – Equipment supporting ODTUk.ts for OPU2 or OPU3 must be backward compatible with equipment which supports only the ODTUjk. ODTUk.ts capable equipment transmitting PT=21 which receives PT=20 from the far end shall revert to PT=20 and operate in ODTUjk only mode. Refer to [ITU-T G.798] for the specification.

NOTE 7 – This bit pattern may appear temporarily due to the presence of the FlexO SquelchText pattern. Refer to [ITU-T G.709.1].

15.9.2.2 OPU mapping specific overhead

Seven bytes are reserved in the OPUk overhead for the mapping and hitless adjustment specific overhead. These bytes are located in rows 1 to 3, columns 15 and 16 and column 16 row 4. In addition, 255 bytes in the PSI #1 and 256 byte in PSI #2 to #n are reserved for specific purposes.

The use of these bytes depends on the specific client signal mapping (defined in clauses 17, 19 and 20) and use of hitless adjustment of ODUflex(GFP) (see [ITU-T G.7044]).

16 Maintenance signals

An alarm indication signal (AIS) is a signal sent downstream as an indication that an upstream defect has been detected. An AIS signal is generated in an adaptation sink function. An AIS signal is detected in a trail termination sink function to suppress defects or failures that would otherwise be detected as a consequence of the interruption of the transport of the original signal at an upstream point.

A forward defect indication (FDI) is a signal sent downstream as an indication that an upstream defect has been detected. An FDI signal is generated in an adaptation sink function. An FDI signal is detected in a trail termination sink function to suppress defects or failures that would otherwise be detected as a consequence of the interruption of the transport of the original signal at an upstream point.

NOTE – AIS and FDI are similar signals. AIS is used as the term when the signal is in the digital domain. FDI is used as the term when the signal is in the optical domain; FDI is transported as a non-associated overhead.

An open connection indication (OCI) is a signal sent downstream as an indication that upstream the signal is not connected to a trail termination source. An OCI signal is generated in a connection function and output by this connection function on each of its output connection points, which are not connected to one of its input connection points. An OCI signal is detected in a trail termination sink function.

Locked (LCK) is a signal sent downstream as an indication that, upstream to the receiver, the connection is administratively "locked", and the normal signal has not passed through.

A payload missing indication (PMI) is a signal sent downstream as an indication that upstream at the source point of the signal none of the frequency slots have an optical tributary signal. This indicates that the transport of the optical tributary signals is interrupted or no optical tributary signals are present.

A PMI signal is generated in the trail termination source function and it is detected in the trail termination sink function which suppresses the LOS defect that arises under this condition.

16.1 OTS maintenance signals

16.1.1 OTS payload missing indication (OTS-PMI)

OTS-PMI is generated as an indication that the OTS payload does not contain an optical signal.

16.2 OMS maintenance signals

Three OMS maintenance signals are defined: OMS-FDI-P, OMS-FDI-O and OMS-PMI.

16.2.1 OMS forward defect indication – Payload (OMS-FDI-P)

OMS-FDI-P is generated as an indication of an OMS server layer defect in the OTS network layer.

16.2.2 OMS forward defect indication – Overhead (OMS-FDI-O)

OMS-FDI-O is generated as an indication when the transport of OMS OH via the OSC is interrupted due to a signal fail condition in the OSC.

16.2.3 OMS payload missing indication (OMS-PMI)

OMS-PMI is generated as an indication when none of the frequency slots contain an optical tributary signal.

16.3 OCh and OTiSA maintenance signals

Three OCh and OTiSA maintenance signals are defined: OCh-FDI-P, OCh-FDI-O, OCh-OCI and OTSiA-FDI-P, OTSiA-FDI-O, OTSiA-OCI.

16.3.1 OCh and OTiSA forward defect indication – Payload (OCh-FDI-P, OTSiA-FDI-P)

OCh-FDI-P and OTSiA-FDI-P are generated as an indication for an OCh and OTSiA server layer defect in the OMS network layer.

When the OTUk or OTUCn is terminated, the OCh-FDI-P and OTSiA_FDI-P is continued as an ODUk-AIS signal.

When the OTUCn is not terminated, the OCh-FDI-P and OTSiA_FDI-P are continued as an OTUCn-AIS signal.

OTSiA-FDI-O is generated as an indication when the transport of OTSiA OH via the OSC is interrupted due to a signal fail condition in the OSC.

16.3.2 OCh and OTiSA forward defect indication – Overhead (OCh-FDI-O, OTSiA-FDI-O)

OCh-FDI-O is generated as an indication when the transport of OCh OH via the OSC or OCC is interrupted due to a signal fail condition in the OSC or OCC.

16.3.3 OCh and OTiSA open connection indication (OCh-OCI, OTSiA-OCI)

The OCh-OCI/OTSiA-OCI signal indicates to downstream transport processing functions that the OCh/OTSiA connection is not bound to, or not connected (via a matrix connection) to a termination source function. The indication is used in order to distinguish downstream between a missing OCh/OTSiA signal due to a defect or due to the open connection (resulting from a management command).

NOTE – OCI is detected at the next downstream OCh or OTSiA trail terminating equipment. If the connection was opened intentionally, the related alarm report from this trail termination should be disabled by using the alarm reporting control mode (refer to [ITU-T M.3100]).

16.4 OTU maintenance signals

16.4.1 OTUk (k=1,2,3) alarm indication signal (OTUk-AIS)

The OTUk-AIS (k=1,2,3) (see Figure 16-1) is a generic-AIS signal (see clause 16.6.1). Since the OTUk capacity (130 560 bits) is not an integer multiple of the PN-11 sequence length (2047 bits), the PN-11 sequence may cross an OTUk frame boundary.

NOTE – OTUk-AIS is defined to support a future server layer application. OTN equipment should be capable of detecting the presence of such a signal within OTUk (k=1,2,3) on OTN point-to-point interface signals; it is not required to generate such a signal. OTU0-AIS is not defined.

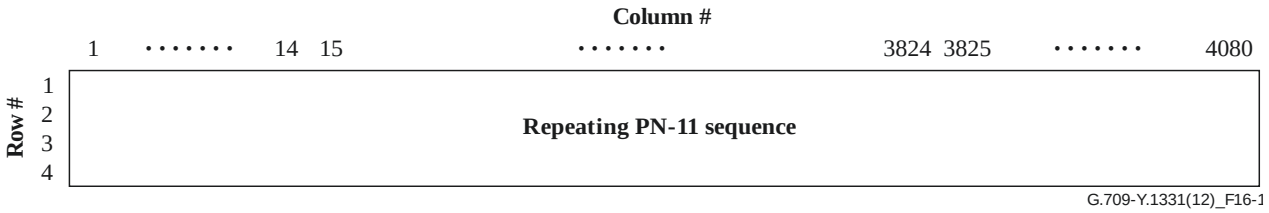


Figure 16-1 – OTUk-AIS

16.4.2 OTUCn alarm indication signal (OTUCn-AIS)

The OTUCn-AIS is specified as all "1"s in the entire OTUCn signal, excluding the frame alignment overhead (FA OH) (see Figure 16-2).

The presence of the OTUCn-AIS is detected by monitoring the OTUCn SM STAT bits in the SM overhead fields.

NOTE – OTUCn-AIS is defined to support a future 3R regenerator application in which the OTUCn is not terminated and an OTUCn subrating application. OTN equipment should be capable of detecting the presence of such a signal within OTUCn signals on OTN optical networking interfaces. ~~OTU0-AIS is not defined.~~

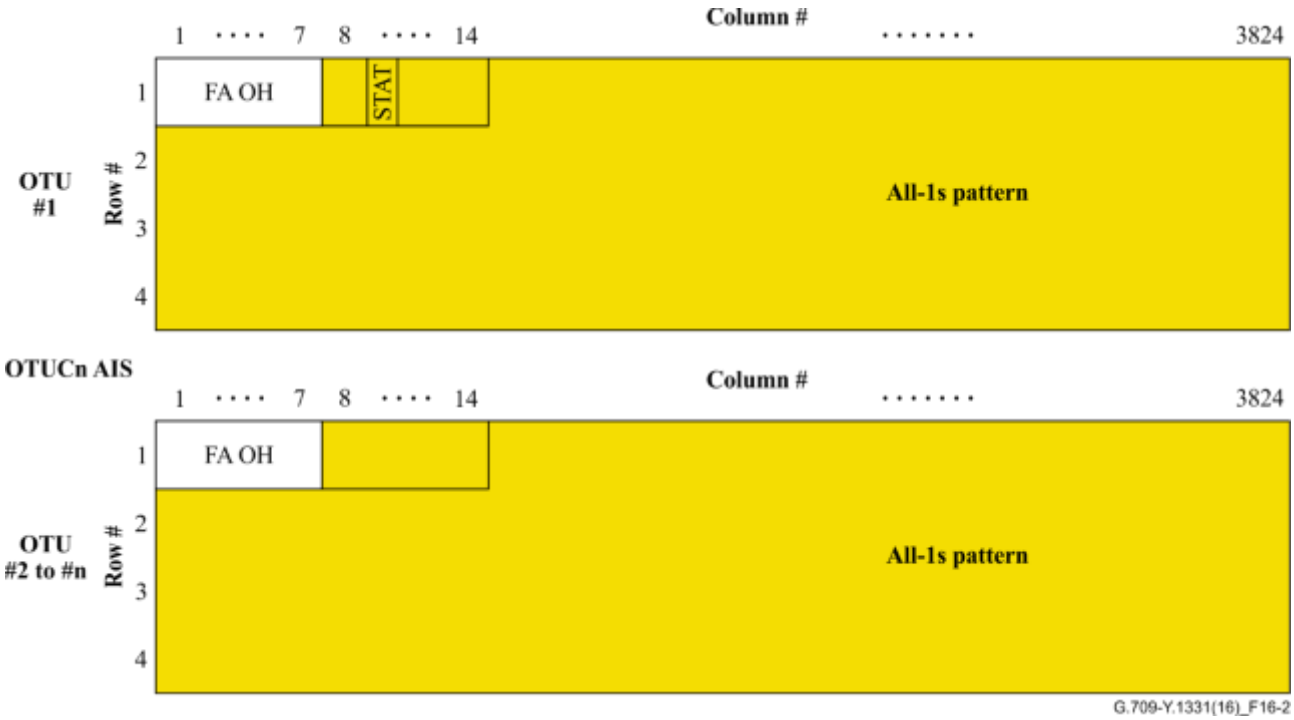


Figure 16-2 – OTUCn-AIS

16.5 ODU maintenance signals

Three ODU maintenance signals are defined: ODU-AIS, ODU-OCI and ODU-LCK.

16.5.1 ODU alarm indication signal (ODU-AIS)

ODUk-AIS is specified as all "1"s in the entire ODU signal, excluding the frame alignment overhead (FA OH), OTU overhead (OTU OH) (see Figure 16-3).

The ODUk contains one instance of ODU AIS. The ODUCn contains n instances of ODU AIS, numbered 1 to n (ODU AIS #1 to #n).

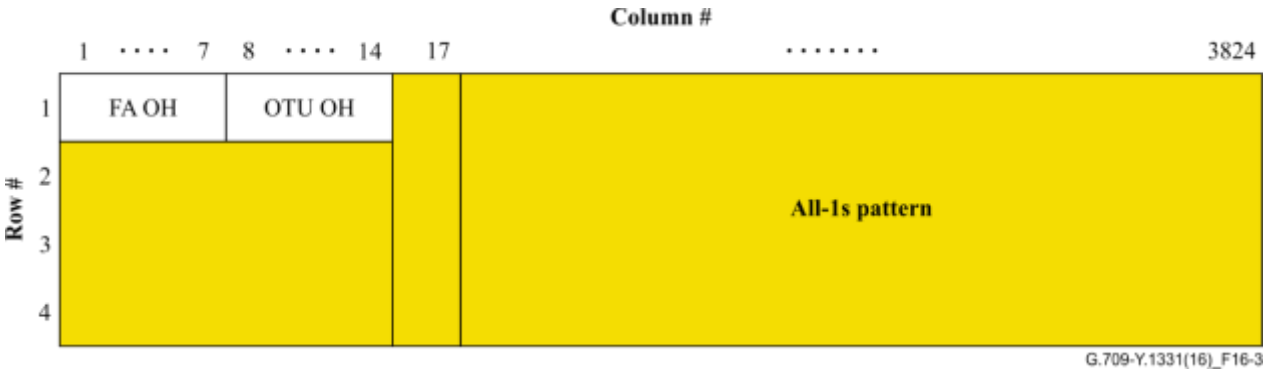


Figure 16-3 – ODU-AIS

In addition, the ODU-AIS signal may be extended with one or more levels of ODU tandem connection, GCC1, GCC2, EXP and/or APS/PCC overhead before it is presented at the OTN interface. This is dependent on the functionality between the ODU-AIS insertion point and the OTN interface.

The presence of the ODU-AIS is detected by monitoring the ODU STAT bits in the PM and TCMi overhead fields.

16.5.2 ODUk open connection indication (ODUk-OCI)

ODUk-OCI is specified as a repeating "0110 0110" pattern in the entire ODUk signal, excluding the frame alignment overhead (FA OH) and OTUk overhead (OTUk OH) (see Figure 16-4).

NOTE – ODUCn OCI is not defined.

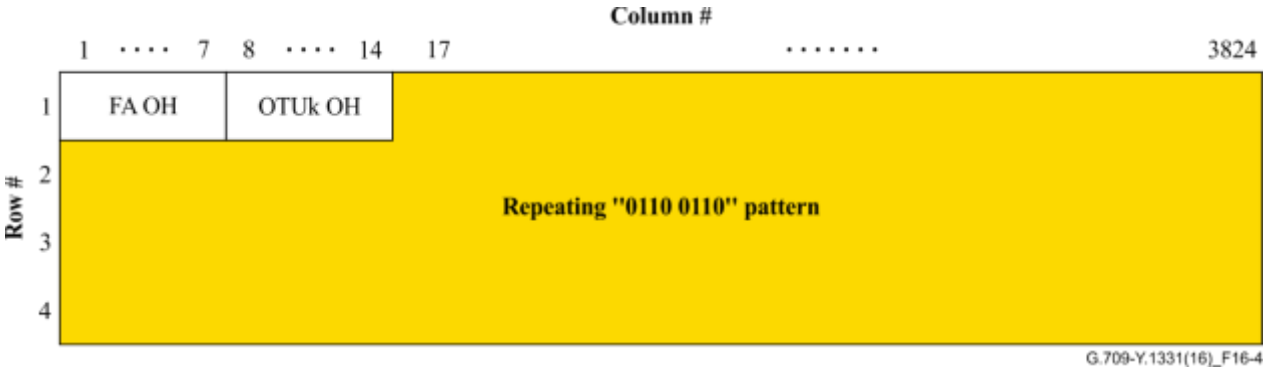


Figure 16-4 – ODUk-OCI

NOTE – The repeating "0110 0110" pattern is the default pattern; other patterns are also allowed as long as the STAT bits in the PM and TCMi overhead fields are set to "110".

In addition, the ODUk-OCI signal may be extended with one or more levels of ODUk tandem connection, GCC1, GCC2, EXP and/or APS/PCC overhead before it is presented at the OTM interface. This is dependent on the functionality between the ODUk-OCI insertion point and the OTM interface.

The presence of ODUk-OCI is detected by monitoring the ODUk STAT bits in the PM and TCMi overhead fields.

16.5.3 ODU locked (ODU-LCK)

ODU-LCK is specified as a repeating "0101 0101" pattern in the entire ODU signal, excluding the frame alignment overhead (FA OH) and OTU overhead (OTU OH) (see Figure 16-5).

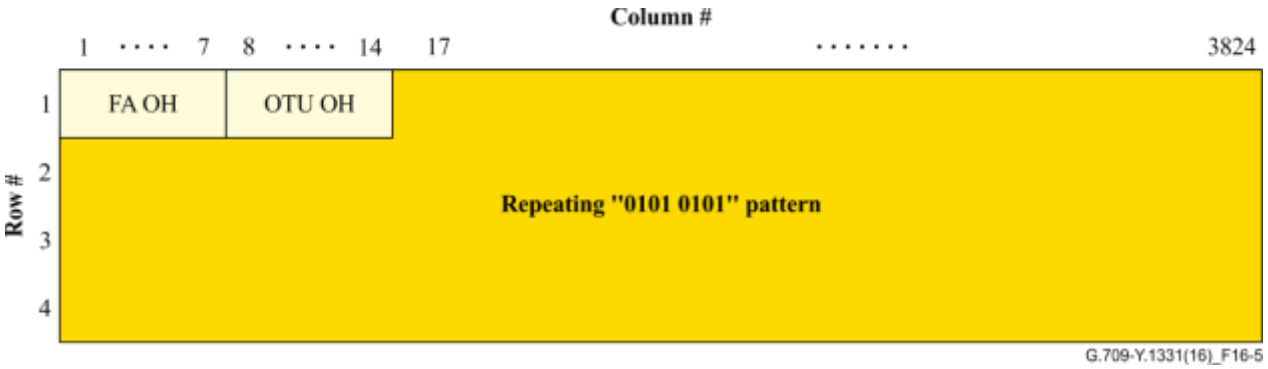


Figure 16-5 – ODU-LCK

NOTE – The repeating "0101 0101" pattern is the default pattern; other patterns are also allowed as long as the STAT bits in the PM and TCMi overhead fields are set to "101".

In addition, the ODU-LCK signal may be extended with one or more additional levels of ODU tandem connection, GCC1, GCC2, EXP and/or the APS/PCC overhead before it is presented at the OTN interface. This is dependent on the functionality between the ODU-LCK insertion point and the OTN interface.

The presence of ODU-LCK is detected by monitoring the ODU STAT bits in the PM and TCMi overhead fields.

16.6 Client maintenance signal

16.6.1 Generic AIS for constant bit rate signals

The generic-AIS signal is a signal with a 2 047-bit polynomial number 11 (PN-11) repeating sequence.

The PN-11 sequence is defined by the generating polynomial $1 + x^9 + x^{11}$ as specified in clause 5.2 of [ITU-T O.150]. (See Figure 16-6.)

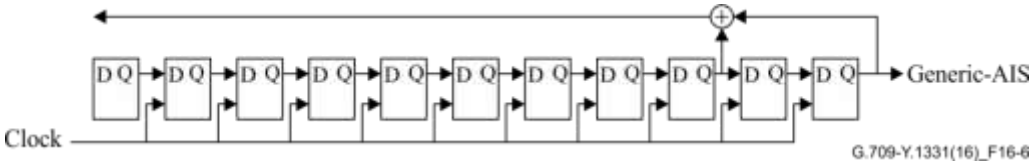


Figure 16-6 – Generic-AIS generating circuit

17 Mapping of client signals

This clause specifies the mapping of:

– STM-16, STM-64, STM-256 constant bit rate client signals into OPUk using client/server specific asynchronous or bit-synchronous mapping procedures (AMP, BMP);

– 10GBASE-R constant bit rate client signal into OPU2e using client/server specific bit-synchronous mapping procedure (BMP);

– FC-1200 constant bit rate client signal after timing transparent transcoding (TTT) providing a 50/51 rate compression into OPU2e using client/server specific byte-synchronous mapping procedure;

– constant bit rate client signals with bit rates up to 1.238 Gbit/s into OPU0 and up to 2.488 Gbit/s into OPU1 using a client agnostic generic mapping procedure (GMP) possibly preceded by a timing transparent transcoding (TTT) of the client signal to reduce the bit rate of the signal to fit the OPUk payload bandwidth;

– constant bit rate client signals into OPU1, OPU2, OPU3 or OPU4 respectively using a client agnostic generic mapping procedure (GMP) possibly preceded by a timing transparent transcoding (TTT) of the client signal to reduce the bit rate of the signal to fit the OPUk payload bandwidth;

– other constant bit rate client signals into OPUflex using a client agnostic bit-synchronous mapping procedure (BMP);

– packet streams (e.g., Ethernet, MPLS, IP) which are encapsulated with the generic framing procedure (GFP-F);

– test signals;

– continuous mode GPON constant bit rate client signal into OPU1 using asynchronous mapping procedure (AMP);

– continuous mode XGPON constant bit rate client signal into OPU2 using asynchronous mapping procedure (AMP);

– FlexE-aware client signal into OPUflex using bit synchronous mapping procedure (BMP);

– FlexE Client client signal into OPUflex using idle mapping procedure;

– packet streams (e.g., Ethernet, MPLS, IP) which are encapsulated with the Idle mapping procedure (IMP) into OPU.

17.1 OPU client signal fail (CSF)

For support of local management systems, a single-bit OPU client signal fail (CSF) indicator is defined to convey the signal fail status of the CBR and Ethernet private line client signal mapped into an OPU at the ingress of the OTN to the egress of the OTN.

OPU CSF is located in bit 1 of the PSI[2] byte of the payload structure identifier. Bits 2 to 8 of the PSI[2] byte are reserved for future international standardization. These bits are set to all-0s.

OPU CSF is set to "1" to indicate a client signal fail indication, otherwise it is set to "0".

NOTE – Equipment designed prior to Edition 3.0 of the Recommendation will generate a "0" in the OPUk CSF and will ignore any value in OPUk CSF.

17.2 Mapping of CBR2G5, CBR10G, CBR10G3 and CBR40G signals into OPUk

The mapping of a CBR2G5, CBR10G or CBR40G signal (with up to ±20 ppm bit-rate tolerance) into an OPUk (k = 1,2,3) may be performed according to the bit-synchronous mapping procedure based on one generic OPUk frame structure (see Figure 17-1). The mapping of a CBR2G5, CBR10G or CBR40G signal (with up to ±45 ppm bit-rate tolerance) into an OPUk (k = 1,2,3) may be performed according to the asynchronous mapping procedure. The mapping of a CBR10G3 signal (with up to ±100 ppm bit-rate tolerance) into an OPUk (k = 2e) is performed using the bit-synchronous mapping procedure.

NOTE 1 – Examples of CBR2G5, CBR10G and CBR40G signals are STM-16 and CMGPON_D/U2 (refer to [ITU-T G.984.6]), STM-64 and CMXGPON_D/U2 [ITU-T G.987.4] and STM-256. An example of a CBR10G3 signal is 10GBASE-R.

NOTE 2 – The maximum bit-rate tolerance between an OPUk and the client signal clock, which can be accommodated by the asynchronous mapping scheme, is ±65 ppm. With a bit-rate tolerance of ±20 ppm for the OPUk clock, the client signal's bit-rate tolerance can be ±45 ppm.

NOTE 3 – For OPUk (k=1,2,3) the clock tolerance is ±20 ppm. For OPU2e the clock tolerance is ±100 ppm and asynchronous mapping cannot be supported with this justification overhead.

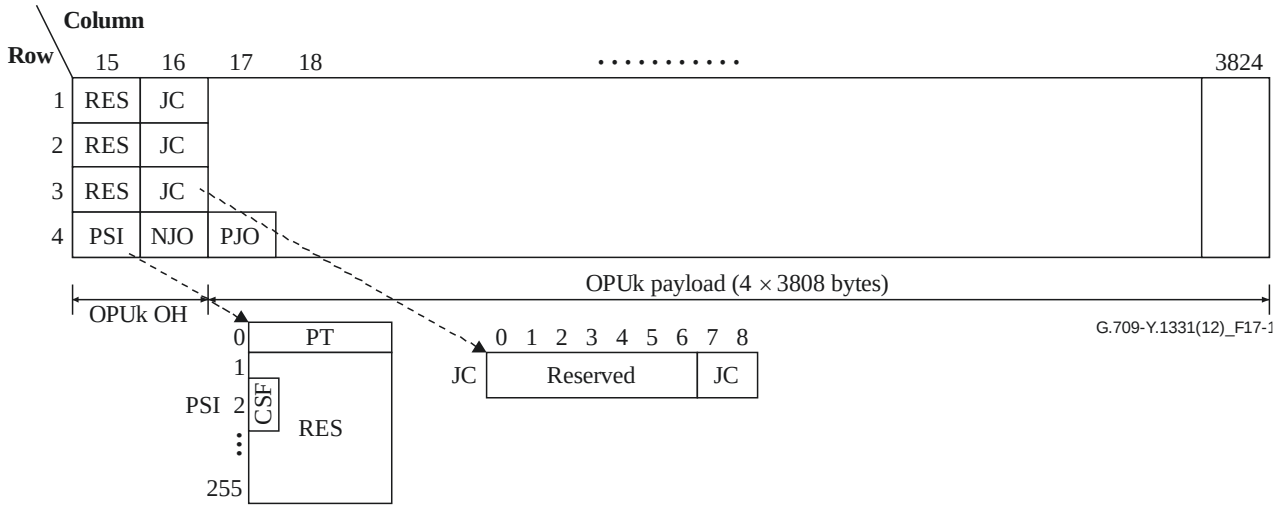


Figure 17-1 – OPUk frame structure for the mapping of a CBR2G5, CBR10G or CBR40G signal

The OPUk overhead for these mappings consists of a payload structure identifier (PSI) including the payload type (PT), a client signal fail (CSF) indicator and 254 bytes plus 7 bits reserved for future international standardization (RES), three justification control (JC) bytes, one negative justification opportunity (NJO) byte, and three bytes reserved for future international standardization (RES).

The JC bytes consist of two bits for justification control and six bits reserved for future international standardization.

The OPUk payload for these mappings consists of 4 × 3808 bytes, including one positive justification opportunity (PJO) byte.

The justification control (JC) signal, which is located in rows 1, 2 and 3 of column 16, bits 7 and 8, is used to control the two justification opportunity bytes NJO and PJO that follow in row 4.

The asynchronous and bit-synchronous mapping processes generate the JC, NJO and PJO according to Tables 17-1 and 17-2, respectively. The de-mapping process interprets JC, NJO and PJO according to Table 17-3. Majority vote (two out of three) shall be used to make the justification decision in the de-mapping process to protect against an error in one of the three JC signals.

Table 17-1 – JC, NJO and PJO generation by an asynchronous mapping process

JC bits 7 8	**NJO**	**PJO**
0 0	justification byte	data byte
0 1	data byte	data byte
1 0	not generated	
1 1	justification byte	justification byte

Table 17-2 – JC, NJO and PJO generation by a bit-synchronous mapping process

JC bits 7 8	**NJO**	**PJO**
0 0	justification byte	data byte
0 1	not generated	
1 0		
1 1		

Table 17-3 – JC, NJO and PJO interpretation

JC bits 7 8	**NJO**	**PJO**
0 0	justification byte	data byte
0 1	data byte	data byte
1 0 (Note)	justification byte	data byte
1 1	justification byte	justification byte
NOTE – A mapper circuit does not generate this code. Due to bit errors a de-mapper circuit might receive this code.		

The value contained in NJO and PJO when they are used as justification bytes is all-0s. The receiver is required to ignore the value contained in these bytes whenever they are used as justification bytes.

During a signal fail condition of the incoming CBR2G5, CBR10G or CBR40G client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the generic-AIS signal as specified in clause 16.6.1, and is then mapped into the OPUk.

During a signal fail condition of the incoming 10GBASE-R type CBR10G3 client signal (e.g., in the case of a loss of input signal), this failed incoming 10GBASE-R signal is replaced by a stream of 66B blocks, with each block carrying two local fault sequence ordered sets (as specified in [IEEE 802.3]). This 66B block stream replacement signal is scrambled before mapping into the OPU2e. A self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$ shall be used, which is identical to the scrambler specified in clause 49.2.6 of [IEEE 802.3].

During the signal fail condition of the incoming ODUk/OPUk signal (e.g., in the case of an ODUk-AIS, ODUk-LCK, ODUk-OCI condition) the generic-AIS pattern as specified in clause 16.6.1 is generated as a replacement signal for the lost CBR2G5, CBR10G or CBR40G signal.

During the signal fail condition of the incoming ODU2e/OPU2e signal (e.g., in the case of an ODU2e-AIS, ODU2e-LCK, ODU2e-OCI condition) a stream of 66B blocks, with each block carrying two local fault sequence ordered sets (as specified in [IEEE 802.3]) is generated as a replacement signal for the lost 10GBASE-R signal. This 66B block stream replacement signal is scrambled using a self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$, which is identical to the scrambler specified in clause 49.2.6 of [IEEE 802.3].

NOTE 4 – Local fault sequence ordered set is /K28.4/D0.0/D0.0/D1.0/. The 66B block contains the following value SH=10 0x55 00 00 01 00 00 00 01.

NOTE 5 – Equipment developed prior to Edition 2.5 of this Recommendation may generate a different 10GBASE-R replacement signal (e.g., Generic-AIS) than the local fault sequence ordered set.

Asynchronous mapping

The OPUk signal for the asynchronous mapping is created from a locally generated clock (within the limits specified in Table 7-3), which is independent of the CBR2G5, CBR10G or CBR40G (i.e., $4^{(k-1)} \times 2\,488\,320$ kbit/s (k = 1,2,3)) client signal.

The CBR2G5, CBR10G, CBR40G (i.e., $4^{(k-1)} \times 2\,488\,320$ kbit/s (k = 1,2,3)) signal is mapped into the OPUk using a positive/negative/zero (pnz) justification scheme.

Bit-synchronous mapping

The OPUk clock for bit-synchronous mapping is derived from the CBR2G5, CBR10G, CBR40G or CBR10G3 client signal. During signal fail conditions of the incoming CBR2G5, CBR10G, CBR40G or CBR10G3 signal (e.g., in the case of a loss of input signal), the OPUk payload signal bit rate shall be within the limits specified in Table 7-3 and the limits defined for the ODCb clock in [ITU-T G.8251] and no OPUk frame phase discontinuity shall be introduced in this case and when resynchronizing on the incoming CBR2G5, CBR10G, CBR40G or CBR10G3 signal.

The CBR2G5, CBR10G, CBR40G or CBR10G3 signal is mapped into the OPUk without using the justification capability within the OPUk frame: NJO contains a justification byte, PJO contains a data byte, and the JC signal is fixed to 00.

17.2.1 Mapping a CBR2G5 signal (e.g., STM-16, CMGPON_D/CMGPON_U2) into OPU1

Groups of eight successive bits (not necessarily being a byte) of the CBR2G5 signal are mapped into a data (D) byte of the OPU1 (see Figure 17-2). Once per OPU1 frame, it is possible to perform either a positive or a negative justification action.

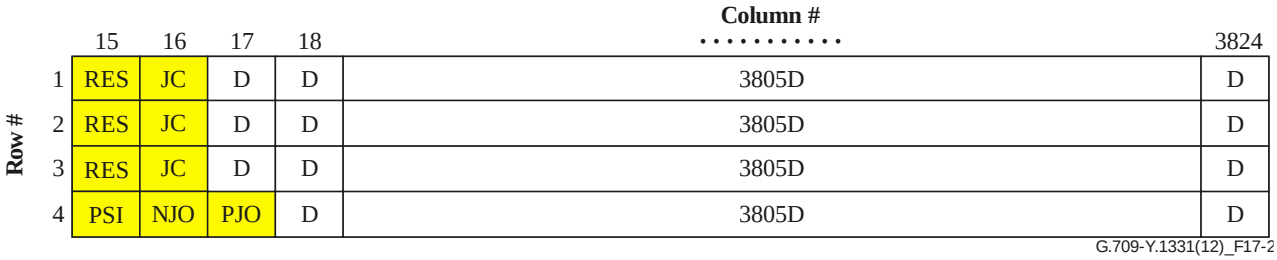


Figure 17-2 – Mapping of a CBR2G5 signal into OPU1

17.2.2 Mapping a CBR10G signal (e.g., STM-64, CMXGPON_D/CMXGPON_U2) into OPU2

Groups of eight successive bits (not necessarily being a byte) of the CBR10G signal are mapped into a data (D) byte of the OPU2 (see Figure 17-3). 64 fixed stuff (FS) bytes are added in columns 1905 to 1920. Once per OPU2 frame, it is possible to perform either a positive or a negative justification action.

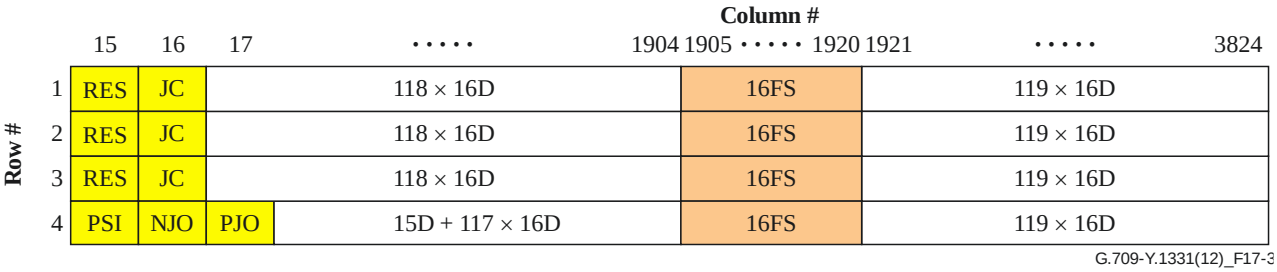


Figure 17-3 – Mapping of a CBR10G signal into OPU2

17.2.3 Mapping a CBR40G signal (e.g., STM-256) into OPU3

Groups of eight successive bits (not necessarily being a byte) of the CBR40G signal are mapped into a data (D) byte of the OPU3 (see Figure 17-4). 128 fixed stuff (FS) bytes are added in columns 1265 to 1280 and 2545 to 2560. Once per OPU3 frame, it is possible to perform either a positive or a negative justification action.

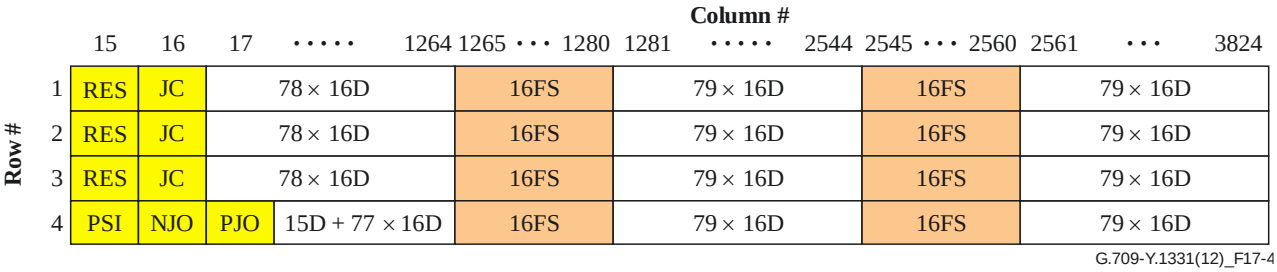


Figure 17-4 – Mapping of a CBR40G signal into OPU3

17.2.4 Mapping a CBR10G3 signal (e.g., 10GBASE-R) into OPU2e

Groups of eight successive bits (not necessarily being a byte) of the CBR10G3 signal are bit-synchronously mapped into a data (D) byte of the OPU2e (see Figure 17-5). 64 fixed stuff (FS) bytes are added in columns 1905 to 1920.

NOTE – The NJO byte will always carry a stuff byte, the PJO byte will always carry a data (D) byte and the JC bytes will always carry the all-0s pattern.

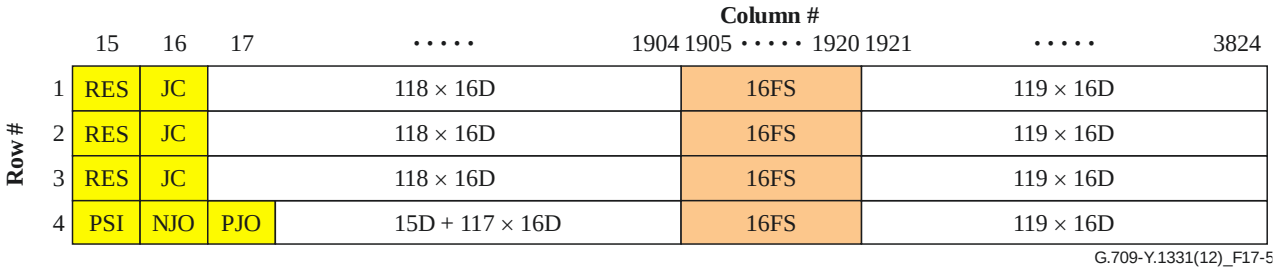


Figure 17-5 – Mapping of a CBR10G3 signal into OPU2e

17.3 Blank clause

This clause is intentionally left blank.

17.4 Mapping of GFP frames into OPUk (k=0,1,2,3,4,flex)

The mapping of generic framing procedure (GFP) frames is performed by aligning the byte structure of every GFP frame with the byte structure of the OPUk payload (see Figure 17-6). Since the GFP frames are of variable length (the mapping does not impose any restrictions on the maximum frame length), a frame may cross the OPUk (k=0,1,2,3,4,flex) frame boundary.

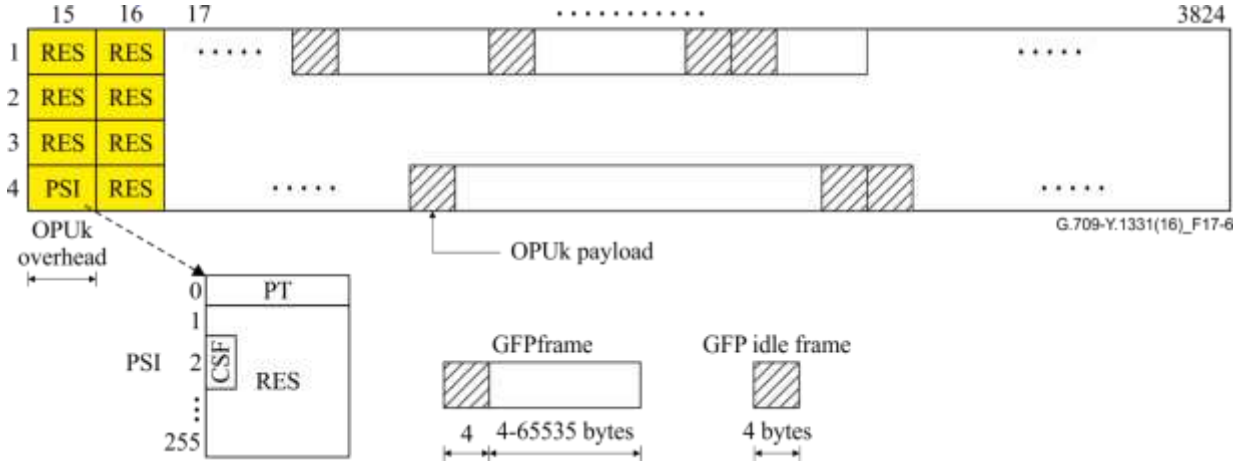


Figure 17-6 – OPUk frame structure and mapping of GFP frames into OPUk

GFP frames arrive as a continuous bit stream with a capacity that is identical to the OPUk payload area, due to the insertion of idle frames at the GFP encapsulation stage. The GFP frame stream is scrambled during encapsulation.

NOTE 1 – There is no rate adaptation or scrambling required at the mapping stage; this is performed by the GFP encapsulation process.

The OPUk overhead for the GFP mapping consists of a payload structure identifier (PSI) including the payload type (PT), a client signal fail (CSF) indicator and 254 bytes plus 7 bits reserved for future international standardization (RES), and seven bytes reserved for future international standardization (RES). The CSF indicator should be used only for Ethernet private line type 1 services; for other packet clients the CSF bit is fixed to 0.

The OPUk payload for the GFP mapping consists of 4 × 3808 bytes.

NOTE 2 – The OPUflex(GFP) bit rate may be any configured bit rate as specified in Tables 7-3 and 7-8.

17.4.1 Mapping of GFP frames into an extended OPU2 payload area

The mapping of generic framing procedure (GFP) frames in an extended OPU2 payload area is performed by aligning the byte structure of every GFP frame with the byte structure of the extended

OPU2 payload (see Figure 17-7). Since the GFP frames are of variable length (the mapping does not impose any restrictions on the maximum frame length), a frame may cross the OPU2 frame boundary.

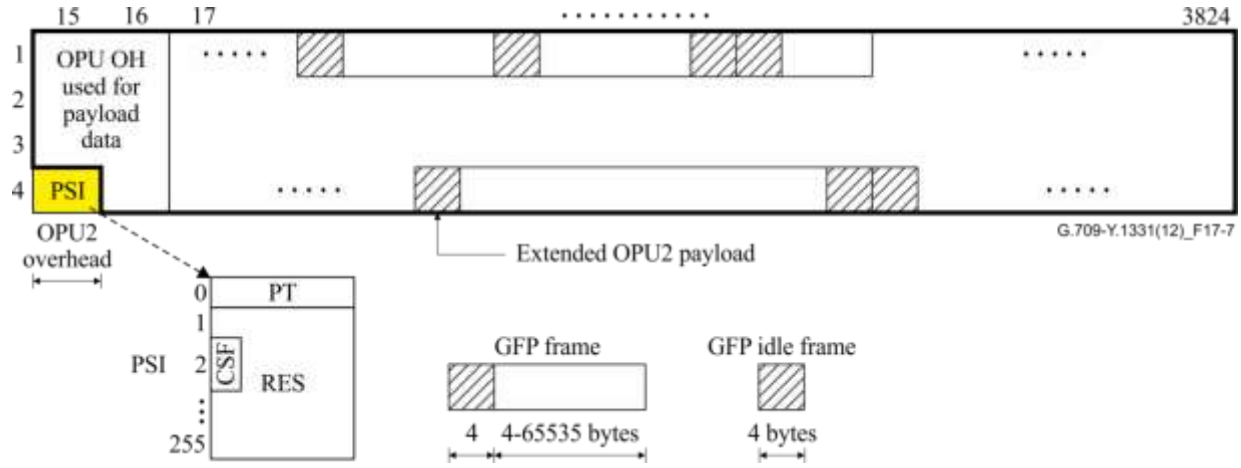


Figure 17-7 – OPU2 frame structure and mapping of GFP frames into an extended OPU2 payload area

GFP frames arrive as a continuous bit stream with a capacity that is identical to the OPU2 payload area, due to the insertion of GFP-idle frames at the GFP encapsulation stage. The GFP frame stream is scrambled during encapsulation.

NOTE – There is no rate adaptation or scrambling required at the mapping stage; this is performed by the GFP encapsulation process.

The OPU2 overhead for the GFP mapping consists of a payload structure identifier (PSI) including the payload type (PT), a client signal fail (CSF) indicator and 254 bytes plus 7 bits of reserved for future international standardization (RES).

The extended OPU2 payload for the GFP mapping consists of 4 × 3808 bytes from the OPU2 payload plus 7 bytes from the OPU2 overhead.

17.5 Mapping of test signal into OPU

17.5.1 Mapping of a NULL client into OPU

An OPU payload signal with an all-0s pattern (see Figure 17-8) is defined for test purposes. This is referred to as the NULL client.

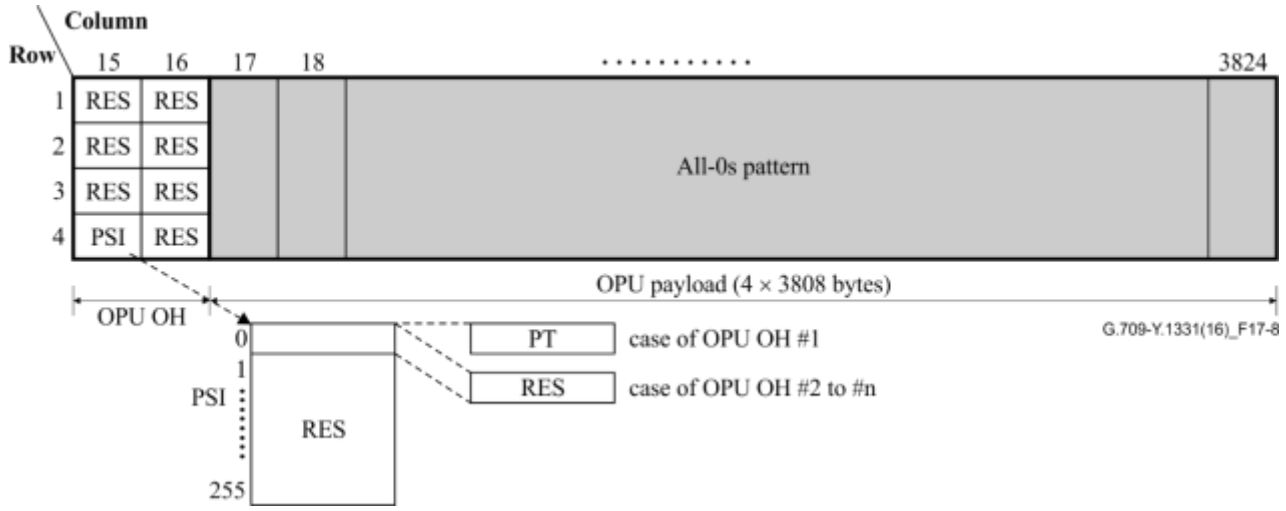


Figure 17-8 – OPU frame structure and mapping of a NULL client into OPU

The OPU overhead for the NULL mapping consists of a payload structure identifier (PSI) including the payload type (PT) and 255 bytes reserved for future international standardization (RES), and seven bytes reserved for future international standardization (RES).

The OPU payload for the NULL mapping consists of 4 × 3808 bytes.

The OPUk contains one instance of the NULL client. The OPUCn contains n instances of the NULL client, numbered 1 to n.

17.5.2 Mapping of PRBS test signal into OPU

For end-to-end and segment turn-up test purposes, a 2 147 483 647-bit pseudo-random test sequence ($2^{31} – 1$) as specified in clause 5.8 of [ITU-T O.150] can be mapped into the OPU payload. Groups of eight successive bits of the 2 147 483 647-bit pseudo-random test sequence signal are mapped into 8 data bits (8D) (i.e., one byte) of the OPU payload (see Figure 17-9).

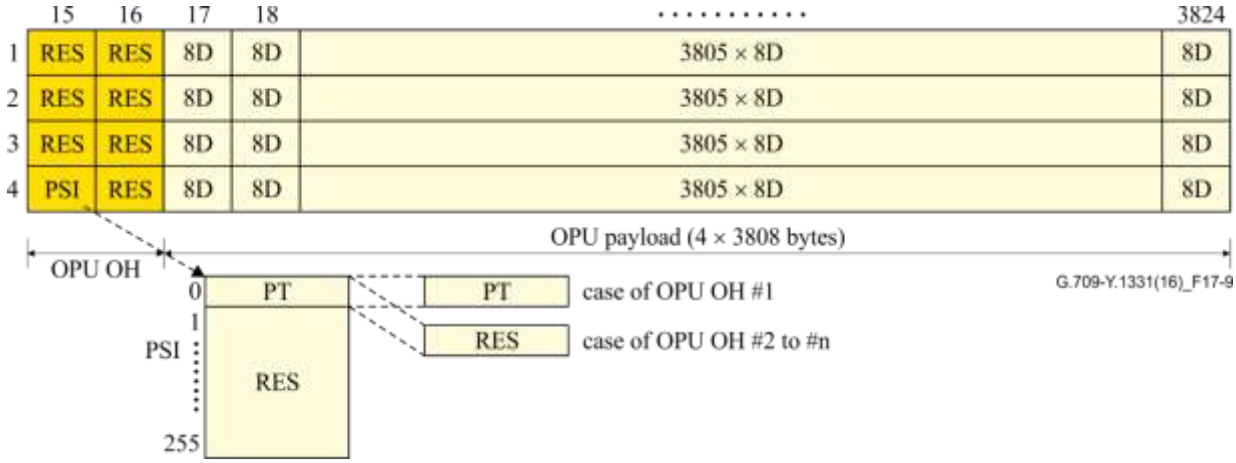


Figure 17-9 – OPU frame structure and mapping of 2 147 483 647-bit pseudo-random test sequence into OPU

The OPU overhead for the PRBS mapping consists of a payload structure identifier (PSI) including the payload type (PT) and 255 bytes reserved for future international standardization (RES), and seven bytes reserved for future international standardization (RES).

The OPU payload for the PRBS mapping consists of 4 × 3808 bytes.

The OPUk contains one instance of the 2 147 483 647-bit pseudo-random test sequence. The OPUCn contains n instances of the 2 147 483 647-bit pseudo-random test sequence, numbered 1 to n.

NOTE – This PRBS test pattern is not intended to be deployed for stress testing of the physical interface.

17.6 Mapping of a non-specific client bit stream into OPUk

In addition to the mappings of specific client signals as specified in the other subclauses of this clause, a non-specific client mapping into OPUk is specified. Any (set of) client signal(s), which after encapsulation into a continuous bit stream with a bit rate of the OPUk payload, can be mapped into the OPUk payload (see Figure 17-10). The bit stream must be synchronous with the OPUk signal. Any justification must be included in the continuous bit stream creation process. The continuous bit stream must be scrambled before mapping into the OPUk payload.

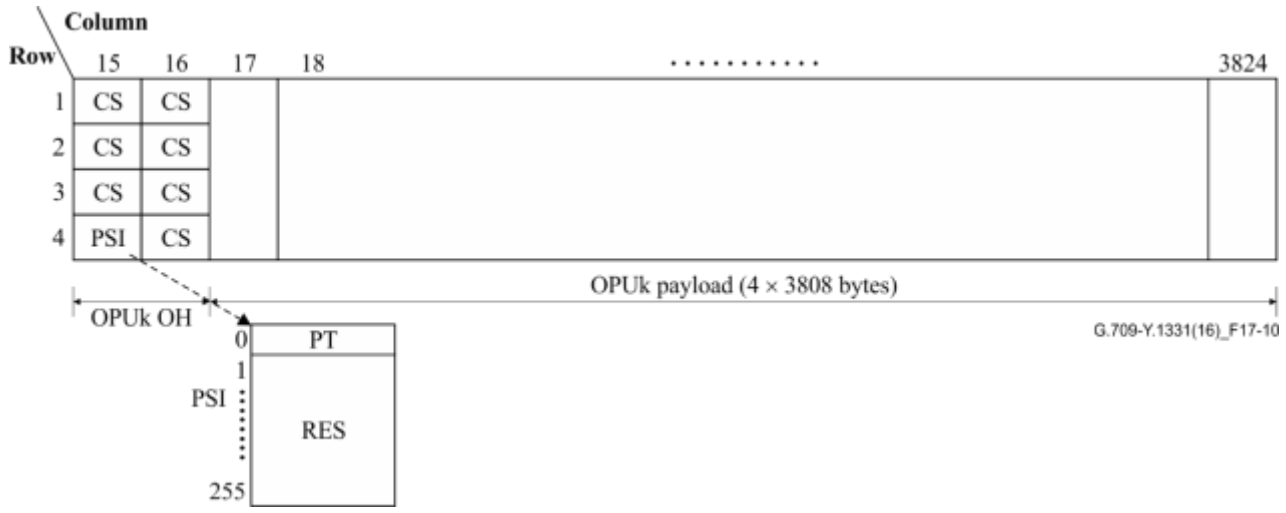


Figure 17-10 – OPUk frame structure for the mapping of a synchronous constant bit stream

The OPUk overhead for the mapping consists of a payload structure identifier (PSI) including the payload type (PT) and 255 bytes reserved for future international standardization (RES), and seven bytes for client-specific (CS) purposes. The definition of these CS overhead bytes is performed within the encapsulation process specification.

The OPUk payload for this non-specific mapping consists of 4 × 3808 bytes.

17.6.1 Mapping bit stream with octet timing into OPUk

If octet timing is available, each octet of the incoming data stream will be mapped into a data byte (octet) of the OPUk payload.

17.6.2 Mapping bit stream without octet timing into OPUk

If octet timing is not available, groups of eight successive bits (not necessarily an octet) of the incoming data stream will be mapped into a data byte (octet) of the OPUk payload.

17.7 Mapping of other constant bit-rate signals with justification into OPUk

Mapping of other CBR client signals (with up to ±100 ppm bit-rate tolerance) into an OPUk (k = 0, 1, 2, 3, 4) is performed by the generic mapping procedure as specified in Annex D.

During a signal fail condition of the incoming CBR client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in the clauses hereafter.

During a signal fail condition of the incoming ODUk/OPUk signal (e.g., in the case of an ODUk-AIS, ODUk-LCK, ODUk-OCI condition), the failed client signal is replaced by the appropriate replacement signal as defined in the clauses hereafter.

The OPUk overhead for this mapping consists of a:

– payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF) and 254 bytes plus 7 bits reserved for future international standardization (RES);

– three justification control (JC1, JC2, JC3) bytes carrying the value of GMP overhead Cm;

– three justification control (JC4, JC5, JC6) bytes carrying the value of GMP overhead ΣCnD and

– one byte reserved for future international standardization (RES).

The JC1, JC2 and JC3 bytes consist of a 14-bit C_m field (bits C1, C2, .., C14), a 1-bit Increment Indicator (II) field, a 1-bit Decrement Indicator (DI) field and an 8-bit CRC-8 field which contains an error check code over the JC1, JC2 and JC3 fields.

The JC4, JC5 and JC6 bytes consist of a 10-bit ΣC_{nD} field (bits D1, D2, .., D10), a 5-bit CRC-5 field which contains an error check code over the bits 4 to 8 in the JC4, JC5 and JC6 fields and nine bits reserved for future international standardization (RES). The default value of n in ΣC_{nD} is 8. The support for n=1 is client dependent and specified in the clauses hereafter when required.

The values of m, $C_{m,min}$, $C_{m,max}$, n, $C_{n,min}$ and $C_{n,max}$ for CBR client into OPUk are as follows:

$$m = 8,16,64,256,640 \tag{17-1}$$

$$c_{m,nom} = \left(\frac{CBR\_nom\_client\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_OPUk}{OPUk\_nom\_bit\_rate} \right) \tag{17-2}$$

$$c_{m,\min} = c_{m,nom} \times \left(\frac{1 - CBR\_client\_bit\_rate\_tolerance}{1 + OPUk\_bit\_rate\_tolerance} \right) \tag{17-3}$$

$$c_{m,\max} = c_{m,nom} \times \left(\frac{1 + CBR\_client\_bit\_rate\_tolerance}{1 - OPUk\_bit\_rate\_tolerance} \right) \tag{17-4}$$

$$C_{m,\min} = floor\left(c_{m,\min}\right) \tag{17-5}$$

$$C_{m,\max} = ceiling\left(c_{m,\min}\right) \tag{17-6}$$

$$n = 8,1 \tag{17-7}$$

$$c_{n,nom} = \left(\frac{CBR\_client\_nom\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_OPUk}{OPUk\_nom\_bit\_rate} \right) \tag{17-8}$$

$$c_{n,\min} = c_{n,nom} \times \left(\frac{1 - CBR\_client\_bit\_rate\_tolerance}{1 + OPUk\_bit\_rate\_tolerance} \right) \tag{17-9}$$

$$c_{n,\max} = c_{n,nom} \times \left(\frac{1 + CBR\_client\_bit\_rate\_tolerance}{1 - OPUk\_bit\_rate\_tolerance} \right) \tag{17-10}$$

$$C_{n,\min} = floor\left(c_{n,\min}\right) \tag{17-11}$$

$$C_{n,\max} = ceiling\left(c_{n,\min}\right) \tag{17-12}$$

$C_{m,min}$, $C_{n,min}$, $C_{m,max}$ and $C_{n,max}$ values represent the boundaries of client/OPU ppm offset combinations (i.e., min. client/max. OPU and max. client/min. OPU). In steady state, given instances of client/OPU offset combinations should not result in generated C_m and C_n values throughout this range but rather should be within as small a range as possible.

Under transient ppm offset conditions (e.g., AIS to normal signal), it is possible that C_n and C_m values outside the range $C_{n,min}$ to $C_{n,max}$ and $C_{m,min}$ to $C_{m,max}$ may be generated and a GMP de-mapper should be tolerant of such occurrences. Refer to Annex D for a general description of the GMP principles.

17.7.1 Mapping a sub-1.238 Gbit/s CBR client signal into OPU0

Table 17-4 specifies the clients ~~defined by this Recommendation~~that use this mapping and their GMP m, n and C_{nD} parameter values. Table 17-5 specifies the replacement signals for those clients.

The support for 1-bit timing information (C_1) is client dependent. Clients for which the 8-bit timing information in C_m with m=8 is sufficient will not deploy the ability to transport ΣC_{1D} and the JC4/5/6 value will be fixed to all-0s.

The OPU0 payload for this mapping consists of 4 × 3808 bytes. The bytes in the OPU0 payload area are numbered from 1 to 15232. The OPU0 payload byte numbering for GMP 1-byte (8-bit) blocks is illustrated in Figure 17-11. In row 1 of the OPU0 frame the first byte will be labelled 1, the next byte will be labelled 2, etc.

Groups of eight successive bits (not necessary being a byte) of the client signal are mapped into a byte of the OPU0 payload area under control of the GMP data/stuff control mechanism. Each byte in the OPU0 payload area may either carry 8 client bits, or carry 8 stuff bits. The stuff bits are set to zero.

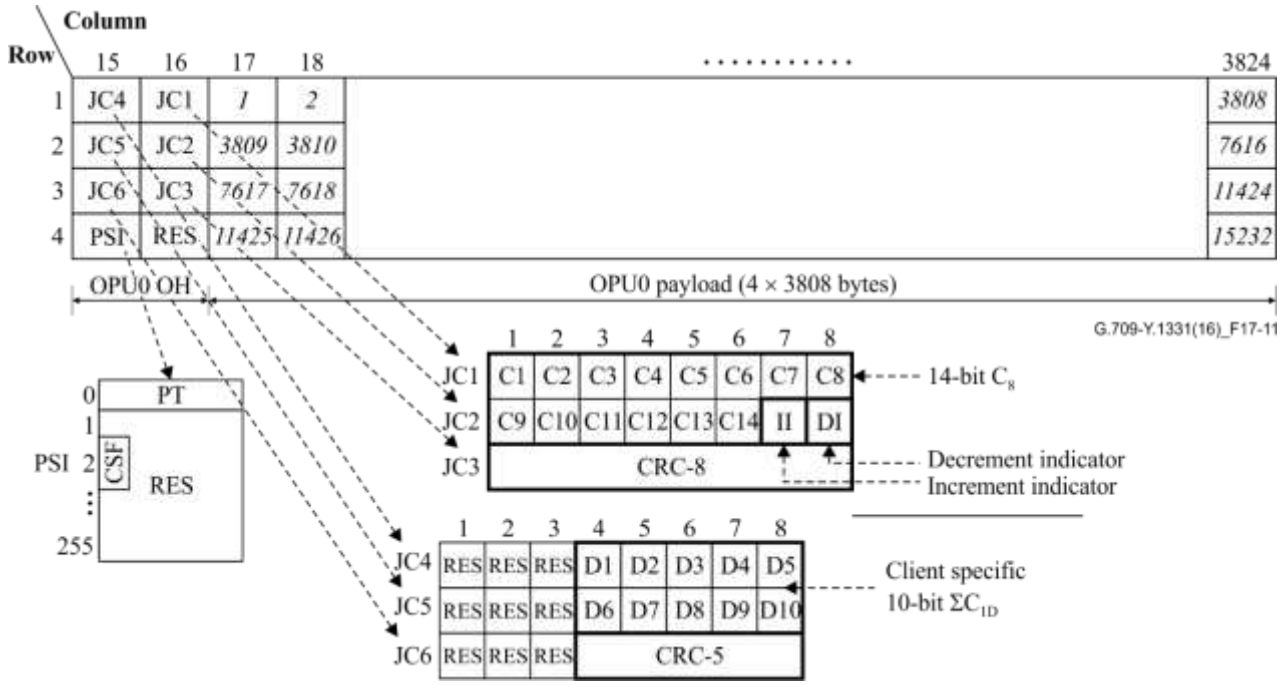


Figure 17-11 – OPU0 frame structure for the mapping of a sub-1.238 Gbit/s client signal

Table 17-4 – m, n and C_{nD} for sub-1.238G clients into OPU0

Client signal	Nominal bit rate (kbit/s)	Bit rate tolerance (ppm)	m	n	C_{nD}
Transcoded 1000BASE-X (see clause 17.7.1.1)	15/16 × 1 250 000	±100	8	8	No
STM-1	155 520	±20	8	1	Yes
STM-4	622 080	±20	8	1	Yes
FC-100	1 062 500	±100	8	8	No
SBCON/ESCON	200 000	±200	8	8	No
DVB-ASI	270 000	±100	8	8	No
SDI	270 000	±2.8	8	TBD	TBD

Table 17-5 – Replacement signal for sub-1.238G clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
STM-1	Generic-AIS	±20
STM-4	Generic-AIS	±20
1000BASE-X	Link Fault	±100
FC-100	NOS	±100
SBCON/ESCON	NOS	±200
DVB-ASI	Generic-AIS	±100
SDI	Generic-AIS	For further study

17.7.1.1 1000BASE-X transcoding

The 1000BASE-X signal (8B/10B coded, nominal bit rate of 1 250 000 kbit/s and a bit-rate tolerance up to ±100 ppm) is synchronously mapped into a 75-octet GFP-T frame stream with a bit rate of 15/16 × 1 250 000 kbit/s ±100 ppm (approximately 1 171 875 kbit/s ±100 ppm). This process is referred to as "timing transparent transcoding (TTT)". The 15/16 × 1 250 000 kbit/s ±100 ppm signal is then mapped into an OPU0 by means of the generic mapping procedure as specified in clause 17.7.1 and Annex D.

For 1000BASE-X client mapping, 1-bit timing information (C_1) is not needed, so OPU0 JC4/JC5/JC6 OH value will be fixed to all-0s.

The mapping of the 1000BASE-X signal into GFP-T is performed as specified in [ITU-T G.7041] with the following parameters:

– Each GFP-T frame contains one superblock

– The 65B_PAD character is not used

– GFP idle frames are not used

– The GFP frame pFCS is not used.

During a signal fail condition of the incoming 1000BASE-X client signal (e.g., in the case of a loss of input signal), either:

– This failed incoming 1000BASE-X signal is replaced by a stream of 10B blocks, with a bit rate of 1 250 000 kbit/s ±100 ppm, each carrying a link fault indication as specified in [IEEE 802.3], which stream is then applied at the GFP-T mapper, or

– The GFP-T signal is replaced by a stream of GFP client signal fail (CSF) and GFP-idle frames as specified in [ITU-T G.7041] with a bit rate of 15/16 x 1 250 000 kbit/s ±100 ppm.

During either

– A signal fail condition of the incoming ODU0/OPU0 signal (e.g., in the case of an ODU0-AIS, ODU0-LCK, ODU0-OCI condition), or

– Incoming CSF frames as specified in [ITU-T G.7041].

The GFP-T de-mapper process generates a stream of 10B blocks, with each block carrying a link fault indication as specified in [IEEE 802.3] as a replacement signal for the lost 1000BASE-X signal.

NOTE – The Ethernet link fault indication is a stream of repeating /C1/C2/C1/C2/ … ordered sets, where C1 = /K28.5/D21.5/D0.0/D0.0/ and C2 = /K28.5/D2.2/D0.0/D0.0/. This character stream is then processed by the GFP-T mapper process in the same manner as if it were the received 8B/10B data stream, mapping it into GFP-T superblocks for transmission.

17.7.1.2 FC-100

During a signal fail condition of the incoming FC-100 signal (e.g., in the case of a loss of input signal), this failed incoming FC-100 signal is replaced by an NOS primitive sequence as specified in [b-INCITS 470].

NOTE – The NOS primitive sequence ordered set is defined as /K28.5/D21.2/D31.5/D5.2/.

During a signal fail condition of the incoming ODU0 signal (e.g., in the case of an ODU0-AIS, ODU0-LCK, ODU0-OCI condition), NOS primitive sequence ordered sets as specified in [b-INCITS 470] are generated as a replacement signal for the lost FC-100 signal.

17.7.1.3 SBCON/ESCON

During a signal fail condition of the incoming SBCON/ESCON signal (e.g., in the case of a loss of input signal), this failed incoming SBCON/ESCON signal is replaced by an NOS sequence as specified in [b-ANSI INCITS 296].

NOTE – The NOS sequence ordered set is defined as /K28.5/D0.2/.

During a signal fail condition of the incoming ODU0 signal (e.g., in the case of an ODU0-AIS, ODU0-LCK, ODU0-OCI condition), NOS sequence ordered sets as specified in [b-ANSI INCITS 296] are generated as a replacement signal for the lost SBCON/ESCON signal.

17.7.2 Mapping a supra-1.238 to sub-2.488 Gbit/s CBR client signal into OPU1

Table 17-6 specifies the clients ~~defined by this Recommendation~~that use this mapping and their GMP m, n and C_{nD} parameter values. Table 17-7 specifies the replacement signals for those clients.

The support for 8-bit timing information (ΣC_{8D}) in the OPU1 JC4/JC5/JC6 OH is required.

The support for 1-bit timing information (ΣC_{1D}) in the OPU1 JC4/JC5/JC6 OH is client dependent.

The OPU1 payload for this mapping consists of 4 × 3808 bytes. The groups of 2 bytes in the OPU1 payload area are numbered from 1 to 7616. The OPU1 payload byte numbering for GMP 2-byte (16-bit) blocks is illustrated in Figure 17-12. In row 1 of the OPU1 frame the first 2-bytes will be labelled 1, the next 2-bytes will be labelled 2, etc.

Groups of sixteen successive bits of the client signal are mapped into a group of 2 successive bytes of the OPU1 payload area under control of the GMP data/stuff control mechanism. Each group of 2 bytes in the OPU1 payload area may either carry 16 client bits, or carry 16 stuff bits. The stuff bits are set to zero.

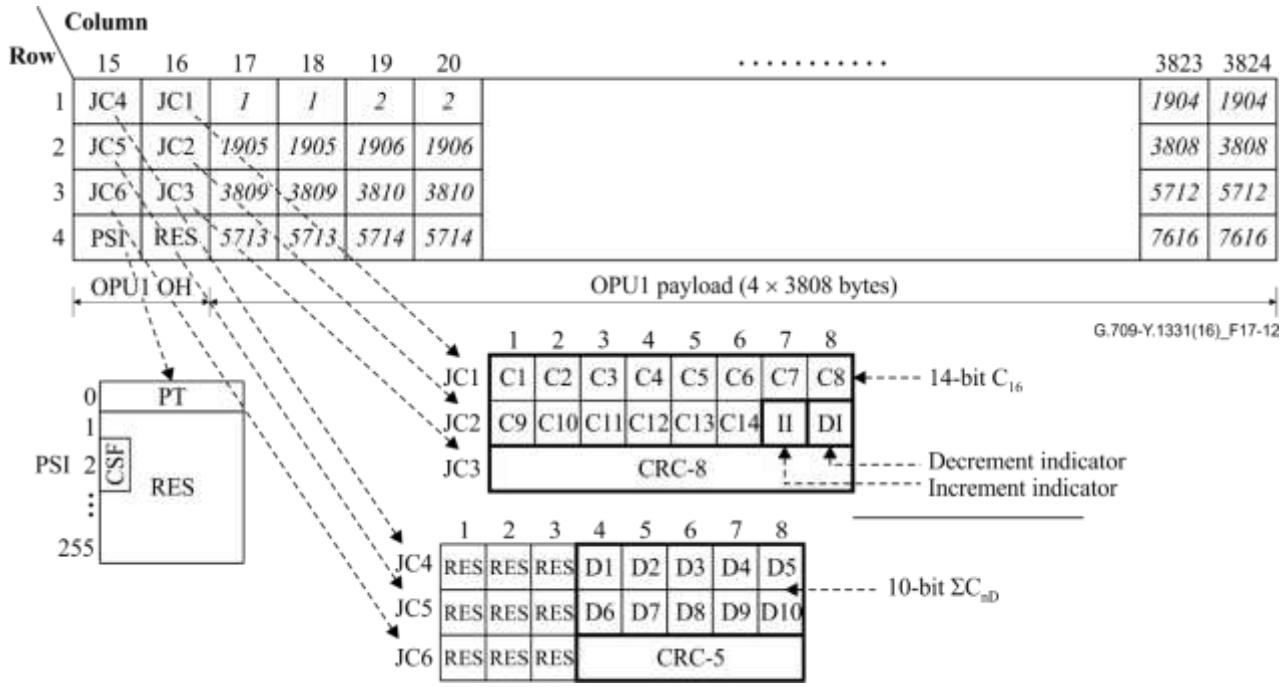


Figure 17-12 – OPU1 frame structure for the mapping of a supra-1.238 to sub-2.488 Gbit/s client signal

Table 17-6 – C_m (m=16) for supra-1.238 to sub-2.488G clients into OPU1

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)	m	n	C_{nD}
FC-200	2 125 000	±100	16	8	Yes
1.5G SDI	1 485 000	±10	16	TBD	Yes
1.5G SDI	1 485 000/1.001	±10	16	TBD	Yes

Table 17-7 – Replacement signal for supra-1.238 to sub-2.488 Gbit/s clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
FC-200	NOS	±100
1.5G SDI	Generic-AIS	For further study

17.7.2.1 FC-200

During a signal fail condition of the incoming FC-200 signal (e.g., in the case of a loss of input signal), this failed incoming FC-200 signal is replaced by an NOS primitive sequence as specified in [b-INCITS 470].

NOTE – The NOS primitive sequence ordered set is defined as /K28.5/D21.2/D31.5/D5.2/.

During a signal fail condition of the incoming ODU1 signal (e.g., in the case of an ODU1-AIS, ODU1-LCK, ODU1-OCI condition), NOS primitive sequence ordered sets as specified in [b-INCITS 470] are generated as a replacement signal for the lost FC-200 signal.

17.7.3 Mapping CBR client signals into OPU2

Table 17-8 specifies the clients ~~defined by this Recommendation~~ that use this mapping and their GMP m, n and C_{nD} parameter values. Table 17-9 specifies the replacement signals for those clients.

The support for 8-bit timing information (ΣC_{8D}) in the OPU2 JC4/JC5/JC6 OH is required.

The support for 1-bit timing information (ΣC_{1D}) in the OPU2 JC4/JC5/JC6 OH is client dependent.

The OPU2 payload for this mapping consists of 4 × 3808 bytes. The groups of eight bytes in the OPU2 payload area are numbered from 1 to 1904. The OPU2 payload byte numbering for GMP 8-byte (64-bit) blocks is illustrated in Figure 17-13. In row 1 of the OPU2 frame the first 8-bytes will be labelled 1, the next 8-bytes will be labelled 2, etc.

Groups of sixty-four successive bits of the client signal are mapped into a group of eight successive bytes of the OPU2 payload area under control of the GMP data/stuff control mechanism. Each group of eight bytes in the OPU2 payload area may either carry 64 client bits, or carry 64 stuff bits. The stuff bits are set to zero.

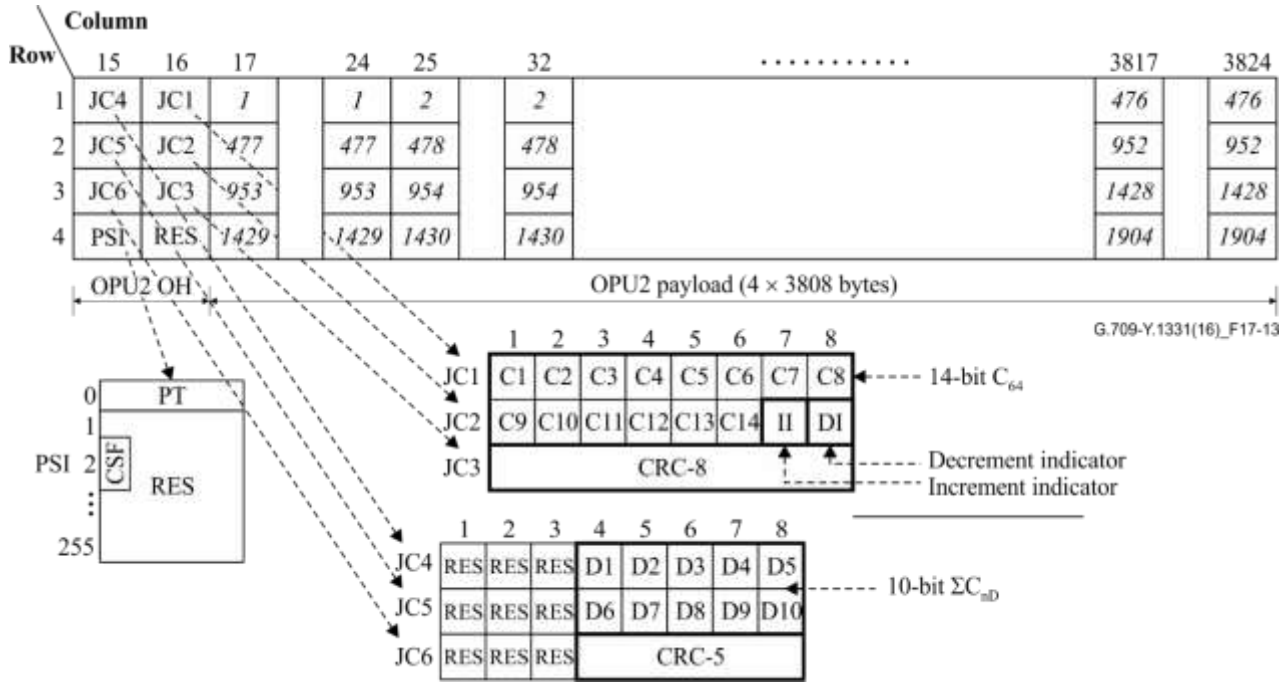


Figure 17-13 – OPU2 frame structure for the mapping of a CBR client signal

Table 17-8 – m, n and C_{nD} for CBR clients into OPU2

Client signal	Nominal bit rate (kbit/s)	Bit rate tolerance (ppm)	m	n	C_{nD}
For further study					

Table 17-9 – Replacement signal for CBR clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
For further study		

17.7.4 Mapping CBR client signals into OPU3

Table 17-10 specifies the clients ~~defined by this Recommendation~~that use this mapping and their GMP m, n and C_{nD} parameter values. Table 17-11 specifies the replacement signals for those clients.

The support for 8-bit timing information (ΣC_{8D}) in the OPU3 JC4/JC5/JC6 OH is required.

The support for 1-bit timing information (ΣC_{1D}) in the OPU3 JC4/JC5/JC6 OH is client dependent.

The OPU3 payload for this mapping consists of 4 × 3808 bytes. The groups of 32 bytes in the OPU3 payload area are numbered from 1 to 476. The OPU3 payload byte numbering for GMP 32-byte (256-bit) blocks is illustrated in Figure 17-14. In row 1 of the OPU3 frame the first 32-bytes will be labelled 1, the next 32-bytes will be labelled 2, etc.

Groups of two hundred-fifty-six successive bits of the client signal are mapped into a group of 32 successive bytes of the OPU3 payload area under control of the GMP data/stuff control mechanism. Each group of 32 bytes in the OPU3 payload area may either carry 256 client bits, or carry 256 stuff bits. The stuff bits are set to zero.

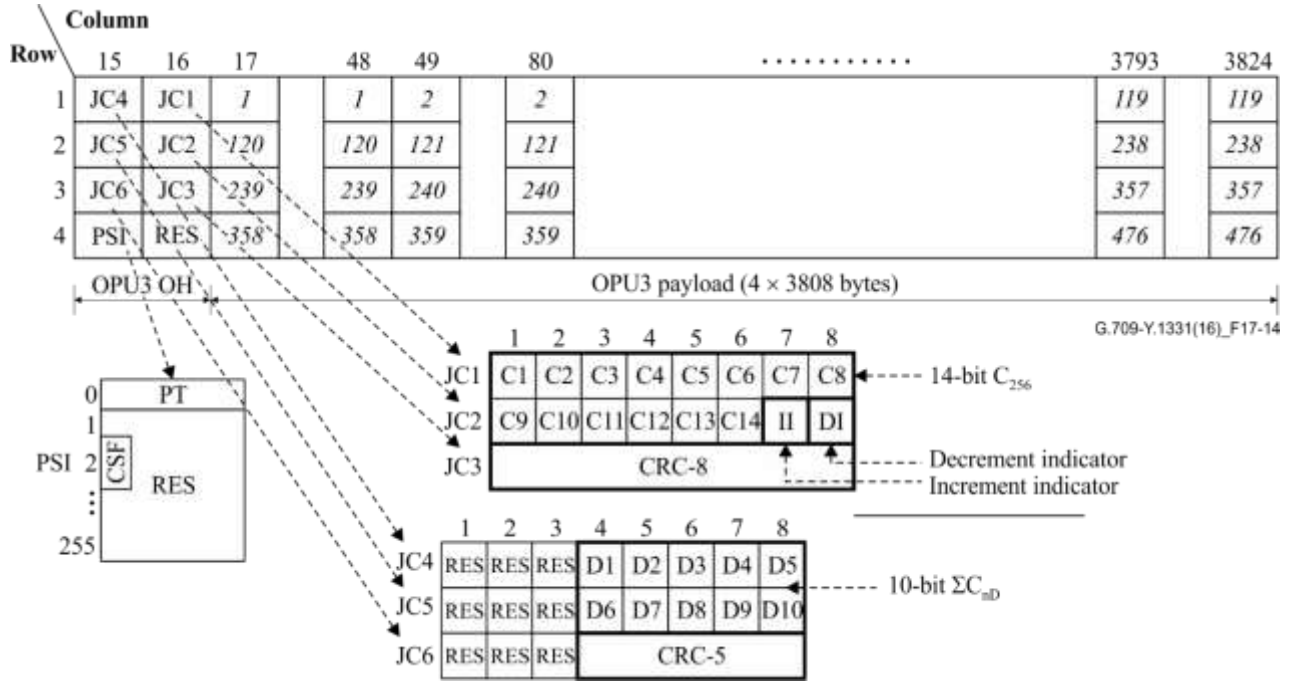


Figure 17-14 – OPU3 frame structure for the mapping of a CBR client signal

Table 17-10 – m, n and C_{nD} for CBR clients into OPU3

Client signal	Nominal bit rate (kbit/s)	Bit rate tolerance (ppm)	m	n	C_{nD}
Transcoded 40GBASE-R (see clause 17.7.4.1)	1027/1024 × 64/66 × 41 250 000	±100	256	8	Yes

Table 17-11 – Replacement signal for CBR clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
40GBASE-R	Continuous 40GBASE-R local fault sequence ordered sets with four PCS lane alignment markers inserted after each 16383 x 4 sixty-six-bit blocks	±100

A 40GBASE-R local fault sequence ordered set is a 66B control block (sync header = 10) with a block type of 0x4B, an "O" code of 0x00, a value of 0x01 to indicate "local fault" in lane 3, and all of the other octets (before scrambling) equal to 0x00.

17.7.4.1 40GBASE-R multi-lane processing and transcoding

The 40GBASE-R client signal (64B/66B encoded, nominal aggregate bit-rate of 41 250 000 kbit/s, ±100 ppm) is recovered using the process described in Annex E for parallel 64B/66B interfaces. The lane(s) of the physical interface are bit-disinterleaved, if necessary, into four streams of 10 312 500 kbit/s. 66B block lock and lane alignment marker lock are acquired on each PCS lane, allowing the 66B blocks to be de-skewed and reordered.

The resulting sequence is descrambled and transcoded according to the process described in Annex B into 513B code blocks. Each pair of two 513B code blocks is combined according to the process described in Annex F into a 1027B block, resulting in a bit stream of 1027/1024 × 40 000 000 kbit/s ±100 ppm (40,117,187.500 kbit/s ±100 ppm). This process is referred to as "timing transparent transcoding (TTT)", mapping a bit stream which is 1027/1056 times the bit-rate of the aggregate Ethernet signal.

In the mapper, the received Ethernet PCS lane BIP may be compared with the expected Ethernet PCS lane BIP as a non-intrusive monitor.

The de-mapper will insert a compensated Ethernet PCS lane BIP as described in Annex E. In addition, as described in Annex E, the combined error mask resulting from the PCS BIP-8 error mask and the OTN BIP-8 error mask may be used as a non-intrusive monitor.

For 40GBASE-R client mapping, 1-bit timing information (C_1) is not needed.

The de-mapper will recover from the output of the GMP processor 1027B block lock, and then trans-decode each 1027B block to sixteen 66B blocks as described in Annex E. Trans-decoded lane alignment markers are constructed with a compensated BIP-8. The 66B blocks are then re-distributed round-robin to PCS lanes. If the number of PCS lanes is greater than the number of physical lanes of the egress interface, the appropriate numbers of PCS lanes are bit-multiplexed onto the physical lanes of the egress interface.

17.7.5 Mapping CBR client signals into OPU4

Table 17-12 specifies the clients ~~defined by this Recommendation~~that use this mapping and their GMP m, n and C_{nD} parameter values. Table 17-13 specifies the replacement signals for those clients.

The support for 8-bit timing information (ΣC_{8D}) in the OPU4 JC4/JC5/JC6 OH is required.

The support for 1-bit timing information (ΣC_{1D}) in the OPU4 JC4/JC5/JC6 OH is client dependent.

The OPU4 payload for this mapping consists of 4 × 3800 bytes for client data and 4 × 8 bytes with fixed stuff. The groups of 80 bytes in the OPU4 payload area are numbered from 1 to 190. The OPU4 payload byte numbering for GMP 80-byte (640-bit) blocks is illustrated in Figure 17-15. In row 1 of the OPU4 frame the first 80-bytes will be labelled 1, the next 80-bytes will be labelled 2, etc.

Groups of six hundred and forty successive bits of the client signal are mapped into a group of 80 successive bytes of the OPU4 payload area under control of the GMP data/stuff control mechanism. Each group of 80 bytes in the OPU4 payload area may either carry 640 client bits, or carry 640 stuff bits. The stuff bits are set to zero.

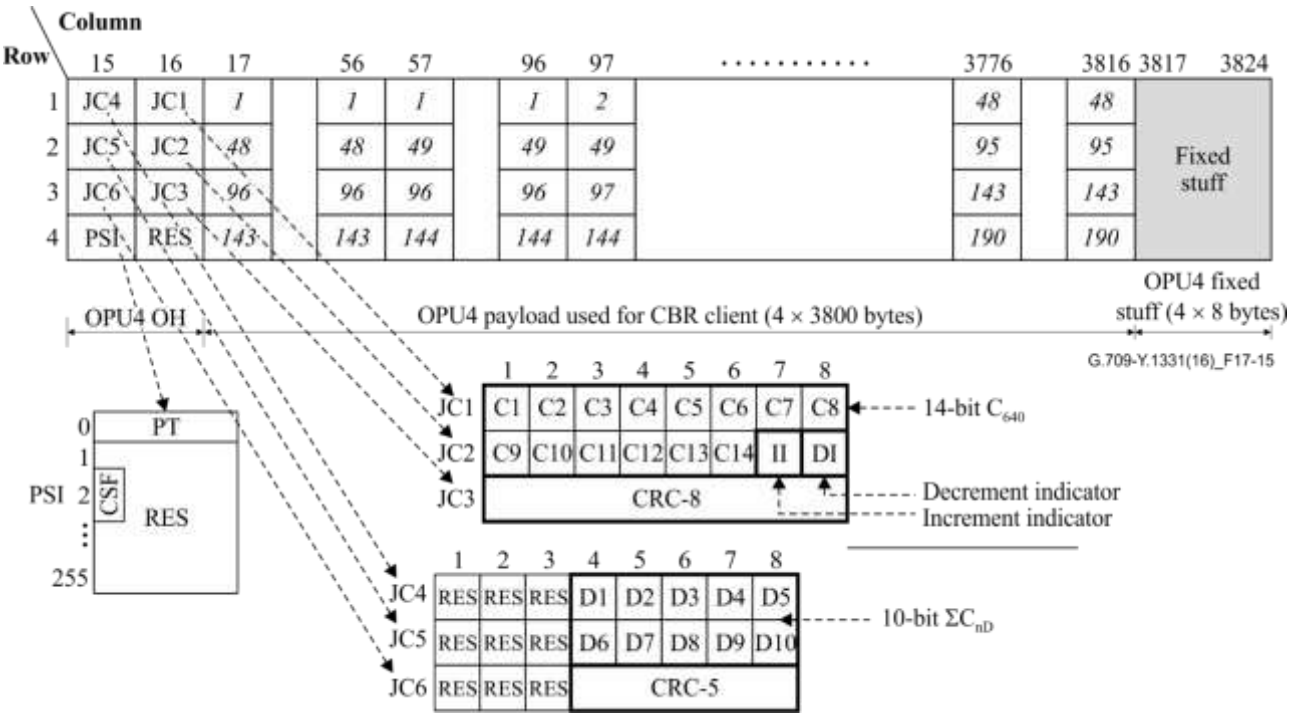


Figure 17-15 – OPU4 frame structure for the mapping of a CBR client signal

Table 17-12 – m, n and C_{nD} for CBR clients into OPU4

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)	m	n	C_{nD}
100GBASE-R (see 17.7.5.1)	103 125 000	±100	640	8	Yes

Table 17-13 – Replacement signal for CBR clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
100GBASE-R (see 17.7.5.1)	Continuous 66B encoded and scrambled 100GBASE-R local fault sequence ordered sets with 20 PCS lane alignment markers inserted after each 16383 x 20 sixty-six-bit blocks	±100

A 66B encoded and scrambled 100GBASE-R local fault sequence ordered set is a 66B control block (sync header = 10) with a block type of 0x4B, an "O" code of 0x00, a value of 0x01 to indicate a "local fault" in lane 3, and all of the other octets (before scrambling) equal to 0x00. A self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$ shall be used, which is identical to the scrambler specified in clause 49.2.6 of [IEEE 802.3].

17.7.5.1 100GBASE-R multi-lane processing

The 100GBASE-R client signal (64B/66B encoded, nominal aggregate bit-rate of 103 125 000 kbit/s ±100 ppm) is recovered using the process described in Annex E for parallel 64B/66B interfaces. The lane(s) of the physical interface are bit-disinterleaved, if necessary, into twenty streams of 5 161 250 kbit/s. 66B block lock and lane alignment marker lock are acquired on each PCS lane, allowing the 66B blocks to be de-skewed and reordered.

In the mapper, the received Ethernet PCS lane BIP may be compared with the expected Ethernet PCS lane BIP as a non-intrusive monitor.

The de-mapper will pass through the PCS lane BIP from the ingress as described in Annex E. In addition, the received Ethernet PCS lane BIP may be compared with the expected Ethernet PCS lane BIP as a non-intrusive monitor.

For 100GBASE-R client mapping, 1-bit timing information (C_1) is not needed.

The de-mapper will recover from the output of the GMP processor 64B/66B block lock per the state diagram in Figure 82-10 [IEEE 802.3]. The 66B blocks are re-distributed round-robin to PCS lanes. If the number of PCS lanes is greater than the number of physical lanes of the egress interface, the appropriate numbers of PCS lanes are bit-multiplexed onto the physical lanes of the egress interface.

17.8 Mapping a 1000BASE-X and FC-1200 signal via timing transparent transcoding into OPUk

17.8.1 Mapping a 1000BASE-X signal into OPU0

Refer to clause 17.7.1 for the mapping of the transcoded 1000BASE-X signal and to clause 17.7.1.1 for the transcoding of the 1000BASE-X signal.

17.8.2 Mapping an FC-1200 signal into OPU2e

The nominal line rate for FC-1200 is 10 518 750 kbit/s ± 100 ppm, and must therefore be compressed to a suitable rate to fit into an OPU2e.

The adaptation of the 64B/66B encoded FC-1200 client is done by transcoding a group of eight 66B blocks into one 513B block (as described in Annex B), assembling eight 513B blocks into one 516-octet superblock and encapsulating seventeen 516-octet superblocks into an 8800 octet GFP frame as illustrated in Figure 17-17. The GFP frame consists of 2200 rows with 32 bits per row. The first row contains the GFP core header, the second row the GFP payload header. The next four rows contain 16 bytes reserved for future international standardization. The next seventeen times 129 rows contain the seventeen superblocks #1 to #17. The last row contains the GFP payload FCS. The flag (F) bit of 513B block #i (i = 0..7) is carried in Flag #i bit located in the superblock flags field. The remaining 512 bits of each of the eight 513B blocks of a superblock are carried in 16 rows of the superblock data field; bits of 513B block #0 in the first 16 rows of the superblock, bits of 513B block #1 in the next 16 rows, etc. Each 513B block contains 'j' (j = 0..8) control blocks (CB1 to CBj) and '8-j' all-data blocks (DB1..DB8-j) as specified in Annex B. Figure 17-17 presents a 513B block with three control blocks and five all-data blocks. A 513B block may contain zero to eight control blocks and a superblock may contain thus zero to sixty-four control blocks.

NOTE 1 – The GFP encapsulation stage does not generate GFP-idle frames and therefore the generated GFP stream is synchronous to the FC-1200 client stream. The adaptation process performs a 50/51 rate compression, so the resulting GFP stream has a signal bit rate of 50/51 × 10.51875 Gbit/s ± 100 ppm (i.e., 10 312 500 kbit/s ± 100 ppm).

The stream of 8800 octet GFP frames is byte-synchronous mapped into the OPU2e payload by aligning the byte structure of every GFP frame with the byte structure of the OPU2e payload (see Figure 17-16). Sixty-four fixed stuff (FS) bytes are added in columns 1905 to 1920 of the OPU2e payload. All the GFP frames have the same length (8800 octets). The GFP frames are not aligned with the OPU2e payload structure and may cross the boundary between two OPU2e frames.

During a signal fail condition of the incoming FC-1200 signal (e.g., in the case of a loss of input signal), this failed incoming FC-1200 signal is replaced by a stream of 66B blocks, with each block carrying two local fault sequence ordered sets as specified in [b-ANSI INCITS 364]. This replacement signal is then applied at the transcoding process.

NOTE 2 – Local fault sequence ordered set is /K28.4/D0.0/D0.0/D1.0/. The 66B block contains the following value SH=10 0x55 00 00 01 00 00 00 01.

During a signal fail condition of the incoming ODU2e/OPU2e signal (e.g., in the case of an ODU2e-AIS, ODU2e-LCK, ODU2e-OCI condition) a stream of 66B blocks, with each block carrying

two local fault sequence ordered sets as specified in [b-ANSI INCITS 364] is generated as a replacement signal for the lost FC-1200 signal.

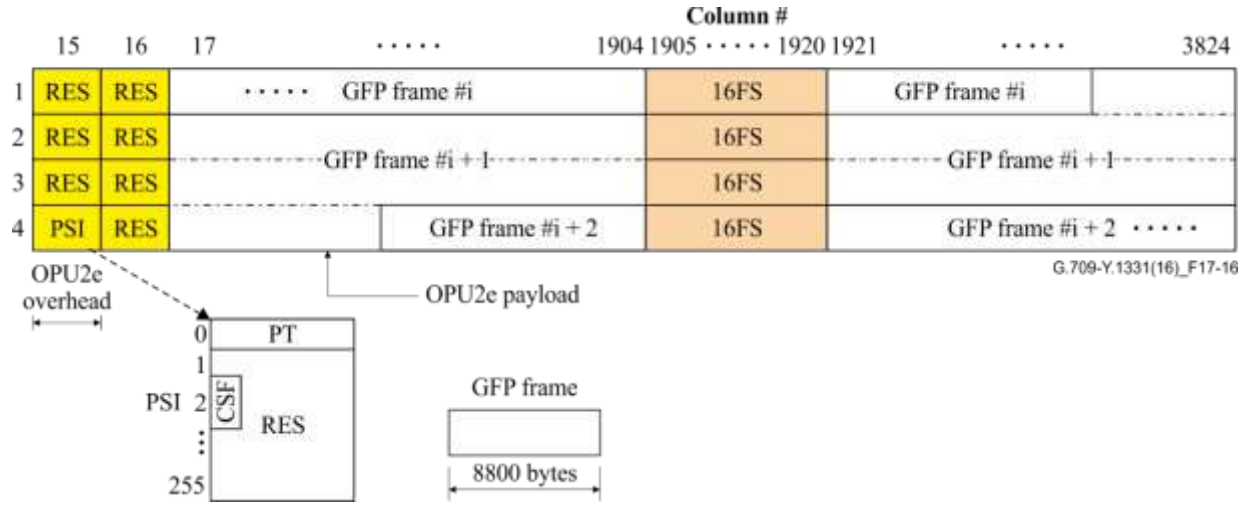


Figure 17-16 – Mapping of transcoded FC-1200 into OPU2e

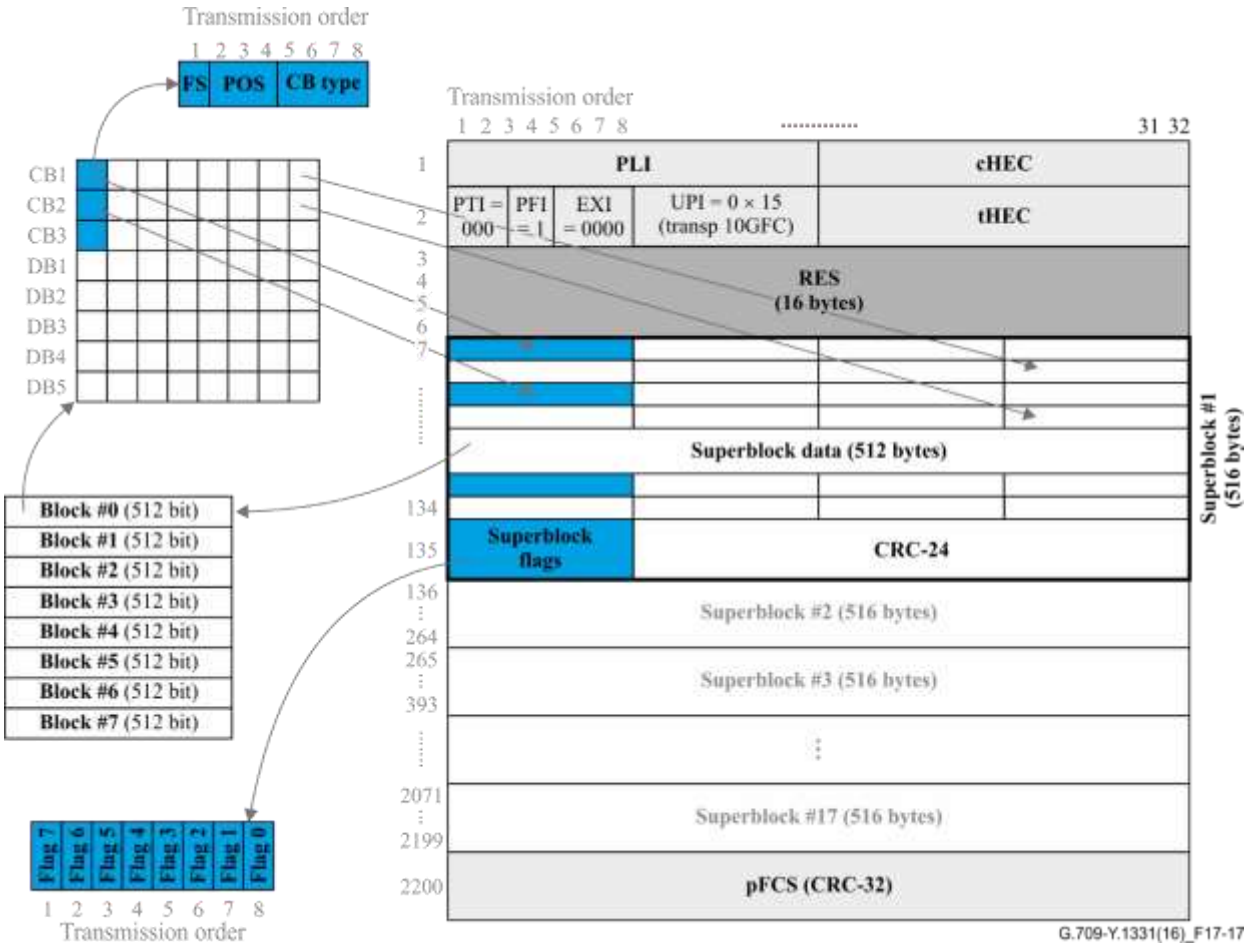


Figure 17-17 – GFP frame format for FC-1200

GFP framing is used to facilitate delineation of the superblock structure by the receiver. The leading flag bits from each of the eight 513B blocks are relocated into a single octet at the end of the 513-octet superblock data field (labelled "Superblock flags").

To minimize the risk of incorrect decoding due to errors in the 1 to 65 octets of "control" information (Flags, FC, POS, CB_Type), a CRC-24 is calculated over the 65 octets within each superblock that may contain such "control" information and appended to form a 516 octet superblock. The 65 octets in the 516-octet superblock over which the CRC-24 is calculated are the octets (1+8n) with n=0..64 (i.e., octets 1, 9, 17, .., 513). The generator polynomial for the CRC-24 is $G(x) = x^{24} + x^{21} + x^{20} + x^{17} + x^{15} + x^{11} + x^{9} + x^{8} + x^{6} + x^{5} + x + 1$ with an all-ones initialization value, where x^{24} corresponds to the MSB and x^{0} to the LSB. This superblock CRC is generated by the source adaptation process using the following steps:

1) The 65 octets of "control" information (Flags, POS, CB_Type) are taken in network octet order (see Figure 17-17), most significant bit first, to form a 520-bit pattern representing the coefficients of a polynomial $M(x)$ of degree 519.
2) $M(x)$ is multiplied by x^{24} and divided (modulo 2) by $G(x)$, producing a remainder $R(x)$ of degree 23 or less.
3) The coefficients of $R(x)$ are considered to be a 24-bit sequence, where x^{23} is the most significant bit.
4) After inversion, this 24-bit sequence is the CRC-24.

Exactly 17 of these 516-octet superblocks are prefixed with the standard GFP core and type headers and 16 octets of "reserved" (padding). Because the number of 516-octet superblocks per GFP frame is known a priori, it is possible for this mapping scheme to operate in a cut-through (as opposed to store and forward) fashion, thus minimizing the mapping latency.

The payload FCS (a CRC-32) is appended to the end of each GFP frame and is calculated across the payload information field of the GFP frame as per [ITU-T G.7041]. The purpose of the payload FCS is to provide visibility of bit errors occurring anywhere in the GFP payload information field and thus augments the coverage provided by the per-superblock CRC-24 (which only provides coverage for the "control" overhead in each superblock). The payload FCS is only for the purposes of gathering statistics.

All octets in the GFP payload area are scrambled using the $X^{43} + 1$ self-synchronous scrambler, again as per [ITU-T G.7041].

17.9 Mapping a supra-2.488 Gbit/s CBR signal into OPUflex using BMP

Mapping of a supra-2.488 Gbit/s CBR client signal (with up to ±100 ppm bit-rate tolerance) into an OPUflex is performed by a bit-synchronous mapping procedure (BMP). Table 17-14 specifies the clients ~~defined by this Recommendation~~that use this mapping.

The bit-synchronous mapping process deployed to map constant bit rate client signals into an OPUflex does not generate any justification control signals.

The OPUflex clock for the bit-synchronous mapping is derived from the client signal. During a signal fail condition of the incoming client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in Table 17-15. The OPUflex payload signal bit rate shall be within the limits specified in Table 7-3 and the limits for the ODCb clock defined in [ITU-T G.8251] and no frame phase discontinuity shall be introduced in this case and when resynchronization on the incoming client signal.

During a signal fail condition of the incoming ODUflex/OPUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), the failed client signal is replaced by the appropriate replacement signal as defined in Table 17-15.

The OPUflex overhead for this mapping consists of:

– A payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF) and 254 bytes plus 7 bits reserved for future international standardization (RES);

– Three justification control (JC) bytes, consisting of two bits for justification control (with fixed 00 value) and six bits reserved for future international standardization;

– One negative justification opportunity (NJO) byte (carrying a justification byte); and

– Three bytes reserved for future international standardization (RES).

NOTE – To allow the use of a common asynchronous/bit-synchronous de-mapper circuit for CBR client signals into ODUk (k=1,2,3 and flex), JC, NJO and PJO fields are assumed to be present in the OPUflex frame structure for the mapping of a supra-2.488G CBR client signal (Figure 17-18). This OPUflex frame structure is now compatible with the OPUk frame structure for the mapping of a CBR2G5, CBR10G or CBR40G signal (Figure 17-1). As a CBR signal is mapped into the OPUflex without justification, the NJO field contains a justification byte (stuff), the PJO field contains a data byte (D), and the JC bits are fixed to 00.

The OPUflex payload for this mapping consists of 4 × 3808 bytes (Figure 17-18). Groups of eight successive bits (not necessarily being a byte) of the client signal are mapped into a data (D) byte of the OPUflex payload area under control of the BMP control mechanism. Each data byte in the OPUflex payload area carries 8 client bits.

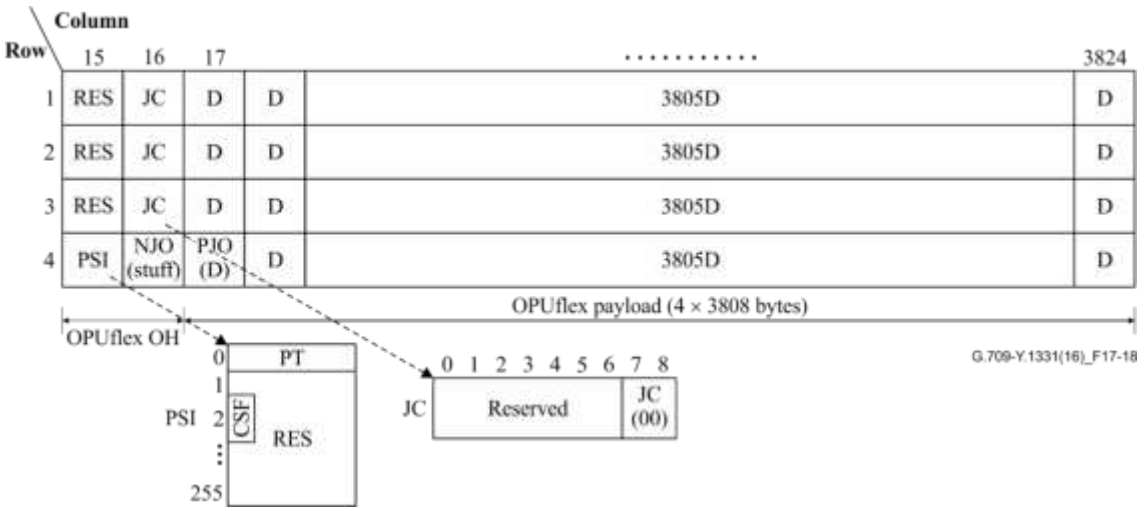


Figure 17-18 – OPUflex frame structure for the mapping of a supra-2.488 Gbit/s client signal

Table 17-14 – supra-2.488G CBR clients

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)
FC-400	4 250 000	±100
FC-800	8 500 000	±100
FC-1600	14 025 000	±100
FC-3200	28 050 000	±100
IB SDR	2 500 000	±100
IB DDR	5 000 000	±100
IB QDR	10 000 000	±100
3G SDI	2 970 000	± 10
3G SDI	2 970 000/1.001	±10

Table 17-15 – Replacement signal for supra-2.488 Gbit/s clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
FC-400	NOS	±100
FC-800	NOS	±100
FC-1600	NOS	±100
FC-3200	NOS	±100
IB SDR	For further study	±100
IB DDR	For further study	±100
IB QDR	For further study	±100
3G SDI	Generic-AIS	For further study

17.9.1 FC-400 and FC-800

During a signal fail condition of the incoming FC-400/FC-800 signal (e.g., in the case of a loss of input signal), this failed incoming FC-400/FC-800 signal is replaced by an NOS primitive sequence as specified in [b-INCITS 470].

NOTE – The NOS primitive sequence ordered set is defined as /K28.5/D21.2/D31.5/D5.2/.

During a signal fail condition of the incoming ODUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), NOS primitive sequence ordered sets as specified in [b-INCITS 470] are generated as a replacement signal for the lost FC-400/FC-800 signal.

17.9.2 FC-1600

The characteristic information of the mapped FC-1600 client signal consists of a sequence of scrambled 64B/66B encoded blocks with a nominal bit-rate of 14 025 000 kbit/s, ±100 ppm.

In case the FC-1600 interface at the mapper has FEC enabled the mapper must recover the FEC code word synchronization, extract the FEC parity bits, perform error correction and transdecode the 64B/65B blocks to 64B/66B blocks as specified in [b-INCITS 470].

In case the FC-1600 interface at the demapper has FEC enabled the demapper must recover 66B block lock from the demapped CBR signal, transcode the 64B/66B blocks to 64/65B blocks, generate and insert the FEC parity bits as specified in [b-INCITS 470].

NOTE − FC-1600 interface ports designed prior to Edition 4.6 may not be able to support termination of the FEC or transdecoding of 64B/65B blocks.

During a signal fail condition of the incoming FC-1600 signal (e.g., in the case of a loss of input signal), this failed incoming FC-1600 signal is replaced by a scrambled NOS primitive sequence as specified in [b-INCITS 470].

During signal fail condition of the incoming ODUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), scrambled NOS primitive sequence ordered sets as specified in [b-INCITS 470] are generated as a replacement signal for the lost FC-1600 signal.

17.9.3 FC-3200

The characteristic information of the mapped FC-3200 client signal consists of a sequence of 64B/66B encoded blocks with a nominal bit-rate of 28 050 000 kbit/s, ±100 ppm.

The mapper must recover the FEC code word synchronization, extract the FEC parity bits, perform error correction and transdecode the 256B/257B blocks to 64B/66B blocks as specified in [b-INCITS 488]. Uncorrectable FEC code words shall be replaced with error control blocks at the output of the transdecoder.

The demapper must recover 66B block lock from the demapped CBR signal, transcode the 64B/66B blocks to 256B/257B blocks, generate and insert the FEC parity bits as specified in [b-INCITS 488].

During a signal fail condition of the incoming FC-3200 signal (e.g., in the case of a loss of input signal), this failed incoming FC-3200 signal is replaced by a scrambled NOS primitive sequence as specified in [b-INCITS 488] ~~at the output of the transdecoder~~.

During signal fail condition of the incoming ODUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), scrambled NOS primitive sequence ordered sets as specified in [b-INCITS 488] are generated as a replacement signal for the lost FC-3200 signal at the input of the transcoder.

17.10 Mapping of packet client signals into OPUk

A set of packet client signals with an aggregated bandwidth of less than or equal to 100 Gbit/s is encapsulated into GFP-F as specified in [ITU-T G.7041] and then mapped into an OPUk, OPUflex(GFP) or OPUflex(GFP,n,k) as specified in clause 17.4 and 12.2.5.

A set of packet client signals with an aggregate bandwidth of any rate may be presented (see Note) as a stream of Ethernet packets (consisting of preamble (PA) to frame check sequence (FCS) fields) and interpacket gaps and then 64B/66B encoded as specified in [IEEE 802.3] Figure 82-5 and illustrated in Figure 17-19. This 64B/66B encoded packet client signal is then mapped into an OPUflex(IMP) as specified in clause 17.11.

A set of packet client signals with an aggregated bandwidth of more than 100 Gbit/s may be presented (see Note) as an n×25 Gbit/s stream of Ethernet packets and interpacket gaps and then 64B/66B encoded as specified in [IEEE 802.3] Figure 82-5 into a FlexE Client signal [OIF FlexE IA] which is then mapped into an OPUflex(IMP,s) as specified in clause 17.11.

A set of packet client signals with an aggregated bandwidth of less than or equal to 100 Gbit/s may be presented (see Note) as a 10, 25, 40, 50, 75 or 100 Gbit/s stream of Ethernet packets and interpacket gaps and then 64B/66B encoded as specified in [IEEE 802.3] Figure 82-5 into a FlexE Client signal [OIF FlexE IA] which is then mapped into an OPUflex(IMP,s) as specified in clause 17.11.

NOTE – Non Ethernet packet clients are assumed to be encapsulated into Ethernet packets before they are presented. Encapsulation is outside the scope of this Recommendation. Ethernet packet clients are presented directly.

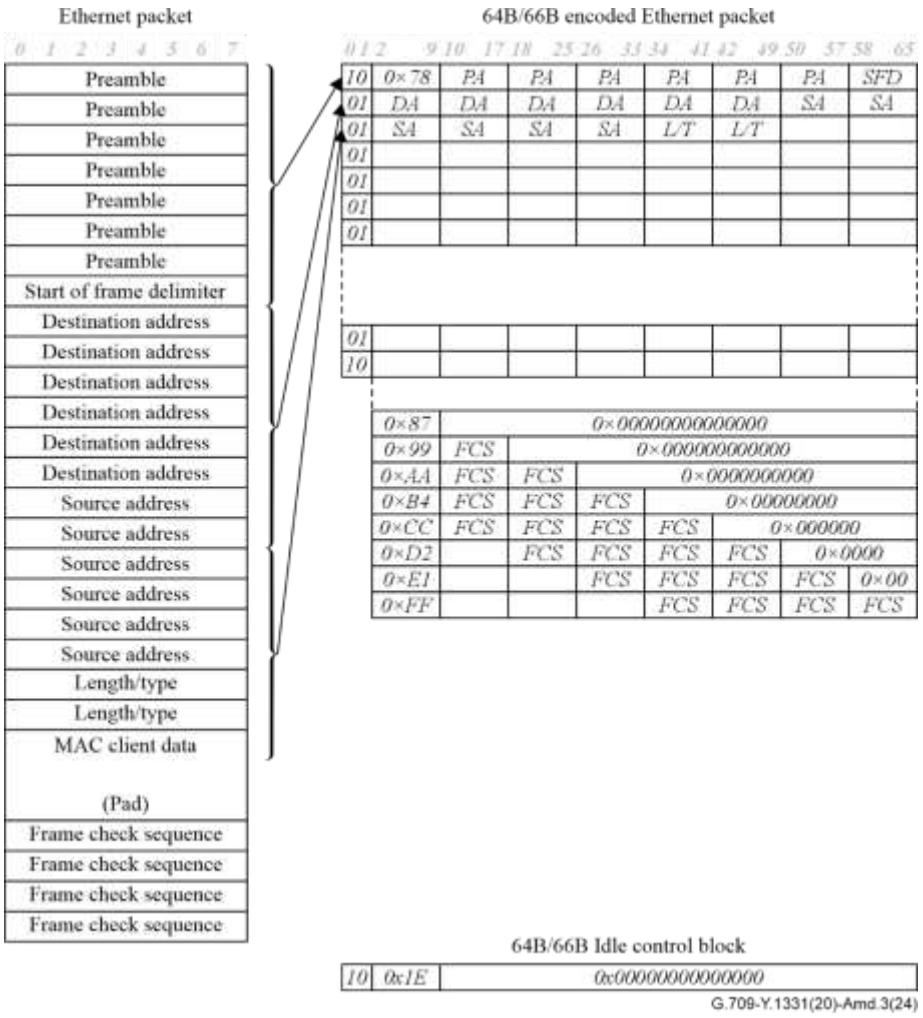


Figure 17-19 – Illustration of 64B/66B encoded Ethernet packet and interpacket gaps

17.11 Mapping of 64B/66B encoded packet client signals into OPUflex using IMP

64B/66B encoded packet client signal bit rates can have any value. Refer to clause 17.10. FlexE Client signal bit rates are s × 5,156,250.000 kbit/s ± 100 ppm, with s = 2, 8, n×5 (n ≥ 1). Refer to [OIF FlexE IA].

The 66B block stream shall be scrambled after rate adaptation and before mapping into the OPUflex. In the reverse operation, following termination of the OPUflex signal, the 66 block stream will be descrambled before being passed to the packet client (e.g., FlexE Client) layer.

A self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$ shall be used, that is identical to the scrambler specified in clause 49.2.6 [IEEE 802.3]. 66B block stream scrambling is required to provide security against false 66B block delineation (as the two sync header bits bypass the scrambler), the 66B code words replicating the OTU and ODU frame alignment signal and the 66B code words combined with the OTU scrambler pattern used for the interface replicating the OTU and ODU frame alignment signal.

Mapping of a 64B/66B encoded packet client (e.g., FlexE Client) signal (with up to ±100 ppm bit-rate tolerance) into an OPUflex is performed by the idle mapping procedure (IMP).

– The OPUflex(IMP) payload bit rate is X kbit/s ± 100 ppm and X is a configured bit rate that can have any value. The ODUflex(IMP) bit rate is 239/238 × X kbit/s ± 100 ppm.

– The OPUflex(IMP,s) payload bit rate is s × 5,156,250.000 kbit/s ± 100 ppm, with s = 2, 8, n×5 (n ≥ 1). The ODUflex bit rate is s × 239/238 × 5,156,250.000 kbit/s ± 100 ppm.

The Idle mapping procedure deploys a clock rate adaptation scheme based on Idle control character (/I/) insert/delete as per clause 82.2.3.6 of [IEEE 802.3] and/or sequence ordered set (/O/) delete as per clause 82.2.3.9 of [IEEE 802.3].

The OPUflex overhead for this mapping consists of a:

– payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF) and 254 bytes plus 7 bits reserved for future international standardization (RES);

– seven bytes reserved for future international standardization (RES).

The OPUflex payload for this mapping consists of 4 × 3808 bytes (Figure 17-20). Scrambled 66B blocks of the client signal are mapped into 66-bits of the OPUflex payload area under control of the IMP control mechanism. The 66B blocks are aligned so that the first bit of the sync header appears in one of the bit positions 1, 3, 5, or 7 of a byte in the OPUflex payload.

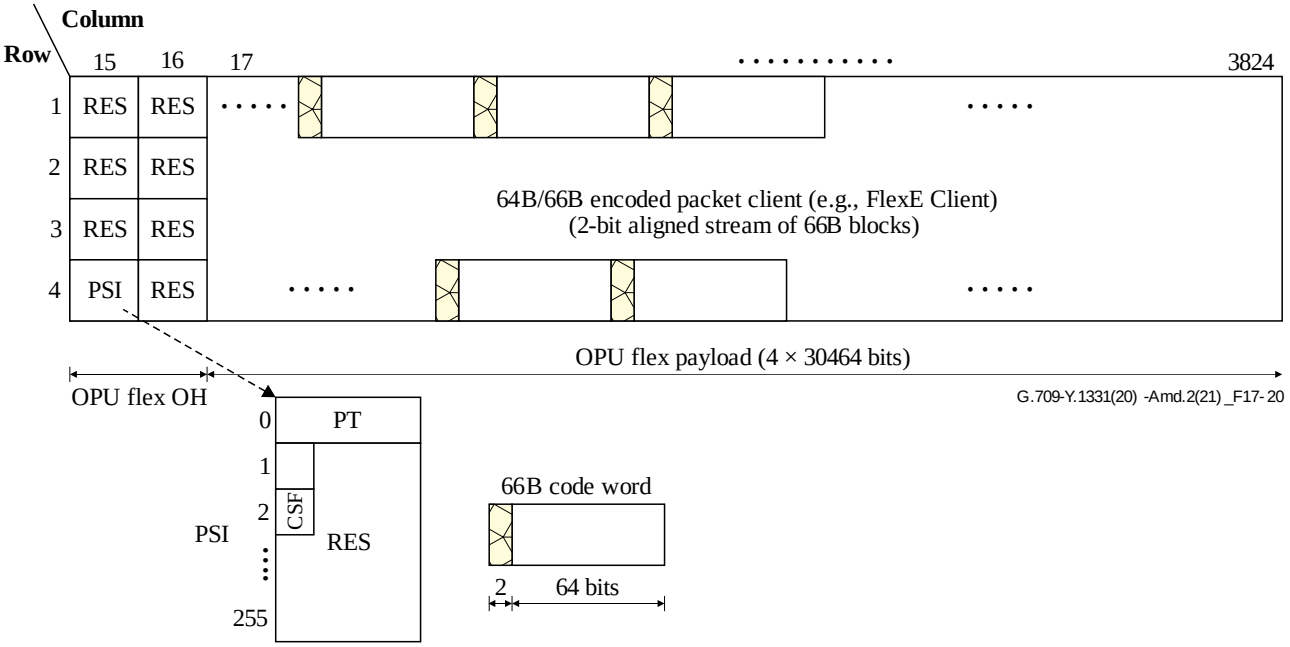


Figure 17-20 – OPUflex frame structure for the mapping of a FlexE Client signal

During a signal fail condition of the incoming 64B/66B encoded packet client (e.g., FlexE Client) signal (e.g., in the case of a loss of input signal), this failed incoming client signal is replaced by a stream of 66B blocks, with each block carrying one local fault sequence ordered set (as specified in [FlexE IA], [IEEE 802.3]). This replacement signal is then mapped into the OPUflex.

During a signal fail condition of the incoming ODUflex/OPUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), a stream of 66B blocks, with each block carrying one local fault sequence ordered sets (as specified in [FlexE IA], [IEEE 802.3]) is generated as a replacement signal for the lost client signal.

NOTE – A 64B/66B encoded packet client (e.g., FlexE Client) signal local fault sequence ordered set is a 66B control block (sync header = 10) with a block type of 0x4B, an "O" code of 0x00, a value of 0x01 to indicate a "local fault" in lane 3, and all of the other octets (before scrambling) equal to 0x00.

17.12 Mapping of FlexE aware signals into OPUflex

In the FlexE aware service p ($1 \leq p \leq m \leq 252$) 100G FlexE Instances – out of an m × 100G FlexE Instance Group signal – are crunched, padded and interleaved as per Figure 17-21. This results in a

FlexE partial rate (sub)group signal with a length of 1024 × n blocks, with $n = n_1 + n_2 + .. + n_p$ (FlexEp-n):

– i = 1 .. p represent the FlexE Instances in ascending FlexE Instance number order.

– n_i (i = 1..p) represents the number of FlexE 5G-equivalent calendar slots that are to be transferred. Supported calendar slots are of 5G or 25G type. A 25G calendar slots is equivalent to five consecutive 5G calendar slots, so if N_i 25G calendar slots must be transferred, $n_i = 5 \times N_i$.

– The value of n_i for each of the p FlexE Instances is negotiated between the FlexE partial rate (sub)group mapping port and the FlexE Shim in the customer equipment. q ($0 \leq q \leq p$) out of the p FlexE Instances may have their n_i set to 20 while the other p-q FlexE Instances may have their n_i set to values of 5, 10 and 15.

– Bits of the Ethernet error control blocks in $20\text{-}n_i$ 5G-equivalent calendar slots that are marked unavailable are located at the end of the sub-calendar and are dropped by the mapper to reduce the rate and reinserted by the demapper to restore the original rate.

– The FlexE partial rate (sub)group interleaving is simplified by adding n_i-1 padding blocks between the overhead block and the first sub-calendar block in each of the p FlexE Instances. The value of each padding block is an Ethernet error control block.

– The p OH blocks are interleaved in the order #1, #2 to #p. The p n_i-1 padding blocks are interleaved in the order #1, #2 to #p. The p sub-calendar blocks (of length n_1, n_2 to n_p) are interleaved in the order #1, #2 to #p.

NOTE 1 – An Ethernet error control block is a 66B control block (sync header = 10) with a block type of 0x1E and the other eight 7-bit characters equal to 0x1E.

NOTE 2 – The remaining m-p 100G FlexE Instances – out of an m × 100G FlexE Instance group signal – are carried via other ODUflex signals carried in other ODUCn/OTUCn/OTSiA signals.

NOTE 3 – Values of p and n (in FlexEp-n) supported and the subset of the p×20 5G-equivalent calendar slots that are included in the FlexEp-n mapped into the OPUflex are vendor specific and outside the scope of this Recommendation.

NOTE 4 – Each 100G FlexE Instance in the FlexE Aware service is an equipped 100G FlexE Instance extracted from a 100G, 200G or 400G Ethernet PHY. The first FlexE Instance of a PHY is always equipped. So, there are one or two equipped FlexE Instances per PHY in a FlexE Group over k × 200GBASE-R PHYs, and one to four equipped FlexE Instances per PHY in a FlexE Group over k × 400GBASE-R PHYs. If the last 100G FlexE Instances of a 200G or 400G Ethernet PHY are unequipped, they do not belong to the FlexE (sub)Group and shall not be carried over the OPUflex payload. Unequipped is indicated with a FlexE Group number set to all-zeros. An unequipped FlexE Instance carries LF in all blocks except in its first OH block (Order Set with 0x5 "O" code used for alignment).

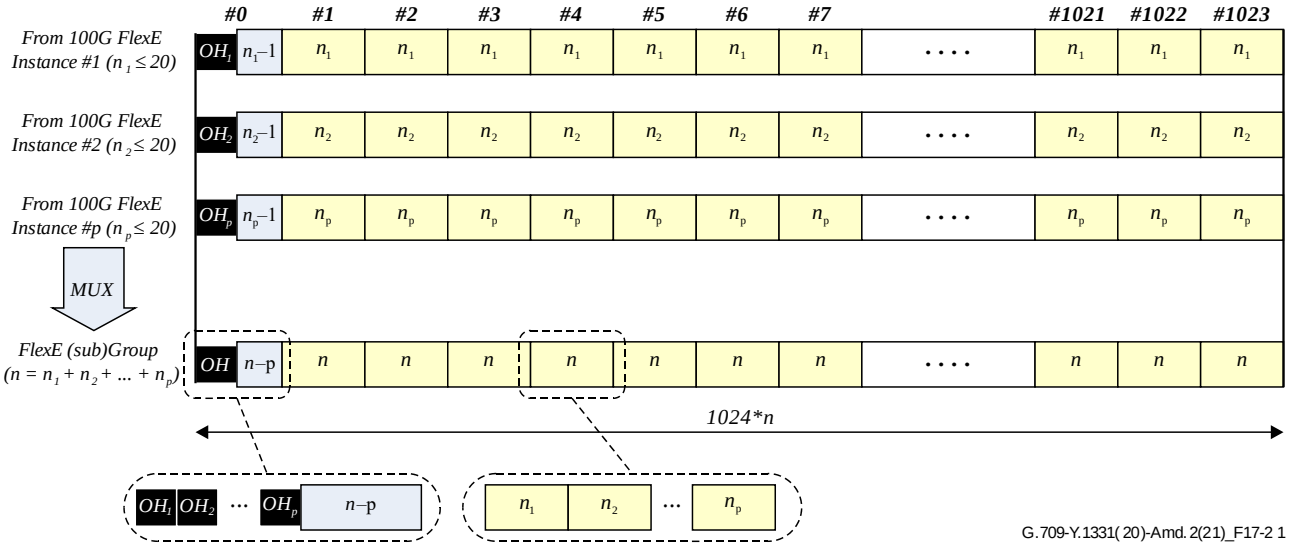


Figure 17-21 – Full/partial rate FlexE Instance interleaving

The 66B block stream of the FlexE partial rate (sub)group signal shall be scrambled after rate adaptation and before mapping into the OPUflex. In the reverse operation, following termination of the OPUflex signal, the 66B block stream will be descrambled before being passed to the FlexE deinterleaving, depadding and decrunching process.

A self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$ shall be used, that is identical to the scrambler specified in clause 49.2.6 [IEEE 802.3]. 66B block stream scrambling is required to provide security against false 66B block delineation (as the two sync header bits bypass the scrambler), the 66B codewords replicating the OTU and ODU frame alignment signal and the 66B codewords combined with the OTU scrambler pattern used for the interface replicating the OTU and ODU frame alignment signal.

The FlexE partial rate (sub)group signal (with up to ±100 ppm bit-rate tolerance) is mapped into the OPUflex as per the encoding and stuffing distribution described below. The OPUflex payload bit rate is 100GE_bit_rate × 240/239 × n/20 kbit/s ± 100 ppm, with $n = n_1 + n_2 + .. + n_p$. The ODUflex bit rate is 100GE_bit_rate × 240/238 × n/20 kbit/s ± 100 ppm.

On the transmit side, the stuffing is inserted as 16-byte blocks and systematically controlled using a 15 bit sigma-delta as:

100GE_rate × (16k-1)/16k × n × 1024/(1+20×1023) = 100GE_rate × n/20 × 240/238 × 238/239 × (4×238-3-s)/(4×238) → s = 3373/23384 (Numerator=3373, Denominator=23384)

This encoding uses a bit synchronous generic mapping procedure (BGMP) for the distribution and encoding of the stuffing. The GMP is operated in a special mode, where the C_m value does not come from a synchronizer but is generated deterministically, and $\sum C_{nD}$ is unused ($C_n = C_m$). See clause D.2.4 for a generic description of BGMP.

As shown in the Figure 17-23, a value of n = m =128 (16-Byte) is chosen. Given the deterministic generation of the stuffing, only two $C_{128}(t)$ values are used for this mapping (C_{128} = 948 or 949). Generating C_{128} = 949 (vs. C_{128} = 948) is deterministic at the source (s = 3373/23384, 15-bit sigma-delta).

$C_m(t)$ is deterministically generated using a sigma-delta C_m=948/C_m=949 justification distribution with s = 3373/23384:

The distribution of the deterministic justification ($C_m(t)$= 948, instead of $C_m(t)$ =949) follows the sigma-delta equation below:

– $C_m(t) = 948$ if $(j \times 3373) \bmod 23384 < 3373$ (17-13)

– $C_m(t) = 949$ if $(j \times 3373) \bmod 23384 \geq 3373$ (17-14)

The index 'j' in equations (17-13) and (17-14) enumerates frames in the 23384 frame sequence. It counts from 1 to 23384.

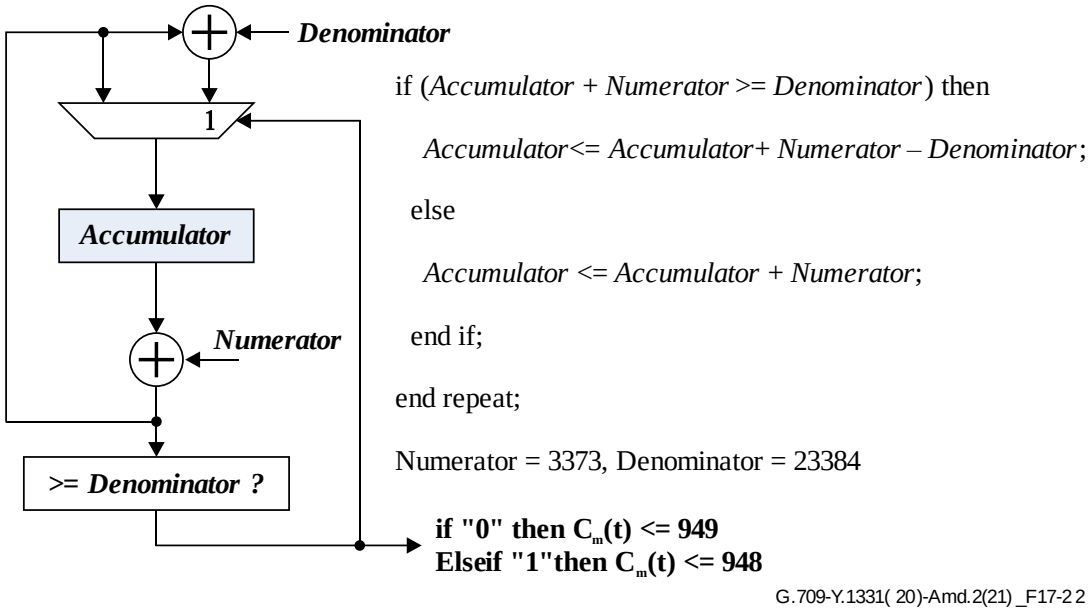


Figure 17-22 – Sigma-delta accumulator for deterministic justification generation

The OPUflex overhead for this mapping consists of a:

– payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF) and 253-p bytes plus 7 bits reserved for future international standardization (RES);

– PSI[3] carries the value of p

– PSI[4] to PSI[3+p] carries the values of n_1 to n_p

– three justification control (JC1, JC2, JC3) bytes carrying the value of GMP overhead Cm;

– four bytes reserved for future international standardization (RES).

The JC1, JC2 and JC3 bytes consist of a 14-bit C_m field (bits C1, C2, .., C14), a 1-bit increment indicator (II) field, a 1-bit decrement indicator (DI) field and an 8-bit CRC-8 field which contains an error check code over the JC1, JC2 and JC3 fields.

The OPUflex payload for this mapping consists of 4 × 3808 bytes (Figure 17-23). Blocks of 16 bytes in the OPUflex payload area are numbered from 1 to 952. The OPUflex payload byte numbering for GMP 16-byte (128-bit) blocks is illustrated in Figure 17-23. In row 1 of the OPUflex frame the first 16-byte block will be labelled 1, the next 16-byte block will be labelled 2, etc.

Groups of one hundred twenty-eight successive bits of the client signal are mapped into a 16-byte block of the OPUflex payload area under control of the BGMP data/stuff control mechanism. For the case of $C_{128} = 948$, 16 byte blocks #1, #239, #477 and #715 are carrying stuff bits and the other 16-bye blocks are carrying client bits. For the case of $C_{128} = 949$, 16 byte blocks #1, #318 and #635 are carrying stuff bits and the other 16-byte blocks are carrying client bits. The stuff bits are set to zero.

Scrambled 64B/66B blocks of the client signal are mapped into 66-bits of the OPUflex payload area skipping the 16-byte Stuff blocks under control of the BGMP control mechanism. The 66B blocks are aligned so that the first bit of the sync header appears in one of the bit positions 1, 3, 5, or 7 of a byte in the OPUflex payload.

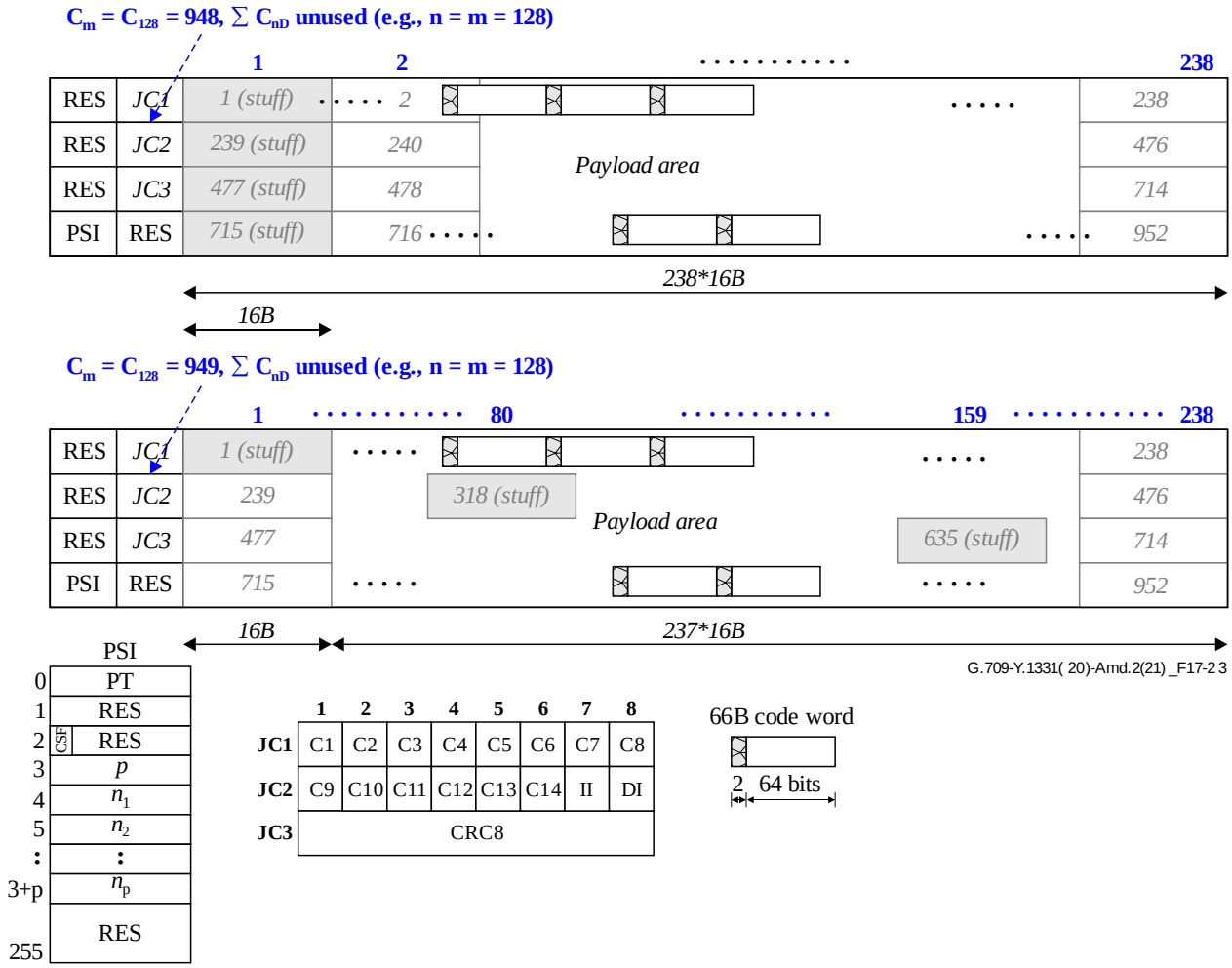


Figure 17-23 – Mapping of the combined FlexE stream into ODUflex via deterministic stuffing and using GMP encoding

During a signal fail condition of an incoming equipped 100G FlexE Instance (e.g., in the case of a loss of 100G/200G/400G input PHY signal carrying this 100G FlexE Instance or FlexE local fault condition), this failed incoming 100G FlexE Instance and the p-1 other 100G FlexE Instances are not mapped into the payload of the OPUflex. Instead the OPUflex payload is filled with the stuff blocks, 66B blocks that carry a local fault sequence ordered set and the OPU CSF bit is set to "1". The OPUflex clock is derived from a local clock. The OPUflex payload signal bit rate shall be within the limits specified in Table 7-3 and the limits for the ODCb defined in [ITU-T G.8251] and no frame phase discontinuity shall be introduced in this case and when resynchronization on the incoming client signal.

During a signal fail condition of the incoming ODUflex/OPUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition) or a CSF condition is present in the OPUflex overhead, a stream of 66B blocks, with each block carrying one local fault sequence ordered set is generated as a replacement signal for each of the p lost FlexE signals.

NOTE 5 – A FlexE Instance local fault sequence ordered set is a 66B control block (sync header = 10) with a block type of 0x4B, an "O" code of 0x00, a value of 0x01 to indicate a "local fault" in lane 3, and all of the other octets (before scrambling) equal to 0x00.

17.13 Mapping a 64B/66B PCS coded signal into OPUflex using BMP and 2-bit alignment of 66B code words

Mapping of a 64B/66B PCS coded (xGBASE-R) client signal (with up to ±100 ppm bit-rate tolerance) into an OPUflex is performed by a bit-synchronous mapping procedure (BMP). Table 17-16 specifies the clients ~~defined by this Recommendation~~that use this mapping.

The bit-synchronous mapping process deployed to map constant bit rate client signals into an OPUflex does not generate any justification control signals.

The 66B block stream of the xGBASE-R client signal shall be scrambled before mapping into the OPUflex. In the reverse operation, following termination of the OPUflex signal, the 66 block stream will be descrambled after demapping from the OPUflex.

A self-synchronizing scrambler with generator polynomial $1+ x^{39} +x^{58}$ shall be used, that is identical to the scrambler specified in clause 49.2.6 of [IEEE 802.3]. 66B block stream scrambling is required to provide security against false 66B block delineation (as the two sync header bits bypass the scrambler), the 66B codewords replicating the OTU and ODU frame alignment signal and the 66B codewords combined with the OTU scrambler pattern used for the interface replicating the OTU and ODU frame alignment signal.

The OPUflex clock for the bit-synchronous mapping is derived from the client signal. During a signal fail condition of the incoming client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in Table 17-17. The OPUflex payload signal bit rate shall be within the limits specified in Table 7-3 and the limits defined in [ITU-T G.8251] and no frame phase discontinuity shall be introduced in this case and when resynchronization on the incoming client signal.

During a signal fail condition of the incoming ODUflex/OPUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), the failed client signal is replaced by the appropriate replacement signal as defined in Table 17-17.

The OPUflex overhead for this mapping consists of:

– a payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF) and 254 bytes plus 7 bits reserved for future international standardization (RES);

– seven bytes reserved for future international standardization (RES).

The OPUflex payload area for this mapping consists of 4 × 3808 bytes (see Figure 17-24). Scrambled 64B/66B blocks of the client signal are mapped into 66-bits of the OPUflex payload area. The 66B blocks are aligned so that the first bit of the sync header appears in one of the bit positions 1, 3, 5, or 7 of a byte in the OPUflex payload.

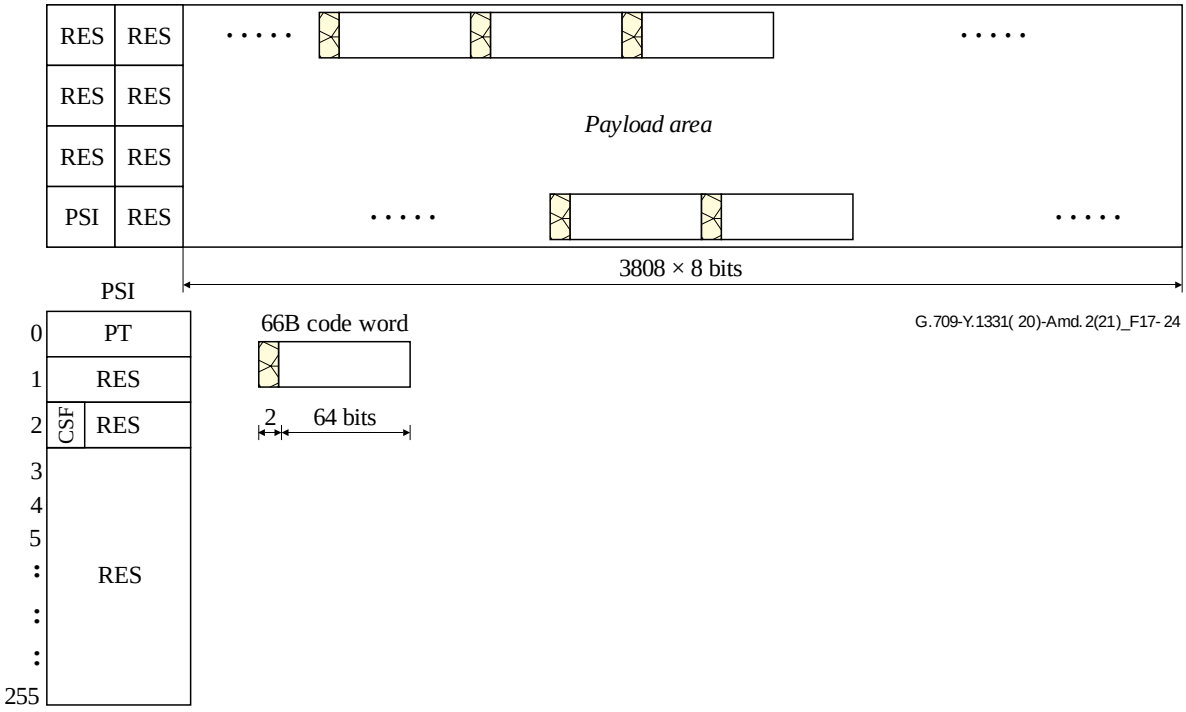


Figure 17-24 – OPUflex frame structure for the mapping of a 64B/66B PCS client signal with 2-bit alignment

Table 17-16 – ~~Supra-2.488G~~64B/66B PCS coded CBR clients

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)
25GBASE-R	25 781 250	±100
50GBASE-R	51 562 500	±100
200GBASE-R (Note)	206 250 000	±100
400GBASE-R (Note)	412 500 000	±100
NOTE – These signals represent a rate adapted 64B/66B encoded signal, which includes RC blocks.		

Table 17-17 – Replacement signal for ~~supra-2.488 Gbit/s~~64B/66B PCS coded clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
25GBASE-R	LF (Note 1)	±100
50GBASE-R	LF (Note 3)	±100
200GBASE-R	LF (Note 2)	±100
400GBASE-R	LF (Note 2)	±100
NOTE 1 – The replacement signal for a 25GBASE-R signal is a continuous stream of 66B encoded and scrambled LF sequence ordered sets encoded according to Figure 49-7 of [IEEE 802.3] using control block type 0x55 (two ordered sets per 66B block) and scrambled according to clause 49.2.6 of [IEEE 802.3]. NOTE 2 – The replacement signal for a 200GBASE-R and 400GBASE-R signal is a continuous stream of LF sequence ordered sets encoded according to Figure 82-5 of [IEEE 802.3] using control block type 0x4B. RC block insertion will be performed on the stream in the mapping direction, while 257B transcoding plus alignment marker (AM) insertion will be performed on the stream in the demapping direction after which these signals are scrambled. NOTE 3 – The replacement signal for a 50GBASE-R signal is a continuous stream of 66B encoded and scrambled LF sequence ordered sets encoded according to Figure 82-5 of [IEEE 802.3] using control block type 0x4B with 4 PCS lane alignment markers inserted after each 20479 x 4 sixty-six-bit blocks and scrambled according to clause 49.2.6 of [IEEE 802.3].		

17.13.1 25GBASE-R

The mapped format for the signal is the 25GBASE-R PCS sublayer as defined in clause 107 of [IEEE 802.3], with a bit-stream at a signalling rate of 25.78125 Gb/s consisting of 64B/66B encoded blocks.

25GBASE-R PHYs may use one of three different operational modes: No FEC, BASE-R FEC, or RS FEC. Only the RS FEC operational mode is applicable to optical PHYs, while any of the three modes may be used for backplane and copper cable PHYs, with the operational mode selected between the endpoints of a PHY link via auto-negotiation.

The No FEC mode of operation sends bits at the physical layer using the same format as the PCS: a continuous stream of 64B/66B blocks at 25.78125 Gb/s ±100 ppm.

In the BASE-R FEC mode of operation, the signal is encoded according to clause 74 of [IEEE 802.3], using a shortened cyclical code (2112,2080). This code removes the redundant sync header bit from each of thirty-two 66B blocks and adds 32 bits of parity to produce the 2112-bit FEC codeword. The FEC is framed by the mapper using a "search and test" algorithm as described in clause 73 of [IEEE 802.3]. Errors are corrected, the FEC parity is removed, and the second sync header bit is restored to produce the stream of 66B blocks to map into OPUflex. The demapper removes the redundant sync header bit from each 66B block and adds the 32 bits of parity to each group of thirty-two, now 65B blocks. Since this process adds and removes the same number of bits in the process of encoding and decoding the FEC, this mapping is timing transparent.

In the RS FEC mode of operation, the signal is encoded according to clause 108 of [IEEE 802.3], using a Reed-Solomon (528,514) error correcting code based on 10-bit symbols. Each group of four 66B blocks from the PCS is transcoded into a 257B block. Twenty 257B blocks (5140 bits) are combined with 140 bits of FEC parity (14 FEC symbols) to produce a 5280-bit FEC codeword. To frame the bit-stream, a 257-bit codeword marker (CWM) is inserted as the first 257 bits of every 1024th FEC codeword, with sufficient idles being deleted from the PCS to make room for the CWM at the same bit-rate as the PCS. The mapper will reverse this process, finding the CWMs in the bit-stream, decoding the FEC and correcting errors, removing the CWMs, trans-decoding 257B to 64B/66B, and inserting idles to maintain the bit-rate as described in the Rx direction data flow of clause 108 of [IEEE 802.3]. The demapper will remove idles to make room for the CWM (maintaining the bit-rate), transcode to 257B, and adding the FEC parity to each group of twenty 257B blocks as described in the Tx direction data flow of clause 108 of [IEEE 802.3]. This process preserves the timing of the client signal, and this mapping is timing transparent.

17.13.2 200GBASE-R and 400GBASE-R

The characteristic information of the adapted and mapped 200GBASE-R or 400GBASE-R client signal consists of a scrambled sequence of 64B/66B encoded blocks with a nominal bit-rate of X kbit/s ±100 ppm.

The mapper shall first recover the 200GBASE-R or 400GBASE-R client stream, which is referred to as the OTN reference signal, that is a sequence of 64B/66B encoded blocks as per clause 119.2.5.7 of [IEEE 802.3], with a bit rate of (528/544 × 20479/20480 × Y kbit/s) ± 100 ppm.

NOTE 1 – Y kbit/s is the nominal bit-rate of the aggregate 200GBASE-R or 400GBASE-R PCS signal consisting of 8 or 16 PCS lanes with 256B/257B encoding and FEC at the PMA service interface.

Table 17-18 – X, Y and Z parameter values

Parameter	200GBASE-R	400G BASE-R
X	206 250 000	412 500 000
Y	212 500 000	425 000 000
Z	16	32

The client stream for the 200GBASE-R or 400G BASE-R OTN reference signal (64B/66B encoded, un-scrambled and without alignment markers) is recovered using the process described in Annex J for parallel 256B/257B interfaces. The lane(s) of the physical interface are bit-disinterleaved, if necessary, into eight or sixteen streams of 26 562 500 kbit/s. Lane alignment marker lock is acquired on each PCS lane, allowing lane deskew and reorder, as well as de-interleave for FEC code word synchronization and error correction. The mapper shall extract the FEC parity bits and alignment markers, perform descrambling and transdecode the 256B/257B blocks to 64B/66B blocks. Uncorrectable FEC code words shall be replaced with error control blocks at the output of the transdecoder.

Then, the client stream for the 200GBASE-R or 400GBASE-R OTN reference signal (or its replacement stream upon signal failure) is rate compensated by inserting Z rate compensation (RC) blocks every (Z × 20479) client blocks (Figure 17-25). The resulting nominal bit-rate of the compensated 64B/66B client stream is [528/544 × Y kbit/s] = X kbit/s ± 100 ppm. The 64B/66B client stream shall be scrambled after rate compensation, and before mapping into the OPUflex.

In the reverse operation, following termination of the OPUflex signal, the demapper must recover 66B block lock from the demapped CBR signal, descramble the 64B/66B client stream, and remove the rate compensation blocks to retrieve the 200GBASE-R or 400GBASE-R client stream at the OTN reference signal as specified in clause 119.2.4.1 of [IEEE 802.3].

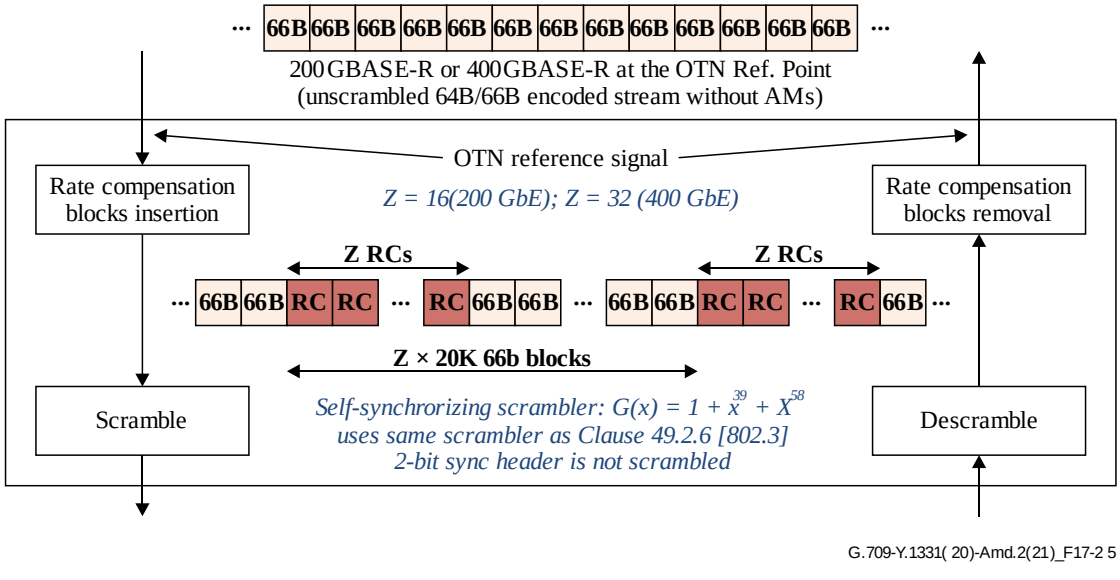


Figure 17-25 – Adapted xGBASE-R (x = 200, 400) client signal for OPUflex mapping

Then, 64B/66B blocks to 256B/257B blocks transcoding and scrambling can be performed, and as described in Annex J, alignment markers and FEC parity bits can be inserted prior to the 200GBASE-R or 400GBASE-R PCS lane distribution, and interleaving, if necessary, into physical lanes at the PMA interface.

The rate compensation blocks (RC) have the form of a specially defined 66-bit block with a control block sync header. The Z/2 first RC blocks carry an RC0 value, and the last Z/2 RC blocks carry an RC1 value as shown in Figure 17-26.

NOTE 2 – The value of the RC blocks (RC0 or RC1) has been chosen so that it does not match any defined control block. The 8b located at the same position as the block type field (bits 2 to 9 in Figure 17-26) carry the unused value 0x00, still maintaining the 4-bit hamming distance with the defined and used block types. At the sink, the RC pattern can be search on a subset of the RC0 and RC1 bits and blocks, using a simple alignment process.

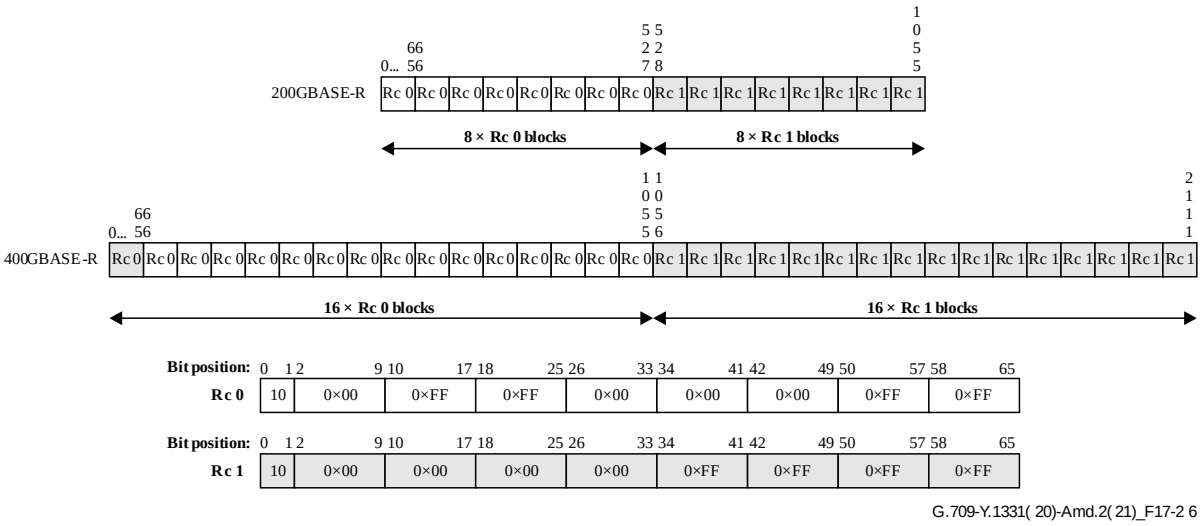


Figure 17-26 – Rate compensation blocks format for 200GBASE-R and 400GBASE-R mapping

17.13.3 50GBASE-R

The mapped format for the signal is the 50GBASE-R PCS sublayer as defined in clause 133 of [IEEE 802.3~~cd~~]. The 50GBASE-R signal (64B/66B encoded, nominal aggregate bit-rate of 51 562 500 kbit/s ±100 ppm, with alignment markers) is recovered using the following process: block lock and lane alignment marker lock are acquired on each of the four PCS lanes, with bit streams of 12 890 625 kbit/s, at the 50GBASE-R sublayer reference point, allowing the 66B blocks to be de-skewed and reordered. The four PCS lanes are serialized with their respective AMs occurring in sequence. Figure 17-27 illustrates the order of 66B blocks mapped into the ODUflex.

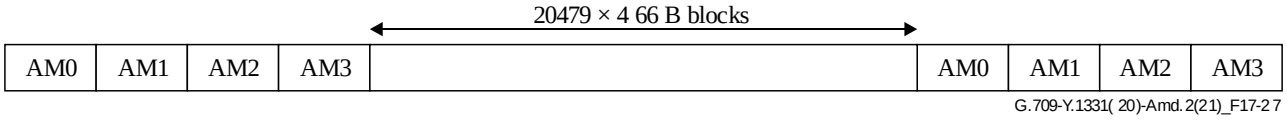


Figure 17-27 – Order of 66B blocks mapped into ODUflex for 50GBASE-R

The payload of the ODUflex is at a nominal rate of 51 562 500 kbit/s ± 100 ppm, which is a precise ratio of 33/34 from the 53 125 000 kbit/s bit-rate of the single-lane FEC encoded signal. For bit-synchronous mapping into an ODUflex, the ODUflex rate is 7887/8092 times the physical line rate of the 50G Ethernet signal; i.e., 7887/8092 × 53 125 000 kbit/s.

In the mapper, the received Ethernet PCS lane BIP may be compared with the expected Ethernet PCS lane BIP as a non-intrusive monitor.

The de-mapper will pass through the PCS lane BIP from the ingress. In addition, the received Ethernet PCS lane BIP may be compared with the expected Ethernet PCS lane BIP as a non-intrusive monitor.

The de-mapper will recover from the output of the BMP processor 64B/66B block lock per the state diagram in Figure 82-10 [IEEE 802.3]. The 66B blocks are re-distributed round-robin to four PCS lanes and these four PCS lanes presented at the 50GBASE-R egress reference point of the PCS layer.

17.14 Mapping a 256B/257B PCS coded signal into OPUflex

Mapping of a 256B/257B PCS coded (*x*GBASE-R) client signal (with up to ±50 ppm bit-rate tolerance) into an OPUflex is performed by a bit-synchronous mapping procedure (BMP) that includes a deterministic rate compensation function to compensate for the removal of the alignment markers and a padding function for 257-bit block alignment. Table 17-19 specifies the clients that use this mapping.

The OPUflex clock for the bit-synchronous mapping is derived from the client signal. During a signal fail condition of the incoming client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in Table 17-20. The OPUflex payload signal bit rate shall be within the limits specified in Table 7-3 and the limits defined in [ITU-T G.8251] and no frame phase discontinuity shall be introduced in this case or when resynchronizing on the incoming client signal.

During a signal fail condition of the incoming ODUflex/OPUflex signal (e.g., in the case of an ODUflex-AIS, ODUflex-LCK, ODUflex-OCI condition), the failed client signal is replaced by the appropriate replacement signal as defined in Table 17-20.

The OPUflex overhead for this mapping consists of:

– a payload structure identifier (PSI) including the payload type (PT) as specified in Table 15-9, the client signal fail (CSF), the client degrade indication (CDI) as specified in Annex K, and 254 bytes plus 4 bits reserved for future international standardization (RES).

– Three justification control (JC) bytes, consisting of one bit for deterministic justification control and seven bits reserved for future international standardization.

– four bytes reserved for future international standardization (RES).

The OPUflex payload area for this mapping consists of 4 × 3808 bytes (see Figure 17-29). It carries 474 unscrambled 257-bit blocks of the adapted client stream, plus 6-bit pad (reserved bits) and a CRC-32. The 257-bit blocks from the adapted client signal are aligned to the OPUflex payload area.

The CRC-32 field is optionally used at the demapper to support error marking in order to guarantee Ethernet client– mean time to false packet acceptance (MTTFPA) requirements are met. It is calculated over 121818 bits corresponding to the mapped client data (including any 257b stuff blocks) in the OPUflex payload area. The 6-bit pad is not covered by the CRC-32. The generator polynomial is as specified in clause 3.2.9 of [IEEE 802.3]:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

This CRC-32 is generated by the source adaptation mapping process using the following steps:

1) The first 32 bits of the 121818 bits in the payload are complemented

2) The 121818 bits (in transmission order) form a pattern representing the coefficients of a polynomial *M(x)* of degree 121817 (the first bit of the 121818 pattern corresponds to the x^{121217} term and the last bit of the 121818 input bits corresponds to the x^0 term).

3) *M(x)* is multiplied by x^{32} and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 31 or less.

4) The coefficients of *R(x)* are considered to be a 32-bit sequence, where x^{31} is the most significant bit.

5) The 32 bits are complemented, and the result is the CRC-32, where the first bit of the CRC-32 to be transmitted is the coefficient x^{31} and the last bit transmitted is the coefficient of x^0.

The sink adaptation demapping process performs steps 1-5 in the same manner as the source adaptation process, except that the $M(x)$ of step 1 includes the CRC-32 in received order and has degree $n + 32$. In the absence of bit errors, the remainder shall be all zeros. If the CRC-32 check process detects an error, then all 257-bit client blocks demapped from that OPUflex frame payload are error marked as shown in Figure 17-28 (each will carry four transcoded 64B/66B error control blocks, set to EBLOCK_R).

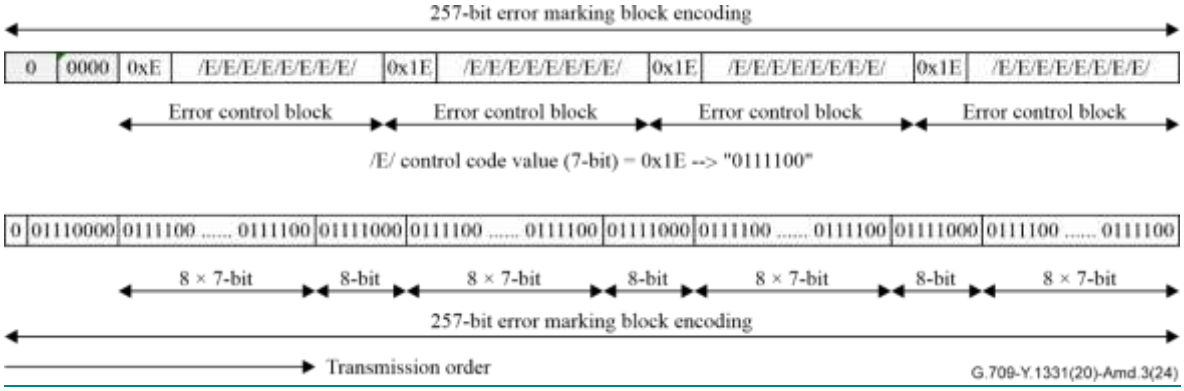


Figure 17-28 – Error marking encoding in transcoded 257-bit block format

Note that the client link status information from the client's terminated alignment markers (AM) fields is processed as specified in Annex K and carried through the client degrade indication (CDI) in the OPUflex overhead.

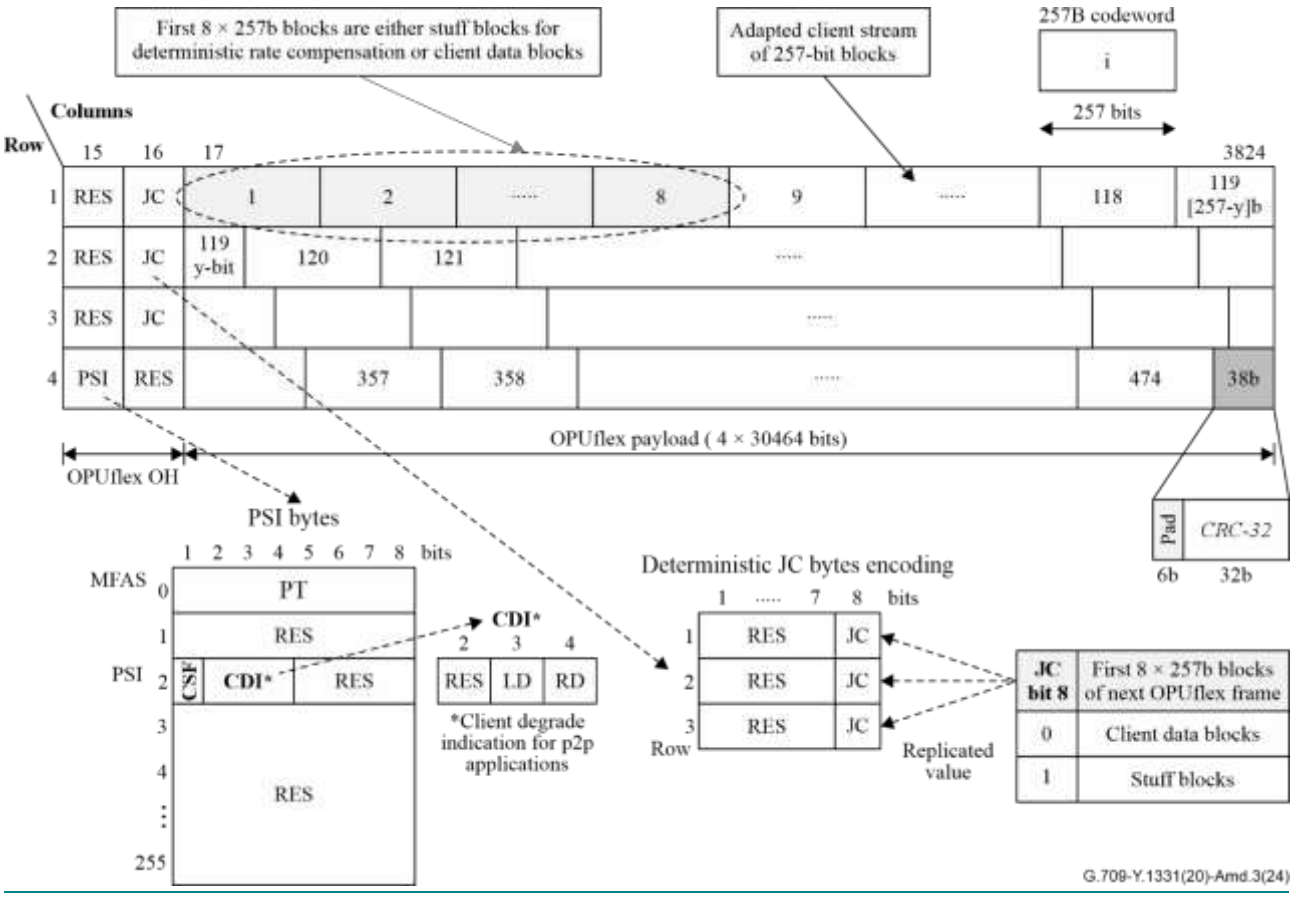


Figure 17-29 – OPUflex frame structure for the mapping of a 256B/257B PCS client signal

Table 17-19 – 256B/257B PCS coded CBR clients

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)
800GBASE-R (Note 1)	2 × 20479/20480 × 401 562 500	±50
NOTE 1 – This signal represents two aligned and unscrambled 256B/257B encoded 400G flows (400G flow –0 and 400G flow– 1), without alignment markers. After 257-bit block round-robin interleaving of the two 400G flows and rate compensation in the mapper, the resulting client nominal bit rate is 803.125 Gbit/s.		

Table 17-20 – Replacement signal for 256B/257B PCS coded CBR clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
800GBASE-R	LF (Note 1)	±50
NOTE 1 – The replacement signal for an 800GBASE-R signal is a continuous stream of LF sequence ordered sets encoded according to Figure 82-5 of [IEEE 802.3] using control block type 0x4B and then 256B/257B transcoded as specified in clause 119.2.4.2 of [IEEE 802.3]. Rate compensation will be performed on the stream of 257-bit blocks in the mapping direction, while 257-bit blocks deinterleaving into two aligned 400G flows and AM insertion will be performed on these two aligned 400G streams of 257-bit blocks in the demapping direction after which these signals are scrambled.		

17.14.1 800GBASE-R

The characteristic information of the adapted and mapped 800GBASE-R client signal consists of an unscrambled sequence of 256B/257B encoded blocks with a nominal bit rate of 803 125 000 kbit/s ±50 ppm.

The mapper shall first recover the 800GBASE-R client stream, which is referred to as the OTN reference signal, that is two 400G unscrambled and aligned streams of 256B/257B encoded blocks without alignment markers as per clause 172.2.4.2 of [IEEE 802.3df], each with a bit rate of (514/544 × 20479/20480 × 425 000 000 kbit/s) ± 50 ppm.

NOTE 1 –The nominal bit rate of the aggregate 800GBASE-R PCS signal consisting of 32 PCS lanes with 256B/257B encoding and [IEEE 802.3df] clause 172 RS FEC at the PMA service interface is 850 000 000 kbit/s.

The aligned two streams of the 800GBASE-R OTN reference signal (two 256B/257B encoded 400G streams, un-scrambled and without alignment markers) are recovered using the process described in Annex J for parallel 256B/257B interfaces. Specifically, the lane(s) of the physical interface are bit-deinterleaved into thirty-two streams of 26 562 500 kbit/s. Lane alignment marker lock is acquired on each PCS lane, followed by lane deskew and reorder, as well as de-interleave into two aligned 400G streams (flow 0 and flow 1) for FEC code word synchronization and error correction. The mapper shall extract the FEC parity bits and alignment markers and perform descrambling of each 400G stream of 257-bit blocks. Uncorrectable FEC code words are optionally replaced with error control blocks in transcoded 257-bit block format per Figure 17-28.

Then, the two 400G streams are 257-bit block round-robin interleaved, rate compensated and mapped into the 257-bit blocks of the OPUflex payload area shown in Figure 17-29, so that the 257-bit blocks of flow 0 are inserted in the odd 257-bit block positions (block positions 2i +1) and the 257-bit blocks of flow 1 are inserted in the even 257-bit block positions (block positions 2i). The 6-bit pad and calculated CRC-32 and are inserted at the end of each OPUflex payload area.

The mapping procedure includes a deterministic rate compensation process inserting eight 257-bit stuff blocks located (when present) at the beginning of some OPUflex payload frames to rate compensate the removal of the alignment markers. The JC OH bytes of OPUflex frame [i] indicates the presence of eight 257b stuff blocks in the next OPUflex frame [i+1] using 1-bit JC value replicated in the three JC bytes as shown in Figure 17-29.

The resulting nominal bit rate of the compensated and padded client stream corresponding to the OPUflex nominal bit rate is [224/237 × 850 000 000 kbit/s] = 238/237 × 800 000 000 kbit/s ± 50 ppm.

In the reverse operation, following termination of the OPUflex signal, the demapper must check the CRC-32 and in case of a detected error, may optionally error mark all 474~~×~~ of the 257-bit blocks in the payload of this OPUflex frame. Then a majority vote (two out of three) of the extracted JC bits in OPUflex frame [i] is used to detect the presence or not of the ~~8×~~eight 257-bit stuff blocks at the beginning of the next OPUflex frame [i+1]; if stuff blocks are detected, they are removed. The remaining 257-bit client data blocks are then deinterleaved into 400G flow 0 (from the 257-bit odd positions in OPUflex payload) and 400G flow 1 (from the 257-bit even positions in OPUflex payload), to retrieve the 800GBASE-R client stream at the OTN reference signal as specified in clause 172.2.4.2 of [IEEE 802.3df]. Then, as described in Annex J, scrambling can be performed and alignment markers and FEC parity bits can be inserted on each 400G flow prior to the 800GBASE-R PCS lane distribution and interleaving into physical lanes at the PMA interface.

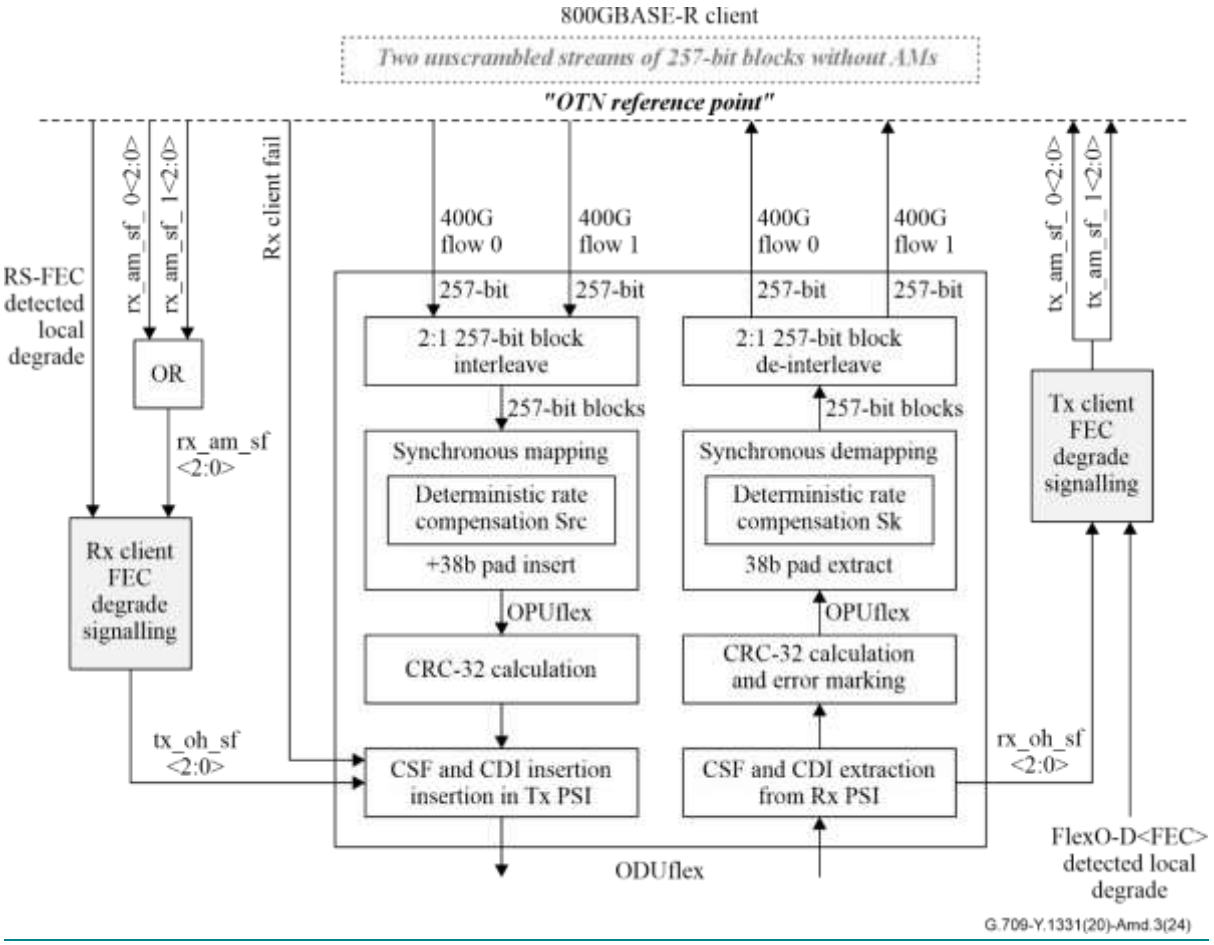


Figure 17-30 – 800GBASE-R client adaptation and mapping to OPUflex

The encoding of the 257-bit stuff block is an all-zero value corresponding to four invalid 64B/66B control blocks that have been 256B/257B transcoded.

NOTE 2 – The value of the deterministic stuff blocks has been chosen so that it does not match any defined set of control block. For each 66b transcoded block, the 8b located at the same position as the block type field (bits 2 to 9) carry the unused value 0x00, still maintaining the 4-bit hamming distance with the defined and used block types; the last 56-bit are all-zeros. At the sink, the deterministic stuff blocks are removed, using simple OPUflex JC overhead bits interpretation based on two out of three majority voting.

The deterministic stuffing (to compensate for AM removal in the ratio of (20k-1)/20k) is inserted as eight 257-bit blocks and can be systematically controlled using simple 17-bit sigma-delta as:

– 800GE_client_with_AM × (20k-1)/20k = 800GE_client_with_AM × (474 - [8×s])/474 → s = 237/80k (Numerator=237, Denominator=81920).

NOTE 3 – 800GE_client_with_AM is the aggregated bit rate of the 256B/257B encoded client before AM removal. It corresponds to the combined bit rate of the two 400G PCS streams of 257-bit blocks after FEC removal and before AM removal. The nominal bit rate of 800GE_client_with_AM = 2 × 514/544 × 425 000 000 kbit/s = 2 × 257/256 × 400 000 000 kbit/s = 803 125 000 kbit/s ± 50 ppm.

The distribution of the deterministic stuffing justification (using 237/81920 stuffing ratio to determine in which OPUflex frame # j to insert 257-bit stuff blocks in the first eight 257-bit locations), could follow the sigma-delta equation below:

– insert eight 257-bit stuff blocks if (j × 237) mod 81920 < 237

– insert eight 257-bit client data blocks if (j × 237) mod 81920 ≥ 237

The index 'j' in the above equations enumerates OPUflex frame in an 81920-frame multiframe sequence. It counts from 1 to 81920.

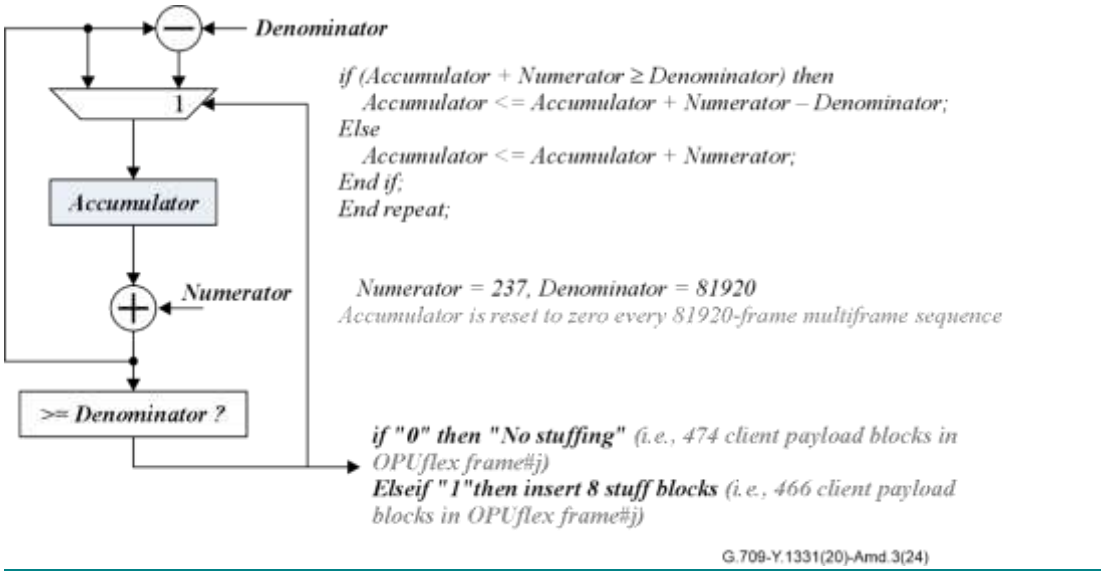


Figure 17-31 – Sigma-delta accumulator for deterministic justification generation

NOTE 4 – The demapping process needs to interpret the extracted OPUflex JC overhead bits to identify the OPUflex frames carrying stuff blocks instead of data blocks in the first eight blocks of the OPUflex payload area; it does not need to check the sigma-delta distribution of the deterministic stuffing ratio.

18 Blank clause

This clause is intentionally left blank.

19 Mapping ODUj signals into the ODTU signal and the ODTU into the OPUk tributary slots

This clause specifies the multiplexing of:

– ODU0 into OPU1, ODU1 into OPU2, ODU1 and ODU2 into OPU3 using client/server specific asynchronous mapping procedures (AMP);

– other ODUj into OPUk using a client agnostic generic mapping procedure (GMP).

This ODUj into OPUk multiplexing is performed in two steps:

1) asynchronous mapping of ODUj into optical data tributary unit (ODTU) using either AMP or GMP;

2) byte-synchronous mapping of ODTU into one or more OPUk tributary slots.

19.1 OPUk tributary slot definition

The OPUk is divided into a number of tributary slots (TS) and these tributary slots are interleaved within the OPUk. A tributary slot includes a part of the OPUk OH area and a part of the OPUk payload area. The bytes of the ODUj frame are mapped into the ODTU payload area and the ODTU bytes are mapped into the OPUk tributary slot or slots. The bytes of the ODTU justification overhead are mapped into the OPUk OH area.

There are two types of tributary slots:

1) Tributary slot with a bandwidth of approximately 2.5 Gbit/s; an OPUk is divided into n tributary slots, numbered 1 to n.

2) Tributary slot with a bandwidth of approximately 1.25 Gbit/s; an OPUk is divided into 2n tributary slots, numbered 1 to 2n.

OPU2 and OPU3 interface ports supporting 1.25 Gbit/s tributary slots must also support the 2.5 Gbit/s tributary slot mode for interworking with interface ports supporting only the 2.5G tributary slot mode (i.e., interface ports compliant with issues of this Recommendation: prior to the definition of 1.25G tributary slots). When operated in 2.5G tributary slot mode, 1.25G tributary slots "i" and "i+n" (i = 1 to n, n = 4 (OPU2) and n = 16 (OPU3)) function as one 2.5G tributary slot.

19.1.1 OPU2 tributary slot allocation

Figure 19-1 presents the OPU2 2.5G tributary slot allocation and the OPU2 1.25G tributary slot allocation. An OPU2 is divided into four 2.5G tributary slots numbered 1 to 4, or in eight 1.25G tributary slots numbered 1 to 8.

– An OPU2 2.5G tributary slot occupies 25% of the OPU2 payload area. It is a structure with 952 columns by 16 (4 × 4) rows (see Figures 19-1 and 19-7) plus a tributary slot overhead (TSOH). The four OPU2 TSs are byte interleaved in the OPU2 payload area and the four OPU2 TSOHs are frame interleaved in the OPU2 overhead area.

– An OPU2 1.25G tributary slot occupies 12.5% of the OPU2 payload area. It is a structure with 476 columns by 32 (8 × 4) rows (see Figures 19-1 and 19-7) plus a tributary slot overhead (TSOH). The eight OPU2 TSs are byte interleaved in the OPU2 payload area and the eight OPU2 TSOHs are frame interleaved in the OPU2 overhead area.

An OPU2 2.5G tributary slot "i" (i = 1,2,3,4) is provided by two OPU2 1.25G tributary slots "i" and "i+4" as illustrated in Figure 19-1.

The tributary slot overhead (TSOH) of OPU2 tributary slots is located in column 16 plus column 15, rows 1, 2 and 3 of the OPU2 frame.

The TSOH for a 2.5G tributary slot is available once every 4 frames. A 4-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 7 and 8 of the MFAS byte as shown in Table 19-1 and Figure 19-1.

The TSOH for a 1.25G tributary slot is available once every 8 frames. An 8-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 6, 7 and 8 of the MFAS byte as shown in Table 19-1 and Figure 19-1.

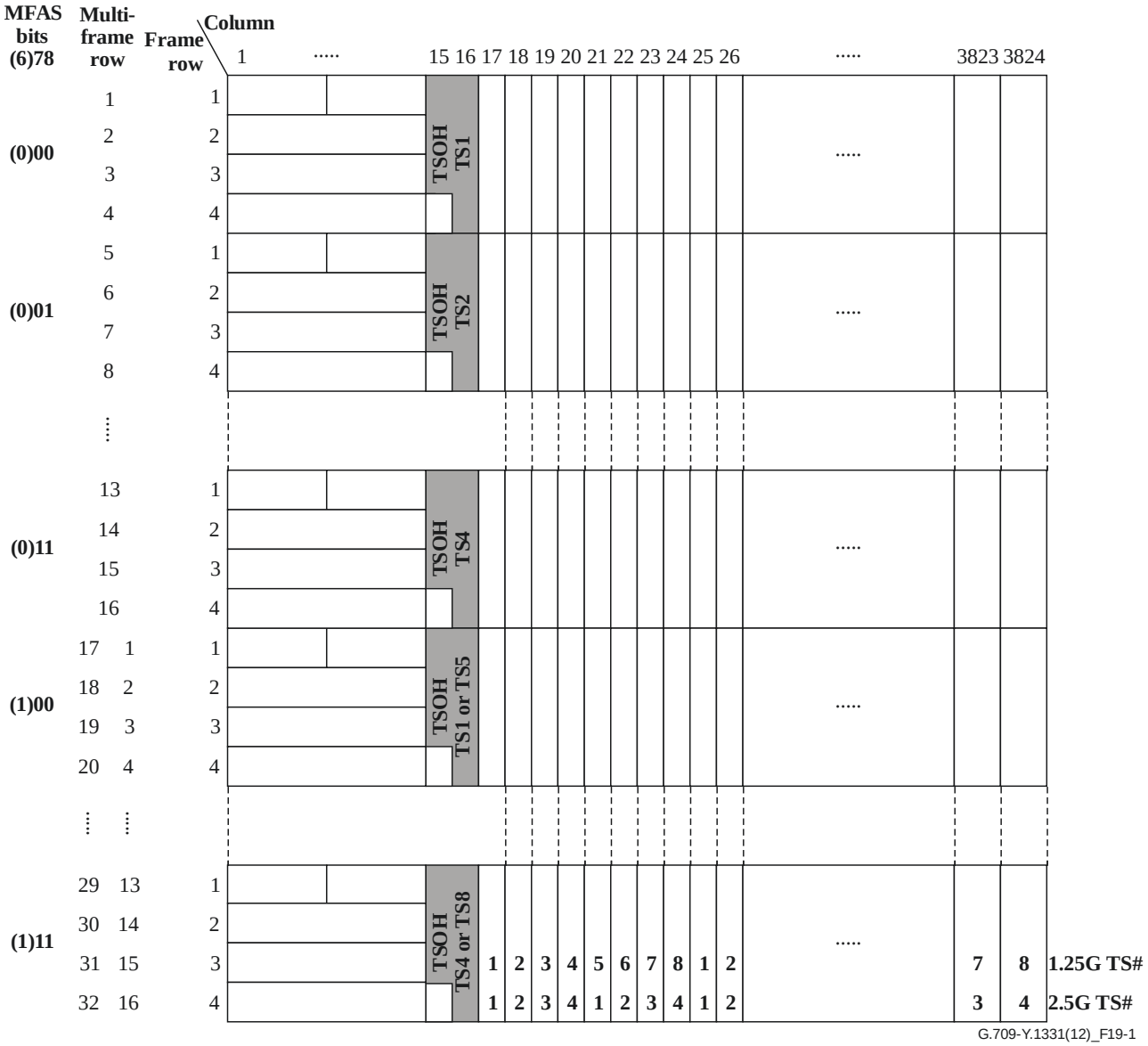


Figure 19-1 – OPU2 tributary slot allocation

Table 19-1 – OPU2 tributary slot OH allocation

MFAS bits 7 8	TSOH 2.5G TS
0 0	1
0 1	2
1 0	3
1 1	4

MFAS bits 6 7 8	TSOH 1.25G TS
0 0 0	1
0 0 1	2
0 1 0	3
0 1 1	4
1 0 0	5
1 0 1	6
1 1 0	7
1 1 1	8

19.1.2 OPU3 tributary slot allocation

Figure 19-2 presents the OPU3 2.5G tributary slot allocation and the OPU3 1.25G tributary slot allocation. An OPU3 is divided into sixteen 2.5G tributary slots numbered 1 to 16, or in thirty-two 1.25G tributary slots numbered 1 to 32.

– An OPU3 2.5G tributary slot occupies 6.25% of the OPU3 payload area. It is a structure with 238 columns by 64 (16 × 4) rows (see Figures 19-2 and 19-8) plus a tributary slot overhead (TSOH). The sixteen OPU3 2.5G TSs are byte interleaved in the OPU3 payload area and the sixteen OPU3 TSOHs are frame interleaved in the OPU3 overhead area.

– An OPU3 1.25G tributary slot occupies 3.125% of the OPU3 payload area. It is a structure with 119 columns by 128 (32 × 4) rows (see Figures 19-2 and 19-8) plus a tributary slot overhead (TSOH). The thirty-two OPU3 1.25G TSs are byte interleaved in the OPU3 payload area and the thirty-two OPU3 TSOHs are frame interleaved in the OPU3 overhead area.

An OPU3 2.5G tributary slot "i" (i = 1,2,..16) is provided by two OPU3 1.25G tributary slots "i" and "i+16" as illustrated in Figure 19-2.

The tributary slot overhead (TSOH) of OPU3 tributary slots is located in column 16 plus column 15, rows 1, 2 and 3 of the OPU3 frame.

The TSOH for a 2.5G tributary slot is available once every 16 frames. A 16-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 5, 6, 7 and 8 of the MFAS byte as shown in Table 19-2 and Figure 19-2.

The TSOH for a 1.25G tributary slot is available once every 32 frames. A 32-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 4, 5, 6, 7 and 8 of the MFAS byte as shown in Table 19-2 and Figure 19-2.

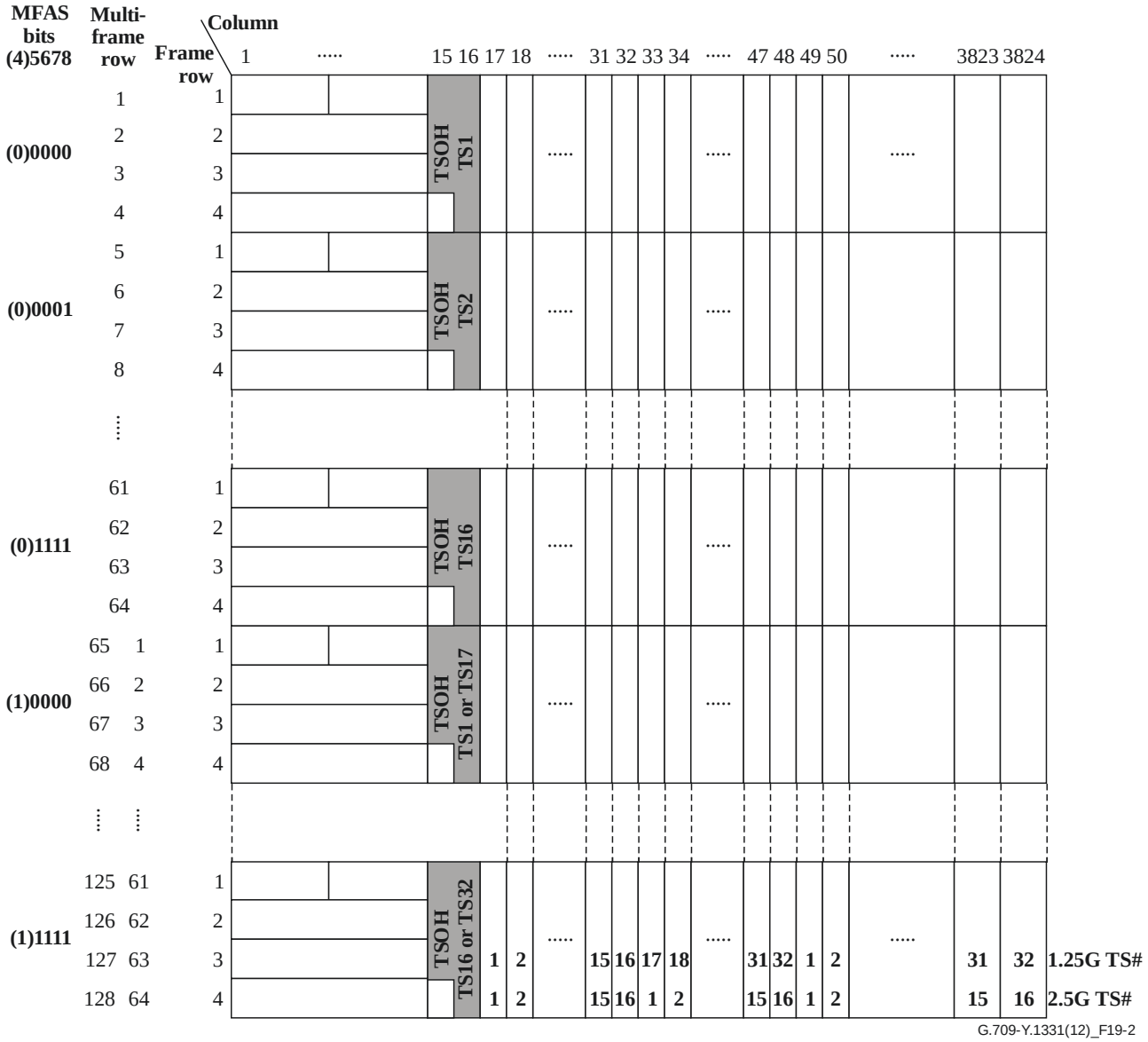


Figure 19-2 – OPU3 tributary slot allocation

Table 19-2 – OPU3 tributary slot OH allocation

MFAS bits 5 6 7 8	TSOH 2.5G TS
0 0 0 0	1
0 0 0 1	2
0 0 1 0	3
0 0 1 1	4
0 1 0 0	5
0 1 0 1	6
0 1 1 0	7
0 1 1 1	8
1 0 0 0	9
1 0 0 1	10
1 0 1 0	11
1 0 1 1	12
1 1 0 0	13
1 1 0 1	14
1 1 1 0	15
1 1 1 1	16

MFAS bits 4 5 6 7 8	TSOH 1.25G TS
0 0 0 0 0	1
0 0 0 0 1	2
0 0 0 1 0	3
0 0 0 1 1	4
0 0 1 0 0	5
0 0 1 0 1	6
0 0 1 1 0	7
0 0 1 1 1	8
0 1 0 0 0	9
0 1 0 0 1	10
0 1 0 1 0	11
0 1 0 1 1	12
0 1 1 0 0	13
0 1 1 0 1	14
0 1 1 1 0	15
0 1 1 1 1	16

MFAS bits 4 5 6 7 8	TSOH 1.25G TS
1 0 0 0 0	17
1 0 0 0 1	18
1 0 0 1 0	19
1 0 0 1 1	20
1 0 1 0 0	21
1 0 1 0 1	22
1 0 1 1 0	23
1 0 1 1 1	24
1 1 0 0 0	25
1 1 0 0 1	26
1 1 0 1 0	27
1 1 0 1 1	28
1 1 1 0 0	29
1 1 1 0 1	30
1 1 1 1 0	31
1 1 1 1 1	32

19.1.3 OPU1 tributary slot allocation

Figure 19-3 presents the OPU1 1.25G tributary slot allocation. An OPU1 is divided into two 1.25G tributary slots numbered 1 to 2.

– An OPU1 1.25G tributary slot occupies 50% of the OPU1 payload area. It is a structure with 1904 columns by 8 (2 × 4) rows (see Figure 19-3) plus a tributary slot overhead (TSOH). The two OPU1 1.25G TSs are byte interleaved in the OPU1 payload area and the two OPU1 TSOHs are frame interleaved in the OPU1 overhead area.

The tributary slot overhead (TSOH) of OPU1 tributary slots is located in column 16 plus column 15, rows 1, 2 and 3 of the OPU1 frame.

The TSOH for a 1.25G tributary slot is available once every 2 frames. A 2-frame multiframe structure is used for this assignment. This multiframe structure is locked to bit 8 of the MFAS byte as shown in Table 19-3 and Figure 19-3.

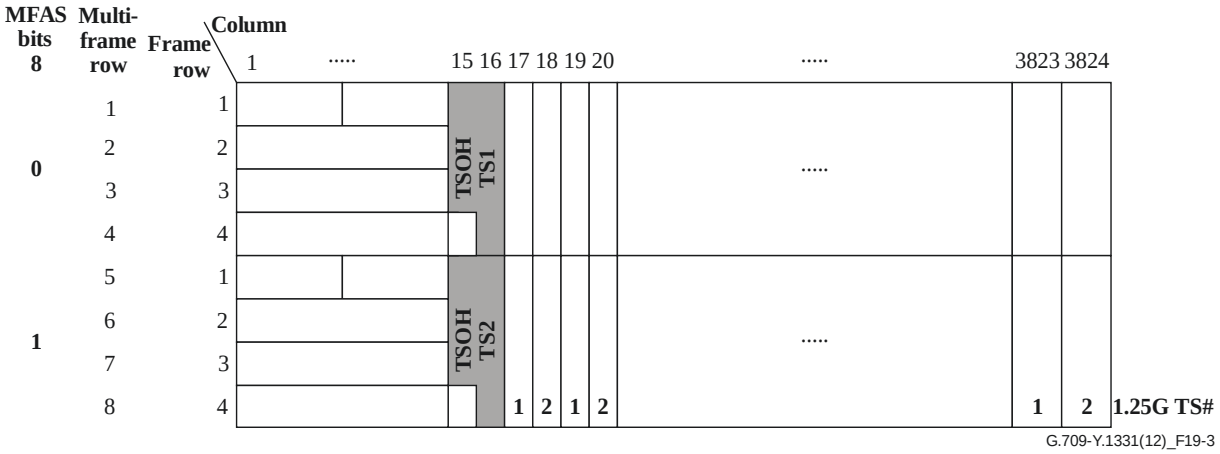


Figure 19-3 – OPU1 tributary slot allocation

Table 19-3 – OPU1 tributary slot OH allocation

MFAS bit 8	TSOH 1.25G TS
0	1
1	2

19.1.4 OPU4 tributary slot allocation

Figures 19-4A and 19-4B present the OPU4 1.25G tributary slot allocation. An OPU4 is divided into eighty 1.25G tributary slots (numbered 1 to 80), which are located in columns 17 to 3816, and 8 columns of fixed stuff located in columns 3817 to 3824. The OPU4 frame may be represented in a 320 row by 3810 column format (Figure 19-4A) and in a 160 row by 7620 column format (Figure 19-4B).

– An OPU4 1.25G tributary slot occupies 1.247% of the OPU4 payload area. It is a structure with 95 columns by 160 (80 × 4/2) rows (see Figure 19-4B) plus a tributary slot overhead (TSOH). The eighty OPU4 1.25G TSs are byte interleaved in the OPU4 payload area and the eighty OPU4 TSOHs are frame interleaved in the OPU4 overhead area.

The tributary slot overhead (TSOH) of OPU4 tributary slots is located in rows 1 to 3, columns 15 and 16 of the OPU4 frame.

The TSOH for a 1.25G tributary slot is available once every 80 frames. An 80-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 2, 3, 4, 5, 6, 7 and 8 of the OMFI byte as shown in Table 19-4.

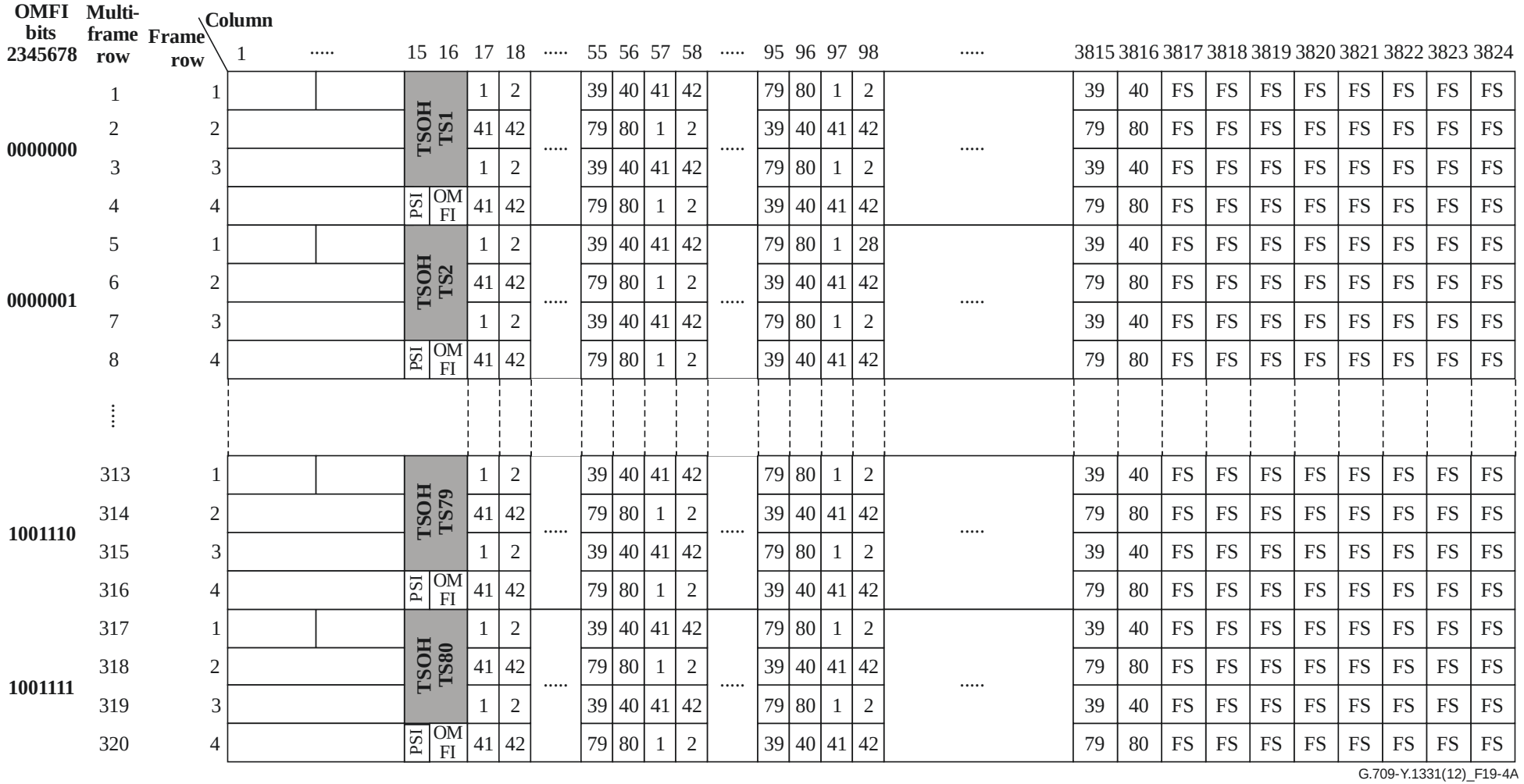


Figure 19-4A – OPU4 1.25G tributary slot allocation

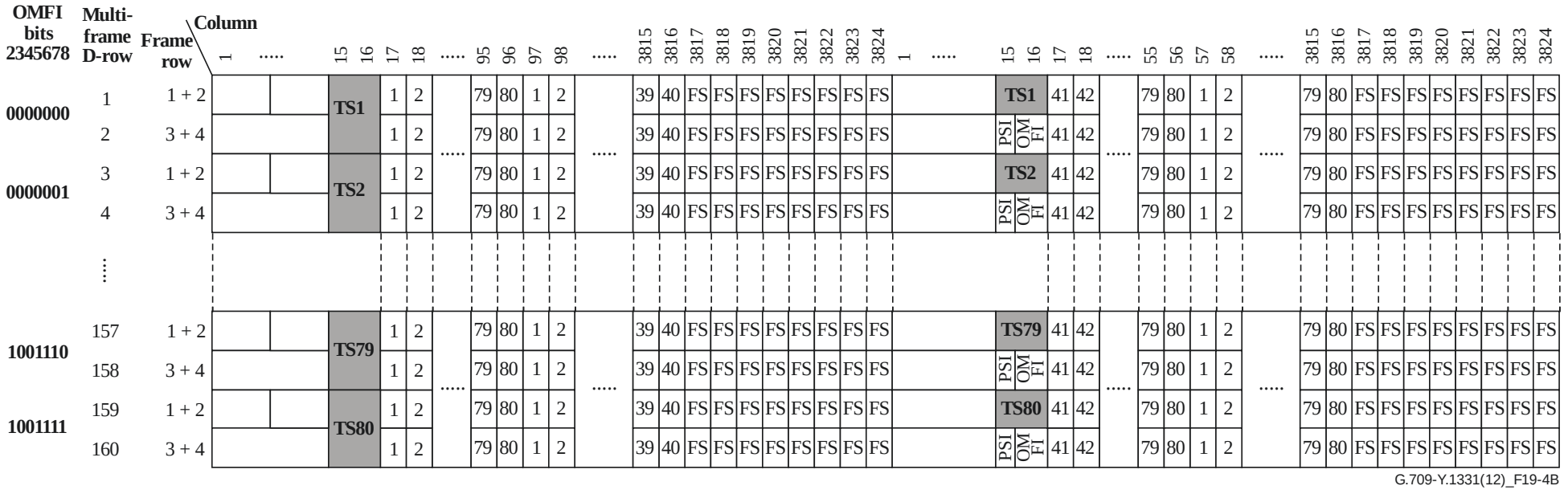


Figure 19-4B – OPU4 tributary slots in 160 row x 7620 column format

Table 19-4 – OPU4 tributary slot OH allocation

OMFI bits 2 3 4 5 6 7 8	TSOH 1.25G TS	OMFI bits 2 3 4 5 6 7 8	TSOH 1.25G TS	OMFI bits 2 3 4 5 6 7 8	TSOH 1.25G TS	OMFI bits 2 3 4 5 6 7 8	TSOH 1.25G TS
0 0 0 0 0 0 0	1	0 0 1 0 1 0 0	21	0 1 0 1 0 0 0	41	0 1 1 1 1 0 0	61
0 0 0 0 0 0 1	2	0 0 1 0 1 0 1	22	0 1 0 1 0 0 1	42	0 1 1 1 1 0 1	62
0 0 0 0 0 1 0	3	0 0 1 0 1 1 0	23	0 1 0 1 0 1 0	43	0 1 1 1 1 1 0	63
0 0 0 0 0 1 1	4	0 0 1 0 1 1 1	24	0 1 0 1 0 1 1	44	0 1 1 1 1 1 1	64
0 0 0 0 1 0 0	5	0 0 1 1 0 0 0	25	0 1 0 1 1 0 0	45	1 0 0 0 0 0 0	65
0 0 0 0 1 0 1	6	0 0 1 1 0 0 1	26	0 1 0 1 1 0 1	46	1 0 0 0 0 0 1	66
0 0 0 0 1 1 0	7	0 0 1 1 0 1 0	27	0 1 0 1 1 1 0	47	1 0 0 0 0 1 0	67
0 0 0 0 1 1 1	8	0 0 1 1 0 1 1	28	0 1 0 1 1 1 1	48	1 0 0 0 0 1 1	68
0 0 0 1 0 0 0	9	0 0 1 1 1 0 0	29	0 1 1 0 0 0 0	49	1 0 0 0 1 0 0	69
0 0 0 1 0 0 1	10	0 0 1 1 1 0 1	30	0 1 1 0 0 0 1	50	1 0 0 0 1 0 1	70
0 0 0 1 0 1 0	11	0 0 1 1 1 1 0	31	0 1 1 0 0 1 0	51	1 0 0 0 1 1 0	71
0 0 0 1 0 1 1	12	0 0 1 1 1 1 1	32	0 1 1 0 0 1 1	52	1 0 0 0 1 1 1	72
0 0 0 1 1 0 0	13	0 1 0 0 0 0 0	33	0 1 1 0 1 0 0	53	1 0 0 1 0 0 0	73
0 0 0 1 1 0 1	14	0 1 0 0 0 0 1	34	0 1 1 0 1 0 1	54	1 0 0 1 0 0 1	74
0 0 0 1 1 1 0	15	0 1 0 0 0 1 0	35	0 1 1 0 1 1 0	55	1 0 0 1 0 1 0	75
0 0 0 1 1 1 1	16	0 1 0 0 0 1 1	36	0 1 1 0 1 1 1	56	1 0 0 1 0 1 1	76
0 0 1 0 0 0 0	17	0 1 0 0 1 0 0	37	0 1 1 1 0 0 0	57	1 0 0 1 1 0 0	77
0 0 1 0 0 0 1	18	0 1 0 0 1 0 1	38	0 1 1 1 0 0 1	58	1 0 0 1 1 0 1	78
0 0 1 0 0 1 0	19	0 1 0 0 1 1 0	39	0 1 1 1 0 1 0	59	1 0 0 1 1 1 0	79
0 0 1 0 0 1 1	20	0 1 0 0 1 1 1	40	0 1 1 1 0 1 1	60	1 0 0 1 1 1 1	80

19.1.5 OPU25 tributary slot allocation

Figures 19-5A and 19-5B present the OPU25 1.25G tributary slot allocation. An OPU25 is divided into twenty 1.25G tributary slots (numbered 1 to 20), which are located in columns 17 to 3824. The OPU25 frame may be represented in an 80 row by 3810 column format (Figure 19-5A) and in a 16 row by 19050 column format (Figure 19-5B).

– An OPU25 1.25G tributary slot occupies 5% of the OPU25 payload area. It is a structure with 952 columns by 16 ($20 \times 4/5$) rows (see Figure 19-5B) plus a tributary slot overhead (TSOH). The twenty OPU25 1.25G TSs are byte interleaved in the OPU25 payload area and the twenty OPU25 TSOHs are frame interleaved in the OPU25 overhead area.

The tributary slot overhead (TSOH) of OPU25 tributary slots is located in rows 1 to 3, columns 15 and 16 of the OPU25 frame.

The TSOH for a 1.25G tributary slot is available once every 20 frames. A 20-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 4, 5, 6, 7 and 8 of the OMFI byte as shown in Table 19-5.

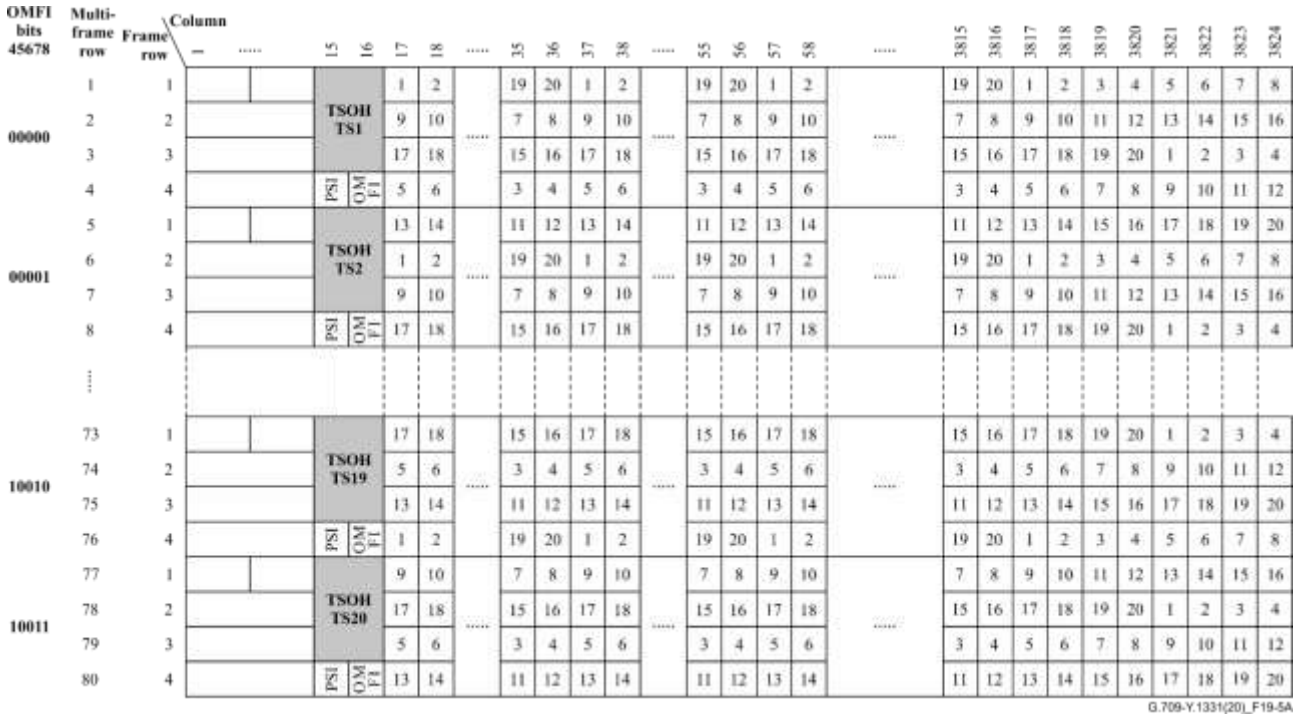


Figure 19-5A – OPU25 1.25G tributary slot allocation

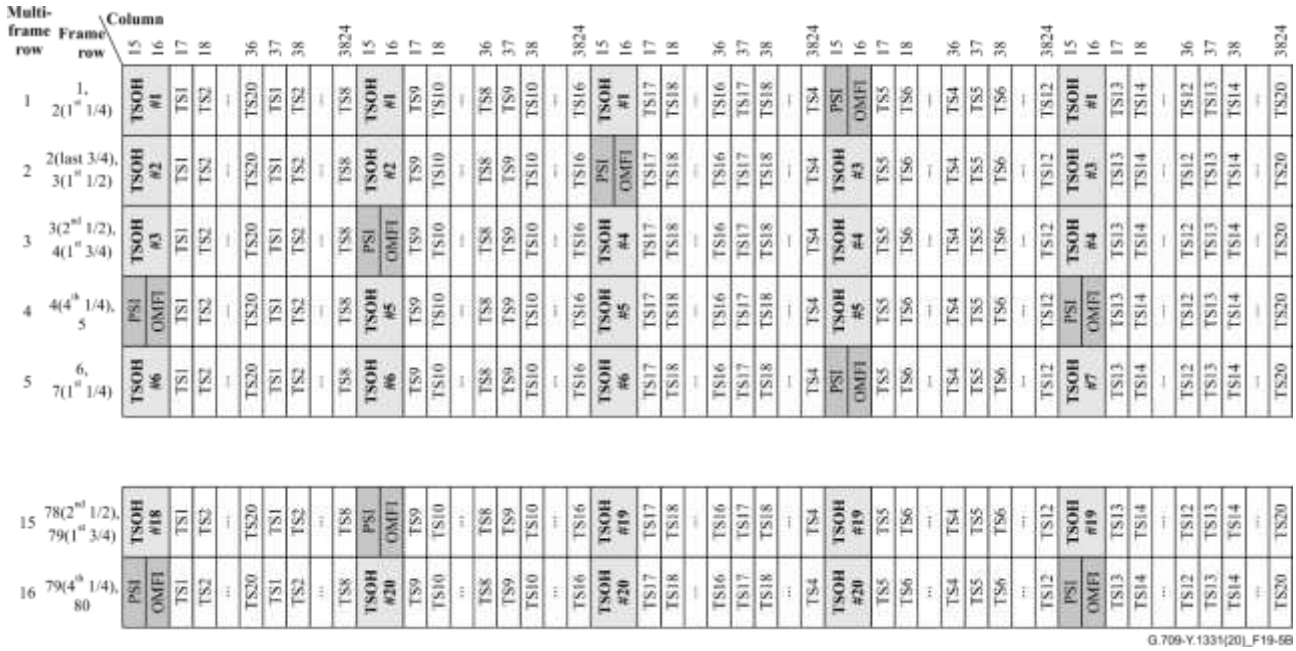


Figure 19-5B – OPU25 tributary slots in 16 row x 19050 column format

Table 19-5 – OPU25 tributary slot OH allocation

OMFI bits 4 5 6 7 8	TSOH 1.25G TS
0 0 0 0 0	1
0 0 0 0 1	2
0 0 0 1 0	3
0 0 0 1 1	4
0 0 1 0 0	5
0 0 1 0 1	6
0 0 1 1 0	7
0 0 1 1 1	8
0 1 0 0 0	9
0 1 0 0 1	10
0 1 0 1 0	11
0 1 0 1 1	12
0 1 1 0 0	13
0 1 1 0 1	14
0 1 1 1 0	15
0 1 1 1 1	16
1 0 0 0 0	17
1 0 0 0 1	18
1 0 0 1 0	19
1 0 0 1 1	20

19.1.6 OPU50 tributary slot allocation

Figures 19-6A and 19-6B present the OPU50 1.25G tributary slot allocation. An OPU50 is divided into forty 1.25G tributary slots (numbered 1 to 40), which are located in columns 17 to 3824. The OPU50 frame may be represented in a 160 row by 3810 column format (Figure 19-6A) and in a 32 row by 19050 column format (Figure 19-6B).

– An OPU25 1.25G tributary slot occupies 2.5% of the OPU50 payload area. It is a structure with 476 columns by 32 (40 × 4/5) rows (see Figure 19-6B) plus a tributary slot overhead (TSOH). The forty OPU50 1.25G TSs are byte interleaved in the OPU50 payload area and the forty OPU25 TSOHs are frame interleaved in the OPU25 overhead area.

The tributary slot overhead (TSOH) of OPU50 tributary slots is located in rows 1 to 3, columns 15 and 16 of the OPU25 frame.

The TSOH for a 1.25G tributary slot is available once every 40 frames. A 40-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 3, 4, 5, 6, 7 and 8 of the OMFI byte as shown in Table 19-6.

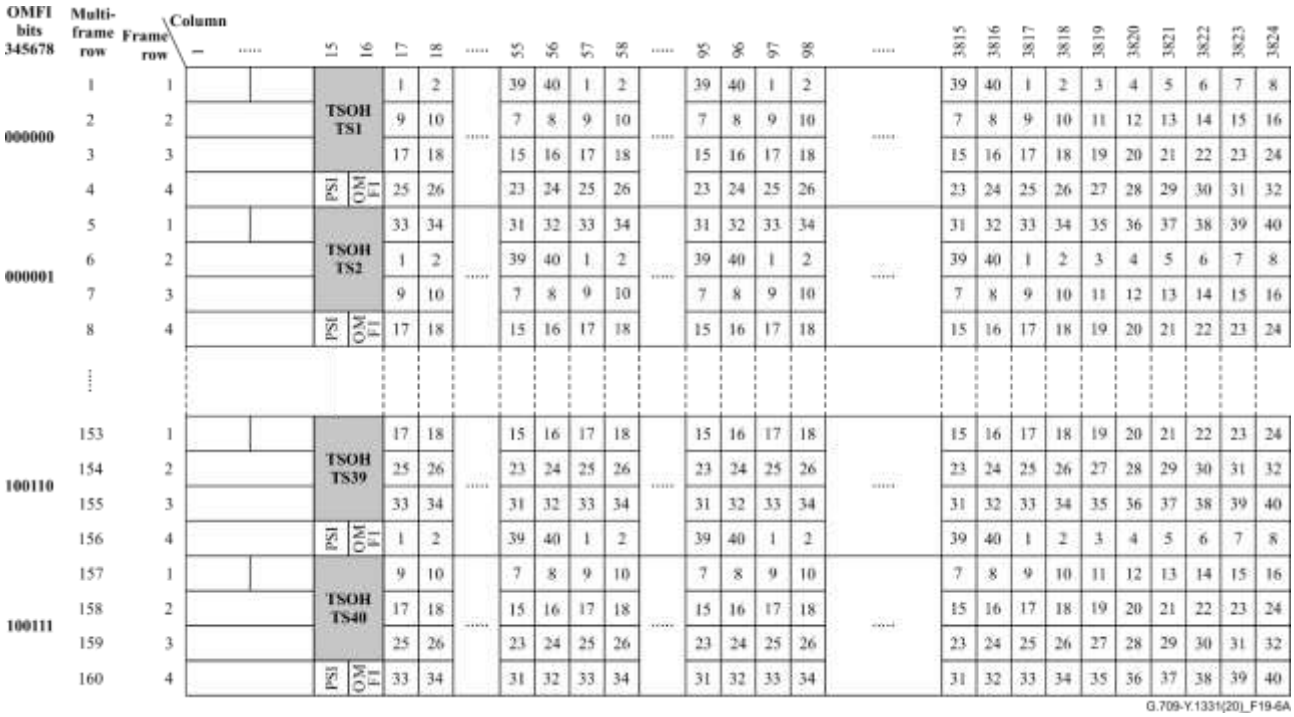


Figure 19-6A – OPU50 1.25G tributary slot allocation

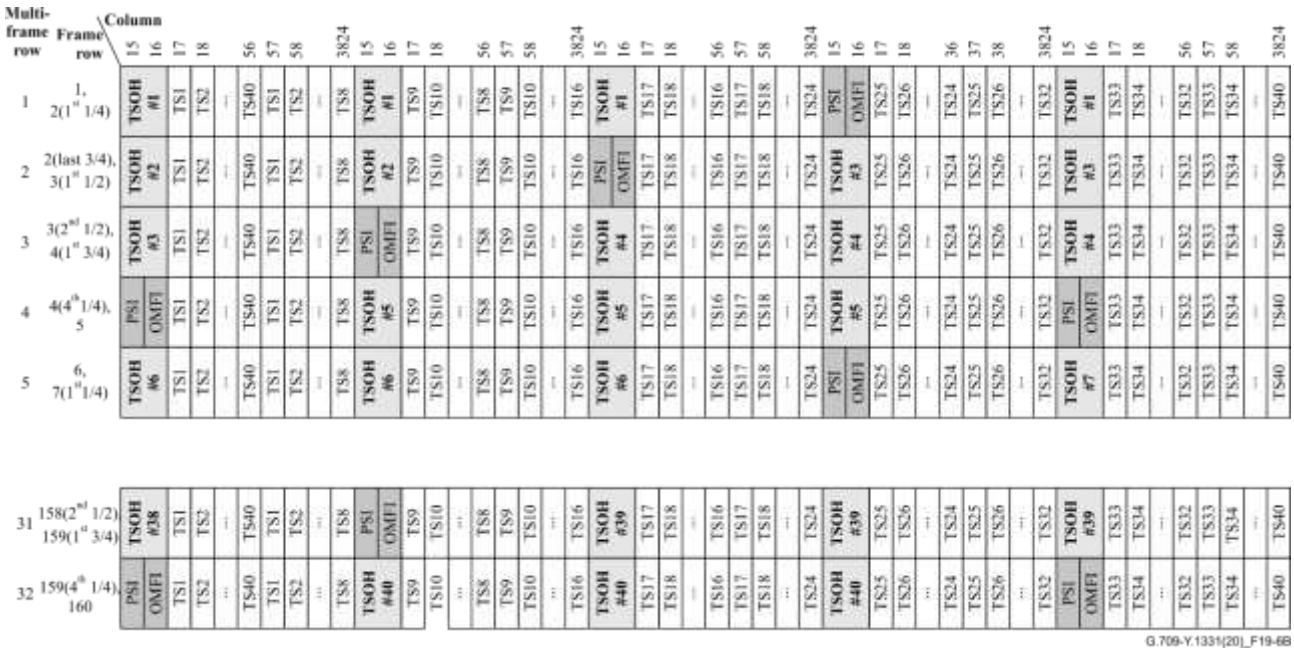


Figure 19-6B – OPU50 tributary slots in 32 row x 19050 column format

Table 19-6 – OPU50 tributary slot OH allocation

OMFI bits 3 4 5 6 7 8	TSOH 1.25G TS	OMFI bits 3 4 5 6 7 8	TSOH 1.25G TS
0 0 0 0 0 0	1	0 1 0 1 0 0	21
0 0 0 0 0 1	2	0 1 0 1 0 1	22
0 0 0 0 1 0	3	0 1 0 1 1 0	23
0 0 0 0 1 1	4	0 1 0 1 1 1	24
0 0 0 1 0 0	5	0 1 1 0 0 0	25
0 0 0 1 0 1	6	0 1 1 0 0 1	26
0 0 0 1 1 0	7	0 1 1 0 1 0	27
0 0 0 1 1 1	8	0 1 1 0 1 1	28
0 0 1 0 0 0	9	0 1 1 1 0 0	29
0 0 1 0 0 1	10	0 1 1 1 0 1	30
0 0 1 0 1 0	11	0 1 1 1 1 0	31
0 0 1 0 1 1	12	0 1 1 1 1 1	32
0 0 1 1 0 0	13	1 0 0 0 0 0	33
0 0 1 1 0 1	14	1 0 0 0 0 1	34
0 0 1 1 1 0	15	1 0 0 0 1 0	35
0 0 1 1 1 1	16	1 0 0 0 1 1	36
0 1 0 0 0 0	17	1 0 0 1 0 0	37
0 1 0 0 0 1	18	1 0 0 1 0 1	38
0 1 0 0 1 0	19	1 0 0 1 1 0	39
0 1 0 0 1 1	20	1 0 0 1 1 1	40

19.2 ODTU definition

The optical data tributary unit (ODTU) carries a justified ODU signal. There are two types of ODTUs:

1) ODTUjk ((j,k) = {(0,1), (1,2), (1,3), (2,3)}; ODTU01, ODTU12, ODTU13 and ODTU23) in which an ODUj signal is mapped via the asynchronous mapping procedure (AMP) as defined in clause 19.5;

2) ODTUk.ts ((k,ts) = (2,1..8), (3,1..32), (4,1..80), (25,1..20), (50,1..40)) in which a ODUj (ODU0, ODU1, ODU2, ODU2e, ODU3, ODUflex) signal is mapped via the generic mapping procedure (GMP) defined in clause 19.6.

Optical data tributary unit jk

The optical data tributary unit jk (ODTUjk) is a structure which consists of an ODTUjk payload area and an ODTUjk overhead area (Figure 19-7). The ODTUjk payload area has c columns and r rows (see Table 19-7) and the ODTUjk overhead area has "ts" times 4 bytes, of which "ts" times 1 byte can carry payload. The ODTUjk is carried in "ts" 1.25G or 2.5G tributary slots of an OPUk.

The location of the ODTUjk overhead depends on the OPUk tributary slot(s) used when multiplexing the ODTUjk in the OPUk (see clauses 19.1.1, 19.1.2, 19.1.3). The ts instances of the ODTUjk overhead might not be equally distributed.

The ODTUjk overhead carries the AMP justification overhead as specified in clause 19.4.

NOTE – The 1.25G and 2.5G tributary slot versions of an ODTU12 are identical when the two 1.25G tributary slots carrying the ODTU12 are TSa and TSa+4. The 1.25G and 2.5G tributary slot versions of an ODTU13 are identical when the two 1.25G tributary slots carrying the ODTU12 are TSa and TSa+16. The 1.25G and 2.5G tributary slot versions of an ODTU23 are identical when the eight 1.25G tributary slots carrying the ODTU23 are TSa, TSb, TSc, TSd, TSa+16, TSb+16, TSc+16 and TSd+16.

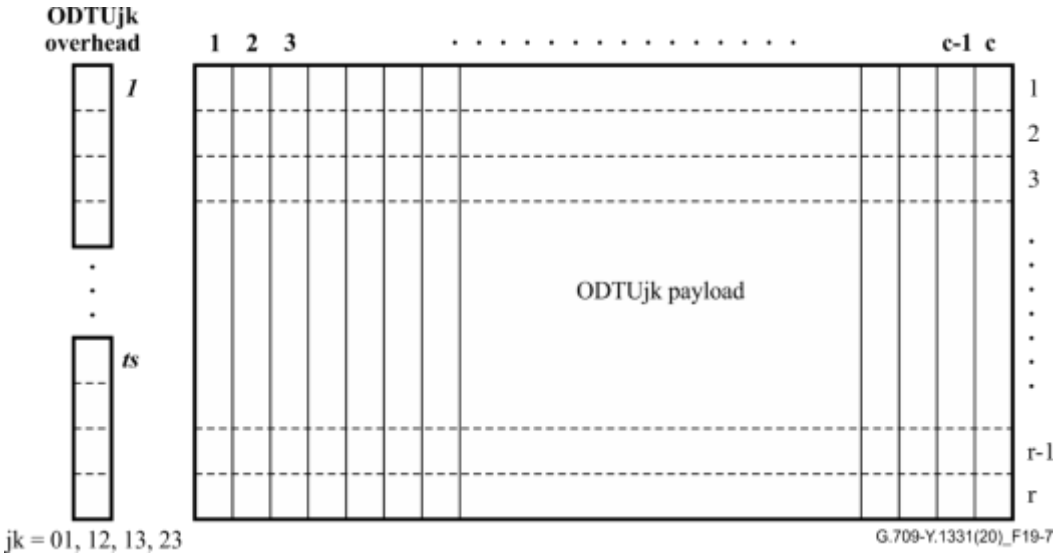


Figure 19-7 – ODTUjk frame formats

Table 19-7 – ODTUjk characteristics for 2.5G and 1.25G tributary slots

2.5G TS	c	r	ts	ODTUjk payload bytes	ODTUjk overhead bytes
ODTU12	952	16	1	15232	1 x 4
ODTU13	238	64	1	15232	1 x 4
ODTU23	952	64	4	60928	4 x 4

1.25G TS	c	r	ts	ODTUjk payload bytes	ODTUjk overhead bytes
ODTU01	1904	8	1	15232	1 × 4
ODTU12	952	32	2	30464	2 × 4
ODTU13	238	128	2	30464	2 × 4
ODTU23	952	128	8	121856	8 × 4

Optical data tributary unit k.ts

The optical data tributary unit k.ts (ODTUk.ts) is a structure which consists of an ODTUk.ts payload area and an ODTUk.ts overhead area (Figure 19-8). The ODTUk.ts payload area has j x ts columns and r rows (see Table 19-8) and the ODTUk.ts overhead area has one times 6 bytes. The ODTUk.ts is carried in "ts" 1.25G tributary slots of an OPUk.

The location of the ODTUk.ts overhead depends on the OPUk tributary slot used when multiplexing the ODTUk.ts in the OPUk (see clauses 19.1.1, 19.1.2, 19.1.4, 19.1.5, 19.1.6). The single instance of an ODTUk.ts overhead is located in the OPUk TSOH of the last OPUk tributary slot allocated to the ODTUk.ts.

The ODTUk.ts overhead carries the GMP justification overhead as specified in clause 19.4.

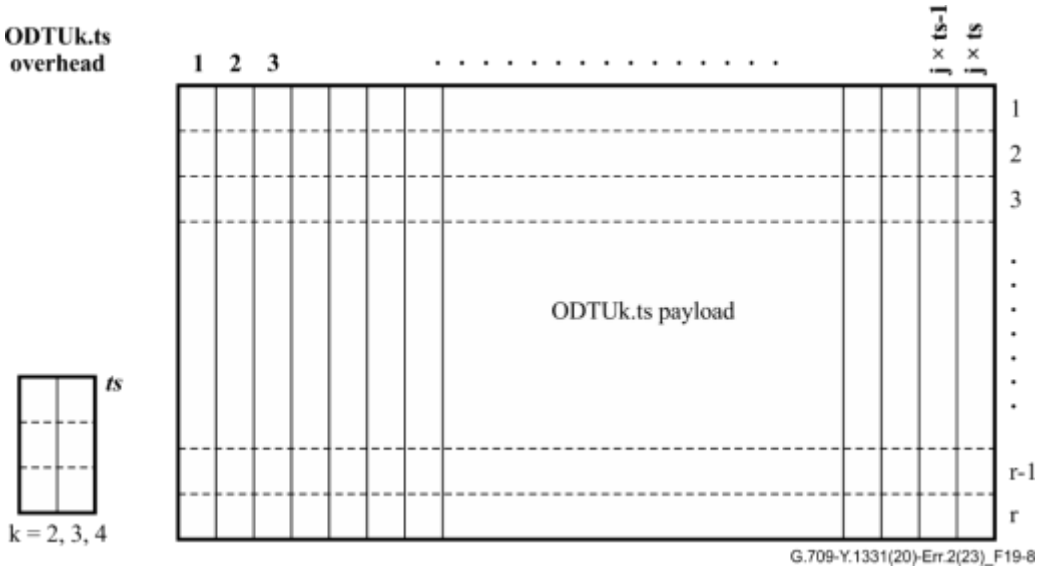


Figure 19-8 – ODTUk.ts frame formats

Table 19-8 – ODTUk.ts characteristics

	j	r	ts	ODTUk.ts payload bytes	ODTUk.ts overhead bytes
ODTU2.ts	476	32	1 to 8	15232 × ts	1 × 6
ODTU3.ts	119	128	1 to 32	15232 × ts	1 × 6
ODTU4.ts	95	160	1 to 80	15200 × ts	1 × 6
ODTU25.ts	952	16	1 to 20	15232 × ts	1 × 6
ODTU50.ts	476	32	1 to 40	15232 × ts	1 × 6

19.3 Multiplexing ODTU signals into the OPUk

Multiplexing an ODTU01 signal into an OPU1 is realized by mapping the ODTU01 signal in one of the two OPU1 1.25G tributary slots.

Multiplexing an ODTU12 signal into an OPU2 is realized by mapping the ODTU12 signal in one of the four OPU2 2.5G tributary slots or in two (of the eight) arbitrary OPU2 1.25G tributary slots: OPU2 TSa and TSb with $1 \le a < b \le 8$.

Multiplexing an ODTU13 signal into an OPU3 is realized by mapping the ODTU13 signal in one of the sixteen OPU3 2.5G tributary slots or in two (of the thirty-two) arbitrary OPU3 1.25G tributary slots: OPU3 TSa and TSb with $1 \le a < b \le 32$.

Multiplexing an ODTU23 signal into an OPU3 is realized by mapping the ODTU23 signal in four (of the sixteen) arbitrary OPU3 2.5G tributary slots: OPU3 TSa, TSb, TSc and TSd with $1 \le a < b < c < d \le 16$ or in eight (of the thirty-two) arbitrary OPU3 1.25G tributary slots: OPU3 TSa, TSb, TSc, TSd, TSe, TSf, TSg and TSh with $1 \le a < b < c < d < e < f < g < h \le 32$.

NOTE – a, b, c, d, e, f, g and h do not have to be sequential (a = i, b = i+1, c = i+2, d = i+3, e=i+4, f=i+5, g=i+6, h=i+7); the values can be arbitrarily selected to prevent bandwidth fragmentation.

Multiplexing an ODTU2.ts signal into an OPU2 is realized by mapping the ODTU2.ts signal in ts (of the eight) arbitrary OPU2 1.25G tributary slots: OPU2 TSa, TSb, .. , TSp with $1 \le a < b < .. < p \le 8$.

Multiplexing an ODTU3.ts signal into an OPU3 is realized by mapping the ODTU3.ts signal in ts (of the thirty-two) arbitrary OPU3 1.25G tributary slots: OPU3 TSa, TSb, .. , TSq with $1 \le a < b < .. < q \le 32$.

Multiplexing an ODTU4.ts signal into an OPU4 is realized by mapping the ODTU4.ts signal in ts (of the eighty) arbitrary OPU4 1.25G tributary slots: OPU4 TSa, TSb, .. , TSr with $1 \le a < b < .. < r \le 80$.

Multiplexing an ODTU25.ts signal into an OPU25 is realized by mapping the ODTU25.ts signal in ts (of the twenty) arbitrary OPU25 1.25G tributary slots: OPU25 TSa, TSb, .. , TSx with $1 \le a < b < .. < x \le 20$.

Multiplexing an ODTU50.ts signal into an OPU50 is realized by mapping the ODTU50.ts signal in ts (of the forty) arbitrary OPU50 1.25G tributary slots: OPU25 TSa, TSb, .. , TSx with $1 \le a < b < .. < x \le 40$.

The OPUk overhead for these multiplexed signals consists of a payload type (PT), the multiplex structure identifier (MSI), the OPU4 multiframe identifier (k=4), the OPUk tributary slot overhead carrying the ODTU overhead and depending on the ODTU type one or more bytes reserved for future international standardization.

19.3.1 ODTU12 mapping into one OPU2 tributary slot

A byte of the ODTU12 payload signal is mapped into a byte of an OPU2 2.5G TS #i (i = 1,2,3,4) payload area, as indicated in Figure 19-9 (left). A byte of the ODTU12 overhead is mapped into a TSOH byte within column 16 of the OPU2 2.5G TS #i.

A byte of the ODTU12 signal is mapped into a byte of one of two OPU2 1.25G TS #A,B (A,B = 1,2,..,8) payload areas, as indicated in Figure 19-9 (right). A byte of the ODTU12 overhead is mapped into a TSOH byte within column 16 of the OPU2 1.25G TS #a,b.

The remaining OPU2 TSOH bytes in column 15 are reserved for future international standardization.

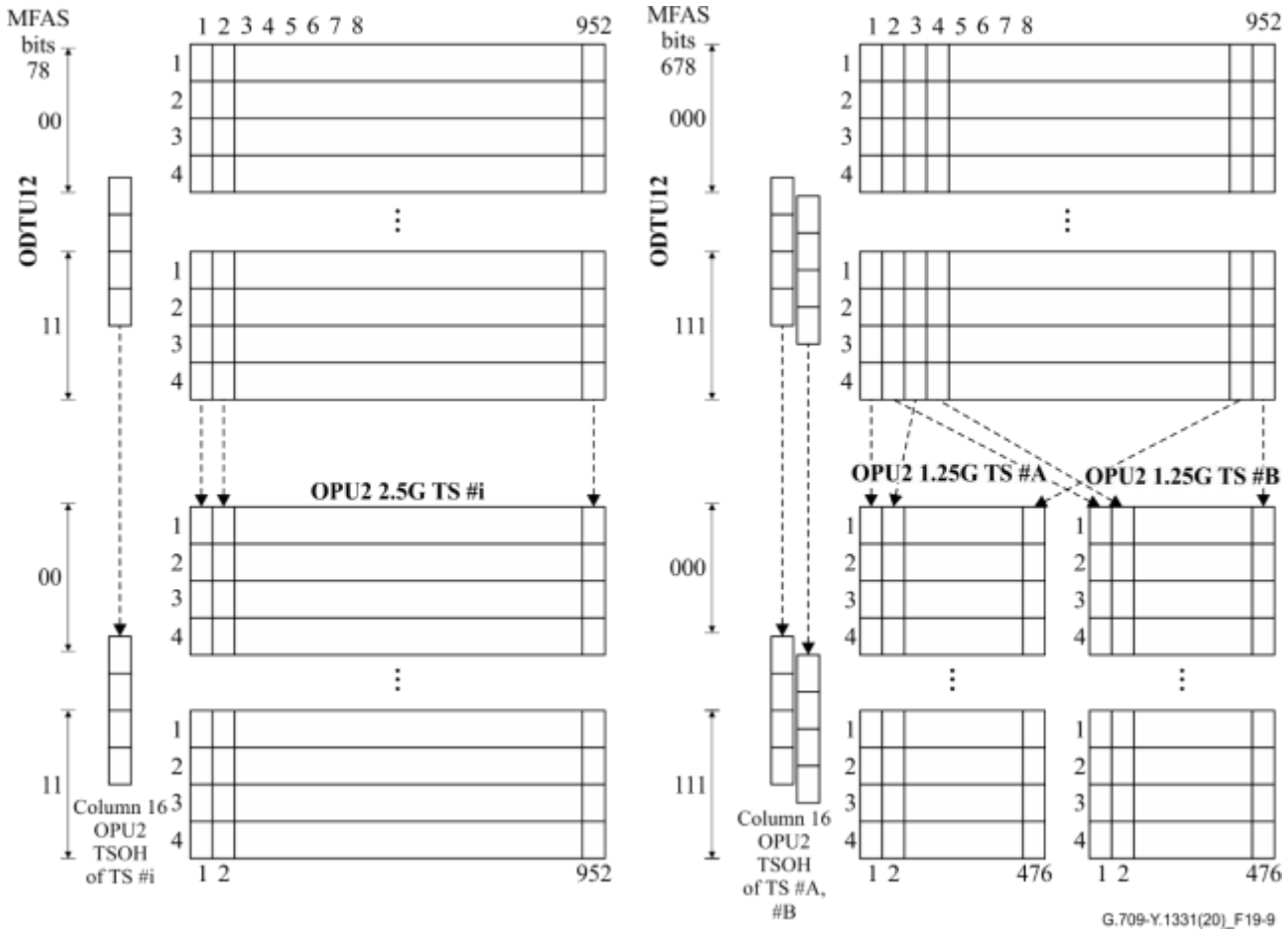


Figure 19-9 – Mapping of ODTU12 into one OPU2 2.5G tributary slot (left) and two OPU2 1.25G tributary slots (right)

19.3.2 ODTU13 mapping into one OPU3 tributary slot

A byte of the ODTU13 signal is mapped into a byte of an OPU3 2.5G TS #i (i = 1,2,..,16) payload area, as indicated in Figure 19-10 (left). A byte of the ODTU13 overhead is mapped into a TSOH byte within column 16 of the OPU3 2.5G TS #i.

A byte of the ODTU13 signal is mapped into a byte of one of two OPU3 1.25G TS #A, B (A,B = 1,2,..,32) payload areas, as indicated in Figure 19-10 (right). A byte of the ODTU13 overhead is mapped into a TSOH byte within column 16 of the OPU3 1.25G TS #a,b.

The remaining OPU3 TSOH bytes in column 15 are reserved for future international standardization.

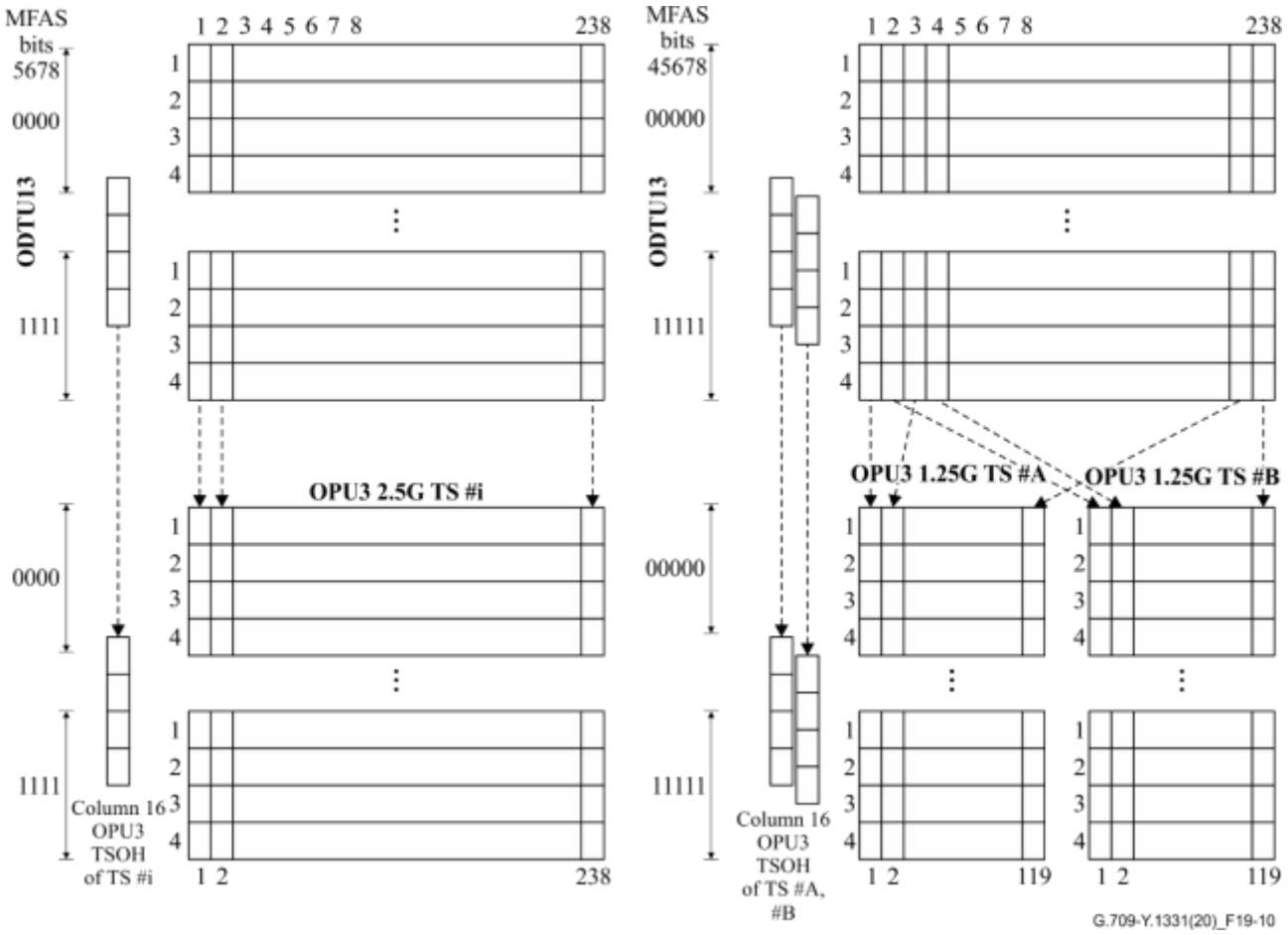


Figure 19-10 – Mapping of ODTU13 into one OPU3 2.5G tributary slot (left) and two OPU3 1.25G tributary slots (right)

19.3.3 ODTU23 mapping into four OPU3 tributary slots

A byte of the ODTU23 signal is mapped into a byte of one of four OPU3 2.5G TS #A,B,C,D (A,B,C,D = 1,2,..,16) payload areas, as indicated in Figure 19-11 (top). A byte of the ODTU23 overhead is mapped into a TSOH byte within column 16 of the OPU3 TS #a,b,c,d.

A byte of the ODTU23 signal is mapped into a byte of one of eight OPU3 1.25G TS #A, B, C, D, E, F, G, H (A,B,C,D,E,F,G,H = 1,2,..,32) payload areas, as indicated in Figure 19-11 (bottom). A byte of the ODTU23 overhead is mapped into a TSOH byte within column 16 of the OPU3 1.25G TS #a,b,c,d,e,f,g,h.

The remaining OPU3 TSOH bytes in column 15 are reserved for future international standardization.

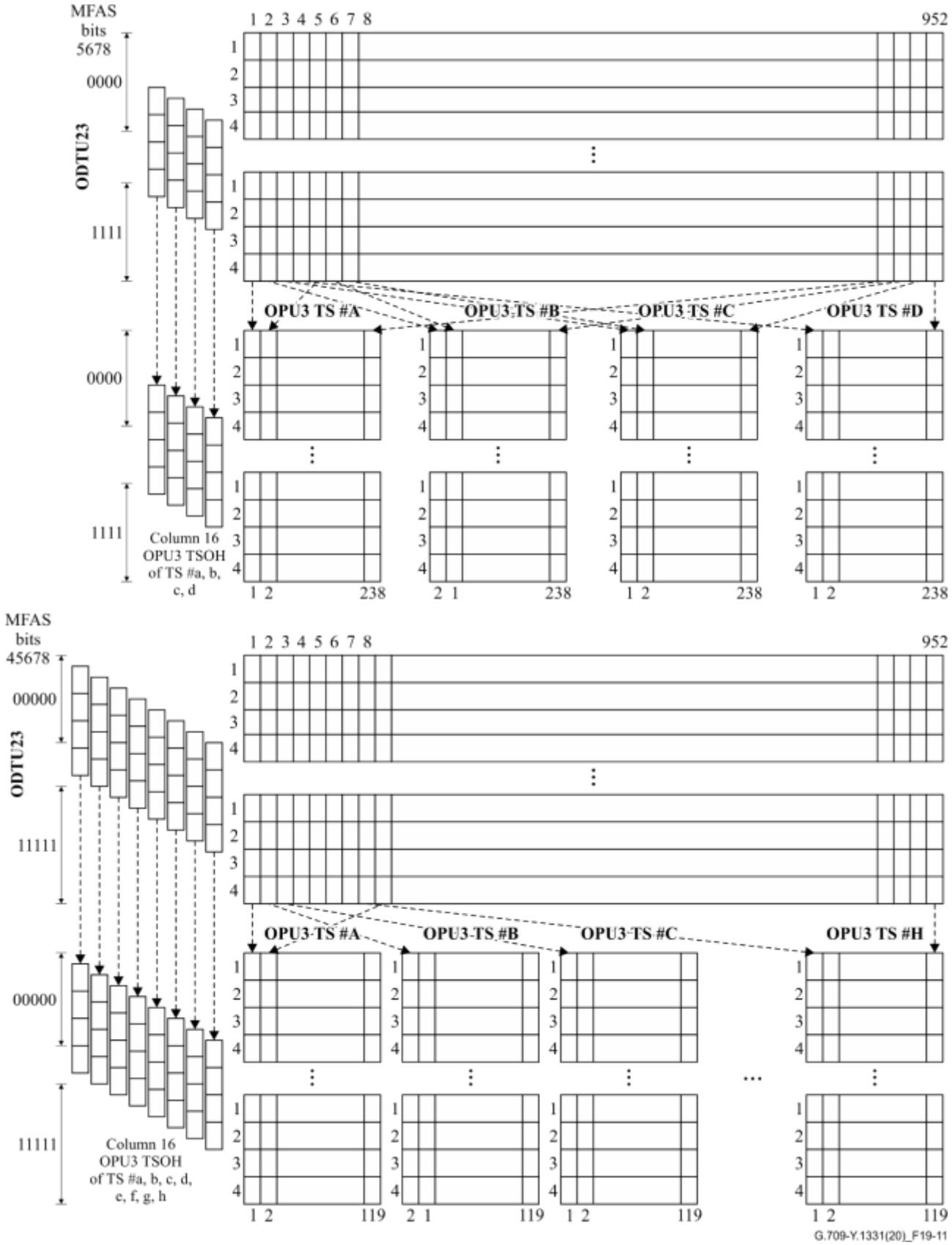


Figure 19-11 – Mapping of ODTU23 into 4 OPU3 2.5G tributary slots (#A, #B, #C, #D with A<B<C<D) (top) and 8 OPU3 1.25G tributary slots (#A, #B, #C, #D, #E, #F, #G, #H with A<B<C<D<E<F<G<H) (bottom)

19.3.4 ODTU01 mapping into one OPU1 1.25G tributary slot

A byte of the ODTU01 signal is mapped into a byte of an OPU1 1.25G TS #i (i = 1,2), as indicated in Figure 19-12 for a group of 4 rows out of the ODTU01.

A byte of the ODTU01 TSOH is mapped into a TSOH byte within column 16 of the OPU1 1.25G TS #i.

The remaining OPU1 TSOH bytes in column 15 are reserved for future international standardization.

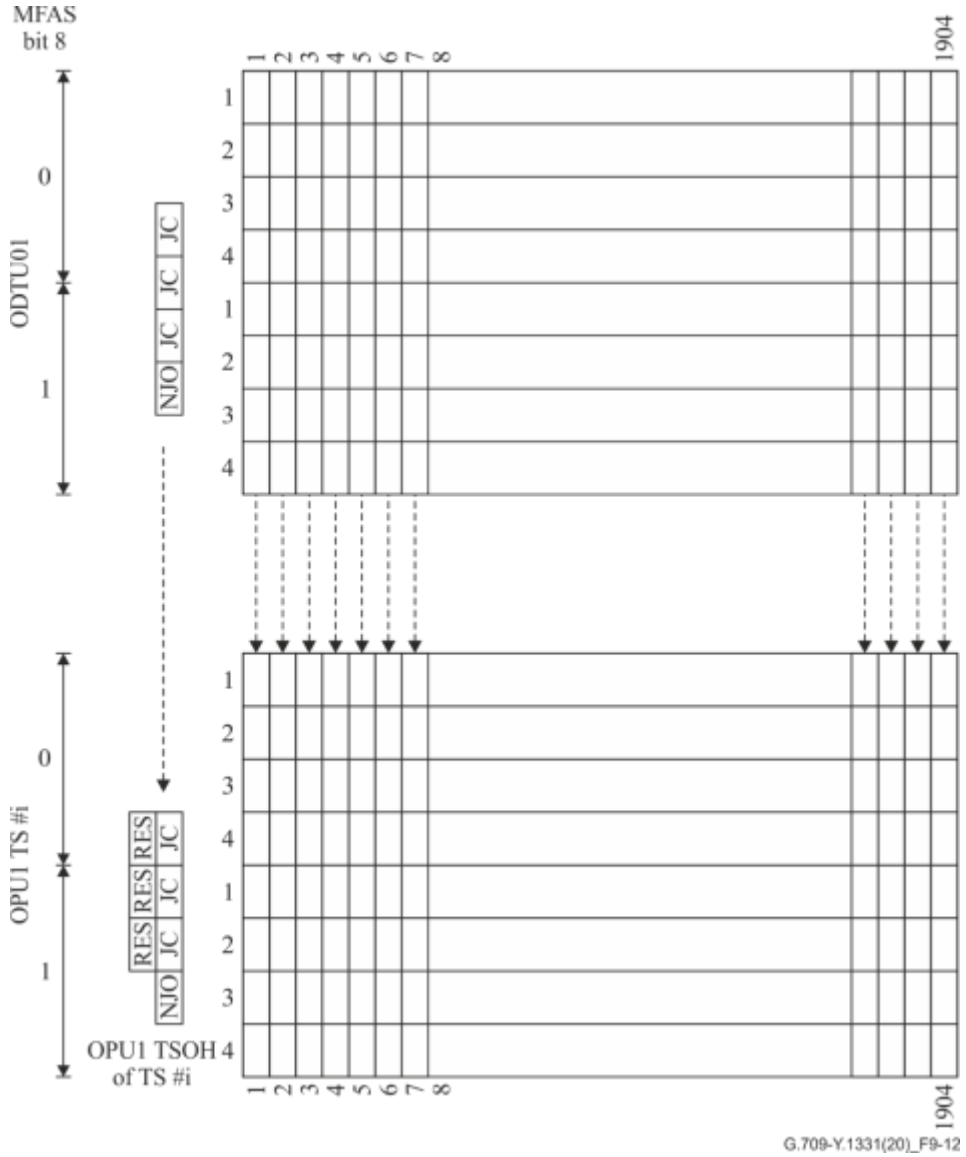


Figure 19-12 – Mapping of ODTU01 (excluding JOH) into OPU1 1.25G tributary slot

19.3.5 ODTU2.ts mapping into ts OPU2 1.25G tributary slots

A byte of the ODTU2.ts payload signal is mapped into a byte of an OPU2 1.25G TS #i (i = 1,..,ts) payload area, as indicated in Figure 19-13.

A byte of the ODTU2.ts overhead is mapped into a TSOH byte within columns 15 and 16, rows 1 to 3 of the last OPU2 1.25G tributary slot allocated to the ODTU2.ts.

The remaining OPU2 TSOH bytes are reserved for future international standardization.

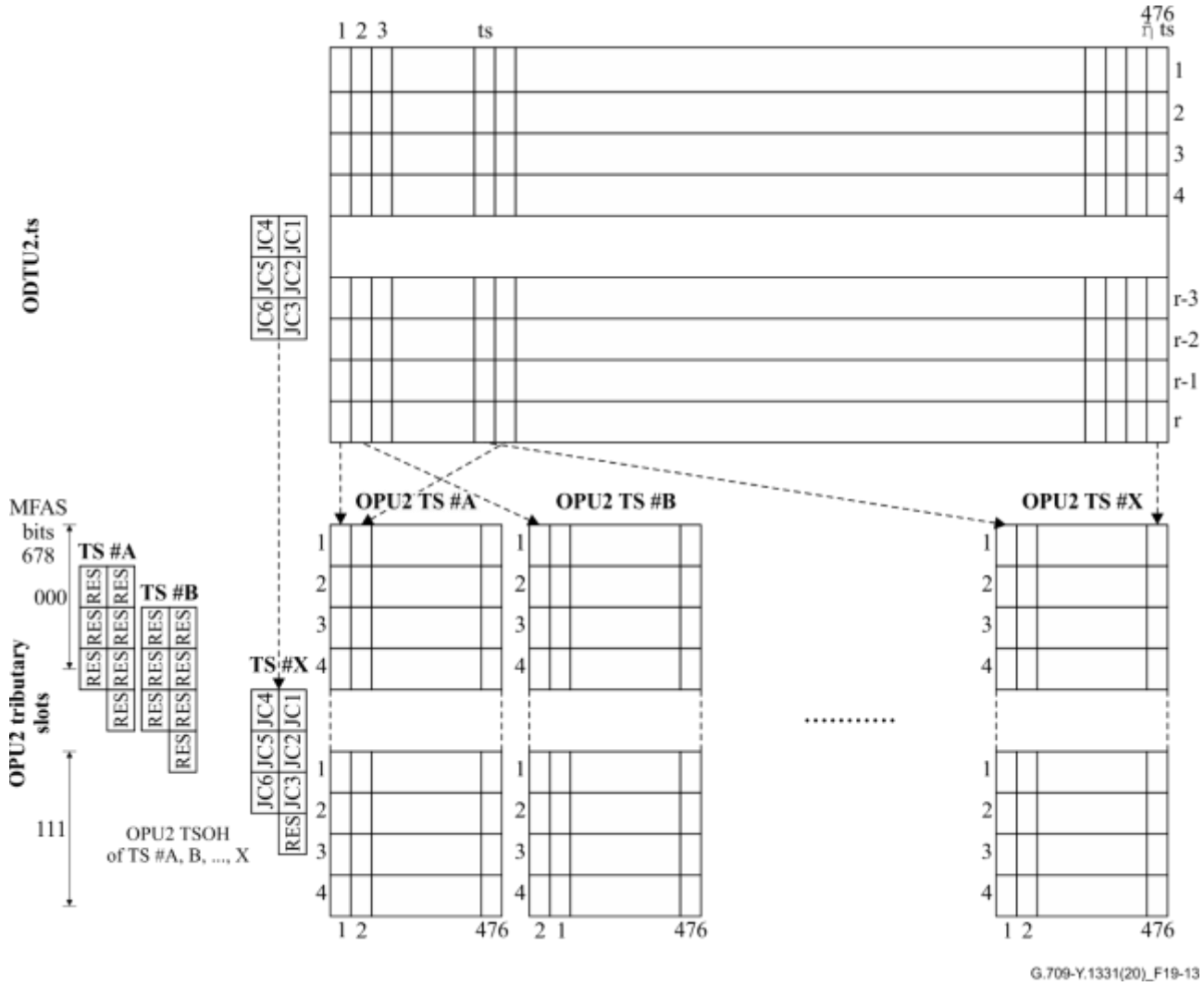


Figure 19-13 – Mapping of ODTU2.ts into 'ts' OPU2 1.25G tributary slots

19.3.6 ODTU3.ts mapping into ts OPU3 1.25G tributary slots

A byte of the ODTU3.ts payload signal is mapped into a byte of an OPU3 1.25G TS #i (i = 1,..,ts) payload area, as indicated in Figure 19-14.

A byte of the ODTU3.ts overhead is mapped into a TSOH byte within columns 15 and 16, rows 1 to 3 of the last OPU3 1.25G tributary slot allocated to the ODTU3.ts.

The remaining OPU3 TSOH bytes are reserved for future international standardization.

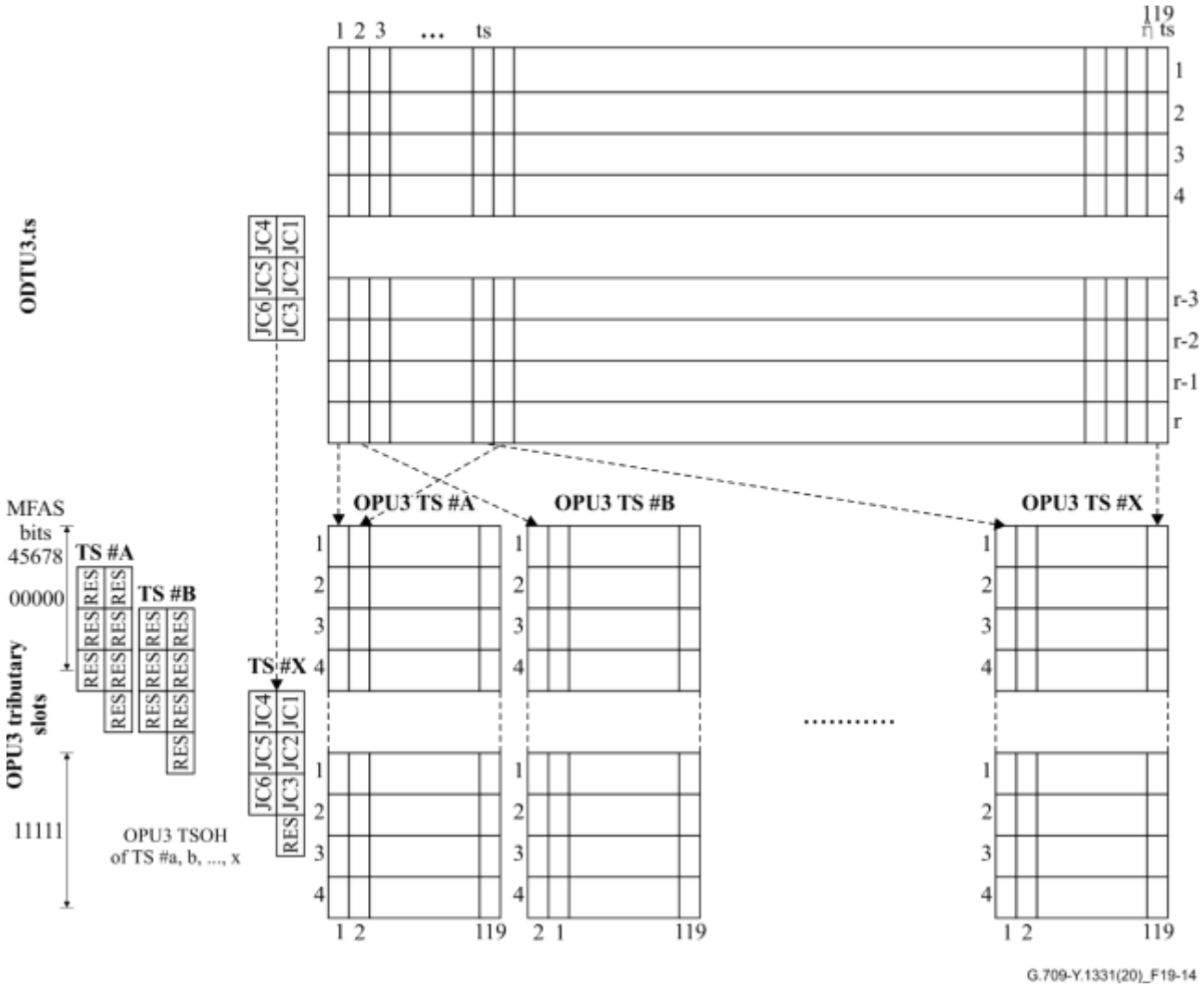


Figure 19-14 – Mapping of ODTU3.ts into 'ts' OPU3 1.25G tributary slots

19.3.7 ODTU4.ts mapping into ts OPU4 1.25G tributary slots

A byte of the ODTU4.ts payload signal is mapped into a byte of an OPU4 1.25G TS #i (i = 1,..,ts) payload area, as indicated in Figure 19-15.

A byte of the ODTU4.ts overhead is mapped into a TSOH byte within columns 15 and 16, rows 1 to 3 of the last OPU4 1.25G tributary slot allocated to the ODTU4.ts.

The remaining OPU4 TSOH bytes are reserved for future international standardization.

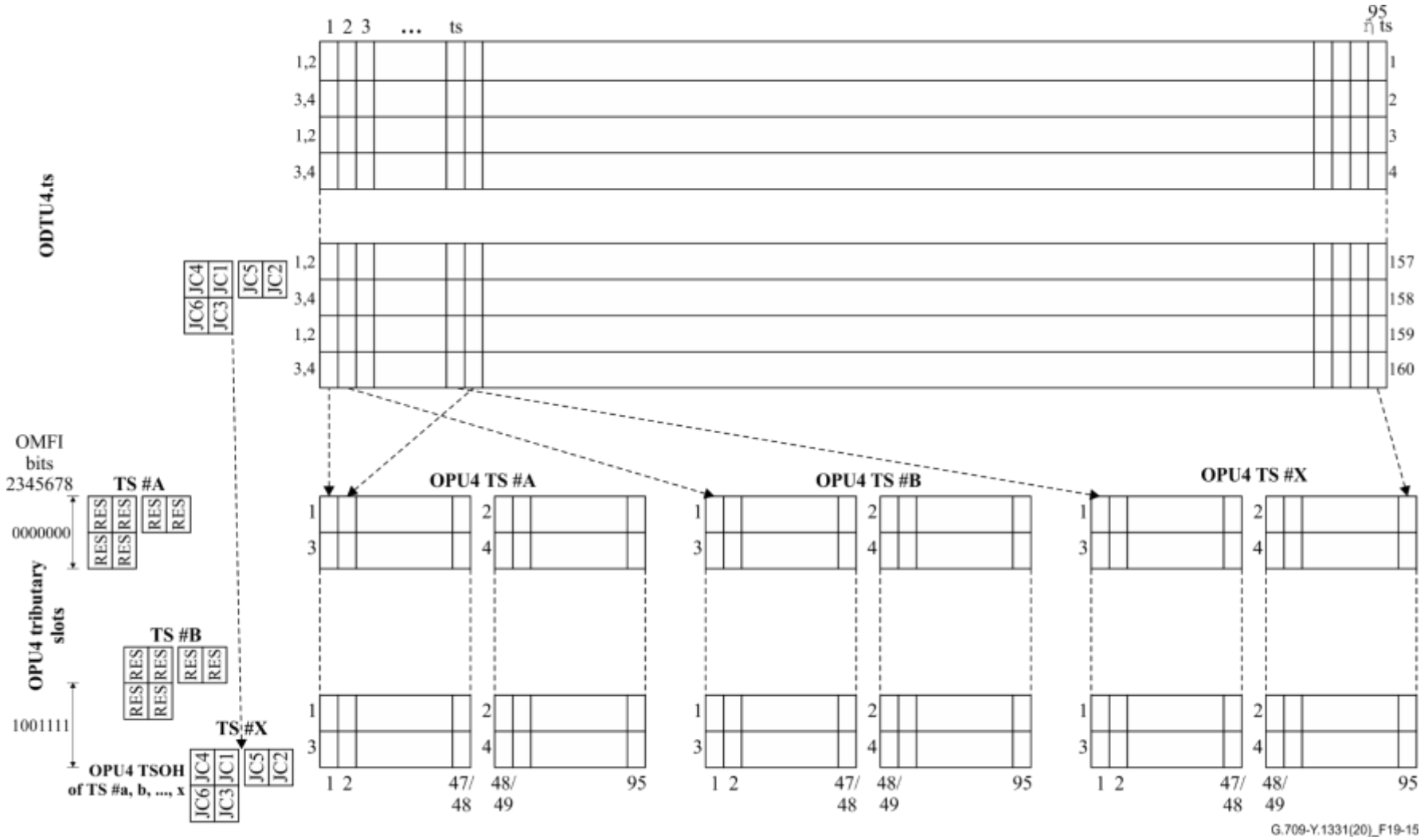


Figure 19-15 – Mapping of ODTU4.ts into 'ts' OPU4 1.25G tributary slots

19.3.8 ODTU25.ts mapping into ts OPU25 1.25G tributary slots

A byte of the ODTU25.ts payload signal is mapped into a byte of an OPU25 1.25G TS #i (i = 1,..,ts) payload area, as indicated in Figure 19-16.

A byte of the ODTU25.ts overhead is mapped into a TSOH byte within columns 15 and 16, rows 1 to 3 of the last OPU25 1.25G tributary slot allocated to the ODTU25.ts.

The remaining OPU25 TSOH bytes are reserved for future international standardization.

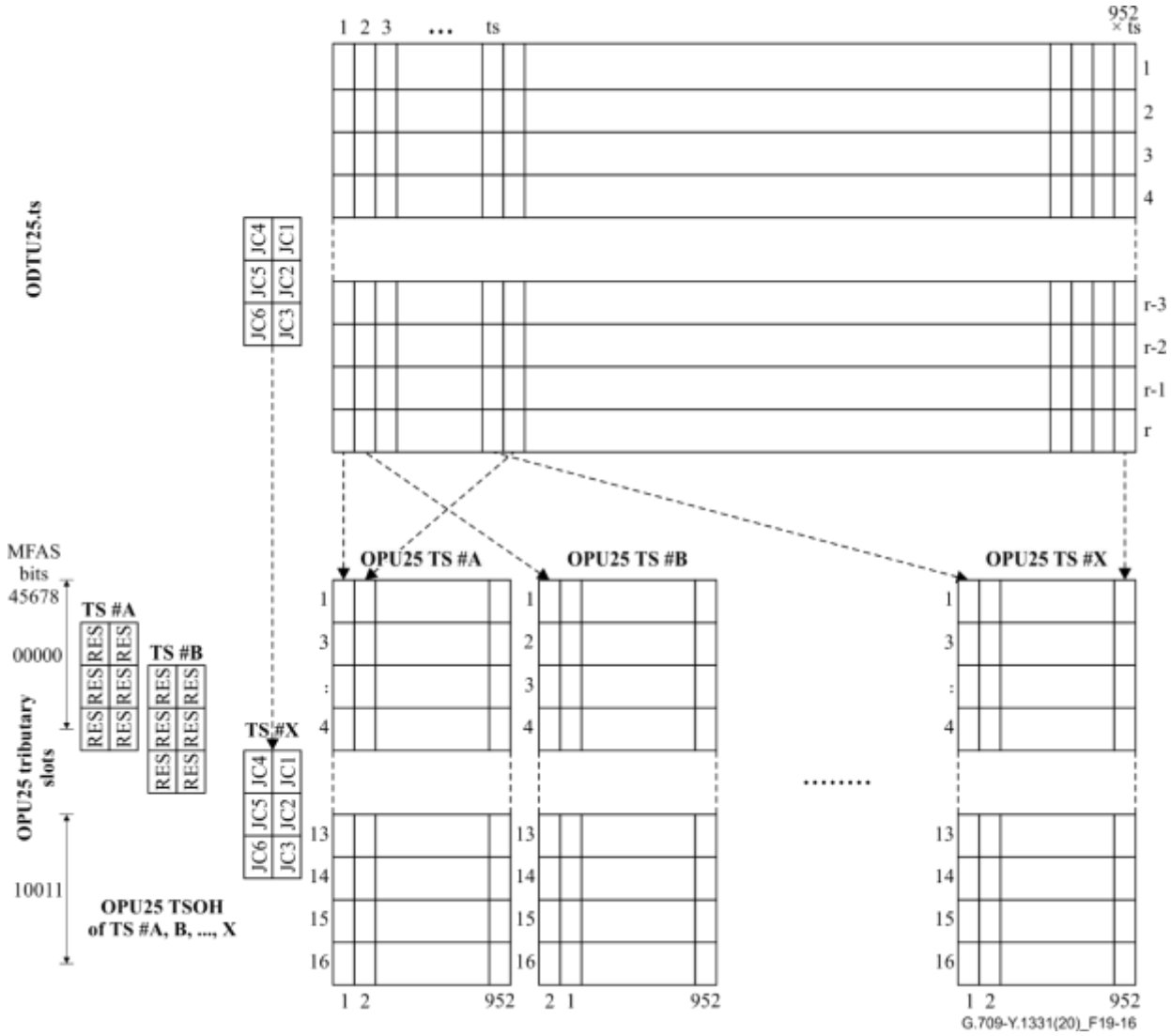


Figure 19-16 – Mapping of ODTU25.ts into 'ts' OPU25 1.25G tributary slots

19.3.9 ODTU50.ts mapping into ts OPU50 1.25G tributary slots

A byte of the ODTU50.ts payload signal is mapped into a byte of an OPU50 1.25G TS #i (i = 1,..,ts) payload area, as indicated in Figure 19-17.

A byte of the ODTU50.ts overhead is mapped into a TSOH byte within columns 15 and 16, rows 1 to 3 of the last OPU50 1.25G tributary slot allocated to the ODTU50.ts.

The remaining OPU50 TSOH bytes are reserved for future international standardization.

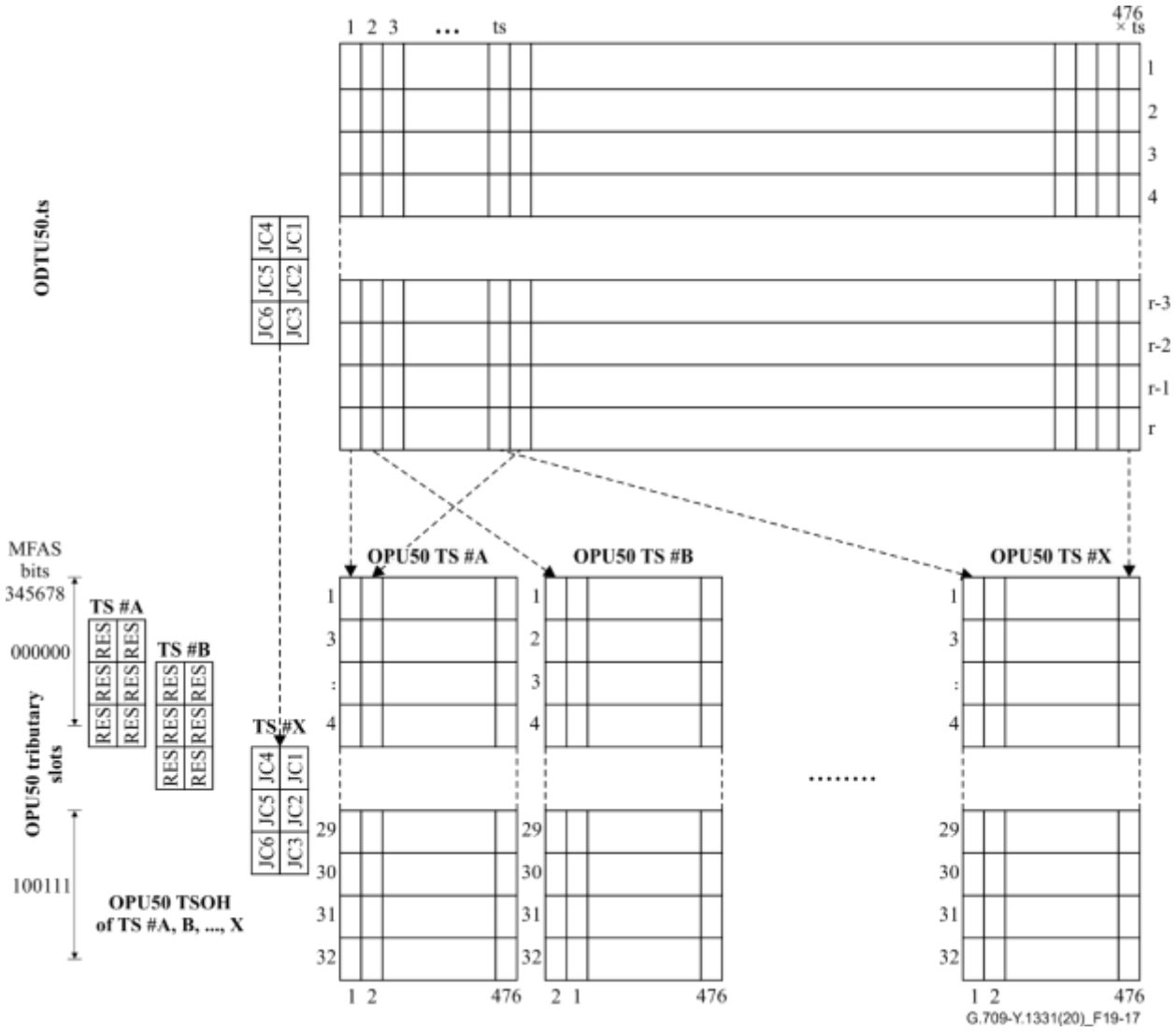


Figure 19-17 – Mapping of ODTU50.ts into 'ts' OPU50 1.25G tributary slots

19.4 OPUk multiplex overhead and ODTU justification overhead

The OPUk (k=1,2,3,4,25,50) multiplex overhead consists of a multiplex structure identifier (MSI) and an ODTU overhead. The OPUk (k=4,25,50) multiplex overhead contains an OPU multiframe identifier (OMFI).

The OPUk MSI overhead locations are shown in Figures 19-18A, 19-18B and 19-18C and the OMFI overhead location is shown in Figure 19-18C.

ODTUjk overhead

The ODTUjk overhead carries the AMP justification overhead consisting of justification control (JC) and negative justification opportunity (NJO) signals in column 16 of rows 1 to 4. ODTUjk overhead bytes in column 15 rows 1, 2 and 3 are reserved for future international standardization.

The ODTUjk overhead consists of 3 bytes of justification control (JC) and 1 byte of negative justification opportunity (NJO) overhead. The JC and NJO overhead locations are shown in Figures 19-18A and 19-18B. In addition, two or n times two positive justification overhead bytes (PJO1, PJO2) are located in the ODTUjk payload area. Note that the PJO1 and PJO2 locations are multiframe, ODUj and OPUk tributary slot dependent.

The PJO1 for an ODU1 in OPU2 or OPU3 2.5G tributary slot #i (i: 1..4 or 1..16 respectively) is located in the first column of OPUk 2.5G tributary slot #i (OPUk column 16+i) and the PJO2 is

located in the second column of OPUk 2.5G tributary slot #i (OPU2 column 20+i, OPU3 column 32+i) in frame #i of the four or sixteen frame multiframe.

EXAMPLE – ODU1 in OPU2 or OPU3 TS(1): PJO1 in column 16+1=17, PJO2 in column 20+1=21 (OPU2) and 32+1=33 (OPU3). ODU1 in OPU2 TS(4): PJO1 in column 16+4=20, PJO2 in column 20+4=24. ODU1 in OPU3 TS(16): PJO1 in column 16+16=32, PJO2 in column 32+16=48.

The four PJO1s for an ODU2 in OPU3 2.5G tributary slots #a, #b, #c and #d are located in the first column of OPU3 2.5G tributary slot #a (OPU3 column 16+a) in frames #a, #b, #c and #d of the sixteen frame multiframe. The four PJO2s for an ODU2 in OPU3 2.5G tributary slots #a, #b, #c and #d are located in the first column of OPU3 2.5G tributary slot #b (OPU3 column 16+b) in frames #a, #b, #c and #d of the sixteen frame multiframe. Figure 19-18A presents an example with four ODU2s in the OPU3 mapped into 2.5G tributary slots (1,5,9,10), (2,3,11,12), (4,14,15,16) and (6,7,8,13).

EXAMPLE – ODU2 in OPU3 TS(1,2,3,4): PJO1 in column 16+1=17, PJO2 in column 16+2=18. ODU2 in OPU3 TS(13,14,15,16): PJO1 in column 16+13=29, PJO2 in column 16+14=30.

The PJO1 for an ODU0 in OPU1 1.25G tributary slot #i (i: 1,2) is located in the first column of OPU1 1.25G tributary slot #i (OPU1 column 16+i) and the PJO2 is located in the second column of OPU1 1.25G tributary slot #i (OPU1 column 18+i) in frame #i of the two frame multiframe.

The PJO1 for an ODU1 in OPU2 or OPU3 1.25G tributary slots #a and #b (a: 1..7 or 1..31 respectively; b: 2..8 or 2..32 respectively) is located in the first column of OPUk 1.25G tributary slot #a (OPUk column 16+a) and the PJO2 is located in the first column of OPUk 1.25G tributary slot #b (OPUk column 16+b) in frames #a and #b of the eight or thirty-two frame multiframe.

EXAMPLE – ODU1 in OPU2 or OPU3 TS(1,2): PJO1 in column 16+1=17, PJO2 in column 16+2=18. ODU1 in OPU2 TS(7,8): PJO1 in column 16+7=23, PJO2 in column 16+8=24. ODU1 in OPU3 TS(31,32): PJO1 in column 16+31=47, PJO2 in column 16+32=48.

The eight PJO1s for an ODU2 in OPU3 1.25G tributary slots #a, #b, #c, #d, #e, #f, #g and #h are located in the first column of OPU3 1.25G tributary slot #a (OPU3 column 16+a) in frames #a, #b, #c, #d, #e, #f, #g and #h of the thirty-two frame multiframe. The eight PJO2s for an ODU2 in OPU3 1.25G tributary slots #a, #b, #c, #d, #e, #f, #g and #h are located in the first column of OPU3 1.25G tributary slot #b (OPU3 column 16+b) in frames #a, #b, #c, #d, #e, #f, #g and #h of the thirty-two frame multiframe. Figure 19-18B presents an example with two ODU2s and two ODU1s in the OPU3 mapped into 1.25G tributary slots (1,5,9,10,17,19,20,21), (25,26,27,28,29,30,31,32), (2,3) and (4,24).

EXAMPLE – ODU2 in OPU3 TS(1,2,3,4,5,6,7,8): PJO1 in column 16+1=17, PJO2 in column 16+2=18. ODU2 in OPU3 TS(25,26,27,28,29,30,31,32): PJO1 in column 16+25=41, PJO2 in column 16+26=42.

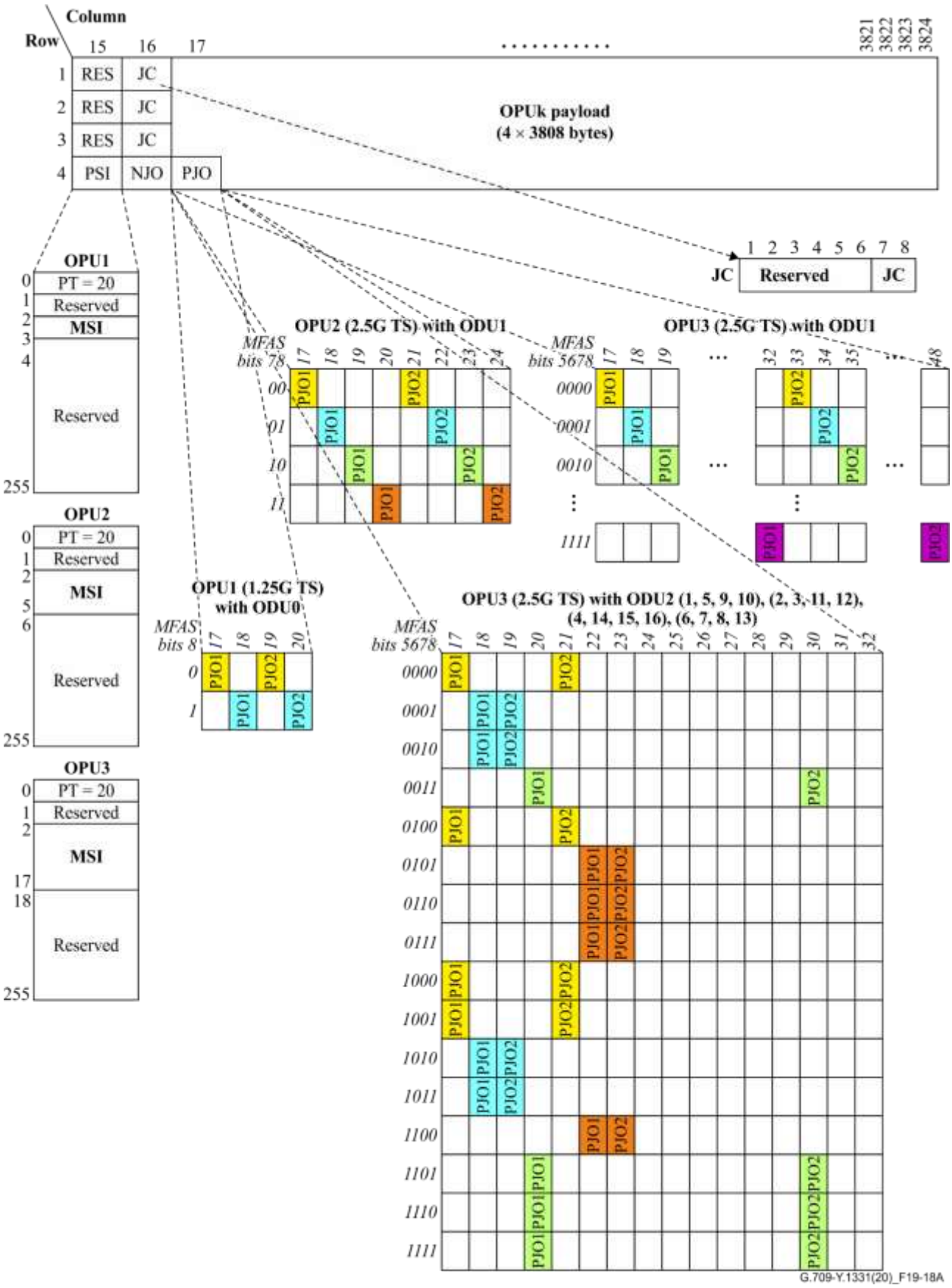


Figure 19-18A – OPUk (k=1,2,3) multiplex overhead associated with an ODTUjk only (payload type = 20)

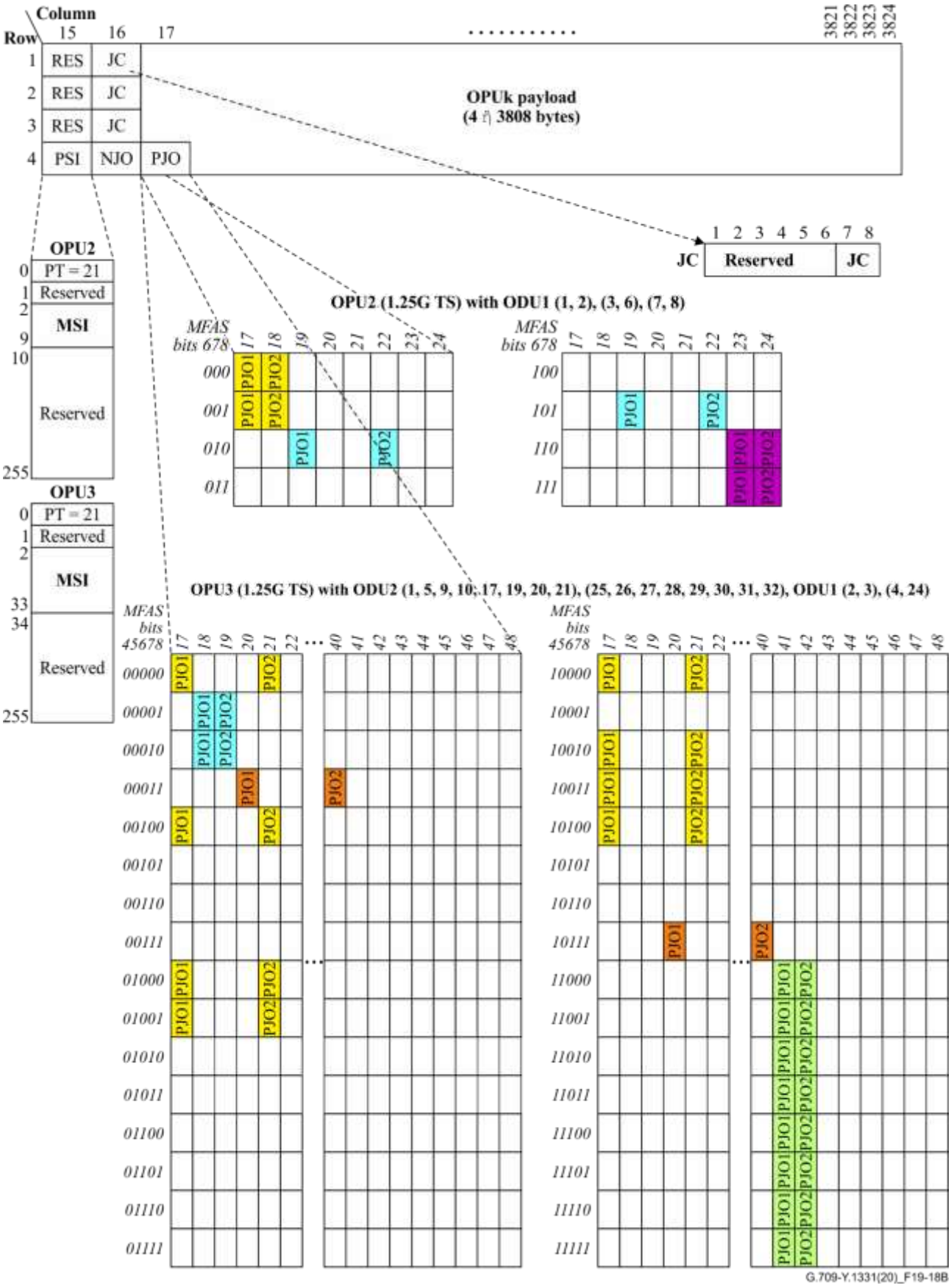


Figure 19-18B – OPUk (k=2,3) multiplex overhead associated with an ODTUjk only (payload type = 21)

ODTUk.ts overhead

The ODTUk.ts overhead carries the GMP justification overhead consisting of 3 bytes of justification control (JC1, JC2, JC3) which carry the 14-bit GMP C_m information and client/ODU specific 3 bytes of justification control (JC4, JC5, JC6) which carry the 10-bit GMP ΣC_{8D} information.

The JC1, JC2, JC3, JC4, JC5 and JC6 overhead locations are shown in Figure 19-18C.

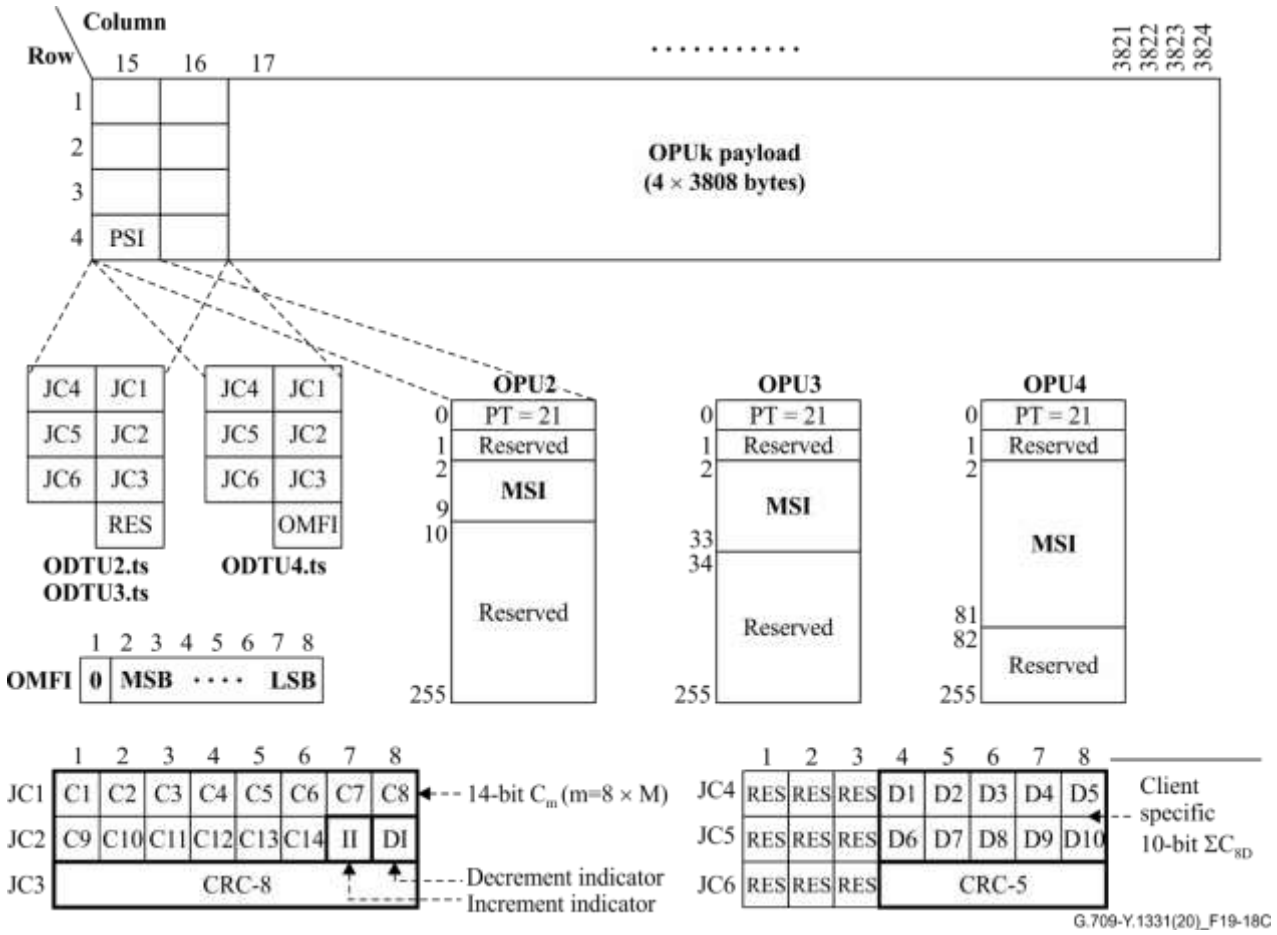


Figure 19-18C – OPUk (k=2,3,4,25,50) multiplex overhead associated with an ODTUk.ts (payload type = 21)

19.4.1 OPUk multiplex structure identifier (MSI)

The OPUk (k=1,2,3,4,25,50) multiplex structure identifier (MSI) overhead, which encodes the ODU multiplex structure in the OPU, is located in the mapping specific area of the PSI signal (refer to Figure 19-18A for the MSI location in OPUk with PT=20, refer to Figures 19-18B and 19-18C for the MSI location in OPUk with PT=21). The MSI has an OPU and tributary slot (2.5G, 1.25G) specific length (OPU1: 2 bytes, OPU2: 4 or 8 bytes, OPU3: 16 or 32 bytes, OPU4: 80 bytes, OPU25: 20 bytes, OPU50: 40 bytes) and indicates the ODTU content of each tributary slot (TS) of an OPU. One byte is used for each TS.

19.4.1.1 OPU2 multiplex structure identifier (MSI) – Payload type 20

For the 4 OPU2 2.5G tributary slots four bytes of the PSI are used (PSI[2] .. PSI[5]) as MSI bytes as shown in Figures 19-18A and 19-19. The MSI indicates the ODTU content of each tributary slot of the OPU2. One byte is used for each tributary slot.

– The ODTU type in bits 1 and 2 is fixed to 00 to indicate the presence of an ODTU12.

– The tributary port # indicates the port number of the ODU1 that is being transported in this 2.5G TS; the assignment of ports to tributary slots is fixed, the port number equals the tributary slot number.

	1 2	3 4 5 6 7 8	
PSI[2]	00	00 0000	*TS1*
PSI[3]	00	00 0001	*TS2*
PSI[4]	00	00 0010	*TS3*
PSI[5]	00	00 0011	*TS4*

Figure 19-19 – OPU2-MSI coding – Payload type 20

19.4.1.2 OPU3 multiplex structure identifier (MSI) – Payload type 20

For the 16 OPU3 2.5G tributary slots 16 bytes of the PSI are used (PSI[2] .. PSI[17]) as MSI bytes as shown in Figures 19-18A, 19-20A and 19-20B. The MSI indicates the ODTU content of each tributary slot of the OPU3. One byte is used for each tributary slot.

– The ODTU type in bits 1 and 2 indicates if the OPU3 TS is carrying ODTU13 or ODTU23. The default ODTU type is ODTU13; it is present when either a tributary slot carries an ODTU13, or is not allocated to carry an ODTU. Refer to Appendix V for some examples.

– The tributary port # in bits 3 to 8 indicates the port number of the ODTU13/23 that is being transported in this 2.5G TS; for the case of ODTU23, a flexible assignment of tributary ports to tributary slots is possible, for the case of ODTU13, this assignment is fixed, the tributary port number equals the tributary slot number. ODTU23 tributary ports are numbered 1 to 4.

	1 2	3 4 5 6 7 8	
PSI[2]	ODTU type	Tributary Port #	*TS1*
PSI[3]	ODTU type	Tributary Port #	*TS2*
PSI[4]	ODTU type	Tributary Port #	*TS3*
PSI[5]	ODTU type	Tributary Port #	*TS4*
PSI[6]	ODTU type	Tributary Port #	*TS5*
PSI[7]	ODTU type	Tributary Port #	*TS6*
PSI[8]	ODTU type	Tributary Port #	*TS7*
PSI[9]	ODTU type	Tributary Port #	*TS8*
PSI[10]	ODTU type	Tributary Port #	*TS9*
PSI[11]	ODTU type	Tributary Port #	*TS10*
PSI[12]	ODTU type	Tributary Port #	*TS11*
PSI[13]	ODTU type	Tributary Port #	*TS12*
PSI[14]	ODTU type	Tributary Port #	*TS13*
PSI[15]	ODTU type	Tributary Port #	*TS14*
PSI[16]	ODTU type	Tributary Port #	*TS15*
PSI[17]	ODTU type	Tributary Port #	*TS16*

Figure 19-20A – OPU3-MSI coding – Payload type 20

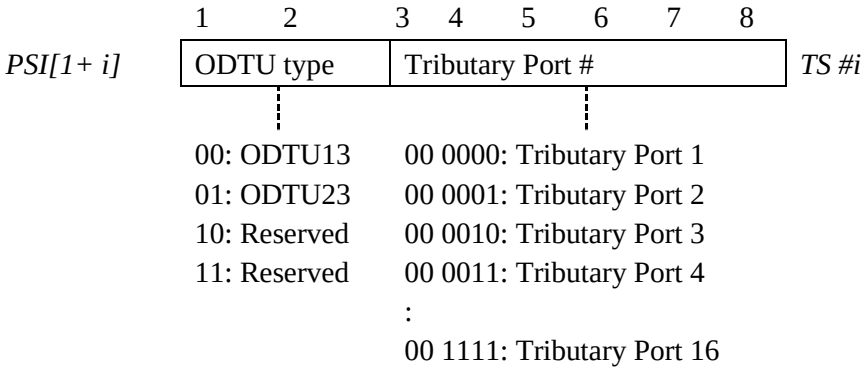


Figure 19-20B – OPU3 MSI coding – Payload type 20

19.4.1.3 OPU1 multiplex structure identifier (MSI) – Payload type 20

For the 2 OPU1 1.25G tributary slots 2 bytes of the PSI are used (PSI[2], PSI[3]) as MSI bytes as shown in Figures 19-18A and 19-21. The MSI indicates the ODTU content of each tributary slot of the OPU1. One byte is used for each tributary slot.

– The ODTU type in bits 1 and 2 is fixed to 11 to indicate the presence of an ODTU01.

– The tributary port # in bits 3 to 8 indicates the port number of the ODTU01 that is being transported in this 1.25G TS; the assignment of ports to tributary slots is fixed, the port number equals the tributary slot number.

	1	2	3 4 5 6 7 8	1.25G TS
PSI[2]	11		00 0000	*TS1*
PSI[3]	11		00 0001	*TS2*

Figure 19-21 – OPU1 MSI coding – Payload type 20

19.4.1.4 OPU4 multiplex structure identifier (MSI) – Payload type 21

For the eighty OPU4 1.25G tributary slots 80 bytes of the PSI are used (PSI[2] to PSI[81]) as MSI bytes as shown in Figures 19-18C, 19-22A and 19-22B. The MSI indicates the ODTU content of each tributary slot of an OPU. One byte is used for each tributary slot.

– The TS occupation bit 1 indicates if the tributary slot is allocated or unallocated.

– The tributary port # in bits 2 to 8 indicates the port number of the ODTU4.ts that is being transported in this TS; for the case of an ODTU4.ts carried in two or more tributary slots, a flexible assignment of tributary port to tributary slots is possible. ODTU4.ts tributary ports are numbered 1 to 80. The value is set to all-0s when the occupation bit has the value 0 (tributary slot is unallocated).

	1	2 3 4 5 6 7 8	1.25G TS
PSI[2]	TS occupied	Tributary Port #	*TS1*
PSI[3]	TS occupied	Tributary Port #	*TS2*
PSI[4]	TS occupied	Tributary Port #	*TS3*
PSI[5]	TS occupied	Tributary Port #	*TS4*
PSI[6]	TS occupied	Tributary Port #	*TS5*
PSI[7]	TS occupied	Tributary Port #	*TS6*
PSI[8]	TS occupied	Tributary Port #	*TS7*
PSI[9]	TS occupied	Tributary Port #	*TS8*
:	:	:	*:*
:	:	:	*:*
PSI[81]	TS occupied	Tributary Port #	*TS80*

Figure 19-22A – OPU4 1.25G TS MSI coding – Payload type 21

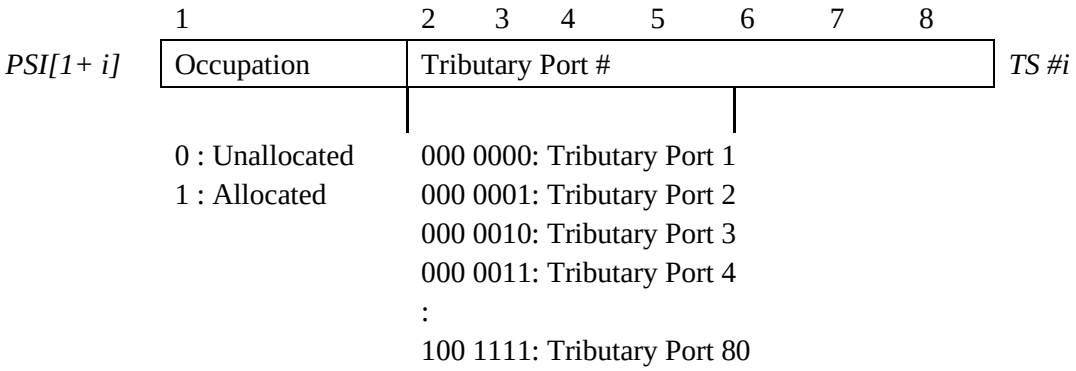


Figure 19-22B – OPU4 MSI coding – Payload type 21

19.4.1.5 OPU2 multiplex structure identifier (MSI) – Payload type 21

For the eight OPU2 1.25G tributary slots 8 bytes of the PSI (PSI[2] to PSI[9]) are used as MSI bytes as show in Figures 19-18B, 19-23A and 19-23B. The MSI indicates the ODTU content of each tributary slot of an OPU. One byte is used for each tributary slot.

– The ODTU type in bits 1 and 2 indicates if the OPU2 1.25G TS is carrying an ODTU12 or ODTU2.ts. The default ODTU type is 11 (unallocated); it is present when a tributary slot is not allocated to carry an ODTU.

– The tributary port # in bits 3 to 8 indicates the port number of the ODTU that is being transported in this TS; a flexible assignment of tributary ports to tributary slots is possible, ODTU12 tributary ports are numbered 1 to 4, and ODTU2.ts tributary ports are numbered 1 to 8. The value is set to all-0s when the ODTU type has the value 11 (tributary slot is unallocated).

	1	2	3	4	5	6	7	8	
PSI[2]	ODTU type		Tributary Port #						*TS1*
PSI[3]	ODTU type		Tributary Port #						*TS2*
PSI[4]	ODTU type		Tributary Port #						*TS3*
PSI[5]	ODTU type		Tributary Port #						*TS4*
PSI[6]	ODTU type		Tributary Port #						*TS5*
PSI[7]	ODTU type		Tributary Port #						*TS6*
PSI[8]	ODTU type		Tributary Port #						*TS7*
PSI[9]	ODTU type		Tributary Port #						*TS8*

Figure 19-23A – OPU2 MSI coding – Payload type 21

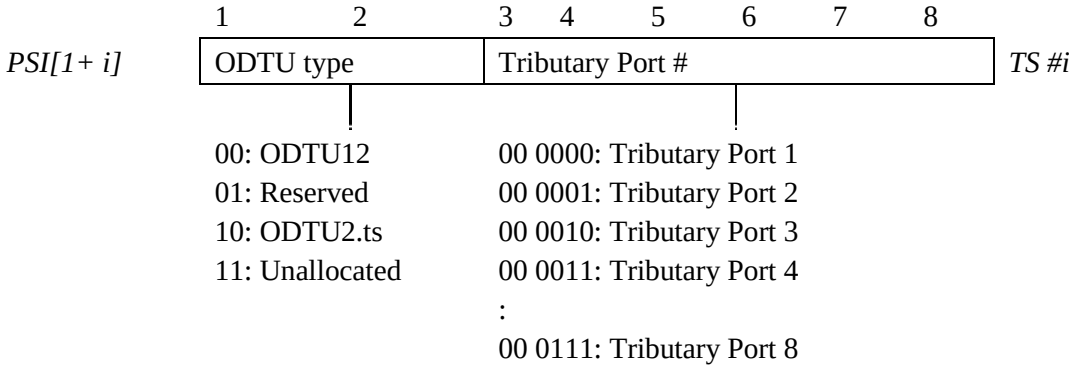


Figure 19-23B – OPU2 MSI coding – Payload type 21

19.4.1.6 OPU3 with 1.25G tributary slots (payload type 21) multiplex structure identifier (MSI)

For the thirty-two OPU3 1.25G tributary slots 32 bytes of the PSI (PSI[2] to PSI[33]) are used as MSI bytes as shown in Figures 19-18B, 19-24A and 19-24B. The MSI indicates the ODTU content of each tributary slot of an OPU. One byte is used for each tributary slot.

– The ODTU type in bits 1 and 2 indicates if the OPU3 1.25G TS is carrying an ODTU13, ODTU23 or ODTU3.ts. The default ODTU type is 11 (unallocated); it is present when a tributary slot is not allocated to carry an ODTU.

– The tributary port # in bits 3 to 8 indicates the port number of the ODTU that is being transported in this TS; a flexible assignment of tributary ports to tributary slots is possible, ODTU13 tributary ports are numbered 1 to 16, ODTU23 tributary ports are numbered 1 to 4 and ODTU3.ts tributary ports are numbered 1 to 32. The value is set to all-0s when the ODTU type has the value 11 (tributary slot is unallocated).

	1	2	3	4	5	6	7	8	
PSI[2]	ODTU type		Tributary Port #						*TS1*
PSI[3]	ODTU type		Tributary Port #						*TS2*
PSI[4]	ODTU type		Tributary Port #						*TS3*
PSI[5]	ODTU type		Tributary Port #						*TS4*
PSI[6]	ODTU type		Tributary Port #						*TS5*
:	:		:						*:*
:	:		:						*:*
PSI[33]	ODTU type		Tributary Port #						*TS32*

Figure 19-24A – OPU3 MSI coding – Payload type 21

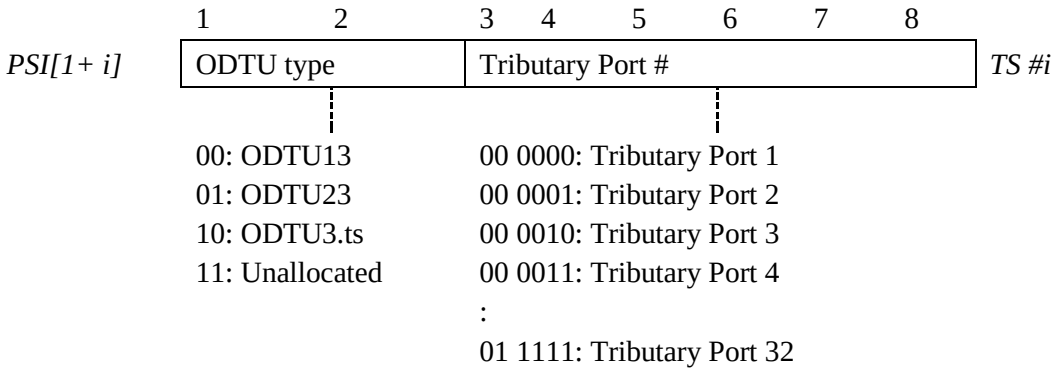


Figure 19-24B – OPU3 MSI coding – Payload type 21

19.4.1.7 OPU25 multiplex structure identifier (MSI) – Payload type 21

For the twenty OPU25 1.25G tributary slots 20 bytes of the PSI are used (PSI[2] to PSI[21]) as MSI bytes as shown in Figures 19-18C, 19-25A and 19-25B. The MSI indicates the ODTU content of each tributary slot of an OPU. One byte is used for each tributary slot.

– The TS occupation bit 1 indicates if the tributary slot is allocated or unallocated.

– The tributary port # in bits 2 to 8 indicates the port number of the ODTU25.ts that is being transported in this TS; for the case of an ODTU25.ts carried in two or more tributary slots, a flexible assignment of tributary port to tributary slots is possible. ODTU25.ts tributary ports are numbered 1 to 20. The value is set to all-0s when the occupation bit has the value 0 (tributary slot is unallocated).

	1	2 3 4 5 6 7 8	1.25G TS
PSI[2]	TS occupied	Tributary Port #	*TS1*
PSI[3]	TS occupied	Tributary Port #	*TS2*
PSI[4]	TS occupied	Tributary Port #	*TS3*
PSI[5]	TS occupied	Tributary Port #	*TS4*
PSI[6]	TS occupied	Tributary Port #	*TS5*
:	:	:	*:*
:	:	:	*:*
PSI[21]	TS occupied	Tributary Port #	*TS20*

Figure 19-25A – OPU25 1.25G TS MSI coding – Payload type 21

	1	2 3 4 5 6 7 8	
PSI[1+ i]	Occupation	Tributary Port #	*TS #i*

0 : Unallocated
1 : Allocated

000 0000: Tributary Port 1
000 0001: Tributary Port 2
000 0010: Tributary Port 3
000 0011: Tributary Port 4
:
001 0011: Tributary Port 20

Figure 19-25B – OPU25 MSI coding – Payload type 21

19.4.1.8 OPU50 multiplex structure identifier (MSI) – Payload type 21

For the forty OPU50 1.25G tributary slots 40 bytes of the PSI are used (PSI[2] to PSI[41]) as MSI bytes as shown in Figures 19-18C, 19-26A and 19-26B. The MSI indicates the ODTU content of each tributary slot of an OPU. One byte is used for each tributary slot.

– The TS occupation bit 1 indicates if the tributary slot is allocated or unallocated.

– The tributary port # in bits 2 to 8 indicates the port number of the ODTU50.ts that is being transported in this TS; for the case of an ODTU50.ts carried in two or more tributary slots, a flexible assignment of tributary port to tributary slots is possible. ODTU50.ts tributary ports are numbered 1 to 40. The value is set to all-0s when the occupation bit has the value 0 (tributary slot is unallocated).

	1	2 3 4 5 6 7 8	1.25G TS
PSI[2]	TS occupied	Tributary Port #	*TS1*
PSI[3]	TS occupied	Tributary Port #	*TS2*
PSI[4]	TS occupied	Tributary Port #	*TS3*
PSI[5]	TS occupied	Tributary Port #	*TS4*
PSI[6]	TS occupied	Tributary Port #	*TS5*
PSI[7]	TS occupied	Tributary Port #	*TS6*
:	:	:	*:*
:	:	:	*:*
PSI[41]	TS occupied	Tributary Port #	*TS40*

Figure 19-26A – OPU50 1.25G TS MSI coding – Payload type 21

	1	2 3 4 5 6 7 8	
PSI[1+ i]	Occupation	Tributary Port #	*TS #i*
	0 : Unallocated 1 : Allocated	000 0000: Tributary Port 1 000 0001: Tributary Port 2 000 0010: Tributary Port 3 000 0011: Tributary Port 4 : 010 0111: Tributary Port 40	

Figure 19-26B – OPU50 MSI coding – Payload type 21

19.4.2 OPUk payload structure identifier reserved overhead (RES)

253 (OPU1), 251 or 247 (OPU2), 239 or 223 (OPU3), 175 (OPU4), 234 (OPU25) and 214 (OPU50) bytes are reserved in the OPUk PSI for future international standardization. These bytes are located in PSI[1] and PSI[4] (OPU1), PSI[6] or PSI[10] (OPU2), PSI[18] or PSI[34] (OPU3), PSI[82] (OPU4), PSI[22] (OPU25), PSI[42] (OPU50) to PSI[255] of the OPUk overhead. These bytes are set to all-0s.

19.4.3 OPUk multiplex justification overhead (JOH)

Two mapping procedures are used for the mapping of ODUj: either the asynchronous mapping procedure (AMP) or generic mapping procedure (GMP) into ODTUjk or ODTUk.ts, respectively. AMP uses ODUj and OPUk specific fixed stuff and justification opportunity definitions (ODTUjk). GMP uses ODUj and OPUk independent stuff and justification opportunity definitions (ODTUk.ts). Stuff locations within an ODTUk.ts are determined by means of a formula which is specified in clause 19.4.3.2.

19.4.3.1 Asynchronous mapping procedures (AMP)

The justification overhead (JOH) located in column 16 of the OPUk (k=1,2,3) as indicated in Figures 19-18A and 19-18B consists of three justification control (JC) bytes and one negative justification opportunity (NJO) byte. The three JC bytes are located in rows 1, 2 and 3. The NJO byte is located in row 4.

Bits 7 and 8 of each JC byte are used for justification control. The other six bits are reserved for future international standardization.

19.4.3.2 Generic mapping procedure (GMP)

The justification overhead (JOH) for the generic mapping procedure consists of two groups of three bytes of justification control; the general (JC1, JC2, JC3) and the client to ODU mapping specific (JC4, JC5, JC6). Refer to Figure 19-18C.

The JC1, JC2 and JC3 bytes consist of a 14-bit C_m field (bits C1, C2, .., C14), a 1-bit increment indicator (II) field, a 1-bit decrement indicator (DI) field and an 8-bit CRC-8 field which contains an error check code over the JC1, JC2 and JC3 fields.

The JC4, JC5 and JC6 bytes consist of a 10-bit ΣC_{nD} field (bits D1, D2, .., D10), a 5-bit CRC-5 field which contains an error check code over bits 4 to 8 in the JC4, JC5 and JC6 fields and nine bits reserved for future international standardization (RES).

The value of 'm' in C_m is 8 × 'ts' (number of tributary slots occupied by the ODTUk.ts).

The value of 'n' represents the timing granularity of the GMP C_n parameter, which is also present in ΣC_{nD}. The value of n is 8.

The value of C_m controls the distribution of groups of 'ts' ODUj data bytes into groups of 'ts' ODTUk.ts payload bytes. Refer to clause 19.6 and Annex D for further specification of this process.

The value of ΣC_{nD} provides additional 'n'-bit timing information, which is necessary to control the jitter and wander performance experienced by the ODUj signal.

The value of C_n (i.e., number of client n-bit data entities per OPUCn multiframe) is computed as follows: $C_n(t) = m/n \times C_m(t) + (\Sigma C_{nD}(t) - \Sigma C_{nD}(t–1))$. Note that the value C_{nD} is effectively an indication of the amount of data in the mapper's virtual queue that it could not send during that multiframe due to it being less than an M-byte word. For the case where the value of ΣC_{nD} in a multiframe 't' is corrupted, it is possible to recover from such error in the next multiframe 't+1'.

19.4.4 OPU multiframe identifier overhead (OMFI)

An OPU4 multiframe identifier (OMFI) byte is defined in row 4, column 16 of the OPU4 overhead (Figure 19-27). The value of bits 2 to 8 of the OMFI byte will be incremented each OPU4 frame to provide an 80 frame multiframe for the multiplexing of ODUj signals into the OPU4.

NOTE 1 – It is an option to align the OMFI = 0 position with MFAS = 0 position every 1280 (the least common multiple of 80 and 256) frame periods.

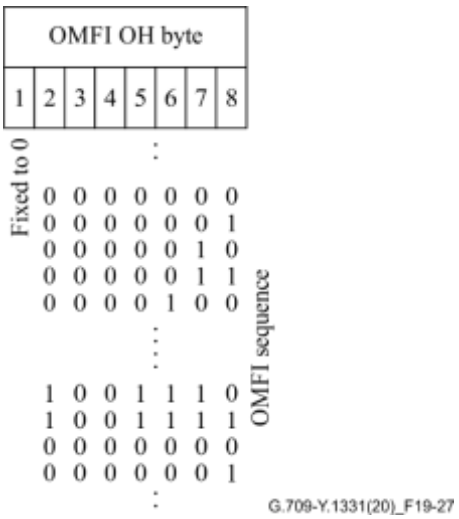


Figure 19-27 – OPU4 multiframe identifier (OMFI) overhead

An OPU25 multiframe identifier (OMFI) byte is defined in row 4, column 16 of the OPU25 overhead (Figure 19-28). The value of bits 4 to 8 of the OMFI byte will be incremented each OPU25 frame to provide a 20 frame multiframe for the multiplexing of ODUj signals into the OPU25.

NOTE 2 – It is an option to align the OMFI = 0 position with MFAS = 0 position every 1280 (the least common multiple of 20 and 256) frame periods.

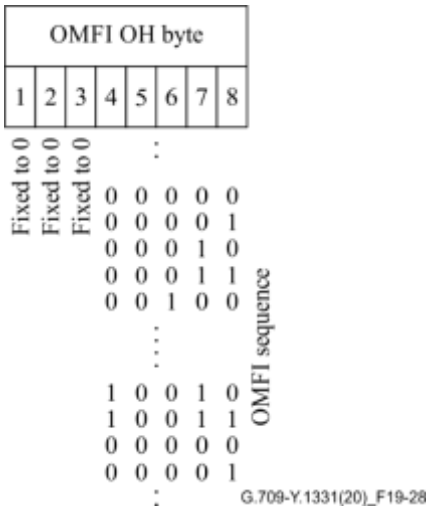


Figure 19-28 – OPU25 multiframe identifier (OMFI) overhead

An OPU50 multiframe identifier (OMFI) byte is defined in row 4, column 16 of the OPU50 overhead (Figure 19-29). The value of bits 3 to 8 of the OMFI byte will be incremented each OPU50 frame to provide a 40 frame multiframe for the multiplexing of ODUj signals into the OPU4.

NOTE 3 – It is an option to align the OMFI = 0 position with MFAS = 0 position every 1280 (the least common multiple of 40 and 256) frame periods.

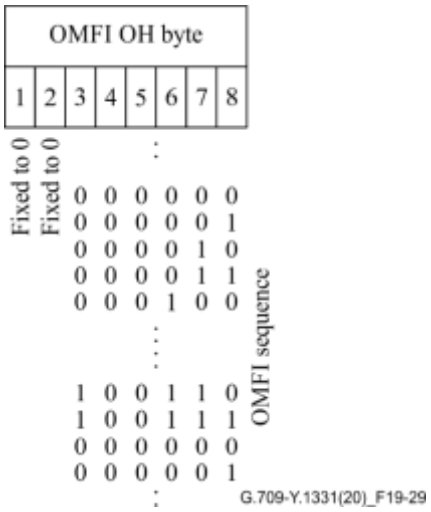


Figure 19-29 – OPU50 multiframe identifier (OMFI) overhead

19.5 Mapping ODUj into ODTUjk

The mapping of ODUj signals (with up to ±20 ppm bit-rate tolerance) into the ODTUjk signal ((j,k) = {(0,1), (1,2); (1,3), (2,3)}) is performed as an asynchronous mapping.

NOTE 1 – The maximum bit-rate tolerance between OPUk and the ODUj signal clock, which can be accommodated by this mapping scheme is –130 to +65 ppm (ODU0 into OPU1), –113 to +83 ppm (ODU1 into OPU2), –96 to +101 ppm (ODU1 into OPU3) and –95 to +101 ppm (ODU2 into OPU3).

The ODUj signal is extended with a frame alignment overhead as specified in clauses 15.6.2.1 and 15.6.2.2 and an all-0s pattern in the OTUj overhead field (see Figure 19-30).

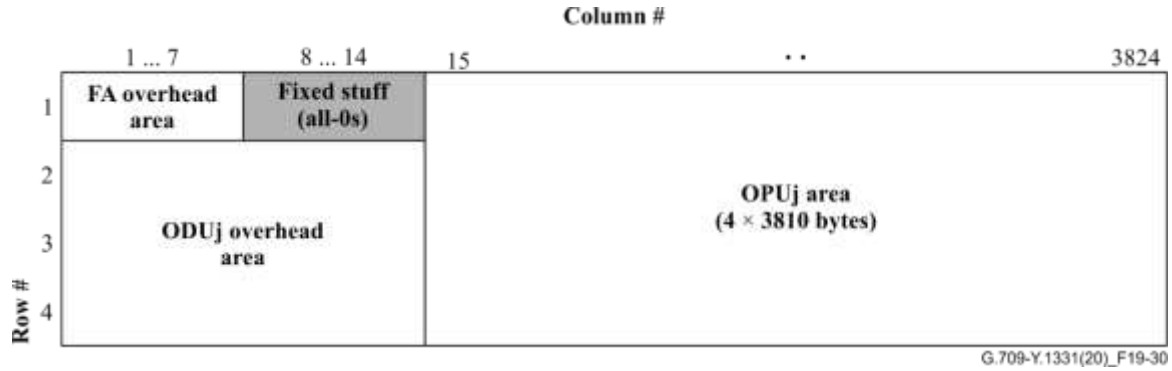


Figure 19-30 – Extended ODUj frame structure (FA OH included, OTUj OH area contains fixed stuff)

The OPUk signal and therefore the ODTUjk (k = 1,2,3) signals are created from a locally generated clock (within the limits specified in Table 7-3), which is independent of the ODUj (j = 0,1,2) client signals.

The extended ODUj (j = 0,1,2) signal is mapped into the ODTUjk (k = 1,2,3) using an asynchronous mapping with –1/0/+1/+2 positive/negative/zero (pnz) justification scheme.

An extended ODUj byte is mapped into an ODTUjk byte.

The asynchronous mapping process generates the JC, NJO, PJO1 and PJO2 according to Table 19-9. The de-mapping process interprets JC, NJO, PJO1 and PJO2 according to Table 19-9. Majority vote (two out of three) shall be used to make the justification decision in the de-mapping process to protect against an error in one of the three JC signals.

Table 19-9 – JC, NJO, PJO1 and PJO2 generation and interpretation

JC 7 8	NJO	PJO1	PJO2	Interpretation
0 0	justification byte	data byte	data byte	no justification (0)
0 1	data byte	data byte	data byte	negative justification (–1)
1 0 (Note)	justification byte	justification byte	justification byte	double positive justification (+2)
1 1	justification byte	justification byte	data byte	positive justification (+1)
NOTE – Note that this code is not used for the case of ODU0 into OPU1.				

The value contained in NJO, PJO1 and PJO2 when they are used as justification bytes is all-0s. The receiver is required to ignore the value contained in these bytes whenever they are used as justification bytes.

During a signal fail condition of the incoming ODUj client signal (e.g., OTUj-LOF), this failed incoming signal will contain the ODUj-AIS signal as specified in clause 16.5.1. This ODUj-AIS is then mapped into the ODTUjk.

For the case where the ODUj is received from the output of a fabric (ODU connection function), the incoming signal may contain (in the case of an open matrix connection), the ODUj-OCI signal as specified in clause 16.5.2. This ODUj-OCI signal is then mapped into the ODTUjk.

NOTE 2 – Not all equipment will have a real connection function (i.e., switch fabric) implemented; instead, the presence/absence of tributary interface port units represents the presence/absence of a matrix connection. If such a unit is intentionally absent (i.e., not installed), the associated ODTUjk signals should carry an ODUj-OCI signal. If such a unit is installed but temporarily removed as part of a repair action, the associated ODTUjk signal should carry an ODUj-AIS signal.

The de-mapping of ODUj signals from the ODTUjk signal (j = 0,1,2; k = 1,2,3) is performed by extracting the extended ODUj signal from the OPUk under control of its justification overhead (JC, NJO, PJO1, PJO2).

NOTE 3 – For the case where the ODUj signal is output as an OTUj signal, frame alignment of the extracted extended ODUj signal is to be recovered to allow frame synchronous mapping of the ODUj into the OTUj signal.

During a signal fail condition of the incoming ODUk/OPUk signal (e.g., in the case of an ODUk-AIS, ODUk-LCK, ODUk-OCI condition) the ODUj-AIS pattern as specified in clause 16.5.1 is generated as a replacement signal for the lost ODUj signal.

19.5.1 Mapping ODU1 into ODTU12

A byte of the ODU1 signal is mapped into an information byte of the ODTU12 (see Figure 19-31A). Once per 4 OPU2 frames, it is possible to perform either a positive or a negative justification action. The frame in which justification can be performed is related to the TSOH of the OPU2 2.5G TS in which the ODTU12 is mapped (Figure 19-1). Figure 19-31A shows the case with mapping in OPU2 2.5G TS1.

A byte of the ODU1 signal is mapped into an information byte of the ODTU12 (see Figure 19-31B). Twice per 8 OPU2 frames, it is possible to perform either a positive or a negative justification action. The frames in which justification can be performed are related to the TSOH of the OPU2 1.25G TSs in which the ODTU12 is mapped (Figure 19-1). Figure 19-31B shows the case with mapping in OPU2 1.25G TS1 and TS4.

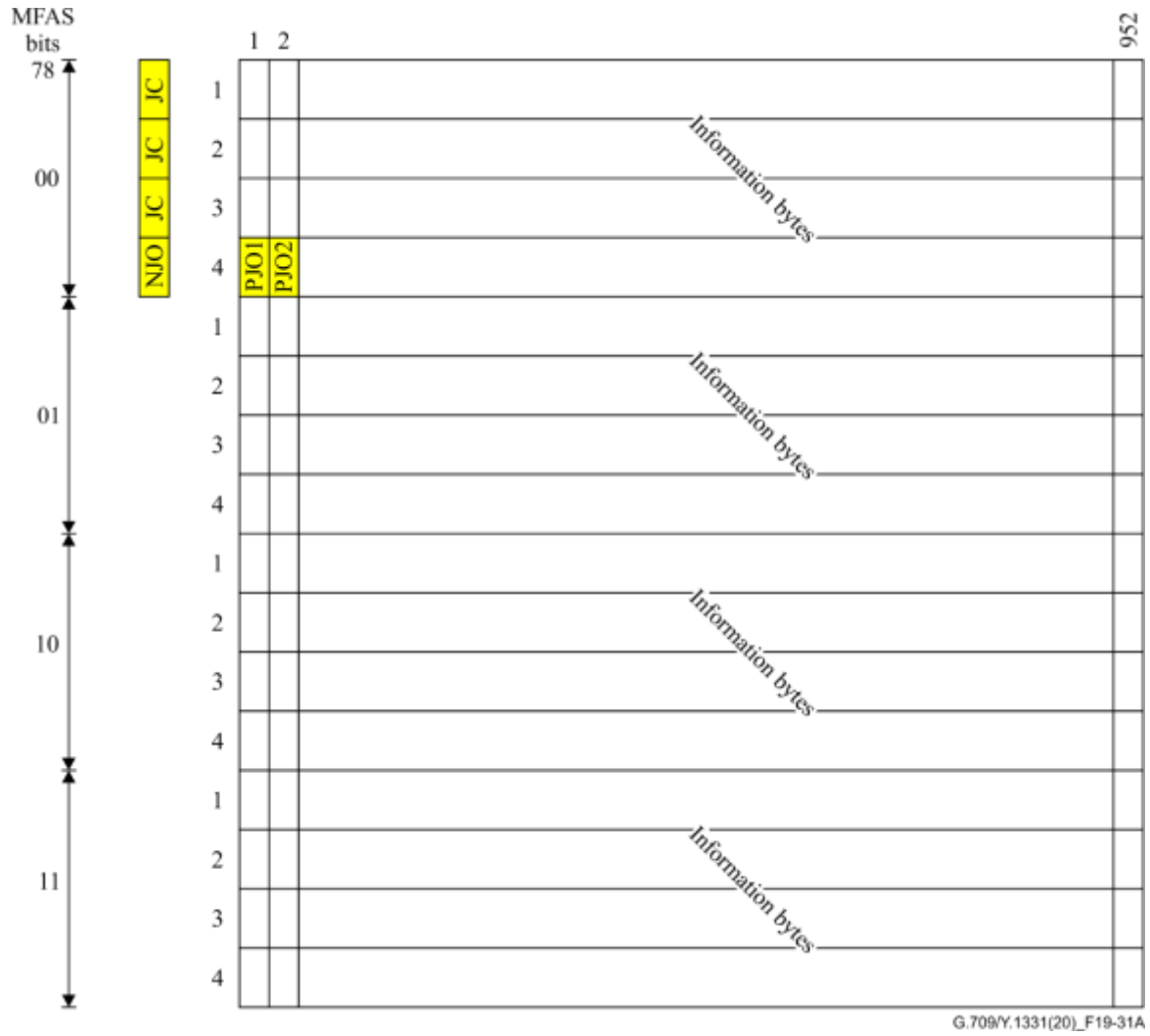


Figure 19-31A – ODTU12 frame format and mapping of ODU1 (mapping in 2.5G TS1)

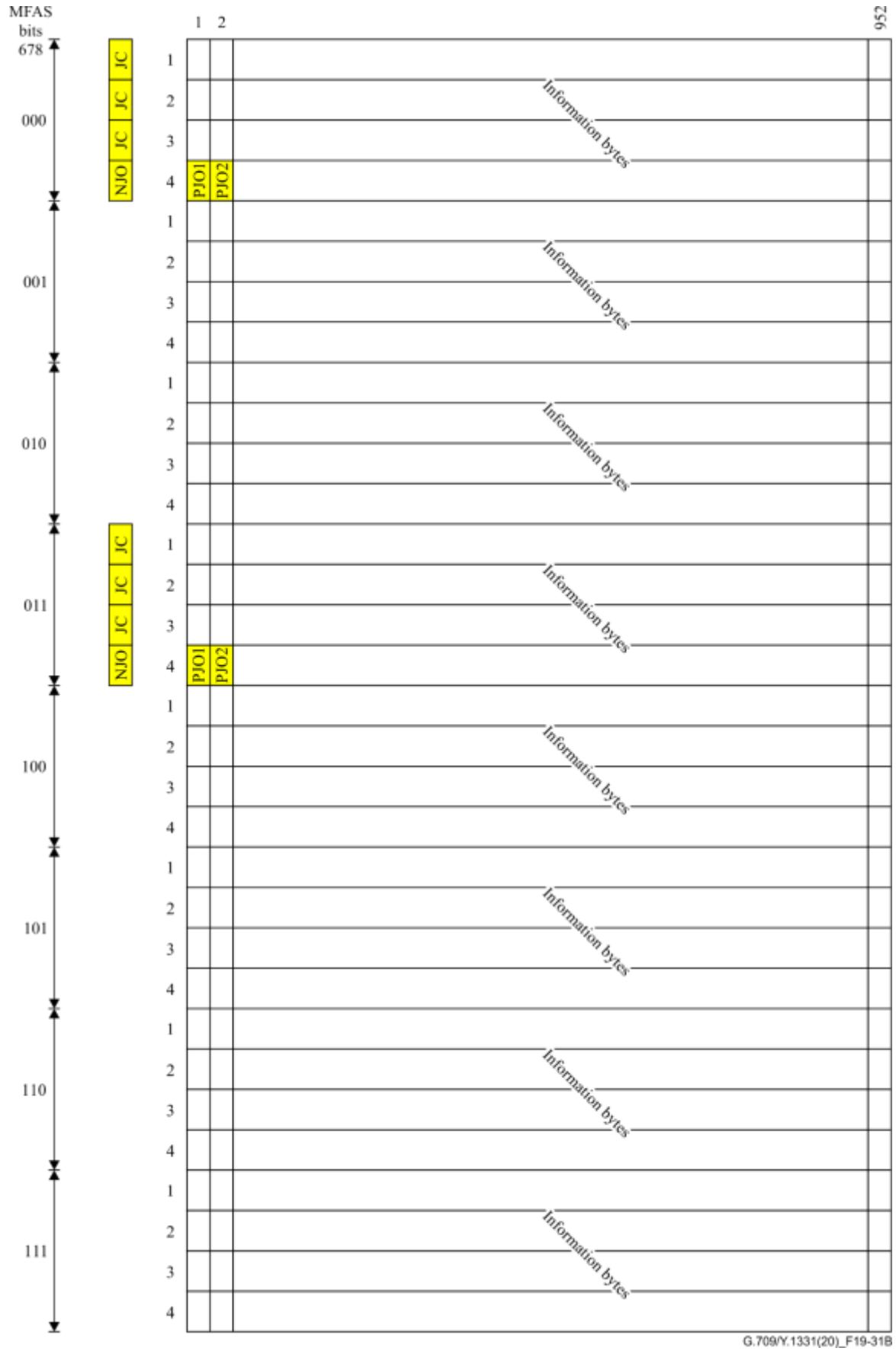


Figure 19-31B – ODTU12 frame format and mapping of ODU1 (mapping in 1.25G TS1 and TS4)

19.5.2 Mapping ODU1 into ODTU13

A byte of the ODU1 signal is mapped into an information byte of the ODTU13 (Figure 19-32A). Column 119 of the ODTU13 is fixed stuff. An all-0s pattern is inserted in the fixed stuff bytes. Once per 16 OPU3 frames, it is possible to perform either a positive or a negative justification action. The frame in which justification can be performed is related to the TSOH of the OPU3 2.5G TS in which the ODTU13 is mapped (Figure 19-2). Figure 19-32A shows the case with mapping in OPU3 2.5G TS3.

A byte of the ODU1 signal is mapped into an information byte of the ODTU13 (see Figure 19-32B). Column 119 of the ODTU13 is fixed stuff. An all-0s pattern is inserted in the fixed stuff bytes. Twice per 32 OPU3 frames, it is possible to perform either a positive or a negative justification action. The frames in which justification can be performed are related to the TSOH of the OPU3 1.25G TSs in which the ODTU13 is mapped (Figure 19-2). Figure 19-32B shows the case with mapping in OPU3 1.25G TS2 and TS25.

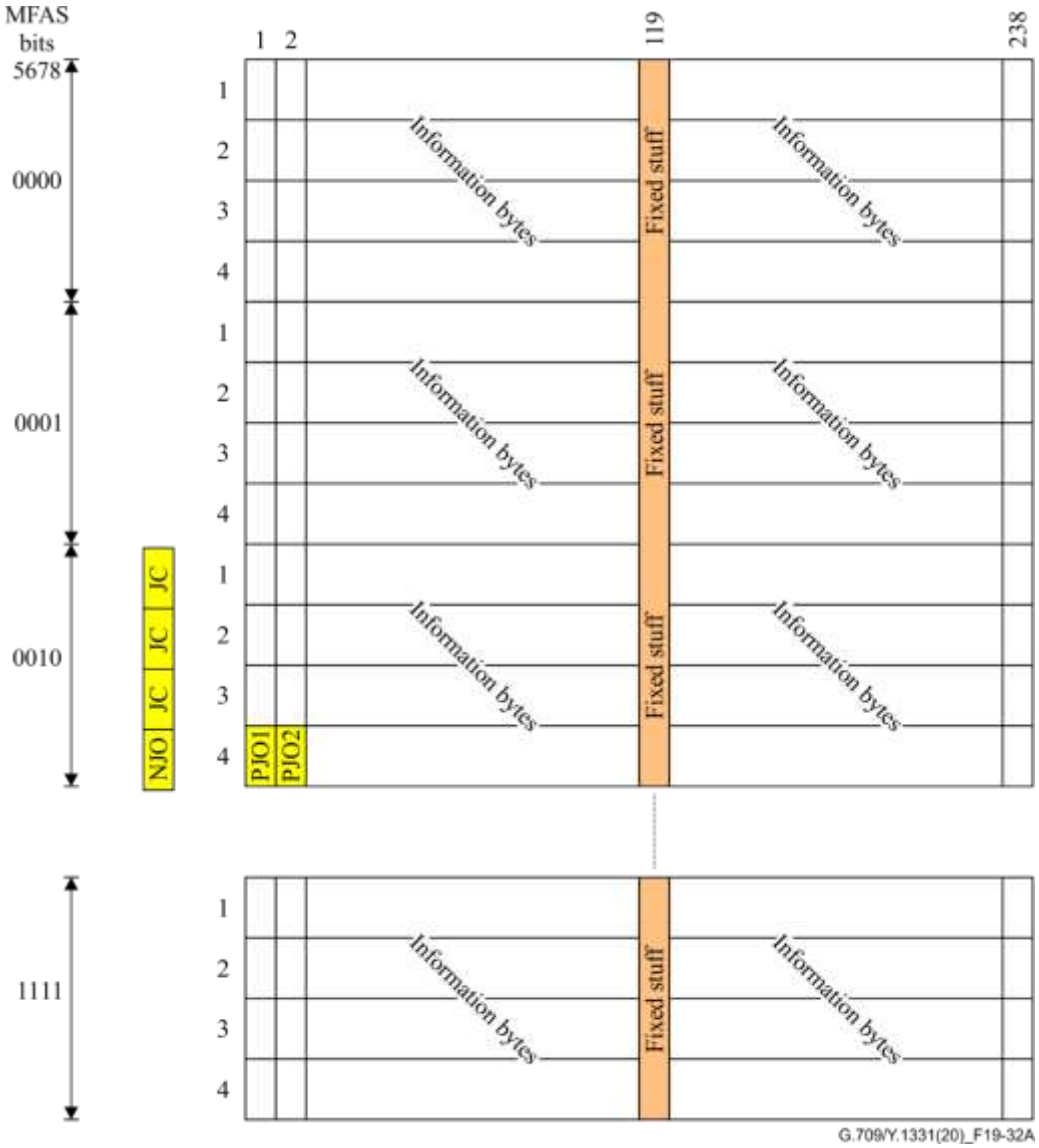


Figure 19-32A – ODTU13 frame format and mapping of ODU1 (mapping in 2.5G TS3)

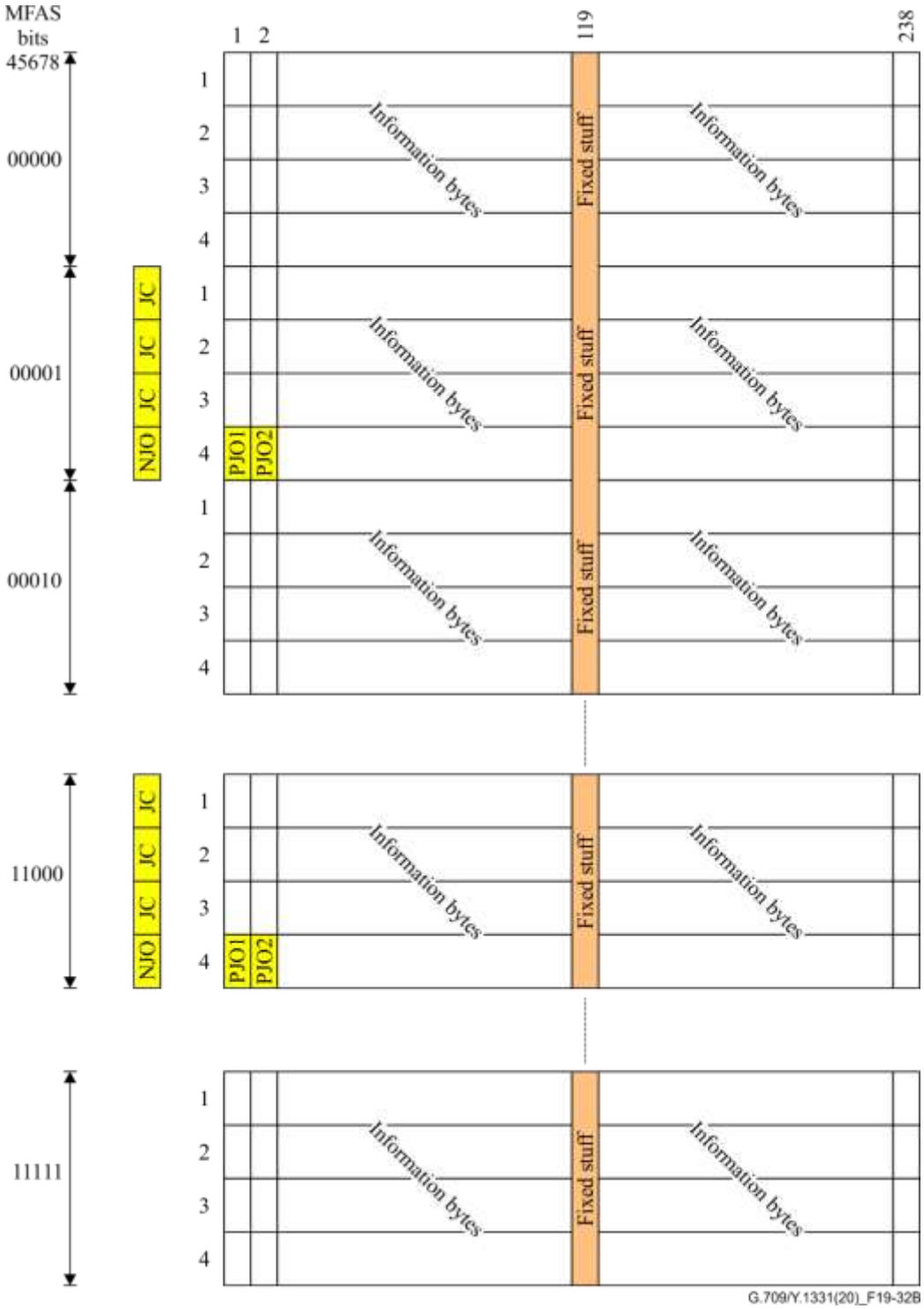


Figure 19-32B – ODTU13 frame format and mapping of ODU1 (mapping in 1.25G TS2 and TS25)

19.5.3 Mapping ODU2 into ODTU23

A byte of the ODU2 signal is mapped into an information byte of the ODTU23 (Figure 19-33A). Four times per sixteen OPU3 frames, it is possible to perform either a positive or a negative justification action. The four frames in which justification can be performed are related to the TSOH of the OPU3 2.5G TSs in which the ODTU23 is mapped (Figure 19-2). Figure 19-33A shows the case with mapping in OPU3 2.5G TS1, TS5, TS9 and TS10.

A byte of the ODU2 signal is mapped into an information byte of the ODTU23 (see Figure 19-33B). Eight times per 32 OPU3 frames, it is possible to perform either a positive or a negative justification action. The frames in which justification can be performed are related to the TSOH of the OPU3 1.25G TSs in which the ODTU23 is mapped (Figure 19-2). Figure 19-33B shows the case with mapping in OPU3 1.25G TS 1, 2, 5, 9, 10, 25, 26 and 32.

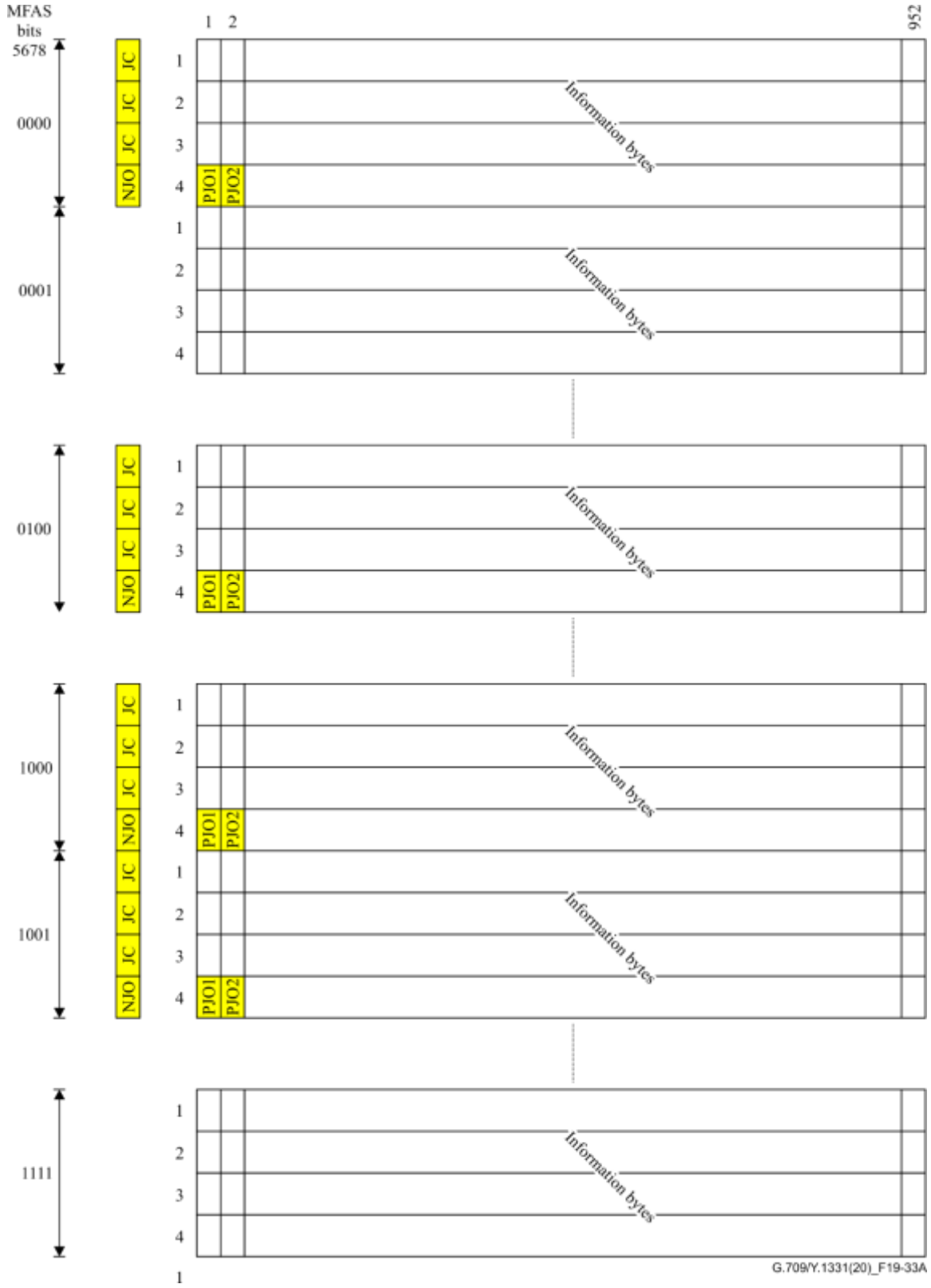


Figure 19-33A – ODTU23 frame format and mapping of ODU2 (mapping in 2.5G TS 1,5,9,10)

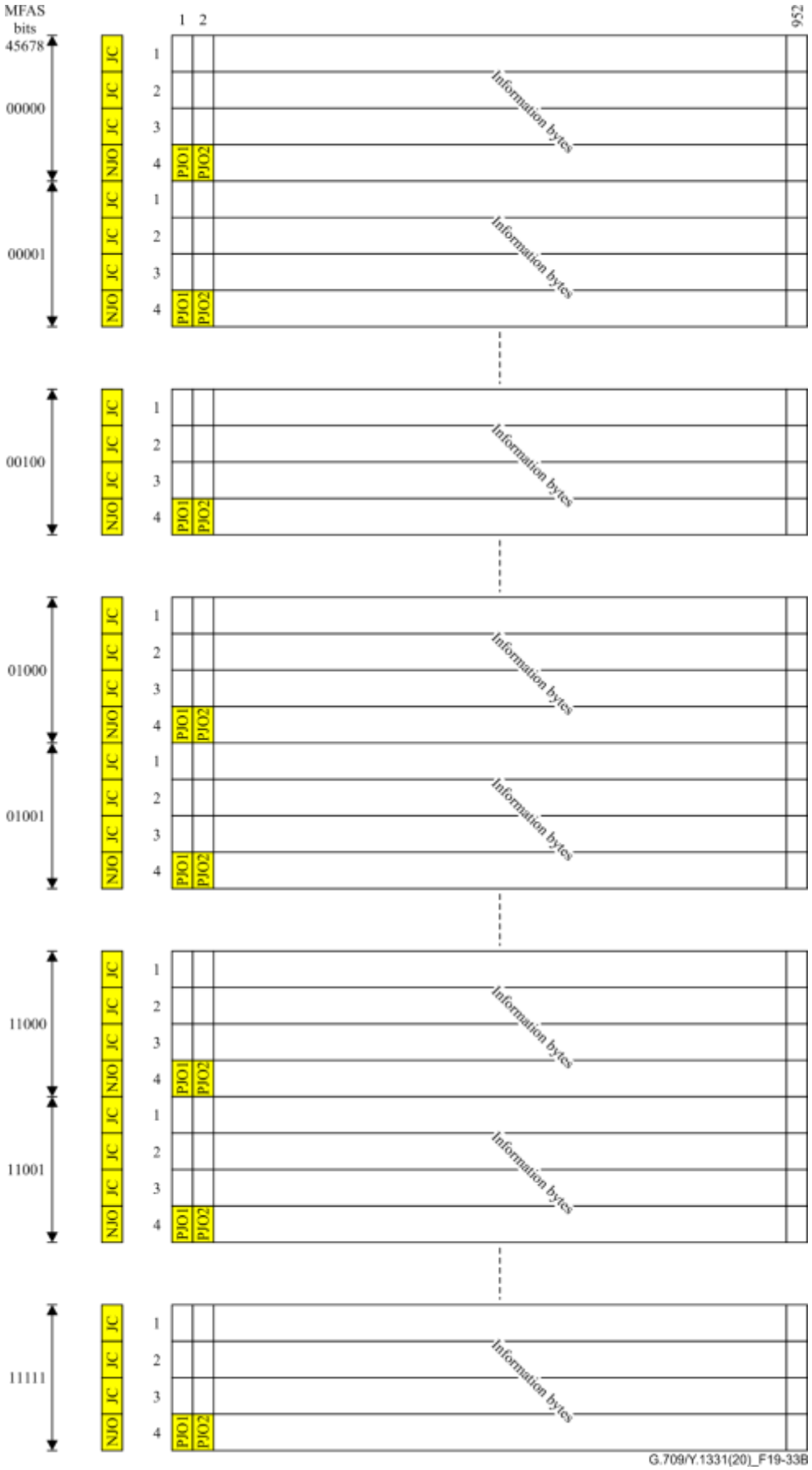


Figure 19-33B – ODTU23 frame format and mapping of ODU2 (mapping in 1.25G TS 1,2,5,9,10,25,26,32)

19.5.4 Mapping ODU0 into ODTU01

A byte of the ODU0 signal is mapped into an information byte of the ODTU01 (see Figure 19-34). Once per 2 OPU1 frames, it is possible to perform either a positive or a negative justification action.

The frame in which justification can be performed is related to the TSOH of the OPU1 TS in which the ODTU01 is mapped (Figure 19-3). Figure 19-34 shows the case with mapping in OPU1 TS1.

NOTE – The PJO2 field will always carry an information byte.

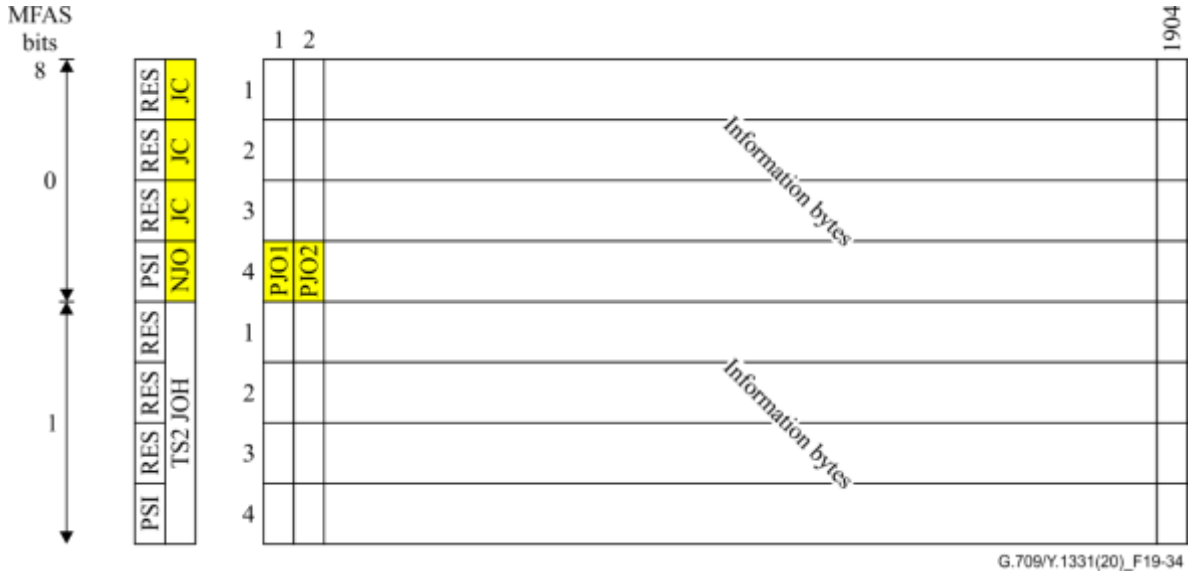


Figure 19-34 – Mapping of ODU0 in OPU1 TS1

19.6 Mapping of ODUj into ODTUk.ts

The mapping of ODUj (j = 0, 1, 2, 2e, 3, flex) signals (with up to ±100 ppm bit-rate tolerance) into the ODTUk.ts (k = 2,3,4,25,50; ts = M) signal is performed by means of a generic mapping procedure as specified in Annex D.

The OPUk and therefore the ODTUk.ts (k = 2,3,4,25,50) signals are created from a locally generated clock (within the limits specified in Table 7-3), which is independent of the ODUj client signal.

The ODUj signal is extended with a frame alignment overhead as specified in clauses 15.6.2.1 and 15.6.2.2 and an all-0s pattern in the OTUj overhead field (see Figure 19-30).

The extended ODUj signal is adapted to the locally generated OPUk/ODTUk.ts clock by means of a generic mapping procedure (GMP) as specified in Annex D. The value of n in c_n and $C_n(t)$ and $C_{nD}(t)$ is specified in Annex D. The value of M is the number of tributary slots occupied by the ODUj; ODTUk.ts = ODTUk.M.

A group of 'M' successive extended ODUj bytes is mapped into a group of 'M' successive ODTUk.M bytes.

The generic mapping process generates for the case of ODUj (j = 0,1,2,2e,3,flex) signals once per ODTUk.M multiframe the $C_m(t)$ and $C_{nD}(t)$ information according to Annex D and encodes this information in the ODTUk.ts justification control overhead JC1/JC2/JC3 and JC4/JC5/JC6. The de-mapping process decodes $C_m(t)$ and $C_{nD}(t)$ from JC1/JC2/JC3 and JC4/JC5/JC6 and interprets $C_m(t)$ and $C_{nD}(t)$ according to Annex D. CRC-8 shall be used to protect against an error in JC1,JC2,JC3 signals. CRC-5 shall be used to protect against an error in JC4,JC5,JC6 signals.

During a signal fail condition of the incoming ODUj signal, this failed incoming signal will contain the ODUj-AIS signal as specified in clause 16.5.1. This ODUj-AIS is then mapped into the ODTUk.M.

For the case where the ODUj is received from the output of a fabric (ODU connection function), the incoming signal may contain (in the case of an open matrix connection) the ODUj-OCI signal as specified in clause 16.5.2. This ODUj-OCI signal is then mapped into the ODTUk.M.

NOTE 1 – Not all equipment will have a real connection function (i.e., switch fabric) implemented; instead, the presence/absence of tributary interface port units represents the presence/absence of a matrix connection. If such a unit is intentionally absent (i.e., not installed), the associated ODTUk.M signals should carry an ODUj-OCI signal. If such a unit is installed but temporarily removed as part of a repair action, the associated ODTUk.M signal should carry an ODUj-AIS signal.

A group of 'M' successive extended ODUj bytes is de-mapped from a group of 'M' successive ODTUk.M bytes.

NOTE 2 – For the case where the ODUj signal is output as an OTUj signal, frame alignment of the extracted extended ODUj signal is to be recovered to allow frame synchronous mapping of the ODUj into the OTUj signal.

During a signal fail condition of the incoming ODUk/OPUk signal (e.g., in the case of an ODUk-AIS, ODUk-LCK, ODUk-OCI condition) the ODUj-AIS pattern as specified in clause 16.5.1 is generated as a replacement signal for the lost ODUj signal.

The values of M, m, $C_{m,min}$, $C_{m,max}$, n, $C_{n,min}$ and $C_{n,max}$ for ODUj into ODTUk.ts are as follows:

$$M = ceiling\left(\frac{ODUj\_nom\_bit\_rate\times(1+ODUj\_bit\_rate\_tolerance+0.00006)}{(ODTUk.1\_nom\_bit\_rate\times(1-ODTUk.ts\_bit\_rate\_tolerance)}\right)$$ for ODUj with j ≠ flex(GFP,n,k') (19-1a)

$$M = \left(\frac{ODUj\_nom\_bit\_rate}{ODUk'.ts}\right) = n$$ for ODUj with j = flex(GFP,n,k') and k' ≤ k with k'∈ {2,3,4} (19-1b)

$$m = 8\times M \quad (19\text{-}2)$$

$$c_{m,nom} = \left(\frac{ODUj\_nom\_bit\_rate\times Number\_of\_GMP\_blocks\_in\_ODTUk.ts}{ODTUk.1\_nom\_bit\_rate\times M}\right) \quad (19\text{-}3)$$

$$c_{m,\min} = c_{m,nom}\times\left(\frac{1-ODUj\_bit\_rate\_tolerance}{1+ODTUk.1\_bit\_rate\_tolerance}\right) \quad (19\text{-}4)$$

$$c_{m,\max} = c_{m,nom}\times\left(\frac{1+ODUj\_bit\_rate\_tolerance}{1-ODTUk.1\_bit\_rate\_tolerance}\right) \quad (19\text{-}5)$$

$$C_{m,\min} = floor\left(c_{m,\min}\right) \quad (19\text{-}6)$$

$$C_{m,max} = ceiling(c_{m,max}) \quad (19\text{-}7)$$

$$n = 8 \quad (19\text{-}8)$$

$$c_{n,nom} = \left(\frac{ODUj\_nom\_bit\_rate\times Number\_of\_GMP\_blocks\_in\_ODTUk.ts}{ODTUk.1\_nom\_bit\_rate}\right) \quad (19\text{-}9)$$

$$c_{n,\min} = c_{n,nom}\times\left(\frac{1-ODUj\_bit\_rate\_tolerance}{1+ODTUk.1\_bit\_rate\_tolerance}\right) \quad (19\text{-}10)$$

$$c_{n,\max} = c_{n,nom}\times\left(\frac{1+ODUj\_bit\_rate\_tolerance}{1-ODTUk.1\_bit\_rate\_tolerance}\right) \quad (19\text{-}11)$$

$$C_{n,\min} = floor\left(c_{n,\min}\right) \quad (19\text{-}12)$$

$$C_{n,max} = ceiling(c_{n,max}) \quad (19\text{-}13)$$

$C_{m,min}$, $C_{n,min}$ (n=8), $C_{m,max}$ and $C_{n,max}$ (n=8) values represent the boundaries of ODU*j*/ODTU*k.M* ppm offset combinations (i.e., min. ODU*j*/max. ODTU*k*.M and max. ODU*j*/min. ODTU*k*.M). In steady state, given instances of ODU*k*/ODTU*k*.M offset combinations should not result in generated C_n and C_m values throughout this range but rather should be within as small a range as possible.

NOTE 3 – Under transient ppm offset conditions (e.g., AIS to normal signal), it is possible that C_n and C_m values outside the range $C_{n,min}$ to $C_{n,max}$ and $C_{m,min}$ to $C_{m,max}$ may be generated and a GMP de-mapper should be tolerant of such occurrences. Refer to Annex D for a general description of the GMP principles.

19.6.1 Mapping ODUj into ODTU2.M

Groups of M successive bytes of the extended ODUj (j = 0, flex) signal are mapped into a group of M successive bytes of the ODTU2.M payload area under control of the GMP data/stuff control mechanism. Each group of M bytes in the ODTU2.M payload area may either carry M ODU bytes, or carry M stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of M bytes in the ODTU2.M payload area are numbered from 1 to 15232.

The ODTU2.M payload byte numbering for GMP M-byte (m-bit) blocks is illustrated in Figure 19-35. In row 1 of the ODTU2.M multiframe the first M-bytes will be labelled 1, the next M-bytes will be labelled 2, etc.

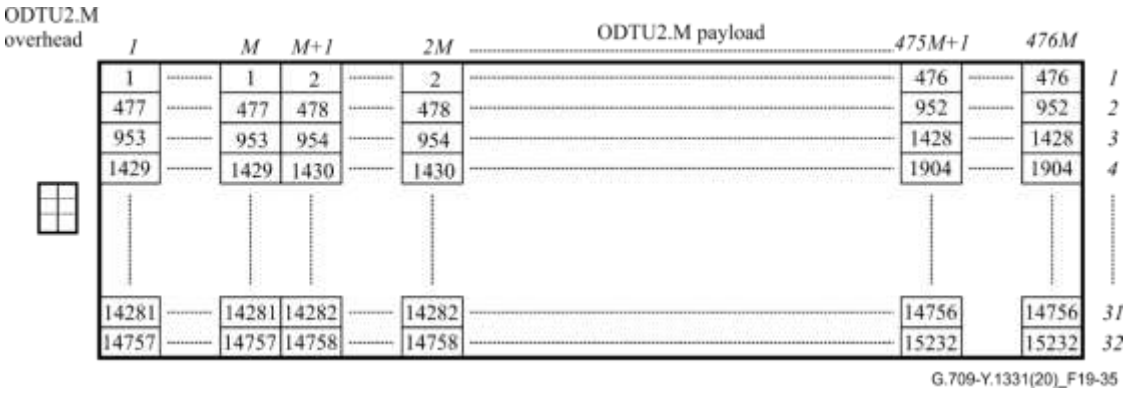


Figure 19-35 – ODTU2.M GMP byte numbering

19.6.2 Mapping ODUj into ODTU3.M

Groups of M successive bytes of the extended ODUj (j = 0, 2e, flex) signal are mapped into a group of M successive bytes of the ODTU3.M payload area under control of the GMP data/stuff control mechanism. Each group of M bytes in the ODTU3.M payload area may either carry M ODU bytes, or carry M stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of M bytes in the ODTU3.M payload area are numbered from 1 to 15232.

The ODTU3.M payload byte numbering for GMP M-byte (m-bit) blocks is illustrated in Figure 19-36. In row 1 of the ODTU3.M multiframe the first M-bytes will be labelled 1, the next M-bytes will be labelled 2, etc.

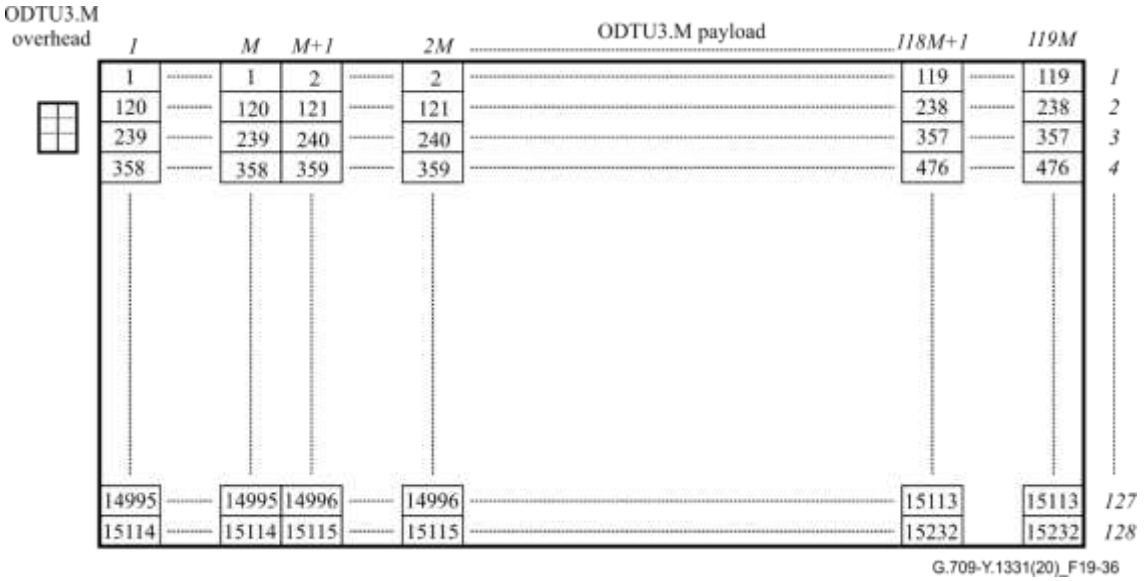


Figure 19-36 – ODTU3.M GMP byte numbering

19.6.3 Mapping ODUj into ODTU4.M

Groups of M successive bytes of the extended ODUj (j = 0, 1, 2, 2e, 3, flex) signal are mapped into a group of M successive bytes of the ODTU4.M payload area under control of the GMP data/stuff control mechanism. Each group of M bytes in the ODTU4.M payload area may either carry M ODU bytes, or carry M stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of M bytes in the ODTU4.M payload area are numbered from 1 to 15200.

The ODTU4.M payload byte numbering for GMP M-byte (m-bit) blocks is illustrated in Figure 19-37. In row 1 of the ODTU4.M multiframe the first M-bytes will be labelled 1, the next M-bytes will be labelled 2, etc.

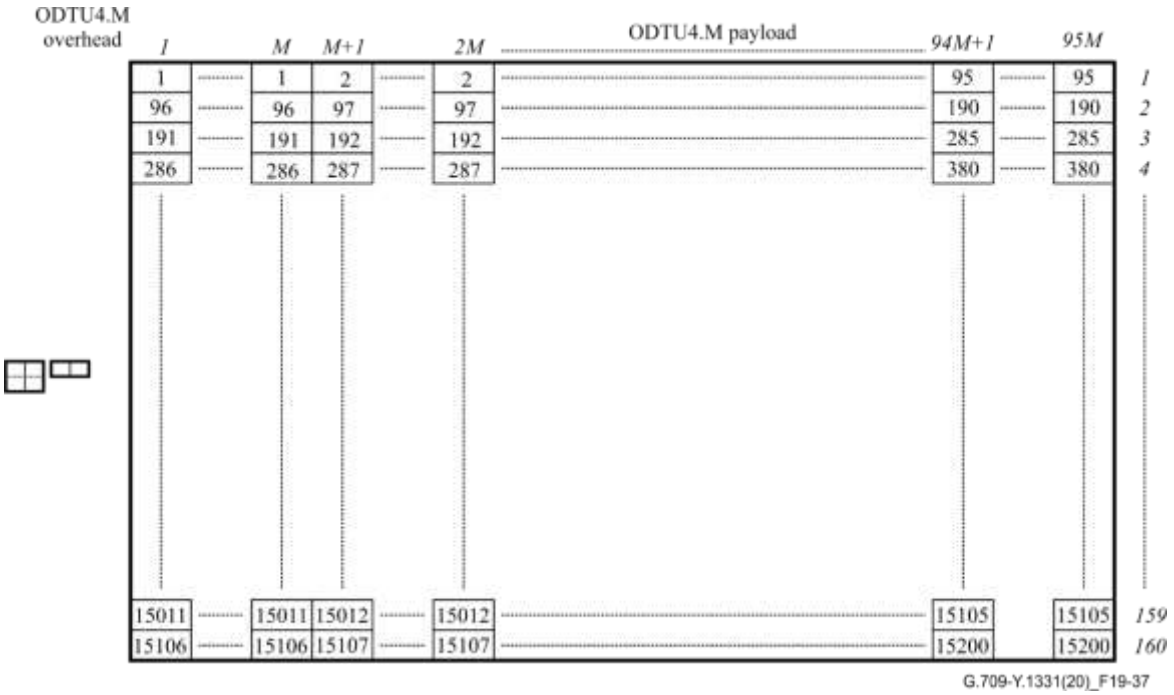


Figure 19-37 – ODTU4.M GMP byte numbering

19.6.4 Mapping ODUj into ODTU25.M

Groups of M successive bytes of the extended ODUj (j = 0, 1, 2, 2e, flex) signal are mapped into a group of M successive bytes of the ODTU25.M payload area under control of the GMP data/stuff control mechanism. Each group of M bytes in the ODTU25.M payload area may either carry M ODU bytes, or carry M stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of M bytes in the ODTU25.M payload area are numbered from 1 to 15232.

The ODTU25.M payload byte numbering for GMP M-byte (m-bit) blocks is illustrated in Figure 19-38. In row 1 of the ODTU25.M multiframe the first M-bytes will be labelled 1, the next M-bytes will be labelled 2, etc.

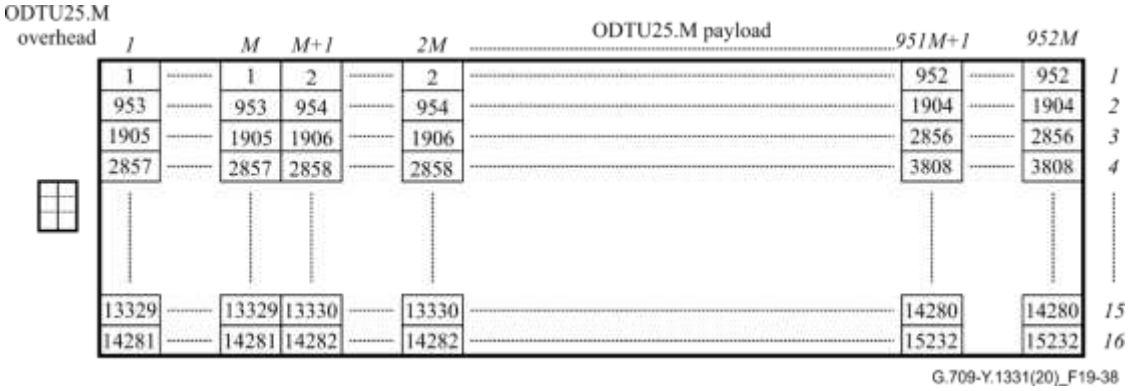


Figure 19-38 – ODTU25.M GMP byte numbering

19.6.5 Mapping ODUj into ODTU50.M

Groups of M successive bytes of the extended ODUj (j = 0, 1, 2, 2e, 3, flex) signal are mapped into a group of M successive bytes of the ODTU50.M payload area under control of the GMP data/stuff control mechanism. Each group of M bytes in the ODTU50.M payload area may either carry M ODU bytes, or carry M stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of M bytes in the ODTU50.M payload area are numbered from 1 to 15232.

The ODTU50.M payload byte numbering for GMP M-byte (m-bit) blocks is illustrated in Figure 19-39. In row 1 of the ODTU50.M multiframe the first M-bytes will be labelled 1, the next M-bytes will be labelled 2, etc.

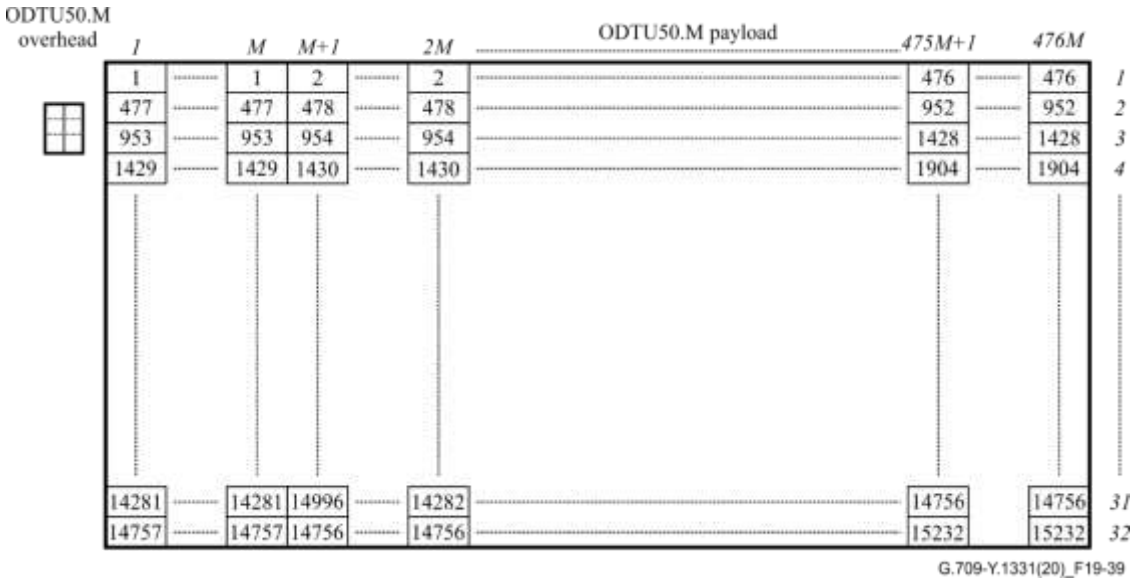


Figure 19-39 – ODTU50.M GMP byte numbering

20 Mapping ODUk signals into the ODTUCn signal and the ODTUCn into the OPUCn tributary slots

This clause specifies the multiplexing of:

– ODU*k* into OPU*Cn* using a client agnostic generic mapping procedure (GMP).

This ODU*k* into OPU*Cn* multiplexing is performed in two steps:

1) asynchronous mapping of ODU*k* into optical data tributary unit (ODTU*Cn*) using GMP;

2) byte-synchronous mapping of ODTU*Cn* into one or more OPUCn tributary slots.

The OPUCn supports up to 10n different ODUk signals.

20.1 OPUCn tributary slot definition

The OPUCn consists of n OPUC. Each OPUC is divided into 20 tributary slots (TS) and these tributary slots are 16-byte interleaved within the OPUC payload area. A tributary slot includes a part of the OPUC OH area and a part of the OPUC payload area. The bytes of the ODUk frame are mapped into the ODTUCn payload area and the ODTUCn bytes are mapped into the OPUCn tributary slot or slots. The bytes of the ODTUCn justification overhead are mapped into the OPUCn OH area.

There is *only* one type of tributary slot:

1) Tributary slot with a bandwidth of approximately 5 Gbit/s; an OPUCn is divided into 20n tributary slots, numbered 1.1 to n.20.

20.1.1 OPUCn tributary slot allocation

Figures 20-1 and 20-2 present the OPUC 5G tributary slot allocation. An OPUC is divided into 20 5G tributary slots (named TS #A.B where A = 1…n which denotes the number of the OPUC within the OPUCn and B = 1…20 which denotes the number of the TS within the OPUC), which are located in columns 17 to 3824. The OPUC multi-frame may be represented in an 80 row by 3810 column format (Figure 20-1) and in a 8 row by 38100 column format (Figure 20-2).

An OPUC 5G tributary slot occupies 5% of the OPUC payload area. It is a structure with 119 16-byte columns by 8*(20×4/10)* rows (see Figure 20-2) plus a tributary slot overhead (TSOH). The 20 OPUC 5G TSs are 16-byte interleaved in the OPUC payload area and the 20 OPUC TSOHs are frame interleaved in the OPUC overhead area.

The tributary slot overhead (TSOH) of OPUC tributary slots is located in rows 1 to 3, columns 15 and 16 of the OPUC frame.

The TSOH for a 5G tributary slot is available once every 20 frames. A 20-frame multiframe structure is used for this assignment. This multiframe structure is locked to bits 4, 5, 6, 7 and 8 of the OMFI byte as shown in Table 20-1 and Figure 20-1.

Figure 20-3 presents the 5G tributary slots in the OPUCn for the case of 16-byte interleaving of the OPUC instances. This interleaving presents the tributary slot order within the OPUCn.

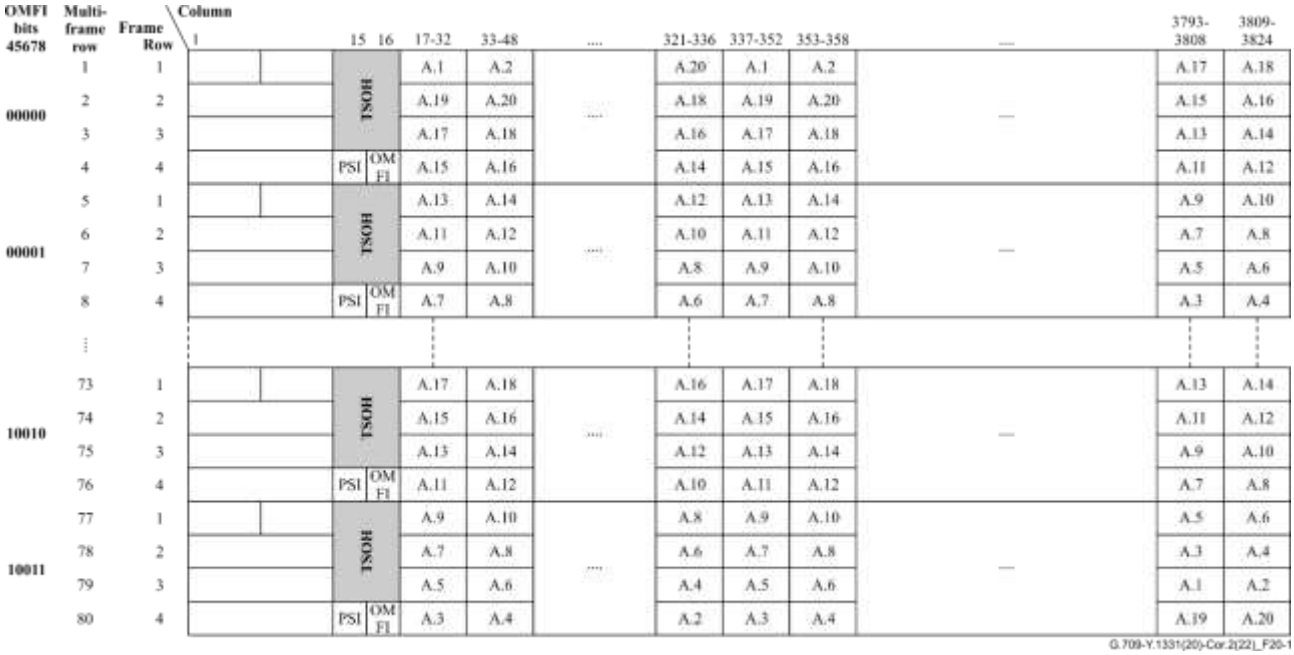


Figure 20-1 – OPUC tributary slot allocation in 80 row x 3810 column format

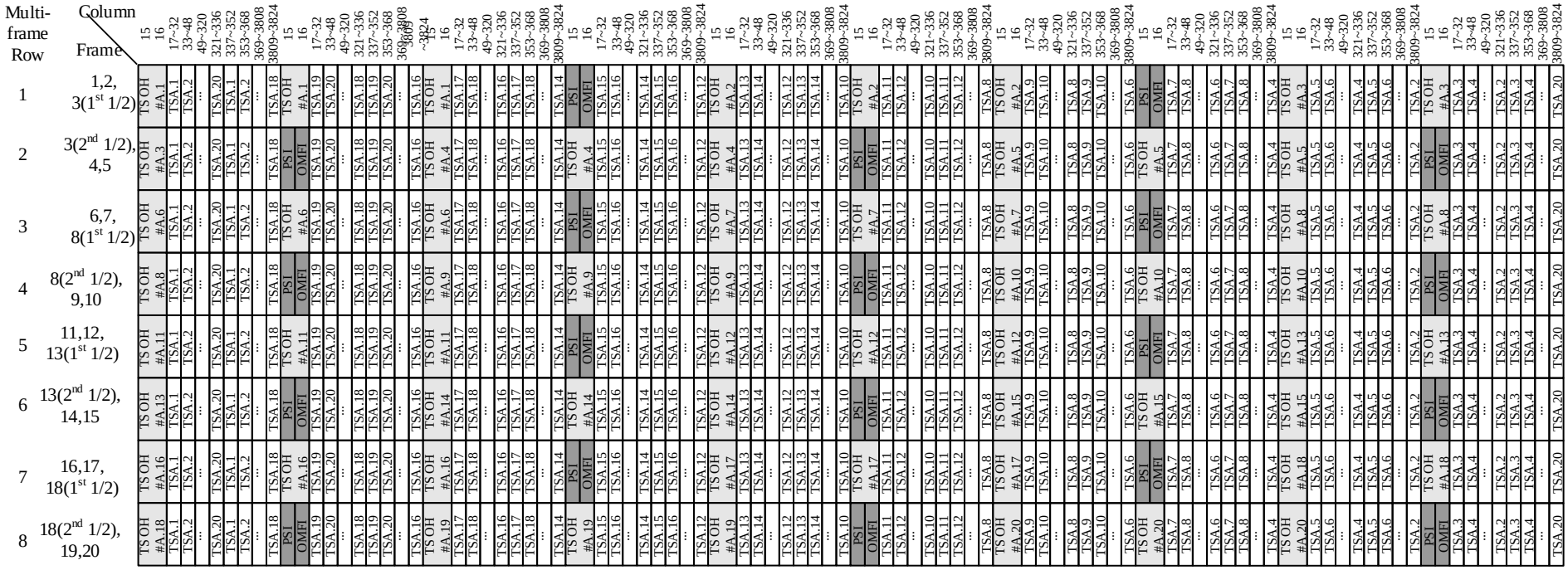


Figure 20-2 – OPUC tributary slot allocation in 8 row x 38100 column format

3808 x n; 20 x16 x n; 218 x16 x n; 2 x n; 16 x n

OMFI bits 45678	Column / Row	2 x n	16 x n				16 x n					16 x n				16 x n					16 x n			
00000	1	OPUCn OH (TSOH)	TS1.1	TS2.1	...	TSn.1	TS1.2	TS2.2	...	TSn.2	...	TS1.20	TS2.20	...	TSn.20	TS1.1	TS2.1	...	TSn.1	...	TS1.18	TS2.18	...	TSn.18
	2		TS1.19	TS2.19	...	TSn.19	TS1.20	TS2.20	...	TSn.20	...	TS1.18	TS2.18	...	TSn.18	TS1.19	TS2.19	...	TSn.19	...	TS1.16	TS2.16	...	TSn.16
	3		TS1.17	TS2.17	...	TSn.17	TS1.18	TS2.18	...	TSn.18	...	TS1.16	TS2.16	...	TSn.16	TS1.17	TS2.17	...	TSn.17	...	TS1.14	TS2.14	...	TSn.14
	4		TS1.15	TS2.15	...	TSn.15	TS1.16	TS2.16	...	TSn.16	...	TS1.14	TS2.14	...	TSn.14	TS1.15	TS2.15	...	TSn.15	...	TS1.12	TS2.12	...	TSn.12
00001	1	OPUCn OH (TSOH)	TS1.13	TS2.13	...	TSn.13	TS1.14	TS2.14	...	TSn.14	...	TS1.12	TS2.12	...	TSn.12	TS1.13	TS2.13	...	TSn.13	...	TS1.10	TS2.10	...	TSn.10
	2		TS1.11	TS2.11	...	TSn.11	TS1.12	TS2.12	...	TSn.12	...	TS1.10	TS2.10	...	TSn.10	TS1.11	TS2.11	...	TSn.11	...	TS1.8	TS2.8	...	TSn.8
	3		TS1.9	TS2.9	...	TSn.9	TS1.10	TS2.10	...	TSn.10	...	TS1.8	TS2.8	...	TSn.8	TS1.9	TS2.9	...	TSn.9	...	TS1.6	TS2.6	...	TSn.6
	4		TS1.7	TS2.7	...	TSn.7	TS1.8	TS2.8	...	TSn.8	...	TS1.6	TS2.6	...	TSn.6	TS1.7	TS2.7	...	TSn.7	...	TS1.4	TS2.4	...	TSn.4
00010	1	OPUCn OH (TSOH)	TS1.5	TS2.5	...	TSn.5	TS1.6	TS2.6	...	TSn.6	...	TS1.4	TS2.4	...	TSn.4	TS1.5	TS2.5	...	TSn.5	...	TS1.2	TS2.2	...	TSn.2
	2		TS1.3	TS2.3	...	TSn.3	TS1.4	TS2.4	...	TSn.4	...	TS1.2	TS2.2	...	TSn.2	TS1.3	TS2.3	...	TSn.3	...	TS1.20	TS2.20	...	TSn.20
	3		TS1.1	TS2.1	...	TSn.1	TS1.2	TS2.2	...	TSn.2	...	TS1.20	TS2.20	...	TSn.20	TS1.1	TS2.1	...	TSn.1	...	TS1.18	TS2.18	...	TSn.18
	4		TS1.19	TS2.19	...	TSn.19	TS1.20	TS2.20	...	TSn.20	...	TS1.18	TS2.18	...	TSn.18	TS1.19	TS2.19	...	TSn.19	...	TS1.16	TS2.16	...	TSn.16
......		…...																						
10011	1	OPUCn OH (TSOH)	TS1.9	TS2.9	...	TSn.9	TS1.10	TS2.10	...	TSn.10	...	TS1.8	TS2.8	...	TSn.8	TS1.9	TS2.9	...	TSn.9	...	TS1.6	TS2.6	...	TSn.6
	2		TS1.7	TS2.7	...	TSn.7	TS1.8	TS2.8	...	TSn.8	...	TS1.6	TS2.6	...	TSn.6	TS1.7	TS2.7	...	TSn.7	...	TS1.4	TS2.4	...	TSn.4
	3		TS1.5	TS2.5	...	TSn.5	TS1.6	TS2.6	...	TSn.6	...	TS1.4	TS2.4	...	TSn.4	TS1.5	TS2.5	...	TSn.5	...	TS1.2	TS2.2	...	TSn.2
	4		TS1.3	TS2.3	...	TSn.3	TS1.4	TS2.4	...	TSn.4	...	TS1.2	TS2.2	...	TSn.2	TS1.3	TS2.3	...	TSn.3	...	TS1.20	TS2.20	...	TSn.20

Figure 20-3 – OPUCn tributary slot allocation with n OPUC 16-byte interleaved

Table 20-1 – OPUCn tributary slot OH allocation

OMFI **bits** **4 5 6 7 8**	**TSOH** **5G TS**
0 0 0 0 0	A.1
0 0 0 0 1	A.2
0 0 0 1 0	A.3
0 0 0 1 1	A.4
0 0 1 0 0	A.5
0 0 1 0 1	A.6
0 0 1 1 0	A.7
0 0 1 1 1	A.8
0 1 0 0 0	A.9
0 1 0 0 1	A.10
0 1 0 1 0	A.11
0 1 0 1 1	A.12
0 1 1 0 0	A.13
0 1 1 0 1	A.14
0 1 1 1 0	A.15
0 1 1 1 1	A.16
1 0 0 0 0	A.17
1 0 0 0 1	A.18
1 0 0 1 0	A.19
1 0 0 1 1	A.20

20.2 ODTUCn definition

The optical data tributary unit *Cn* (ODTU*Cn*) carries a justified ODUk signal. There is one type of ODTU*Cn*:

– ODTU*Cn.ts* (ts = 1 to 20n) in which a ODU*k* (k = 0,1,2,2e,3,4,flex) signal is mapped via the generic mapping procedure (GMP) defined in clause 20.5.

Optical data tributary unit Cn.ts (ODTU*Cn.ts*)

The optical data tributary unit Cn.ts (ODTU*Cn*.ts) is a structure which consists of an ODTU*Cn*.ts payload area and an ODTU*Cn.ts* overhead area (Figure 20-4). The ODTU*Cn.ts* payload area has 119×ts (ts=1 to 20n) 16-byte-columns and 8 rows (15232×ts bytes) and the ODTU*Cn.ts* overhead area has one times 6 bytes. The ODTU*Cn.ts* is carried in "ts" 5G tributary slots of an OPU*Cn*.

The location of the ODTU*Cn.ts* overhead depends on the OPU*Cn* tributary slot used when multiplexing the ODTU*Cn.ts* in the OPU*Cn* (see clause 20.1.1). The single instance of an ODTU*Cn*.ts overhead is located in the OPU*Cn* TSOH of the last OPU*Cn* tributary slot allocated to the ODTU*Cn.ts*.

The ODTU*Cn.ts* overhead carries the GMP justification overhead as specified in clause 20.4.

The ODTU*Cn.ts* overhead carries the GMP justification overhead consisting of 6 bytes of justification control (JC1, JC2, JC3, JC4, JC5, JC6) which carry the GMP C_m and ΣC_{8D} information.

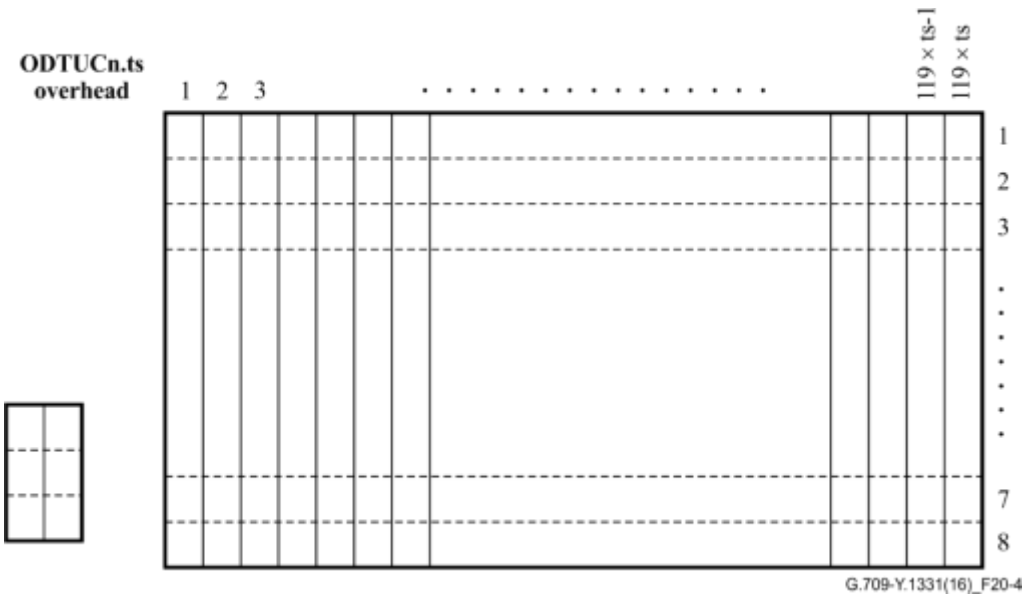


Figure 20-4 – ODTUCn.ts frame formats

20.3 Multiplexing ODTUCn signals into the OPUCn

Multiplexing an ODTU*Cn.ts* signal into an OPU*Cn* is realized by mapping the ODTU*Cn.ts* signal into ts (of 20n) arbitrary OPU*Cn* 5G tributary slots: OPUCn TS #A_1.B_1, TS #A_2.B_2, .. , TS #A_{ts}.B_{ts} with $1 \leq n\times(B_1-1)+A_1 < n\times(B_2-1)+A_2 < .. < n\times (B_{ts}-1)+A_{ts} \leq 20n$.

NOTE 1 – TS #A_1.B_1, TS #A_2.B_2, ..., TS #A_{ts}.B_{ts} do not have to be sequential in the OPU*Cn* TS sequence; the TSs can be arbitrarily selected to prevent bandwidth fragmentation.

NOTE 2 – TS order is illustrated in Figure 20-5.

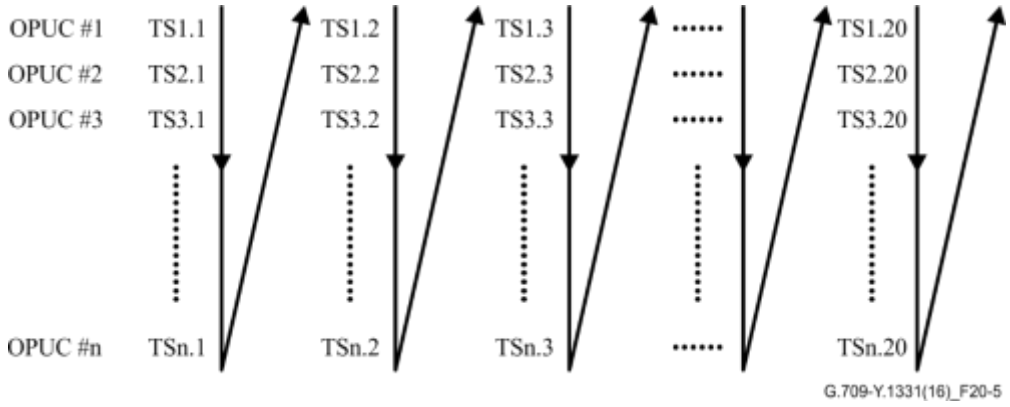


Figure 20-5 – OPUCn TS order

The OPU*Cn* overhead for these multiplexed signals consists of a payload type (PT), the multiplex structure identifier (MSI), the OPU*Cn* multiframe identifier, the OPUCn tributary slot overhead carrying the ODTU*Cn* overhead and depending on the ODTU type one or more bytes reserved for future international standardization.

20.3.1 ODTUCn.ts mapping into ts OPUCn 5G tributary slots

A 16-byte of the ODTU*Cn.ts* payload signal is mapped into a 16-byte of an OPU*Cn* 5G TS #Ai.Bi (i = 1,..,ts) payload area, as indicated in Figure 20-6.

A byte of the ODTU*Cn.ts* overhead is mapped into a TSOH byte of TS #A_{ts}.B_{ts} within columns 15 and 16, rows 1 to3 of the last OPU*Cn* 5G tributary slot allocated to the ODTU*Cn.ts*.

The remaining OPU*Cn* TSOH bytes are reserved for future international standardization.

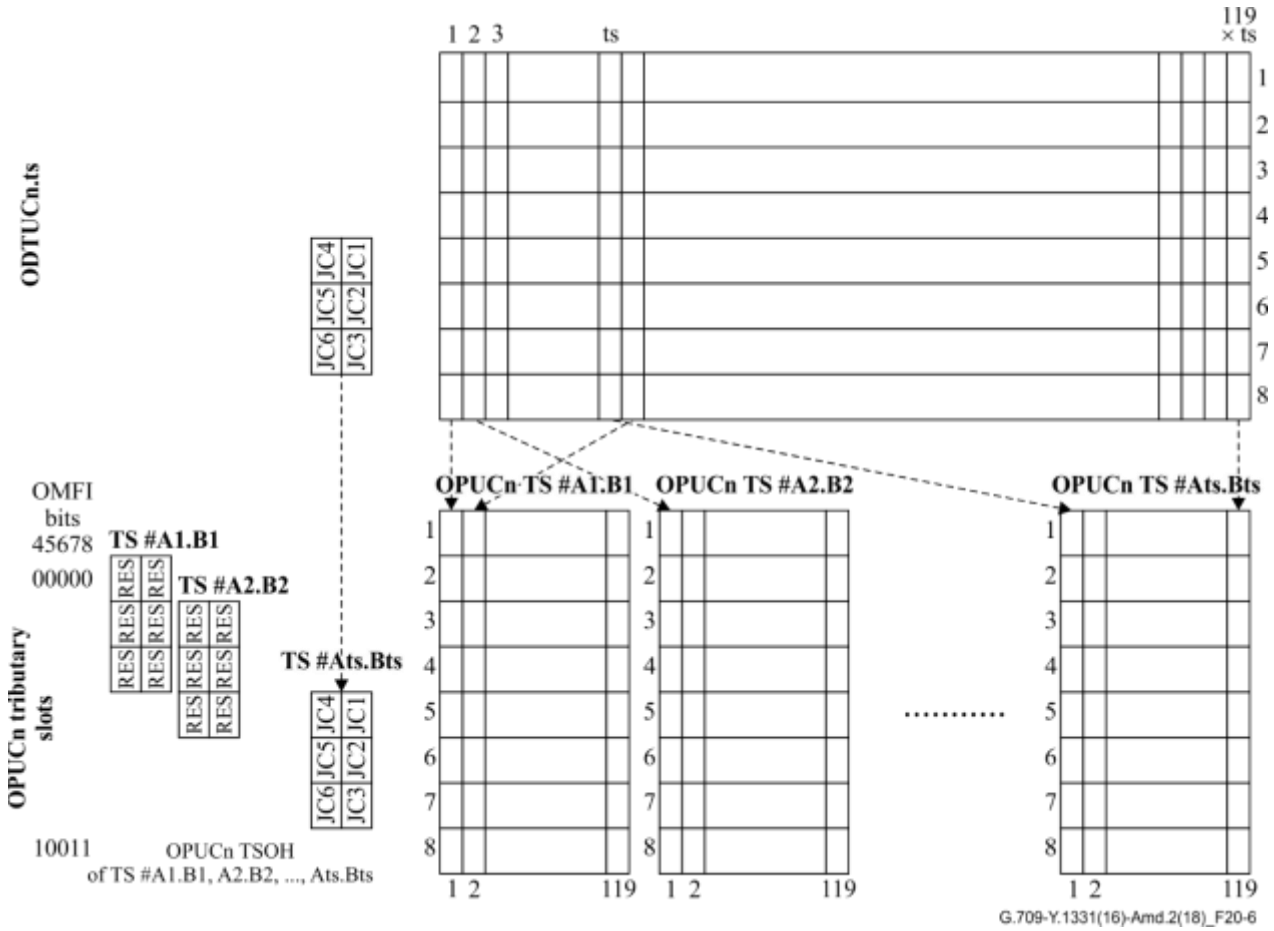


Figure 20-6 – Mapping of ODTU*Cn.ts* into 'ts' OPU*Cn* 5G tributary slots

20.4 OPUCn multiplex overhead and ODTU justification overhead

The OPU*Cn* multiplex overhead consists of a multiplex structure identifier (MSI), an OPU multiframe identifier (OMFI), an ODTU overhead and bytes reserved for future international standardization.

The OPU*Cn* MSI, OMFI and RES overhead locations are shown in Figure 20-7.

ODTUCn.ts overhead

The ODTU*Cn.ts* overhead carries the GMP justification overhead consisting of 18 bits of justification control (JC1[3-8], JC2[3-8], JC3[3-8]) which carry the 10-bit GMP C_m information and ODU*k* (k=0,1,2,2e,3,4,flex) specific 30 bits of justification control (JC1[1-2], JC2[1-2], JC3[1-2], JC4, JC5, JC6) which carry the 18-bit GMP ΣC_{8D} information.

The JC1, JC2, JC3, JC4, JC5 and JC6 overhead locations are shown in Figure 20-7.

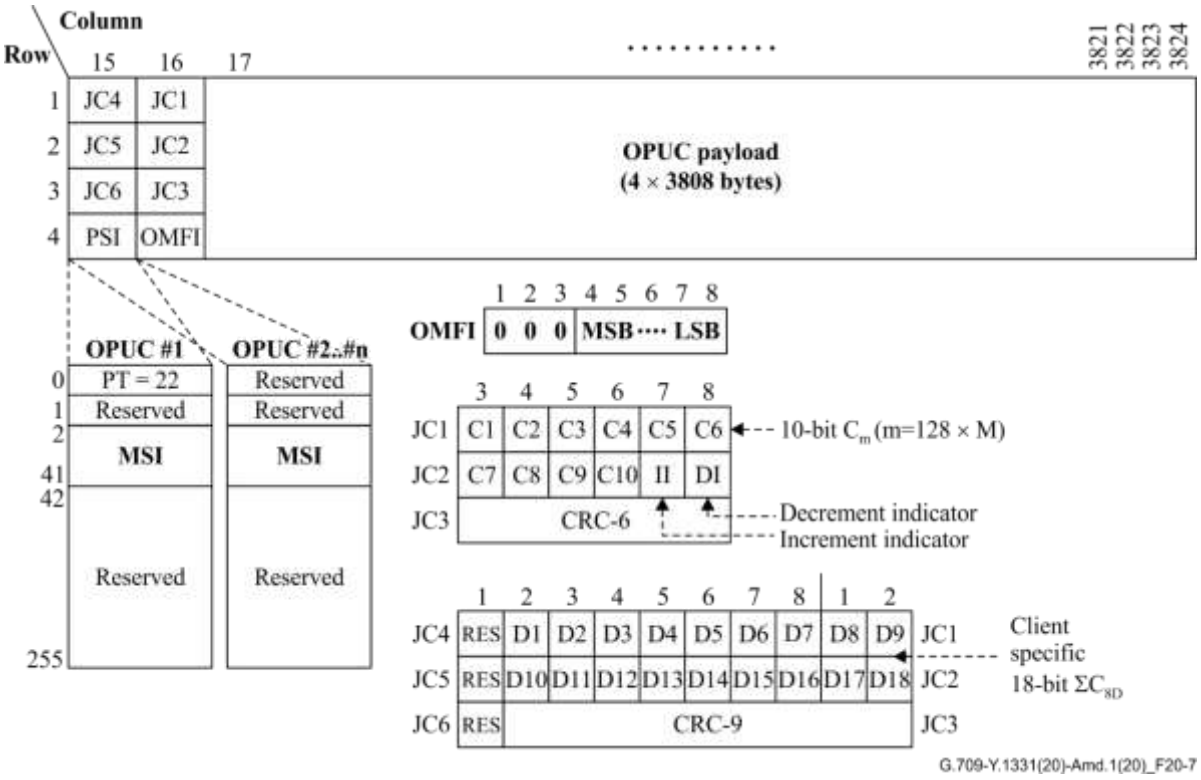


Figure 20-7 – OPU*Cn* multiplex overhead (payload type = 22)

20.4.1 OPUCn multiplex structure identifier (MSI)

The OPU*Cn* multiplex structure identifier (MSI) overhead, which encodes the ODU multiplex structure in the OPU, is located in the mapping specific area of the PSI signal (refer to Figure 20-7 for the MSI location in OPU*Cn* with PT=22. The MSI has a fixed length of 40n bytes and indicates the ODTU content of each tributary slot (TS) of an OPU*Cn*. Two bytes are used for each TS.

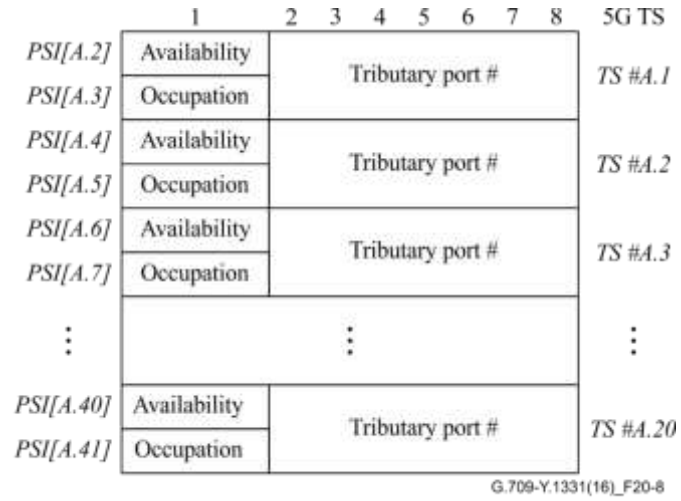
20.4.1.1 OPUCn multiplex structure identifier (MSI) – Payload type 22

For the 20n OPU*Cn* 5G tributary slots n times 40 bytes of the PSI are used (named PSI[x.y] where x = 1…n and y = 2…41, that is PSI[1.2], PSI[1.3],…PSI[1.41], PSI[2.2], PSI[2.3],…, PSI[2.41], PSI[3.2],…,PSI[n-1.41], PSI[n.2]…, PSI[n.41]) as MSI bytes as shown in Figure 20-7. The MSI indicates the ODTU content of each tributary slot of an OPU. Two bytes are used for each tributary slot as illustrated in Figure 20-8 and the encoding of the fields of each MSI instance is illustrated in Figure 20-9.

– The TS availability bit 1 indicates if the tributary slot is available or unavailable.

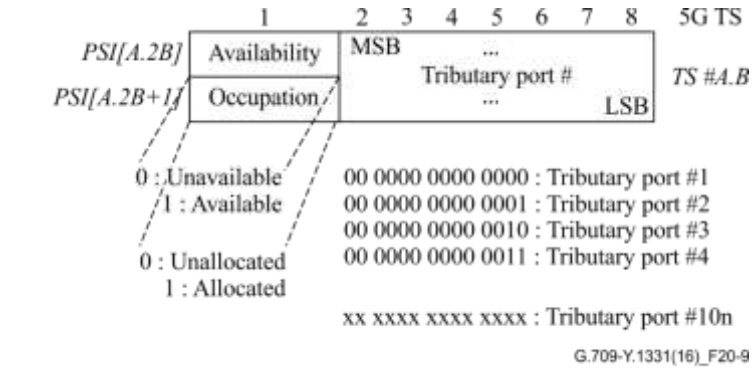
– The TS occupation bit 9 indicates if the tributary slot is allocated or unallocated.

– The tributary port # in bits 2 to 8 and 10 to 16 indicates the port number of the ODTUCn.ts that is being transported in this TS; a flexible assignment of tributary port to tributary slots is possible. ODTUCn.ts tributary ports are numbered 1 to 10n. The value is set to all-0s when the occupation bit has the value 0 (tributary slot is unallocated).



NOTE – $A = 1...n$

Figure 20-8 – OPU*Cn* 5G TS MSI coding – Payload type 22



NOTE – $A = 1...n$, $B = 1...20$

Figure 20-9 – OPU*Cn* MSI coding – Payload type 22

20.4.2 OPUCn payload structure identifier reserved overhead (RES)

216×n-1 (OPUCn) bytes are reserved in the OPUCn PSI for future international standardization. These bytes are located in PSI[x.1] and PSI[x.42] to PSI[x.255] of the OPUCn overhead where x = 1…n and PSI[x.0], x=2 to n. These bytes are set to all-0s.

20.4.3 OPUCn multiplex justification overhead (JOH)

The generic mapping procedure (GMP) is used for the mapping of an ODU*k* into ODTU*Cn.ts*. GMP uses ODU*k* and OPU*Cn* independent stuff and justification opportunity definitions (ODTU*Cn.ts*). Stuff locations within an ODTU*Cn.ts* are determined by means of a formula which is specified in clause 20.4.3.1.

20.4.3.1 Generic mapping procedure (GMP)

The justification overhead (JOH) for the generic mapping procedure consists of two groups of three bytes of justification control; JC1, JC2, JC3 and JC4, JC5, JC6. Refer to Figure 20-7.

The bits 3 to 8 of the JC1, JC2 and JC3 bytes consist of a 10-bit C_m field (bits C1, C2, .., C10), a 1-bit increment indicator (II) field, a 1-bit decrement indicator (DI) field and a 6-bit CRC-6 field which contains an error check code over the JC1, JC2 and JC3 bits 3 to 8 fields.

The bits 1 and 2 of the JC1, JC2, JC3 bytes and bits 2 to 8 of the JC4, JC5 and JC6 bytes consist of a 18-bit ΣC_{nD} field (bits D1, D2, .., D18), a 9-bit CRC-9 field which contains an error check code over

bits 2 to 8 in the JC4, JC5 and JC6 fields plus bits 1 and 2 in the JC1, JC2 and JC3 fields and three reserved bits.

The value of 'm' in C_m is 128×'ts' (number of tributary slots occupied by the ODTUCn.ts).

The value of 'n' represents the timing granularity of the GMP C_n parameter, which is also present in ΣC_{nD}. The value of n is 8.

The value of C_m controls the distribution of groups of '16ts' ODUk data bytes into groups of '16ts' ODTUCn.ts payload bytes. Refer to clause 20.5 and Annex D for further specification of this process.

The value of ΣC_{nD} provides additional 'n'-bit timing information, which is necessary to control the jitter and wander performance experienced by the ODUk signal.

The value of C_n (i.e., number of client n-bit data entities per OPUCn multiframe) is computed as follows: $C_n(t) = m/n \times C_m(t) + (\Sigma C_{nD}(t) - \Sigma C_{nD}(t–1))$. Note that the value C_{nD} is effectively an indication of the amount of data in the mapper's virtual queue that it could not send during that multiframe due to it being less than a 16M-byte word. For the case where the value of ΣC_{nD} in a multiframe 't' is corrupted, it is possible to recover from such error in the next multiframe 't+1'.

20.4.4 OPUCn multiframe identifier overhead (OMFI)

An OPU*Cn* multiframe identifier (OMFI) byte is defined in row 4, column 16 of the OPUC #1 to #n overhead (Figure 20-10). The value of bits 4 to 8 of the OMFI byte will be incremented each OPU*Cn* frame to provide a 20 frame multiframe for the multiplexing of ODUk signals into the OPU*Cn*.

NOTE 1 – It is an option to align the OMFI = 0 position with MFAS = 0 position every 1280 (the least common multiple of 20 and 256) frame periods.

NOTE 2 – OMFI must be copied in all n OPUC instances at the source, and only 1 need to be processed at the sink.

Figure 20-10 – OPU*Cn* multiframe identifier (OMFI) overhead

20.5 Mapping ODUk into ODTUCn.ts

The mapping of ODU*k* (k = 0, 1, 2, 2e, 3, 4, flex) signals (with up to ±100 ppm bit-rate tolerance) into the ODTU*Cn.ts* (ts = M) signal is performed by means of a generic mapping procedure as specified in Annex D.

The OPU*Cn* and therefore the ODTU*Cn.ts* signals are created from a locally generated clock (within the limits specified in Table 7-3), which is independent of the ODU*k* signal.

The ODU*k* signal is extended with a frame alignment overhead as specified in clauses 15.6.2.1 and 15.6.2.2 and an all-0s pattern in the OTU*k* overhead field (see Figure 19-30, read "j" by "k").

The extended ODU*k* signal is adapted to the locally generated OPU*Cn*/ODTU*Cn.ts* clock by means of a generic mapping procedure (GMP) as specified in Annex D. The value of n in c_n and $C_n(t)$ and $C_{nD}(t)$ is specified in Annex D. The value of M is the number of tributary slots occupied by the ODU*k*; ODTU*Cn.ts* = ODTU*Cn.M*.

A group of M successive extended ODUk 16-byte (128-bit) words is mapped into a group of M successive ODTUCn.M 16-byte (128-bit) words.

NOTE 1 – The 16-byte word alignment of the extended ODUk is preserved through the mapping procedure; e.g., the position of the first 16 OH bytes of the ODUk is always located after an integer number of 16-byte words from the start of the ODTUCn.M structure.

The generic mapping process generates for the case of ODU*k* signals once per ODTU*Cn.M* multiframe the $C_m(t)$ and $C_{nD}(t)$ information according to Annex D and encodes this information in the ODTU*Cn.ts* justification control overhead JC1/JC2/JC3 and JC4/JC5/JC6. The de-mapping process decodes $C_m(t)$ and $C_{nD}(t)$ from JC1/JC2/JC3 and JC4/JC5/JC6 and interprets $C_m(t)$ and $C_{nD}(t)$ according to Annex D. CRC-6 shall be used to protect against an error in bits 3 to 8 of the JC1, JC2, JC3 signals. CRC-9 shall be used to protect against an error in bits 1 and 2 of JC1, JC2, JC3 and bits 2 to 8 of JC4, JC5, JC6 signals.

During a signal fail condition of the incoming ODU*k* signal, this failed incoming signal will contain the ODU*k*-AIS signal as specified in clause 16.5.1. This ODU*k*-AIS is then mapped into the ODTU*Cn.M*.

For the case where the ODU*k* is received from the output of a fabric (ODU connection function), the incoming signal may contain (in the case of an open matrix connection) the ODU*k*-OCI signal as specified in clause 16.5.2. This ODU*k*-OCI signal is then mapped into the ODTU*Cn.M*.

NOTE 2 – Not all equipment will have a real connection function (i.e., switch fabric) implemented; instead, the presence/absence of tributary interface port units represents the presence/absence of a matrix connection. If such a unit is intentionally absent (i.e., not installed), the associated ODTU*Cn.M* signals should carry an ODU*k*-OCI signal. If such a unit is installed but temporarily removed as part of a repair action, the associated ODTU*Cn.M* signal should carry an ODU*k*-AIS signal.

A group of M successive extended ODUk 16-byte words is de-mapped from a group of M successive ODTU*Cn.M 16*-byte blocks.

NOTE 3 – For the case where the ODU*k* signal is output as an OTU*k* signal, frame alignment of the extracted extended ODU*k* signal is to be recovered to allow frame synchronous mapping of the ODU*k* into the OTU*k* signal.

During a signal fail condition of the incoming ODU*Cn*/OPU*Cn* signal (e.g., in the case of an ODU*Cn*-AIS condition) the ODU*k*-AIS pattern as specified in clause 16.5.1 is generated as a replacement signal for the lost ODU*k* signal.

The values of M, m, $C_{m,min}$, $C_{m,max}$, n, $C_{n,min}$ and $C_{n,max}$ for ODUk into ODTUCn.ts are as follows:

$$M = ceiling\left(\frac{ODUk\_nom\_bit\_rate\times(1+ODUk\_bit\_rate\_tolerance+0.00006)}{(ODTUCn.1\_nom\_bit\_rate\times(1-ODTUCnts\_bit\_rate\_tolerance)}\right) \text{ for ODUk with } k \neq flex(GFP,n,k') \quad (20\text{-}1a)$$

$$M = ceiling\left(\frac{ODUk\_nom\_bit\_rate}{ODUk'.ts \times 4}\right) = ceiling\left(n/4\right)$$ for ODUk with k = flex(GFP,n,k') and k' ∈ {2,3,4} (20-1b)

$$m = 128 \times M \quad (20\text{-}2)$$

NOTE 4 – While Equation 20-1a will produce the value M = 154 for an ODUflex(800GBASE-R) being multiplexed into an ODTUCn.ts, in some applications, M = 160 is used for this client. In these applications, the 160 tributary slots are selected from eight OPUC instances within the OPUCn.

$$c_{m,nom} = \left(\frac{ODUk\_nom\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_ODTUCn.ts}{ODTUCn.1\_nom\_bit\_rate \times M}\right) \quad (20\text{-}3)$$

$$c_{m,\min} = c_{m,nom} \times \left(\frac{1 - ODUk\_bit\_rate\_tolerance}{1 + ODTUCn.1\_bit\_rate\_tolerance}\right) \quad (20\text{-}4)$$

$$c_{m,\max} = c_{m,nom} \times \left(\frac{1 + ODUk\_bit\_rate\_tolerance}{1 - ODTUCn.1\_bit\_rate\_tolerance}\right) \quad (20\text{-}5)$$

$$C_{m,\min} = floor\left(c_{m,\min}\right) \quad (20\text{-}6)$$

$$C_{m,max} = ceiling\left(c_{m,max}\right) \quad (20\text{-}7)$$

$$n = 8 \quad (20\text{-}8)$$

$$c_{n,nom} = \left(\frac{ODUk\_nom\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_ODTUCn.ts \times 16}{ODTUCn.1\_nom\_bit\_rate}\right) \quad (20\text{-}9)$$

$$c_{n,\min} = c_{n,nom} \times \left(\frac{1 - ODUk\_bit\_rate\_tolerance}{1 + ODTUCn.1\_bit\_rate\_tolerance}\right) \quad (20\text{-}10)$$

$$c_{n,\max} = c_{n,nom} \times \left(\frac{1 + ODUk\_bit\_rate\_tolerance}{1 - ODTUCn.1\_bit\_rate\_tolerance}\right) \quad (20\text{-}11)$$

$$C_{n,\min} = floor\left(c_{n,\min}\right) \quad (20\text{-}12)$$

$$C_{n,max} = ceiling\left(c_{n,max}\right) \quad (20\text{-}13)$$

$C_{m,min}$, $C_{n,min}$ (n=8), $C_{m,max}$ and $C_{n,max}$ (n=8) values represent the boundaries of ODU*k*/ODTU*Cn.M* ppm offset combinations (i.e., min. ODU*k*/max. ODTUCn.M and max. ODU*k*/min. ODTUCn.M). In steady state, given instances of ODU*k*/ODTUCn.M offset combinations should not result in generated C_n and C_m values throughout this range but rather should be within as small a range as possible.

NOTE ~~4~~ 5 – Under transient ppm offset conditions (e.g., AIS to normal signal), it is possible that C_n and C_m values outside the range $C_{n,min}$ to $C_{n,max}$ and $C_{m,min}$ to $C_{m,max}$ may be generated and a GMP de-mapper should be tolerant of such occurrences. Refer to Annex D for a general description of the GMP principles.

20.5.1 Mapping ODUk into ODTUCn.M

Groups of M successive 16-byte words of the extended ODU*k* (k = 0, 1, 2, 2e, 3, 4, flex) signal are mapped into a group of M successive 16-byte blocks of the ODTU*Cn.M* payload area under control of the GMP data/stuff control mechanism. Each group of M 16-byte blocks in the ODTU*Cn.M* payload area may either carry M ODU*k* 16-byte words, or carry M stuff 16-byte words. The value of the stuff bytes is set to all-0s.

The groups of M 16-byte blocks in the ODTU*Cn.M* payload area are numbered from 1 to 952.

The ODTU*Cn.M* payload 16-byte numbering for GMP M 16-byte (m-bit) blocks is illustrated in Figure 20-11. In row 1 of the ODTU*Cn.M* multiframe the first M 16-byte blocks will be labeled 1, the next M 16-byte blocks will be labeled 2, etc.

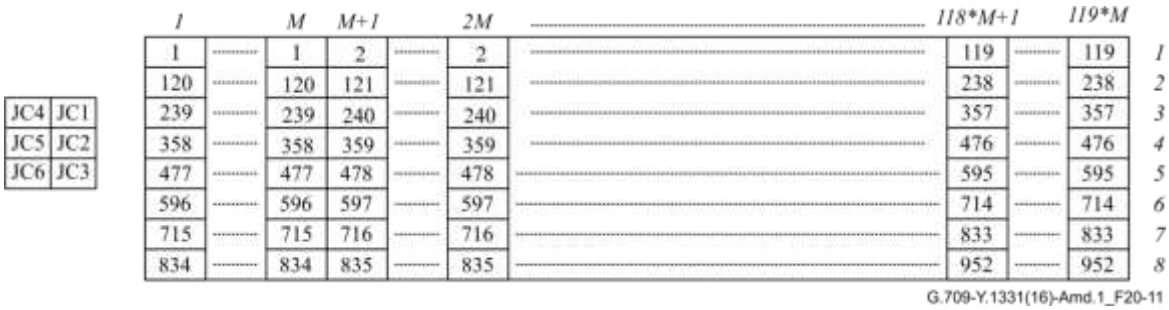


Figure 20-11 – ODTU*Cn.M* GMP 16-byte block numbering

Annex A

Forward error correction using 16-byte interleaved RS(255,239) codecs

(This annex forms an integral part of this Recommendation.)

The forward error correction for the OTU-k uses 16-byte interleaved codecs using a Reed-Solomon RS(255,239) code. The RS(255,239) code is a non-binary code (the FEC algorithm operates on byte symbols) and belongs to the family of systematic linear cyclic block codes.

For FEC processing, an OTU row is separated into 16 sub-rows using byte-interleaving as shown in Figure A.1. Each FEC encoder/decoder processes one of these sub-rows. The FEC parity check bytes are calculated over the information bytes 1 to 239 of each sub-row and transmitted in bytes 240 to 255 of the same sub-row.

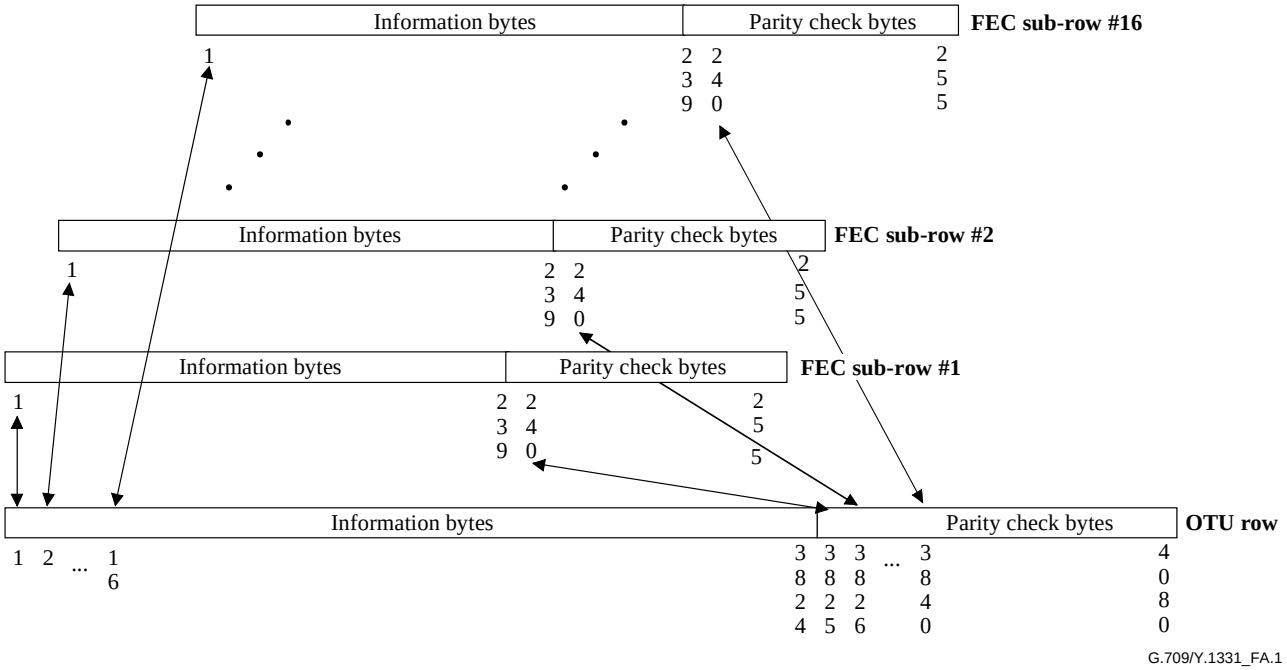


Figure A.1 – FEC sub-rows

The bytes in an OTU row belonging to FEC sub-row X are defined by: $X + 16 \times (i - 1)$ (for i = 1...255).

The generator polynomial of the code is given by:

$$G(z) = \prod_{i=0}^{15} (z - \alpha^i)$$

where α is a root of the binary primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$.

The FEC code word (see Figure A.2) consists of information bytes and parity bytes (FEC redundancy) and is represented by the polynomial:

$$C(z) = I(z) + R(z)$$

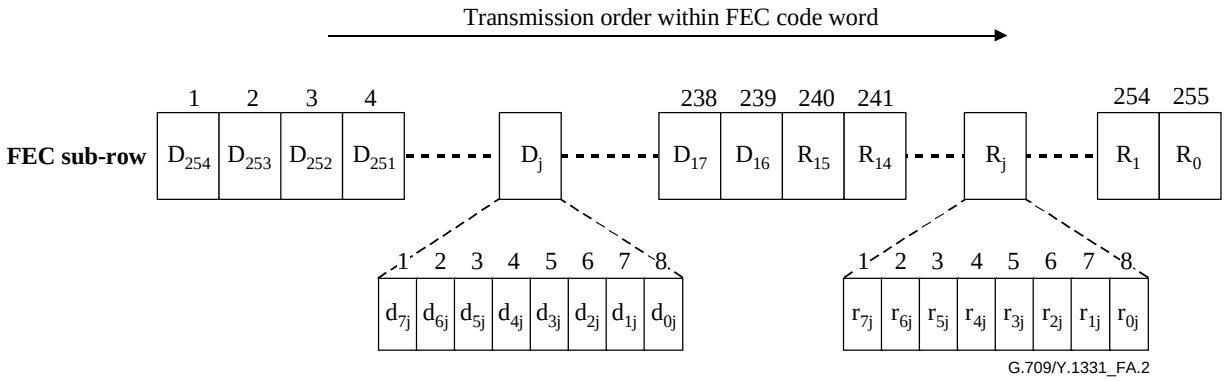


Figure A.2 – FEC code word

Information bytes are represented by:

$$I(z) = D_{254}.z^{254} + D_{253}.z^{253} + ... + D_{16}.z^{16}$$

Where D_j (j = 16 to 254) is the information byte represented by an element out of GF(256) and:

$$D_j = d_{7j} \cdot \alpha^7 + d_{6j} \cdot \alpha^6 + ... + d_{1j} \cdot \alpha^1 + d_{0j}$$

Bit d_{7j} is the MSB and d_{0j} the LSB of the information byte.

D_{254} corresponds to byte 1 in the FEC sub-row and D_{16} to byte 239.

Parity bytes are represented by:

$$R(z) = R_{15} \cdot z^{15} + R_{14} \cdot z^{14} + ... + R_1 \cdot z^1 + R_0$$

Where R_j (j = 0 to 15) is the parity byte represented by an element out of GF(256) and:

$$R_j = r_{7j} \cdot \alpha^7 + r_{6j} \cdot \alpha^6 + ... + r_{1j} \cdot \alpha^1 + r_{0j}$$

Bit r_{7j} is the MSB and r_{0j} the LSB of the parity byte.

R_{15} corresponds to the byte 240 in the FEC sub-row and R_0 to byte 255.

R(z) is calculated by:

$$R(z) = I(z) \bmod G(z)$$

where "mod" is the modulo calculation over the code generator polynomial *G(z)* with elements out of the GF(256). Each element in GF(256) is defined by the binary primitive polynomial

$$x^8 + x^4 + x^3 + x^2 + 1.$$

The Hamming distance of the RS(255,239) code is d_{min} = 17. The code can correct up to 8 symbol errors in the FEC code word when it is used for error correction. The FEC can detect up to 16 symbol errors in the FEC code word when it is used for error detection capability only.

Annex B

Adapting 64B/66B encoded clients via transcoding into 513B code blocks

(This annex forms an integral part of this Recommendation.)

Clients using 64B/66B coding can be adapted in a codeword and timing transparent mapping via transcoding into 513B code blocks to reduce the bit rate that is required to transport the signal. The resulting 513B blocks can be mapped in one of several ways depending on the requirements of the client and the available bandwidth of the container into which the client is mapped. This mapping can be applied to serial or parallel client interfaces.

B.1 Transmission order

The order of transmission of information in all the diagrams in this annex is first from left to right and then from top to bottom.

B.2 Client frame recovery

For 40GBASE-R and 100GBASE-R clients, framing recovery consists of the recovering 64B/66B block lock per the state diagram in Figure 82-10 of [IEEE 802.3]. For other 64B/66B encoded clients, block lock is achieved as per the state diagram in Figure 49-12 of [IEEE 802.3]. Descrambling is performed as per the process shown in Figure 49-10 of [IEEE 802.3].

Each 66B codeword (after block lock) is one of the following:

– a set of eight data bytes with a sync header of "01";

– a control block (possibly including seven or fewer data octets) beginning with a sync header of "10".

The 64 bits following the sync header are scrambled as a continuous bit-stream (skipping sync headers and PCS lane markers) according to the polynomial $G(x) = 1 + x^{39} + x^{58}$. The 64B/66B PCS receive process will descramble the bits other than (1) the sync header of 66B data and control blocks, and (2) the PCS lane markers.

Figure B.1 illustrates the ordering of 64B/66B code blocks after the completion of the recovering process for an interface.

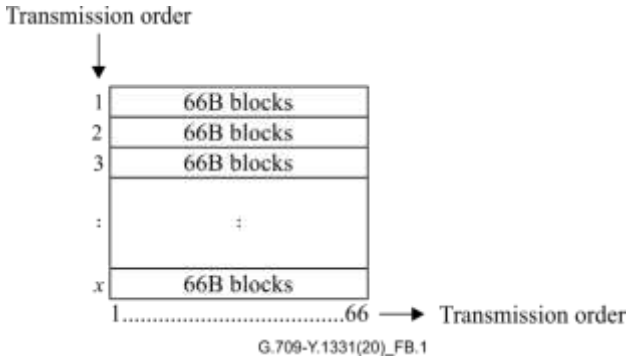


Figure B.1 – Stream of 64B/66B code blocks for transcoding

B.3 Transcoding from 66B blocks to 513B blocks

The transcoding process at the encoder operates on an input sequence of 66B code blocks.

66B control blocks (after descrambling) follow the format shown in Figure B.2.

A group of eight 66B blocks is encoded into a single 513B block. The format is illustrated in Figure B.3.

Input Data	SYNC		Block Payload									
Bit Position / Data Block Format	0	1	2–65									
$D_0D_1D_2D_3D_4D_5D_6D_7$	0	1	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		
Control block formats			**Block type field**									**4-bit code**
$C_0C_1C_2C_3C_4C_5C_6C_7$	1	0	0x1e	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	0001
$C_0C_1C_2C_3O_4D_5D_6D_7$	1	0	0x2d	C_0	C_1	C_2	C_3	O_4	D_5	D_6	D_7	0010
$C_0C_1C_2C_3S_4D_5D_6D_7$	1	0	0x33	C_0	C_1	C_2	C_3		D_5	D_6	D_7	0111
$O_0D_1D_2D_3S_4D_5D_6D_7$	1	0	0x66	D_1	D_2	D_3	O_0		D_5	D_6	D_7	1011
$O_0D_1D_2D_3O_4D_5D_6D_7$	1	0	0x55	D_1	D_2	D_3	O_0	O_4	D_5	D_6	D_7	1101
$S_0D_1D_2D_3D_4D_5D_6D_7$	1	0	0x78	D_1	D_2	D_3	D_4	D_5	D_6	D_7		1110
$O_0D_1D_2D_3C_4C_5C_6C_7$	1	0	0x4b	D_1	D_2	D_3	O_0	C_4	C_5	C_6	C_7	1000
$T_0C_1C_2C_3C_4C_5C_6C_7$	1	0	0x87		C_1	C_2	C_3	C_4	C_5	C_6	C_7	0011
$D_0T_1C_2C_3C_4C_5C_6C_7$	1	0	0x99	D_0		C_2	C_3	C_4	C_5	C_6	C_7	0101
$D_0D_1T_2C_3C_4C_5C_6C_7$	1	0	0xaa	D_0	D_1		C_3	C_4	C_5	C_6	C_7	1001
$D_0D_1D_2T_3C_4C_5C_6C_7$	1	0	0xb4	D_0	D_1	D_2		C_4	C_5	C_6	C_7	1010
$D_0D_1D_2D_3T_4C_5C_6C_7$	1	0	0xcc	D_0	D_1	D_2	D_3		C_5	C_6	C_7	1100
$D_0D_1D_2D_3D_4T_5C_6C_7$	1	0	0xd2	D_0	D_1	D_2	D_3	D_4		C_6	C_7	0110
$D_0D_1D_2D_3D_4D_5T_6C_7$	1	0	0xe1	D_0	D_1	D_2	D_3	D_4	D_5		C_7	0000
$D_0D_1D_2D_3D_4D_5D_6T_7$	1	0	0xff	D_0	D_1	D_2	D_3	D_4	D_5	D_6		1111

Figure B.2 – 66B Block coding

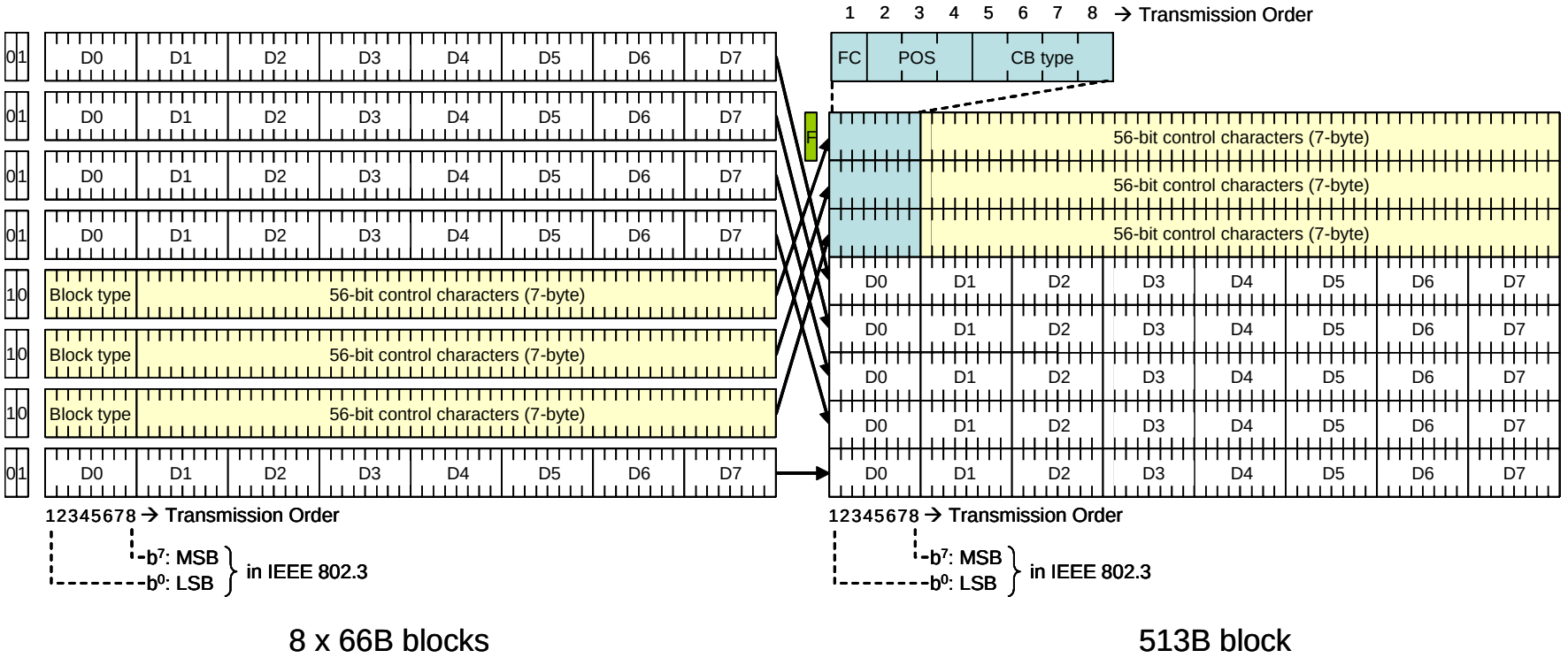


Figure B.3 – 513B block code format

Each of the 66B blocks is encoded into a row of the 8-byte by 8-row structure. Any 66B control blocks (CBi) are placed into the uppermost rows of the structure in the order received, while any all-data 66B blocks (DBi) are placed into the lowermost rows of the structure in the order received.

The flag bit "F" is 1 if the 513B structure contains at least one 66B control block, and 0 if the 513B structure contains eight all-data 66B blocks. Because the 66B control blocks are placed into the uppermost rows of the 513B block, if the flag bit "F" is 1, then the first row will contain a mapping of a 66B control block.

A 66B control block is encoded into a row of the structure shown in Figure B.3 as follows: the sync header of "10" is removed. The byte representing the block type field (see Figure B.2) is replaced by the structure shown in Figure B.4:

Figure B.4 – 513B block's control block header

The byte indicating the control block type (one of 15 legal values) is translated into a 4-bit code according to the rightmost column of Figure B.2. The 3-bit POS field is used to encode the position in which this control block was received in the sequence of eight 66B blocks. The flag continuation bit "FC" will be set to a 0 if this is the final 66B control block or PCS lane alignment marker encoded in this 513B block, or to a 1 if one or more 66B control blocks or PCS lane alignment markers follow this one. At the decoder, the flag bit for the 513B block as a whole, plus the flag continuation bits in each row containing the mapping of a 66B control block or PCS lane alignment marker will allow identification of those rows, which can then be restored to their original position amongst any all-data 66B blocks at the egress according to the POS field. The remaining 7 bytes of the row are filled with the last 7 bytes of the 66B control block.

An all-data 66B block is encoded into a row of the 513B block by dropping the sync header and copying the remaining eight bytes into the row. If all eight rows of the 513B block are placements of 66B all-data blocks, the flag bit "F" will be 0. If fewer than eight rows of the 513B block are placements of 66B all-data blocks, they will appear at the end, and the row containing the placement of the final 66B control block will have a flag continuation bit "FC" value of 0.

The decoder operates in the reverse of the encoder to reconstruct the original sequence of 66B blocks. If flag bit "F" is 1, then 66B control blocks starting from the first row of the block are reconstructed and placed in the position indicated by the POS field. This process continues through all of the control blocks working downward from the top row. The final 66B control block placed within the 513B block will be identified when the flag continuation bit "FC" is zero.

The structure of the 512B/513B code block is shown in Figure B.5. For example, if there is a single 64B/66B control block CB1 in a 512B/513B code block and it was originally located between 64B/66B data blocks DB2 and DB3, the first octet of the 64B character will contain 0.010.1101.CB1; the leading bit in the control octet of 0 indicates the flag continuation "FC" that this 64B control block is the last one in the 512B/513B code block, the value of 010 indicates CB1's position "POS" between DB2 and DB3, and the value of 1101 is a four-bit representation of the control code's block type "CB TYPE" (of which the eight-bit original block type is 0x55).

Input client characters	Flag bit	512-bit (64-Octet) field							
All data block	0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	DB8
7 data block 1 control block	1	0 AAA aaaa CB1	DB1	DB2	DB3	DB4	DB5	DB6	DB7
6 data block 2 control block	1	1 AAA aaaa CB1	0 BBB bbbb CB2	DB1	DB2	DB3	DB4	DB5	DB6
5 data block 3 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	0 CCC cccc CB3	DB1	DB2	DB3	DB4	DB5
4 data block 4 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	1 CCC cccc CB3	0 DDD dddd CB4	DB1	DB2	DB3	DB4
3 data block 5 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	1 CCC cccc CB3	1 DDD dddd CB4	0 EEE eeee CB5	DB1	DB2	DB3
2 data block 6 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	1 CCC cccc CB3	1 DDD dddd CB4	1 EEE eeee CB5	0 FFF ffff CB6	DB1	DB2
1 data block 7 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	1 CCC cccc CB3	1 DDD dddd CB4	1 EEE eeee CB5	1 FFF ffff CB6	0 GGG gggg CB7	DB1
8 control block	1	1 AAA aaaa CB1	1 BBB bbbb CB2	1 CCC cccc CB3	1 DDD dddd CB4	1 EEE eeee CB5	1 FFF ffff CB6	1 GGG gggg CB7	0 HHH hhhh CB8

- Leading bit in a 66B control block FC = 1 if there are more than 66Bcontrol block and = 0 if this payload contains the last control block in that 513B block
- AAA = 3-bit representation of the first control code's original position (First control code loca tor: POS)
- BBB = 3-bit representation of the second control code's original position (Second control code lo cator: POS)

.....

- HHH = 3-bit representation of the eighth control code's original position (Eighth control code lo cator: POS)
- aaa = 4-bit representation of the first control code's type (first control block type: CB TYPE)
- bbb = 4-bit representation of the second control code's type (Second control block type: CB TYPE)

.....

-hhh = 4-bit representation of the eighth control code's type (Eighth control block type: CB TYPE)
- CBi = 56-bit representation of the i-th control code caracters
- DBi = 64-bit representation of the i-th data value in order of transmission

G.709-Y.1331(12)_FB.5

Figure B.5 – 513B code block components

B.3.1 Errors detected before 512B/513B encoder

A set of errors might be detected at the 64B/66B PCS receive process which, in addition to appropriate alarming, needs to send the appropriate signal downstream.

Errors encountered before the encoder, such as loss of client signal, will result in the insertion of an Ethernet LF sequence ordered set prior to this process, which will be transcoded as any other control block. The same action should be taken as a result of failure to achieve 66B block lock on an input signal.

An invalid 66B block will be converted to an error control block before transcoding and the OTN BIP-8 calculation as described in clause E.4.1. An invalid 66B block is one which does not have a sync header of "01" or "10", or one which has a sync header of "10" and a control block type field which does not appear in Figure B.2. An error control block has sync bits of "10", a block type code of 0x1E, and 8 seven-bit/E/error control characters. This will prevent the Ethernet receiver from interpreting a sequence of bits containing this error as a valid packet.

B.3.2 Errors detected by 512B/513B decoder

Several mechanisms will be employed to reduce the probability that the decoder constructs erroneous 64B/66B encoded data at the egress if bit errors have corrupted. Since detectable corruption normally means that the proper order of 66B blocks to construct at the decoder cannot be reliably determined, if any of these checks fail, the decoder will transmit eight 66B error control blocks (sync="10", control block type=0x1e, and eight 7-bit/E/control characters).

Mechanisms for improving the robustness and for 513B block lock are discussed in Annex F.

B.4 Link fault signalling

In-band link fault signalling in the client 64B/66B code (e.g., if a local fault or remote fault sequence ordered set is being transmitted between Ethernet equipments) is carried transparently according to this transcoding.

Annex C

Adaptation of OTU3 and OTU4 over multichannel parallel interfaces

(This annex forms an integral part of this Recommendation.)

NOTE 1 – This mechanism is designed to allow the use of the optical modules being developed for IEEE 40GBASE-R and 100GBASE-R signals for short-reach client-side OTU3 and OTU4 interfaces respectively. The corresponding physical layer specifications are being added to [ITU-T G.695] and [ITU-T G.959.1].

OTU3 signals may be carried over parallel interfaces consisting of four lanes. This four-lane format is referred to as the OTL3.4 format.

OTU4 signals may be carried over parallel interfaces consisting of four or ten lanes, which are formed by bit multiplexing of 20 logical lanes. The four-lane format is referred to as the OTL4.4 signal format and the ten-lane format is referred to as the OTL4.10 signal format.

NOTE 2 – Ten lane IEEE 100GBASE-R interfaces have no corresponding ITU-T physical layer interface specification.

The OTU3 and OTU4 frames are inversely multiplexed over physical/logical lanes on a 16-byte boundary aligned with the OTUk frame as illustrated in Figure C.1. The OTUk frame is divided into 1020 groups of 16-bytes.

	1					*4080*
1	1:16 (FAS)	17:32	33:48	49:64	.....	4065:4080
2	4081:4096	4097:5012	5013:5028	5029:5044		9145:9160
3	9161:9176	9177:9192	9193:9208	9209:9224		12225:12240
4	12241:12256	12257:12272	12273:12288	12289:13304		16305:16320

G.709-Y.1331(20)_FC.1

Figure C.1 – OTU3 and OTU4 frames divided on 16-byte boundary

OTU3 16-byte increment distribution

Each 16-byte increment of an OTU3 frame is distributed round robin, to each of the four physical lanes. On each OTU3 frame boundary the lane assignments are rotated.

For OTU3, the lane rotation and assignment is determined by the two LSBs of the MFAS as described in Table C.1 and Figure C.2, which indicates the starting group of bytes of the OTU3 frame that are sent on each lane.

NOTE 3 – MFAS is scrambled as defined in clause 11.2.

The pattern repeats every 64 bytes until the end of the OTU3 frame. The following OTU3 frame will use different lane assignments according to the MFAS.

Table C.1 – Lane rotation assignments for OTU3

MFAS 7-8	Lane 0	Lane 1	Lane 2	Lane 3
*00	1:16	17:32	33:48	49:64
*01	49:64	1:16	17:32	33:48
*10	33:48	49:64	1:16	17:32
*11	17:32	33:48	49:64	1:16

The distribution of 16-byte blocks from the sequence of OTU3 frames is illustrated in Figure C.2:

The parallel lanes can be reassembled at the sink by first recovering framing on each of the parallel lanes, then recovering the lane identifiers and then performing lane de-skewing. Frame alignment, lane identifier recovery and multi-lane alignment should operate under 10^{-3} bit error rate conditions before error correction. Refer to [ITU-T G.798] for the specific processing details.

The lane rotation mechanism will place the first 16 bytes of the OTU3 frame on each lane once per 4080 × 4 (i.e., 16320) bytes (the same as an OTU3 itself). The two LSBs of the MFAS will be the same in each FAS on a particular lane, which allows the lane to be identified. Since the MFAS cycles through 256 distinct values, the lanes can be de-skewed and reassembled by the receiver as long as the total skew does not exceed 127 OTU3 frame periods (approximately 385 μs). The receiver must use the MFAS to identify each received lane, as lane positions may not be preserved by the optical modules to be used for this application.

OTU4 16-byte increment distribution

Each 16-byte increment of an OTU4 frame is distributed, round robin, to each of the 20 logical lanes. On each OTU4 frame boundary the lane assignments are rotated.

For distribution of OTU4 to twenty logical lanes, since the MFAS is not a multiple of 20, a different marking mechanism must be used. Since the frame alignment signal is 6 bytes (48 bits) and as per [ITU-T G.798] only 32 bits must be checked for frame alignment, the third OA2 byte position will be borrowed as a logical lane marker (LLM). For maximum skew detection range, the lane marker value will increment on successive frames from 0-239 (240 values being the largest multiple of 20 that can be represented in 8-bits). LLM = 0 position shall be aligned with MFAS = 0 position every 3840 (the least common multiple of 240 and 256) frame periods. The logical lane number can be recovered from this value by a modulo 20 operation. Table C.2 and Figure C.3 illustrate how bytes of the OTU4 are distributed in 16-byte increments across the 20 logical lanes.

The pattern repeats every 320 bytes until the end of the OTU4 frame.

The following OTU4 frame will use different lane assignment according to the LLM MOD 20.

Table C.2 – Lane rotation assignments for OTU4

LLM MOD 20	Lane 0	Lane 1	…….	Lane 18	Lane 19
0	1:16	17:32		289:304	305:320
1	305:320	1:16		273:288	289:304
:					
18	33:48	49:64		1:16	17:32
19	17:32	33:48		305:320	1:16

The distribution of 16-byte blocks from the sequence of OTU4 frames is illustrated in Figure C.3.

The parallel lanes can be reassembled at the sink by first recovering framing on each of the parallel lanes, then recovering the lane identifiers and then performing de-skewing of the lanes. Frame alignment, lane identifier recovery and multi-lane alignment should operate under 10^{-3} bit error rate conditions before error correction. Refer to [ITU-T G.798] for specific processing details.

The lane rotation mechanism will place the first 16 bytes of the OTU4 frame on each lane once per 4080×4 (i.e., 16320) bytes (the same as an OTU4 itself). The "LLM MOD 20" will be the same in each FAS on a particular lane, which allows the lane to be identified. Since the LLM cycles through 240 distinct values, the lanes can be de-skewed and reassembled by the receiver as long as the total skew does not exceed 119 OTU4 frame periods (approximately 139 μs). The receiver must use the

"LLM MOD 20" to identify each received lane, as lane positions may not be preserved by the optical modules to be used for this application.

The lanes are identified, de-skewed, and reassembled into the original OTU4 frame according to the lane marker. The MFAS can be combined with the lane marker to provide additional skew detection range, the maximum being up to the least common multiple "LCM(240, 256)/2 – 1" or 1919 OTU4 frame periods (approximately 2.241 ms). In mapping from lanes back to the OTU4 frame, the sixth byte of each OTU4 frame which was borrowed for lane marking is restored to the value OA2.

Each physical lane of an OTL4.4 carried over a multi-lane interface is formed by simple bit multiplexing of five logical lanes. At the sink, the bits are disinterleaved into five logical lanes from each physical lane. The sink will identify each logical lane according to the lane marker in the LLM byte. The sink must be able to accept the logical lanes in any position as the ordering of bit multiplexing on each physical lane is arbitrary; the optical module hardware to be used for this application is permitted full flexibility concerning which physical lane will be used for output of each logical lane, and the order of bit multiplexing of logical lanes on each physical output lane.

NOTE 4 – Ten-lane IEEE 100GBASE-R interfaces are specified, although not with ITU-T physical layer specifications. These interfaces may be compatible with a 10-lane interface for OTU4 (OTL4.10), each lane consisting of two bit-multiplexed logical lanes. Refer to[b-ITU-T G-Sup.58].

This mechanism handles any normally framed OTU3 or OTU4 sequence.

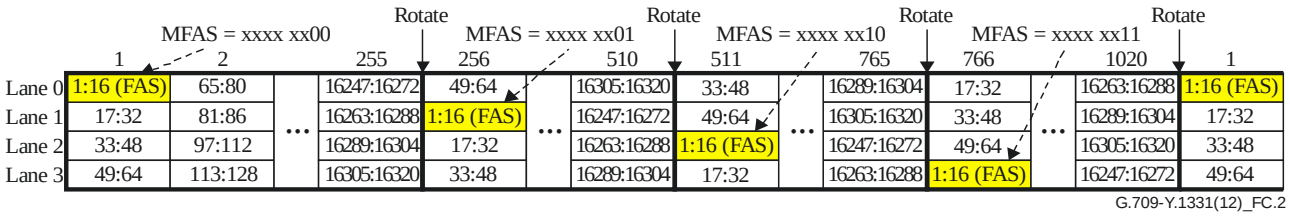


Figure C.2 – Distribution of bytes from OTU3 to parallel lanes

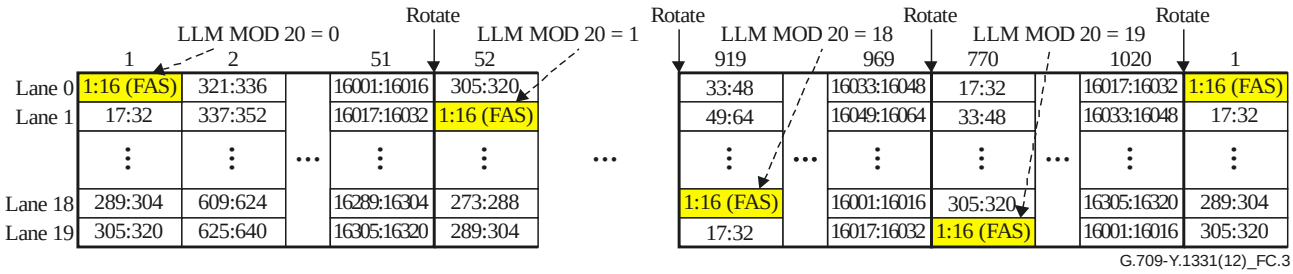


Figure C.3 – Distribution of bytes from OTU4 to parallel lanes

Annex D

Generic mapping procedure principles

(This annex forms an integral part of this Recommendation.)

This annex introduces the principles of the generic mapping procedure. Clause D.1 presents a technology-agnostic description of the basic principle. Clause D.2 presents a technology-agnostic description of the GMP application principles. Clause D.3 presents application principles of GMP within the OTN cases when a client is mapped into an OPU and when an ODUk is mapped into an ODTU and this ODTU is multiplexed into an OPU.

D.1 Basic principle

For any given CBR client signal, the number of n-bit (e.g., n = 1, 8) data entities that arrive during one server frame period is defined by:

$$c_n = \left(\frac{f_{client}}{n} \times T_{server} \right) \quad \text{(D-1)}$$

f_{client}: bit rate of the received client signal

T_{server}: frame period of the server signal

c_n: number of client n-bit data entities per server frame period

As only an integer number of n-bit data entities can be transported per server frame period, c_n must be converted to an integer value $C_n(t)$ for each frame period. Since it is required that no client information is lost, the rounding process has to take care of the truncated part by occasionally transmitting an additional n-bit data entity during a server frame period. As such, $C_n(t)$ will typically vary between the integer portion of c_n and the next larger integer:

$$C_n(t) = \lfloor c_n \rfloor \text{ or } C_n(t) = \lceil c_n \rceil \quad \text{(D-2)}$$

$C_n(t)$: number of client n-bit data entities per server frame period t (integer)

Note that jitter/wander could impact the actual instantaneous value of c_n. The server frame period is defined by the server bit rate and the number of bits per server frame:

$$T_{server} = \frac{B_{server}}{f_{server}} \quad \text{(D-3)}$$

f_{server}: server bit rate

B_{server}: bits per server frame

Combining (D-3) with (D-1) and (D-2) results in:

$$C_n(t) = \left\lfloor \frac{f_{client}}{f_{server}} \times \frac{B_{server}}{n} \right\rfloor \text{ or } C_n(t) = \left\lceil \frac{f_{client}}{f_{server}} \times \frac{B_{server}}{n} \right\rceil \quad \text{(D-4)}$$

As the client data has to fit into the payload area of the server signal, the maximum value of $C_n(t)$ and consequently the maximum client bit rate is limited by the size of the server payload area.

$$C_n(t) \le P_{server} \quad \text{(D-5)}$$

$$f_{client} \le f_{server} \times \frac{P_{server}}{B_{server}} \times n \quad \text{(D-6)}$$

P_{server}: maximum number of (n bits) data entities in the server payload area

The client and server bit rate are typically independent, which results in client clock impairments not being seen at the server clock.

Taking the frame overhead into account, the relationship between P_{server} and B_{server} is:

$$P_{server} = P_{n,server} = \left\lfloor \frac{B_{server} - O_{server}}{n} \right\rfloor \quad \text{(D-7)}$$

O_{server}: overhead bits per server frame

Note that in some cases, as indicated in equation D-7, the term P_{server} is written as $P_{n,server}$ to provide an explicit reminder that it represents n-bit data entities, analogous to D-18 defining $P_{m,server}$ in terms of m-bit data entities.

D.1.1 Impact of client or server frequency tolerances

If the client or server bit rate changes due to client or server frequency tolerances, c_n and $C_n(t)$ change accordingly. f_{client} and f_{server} have a time dependent value between a minimum and maximum value, determined by their nominal value ± their tolerance value. As such, c_n will vary between:

$$c_{n,min} = \left(\frac{f_{client,min}}{f_{server,max}} \times \frac{B_{server}}{n} \right) \text{ and } c_{n,max} = \left(\frac{f_{client,max}}{f_{server,min}} \times \frac{B_{server}}{n} \right) \quad \text{(D-8)}$$

And consequently, $C_n(t)$ will vary between

$$C_{n,min}(t) = \left\lfloor \frac{f_{client,min}}{f_{server,max}} \times \frac{B_{server}}{n} \right\rfloor \text{ and } C_{n,max}(t) = \left\lceil \frac{f_{client,max}}{f_{server,mmin}} \times \frac{B_{server}}{n} \right\rceil \leq P_{server} \quad \text{(D-9)}$$

with the maximum value of $C_n(t)$ and consequently the maximum client bit rate is limited by the size of the server payload area given in equations D-10 and D-11.

$$Min\ Server\ payload\ capacity = f_{server,min} \times \left(\frac{P_{server}}{B_{server}} \right) \times n \quad \text{(D-10)}$$

$$f_{client,max} \leq Min\ Server\ payload\ capacity \quad \text{(D-11)}$$

NOTE 1 – During a step change in the client bit rate (e.g., when the client signal is replaced by its AIS signal, or the AIS signal is replaced by the client signal), the $C_n(t)$ may temporarily exceed the above range. A transparent mapping has to determine $C_n(t)$ once per server frame period.

NOTE 2 – Any given instance of client and server frequency tolerance will have a relatively constant c_n value (i.e., $c_{n,x}$) so that $C_{n,x}(t)$ will vary in the range $int(c_{n,x}) \leq C_n(t) \leq int(c_{n,x}) + 1$.

D.1.2 Synchronizing the mapper and de-mapper

In order to extract the correct number of client data entities at the de-mapper, $C_n(t)$ is transported in the overhead area of the server frame from the mapper to the de-mapper.

Figure D.1 shows the generic functionality of the mapper and de-mapper circuit.

At the mapper, $C_n(t)$ is determined based on the client and server clocks. The client data is constantly written into the buffer memory at the rate of c_n *n*-bit client data entities arriving every frame period. The read out is controlled by the value of $C_n(t)$, which is inserted into the overhead of frame *t*-1. $C_n(t)$ corresponds to the integer number of *n*-bit data entities that will be read from the buffer for transmission during server frame period *t*. The long-term average of $C_n(t)$ is intended to track c_n. For example, during frames in which the number of n-bit data entities in the buffer is less than a certain threshold, the transmitted $C_n(t)$ will be $\lfloor c_n \rfloor$. The non-integer portions of c_n accumulate over time until the number of n-bit data entities in the buffer exceeds the threshold and the transmitted $C_n(t)$ will be increased to $\lceil c_n \rceil$. In this manner GMP avoids losing any client data due to the process of rounding c_n to an integer value for transmission. Note that this is a conceptual description that does not preclude functionally equivalent implementations.

As shown in Figure D.2, the determined $C_n(t)$ value is carried in the overhead of a server frame (i.e., frame *t-1*). The transmitted $C_n(t)$ value effectively represents the amount of client data phase that will

be carried within the next server frame *t*, which corresponds to the number of *n*-bit client data entities that will be carried in within the server payload area of the next frame (i.e., frame *t*). Consequently, the GMP sink uses the updated encoded $C_n(t)$ information of each server frame to extract the client data from the next server frame. Note that server frame *t* carries in its overhead the determined $C_n(t+1)$ value, in addition to the $C_n(t)$ *n*-bit client data entities in its payload. $C_n(t+1)$ *n*-bit client data entities are carried in server frame *t*+1.

At the de-mapper, $C_n(t)$ is extracted from the overhead. $C_n(t)$ controls the write enable signal for the buffer. The client clock is generated based on the server clock and the value of $C_n(t)$.

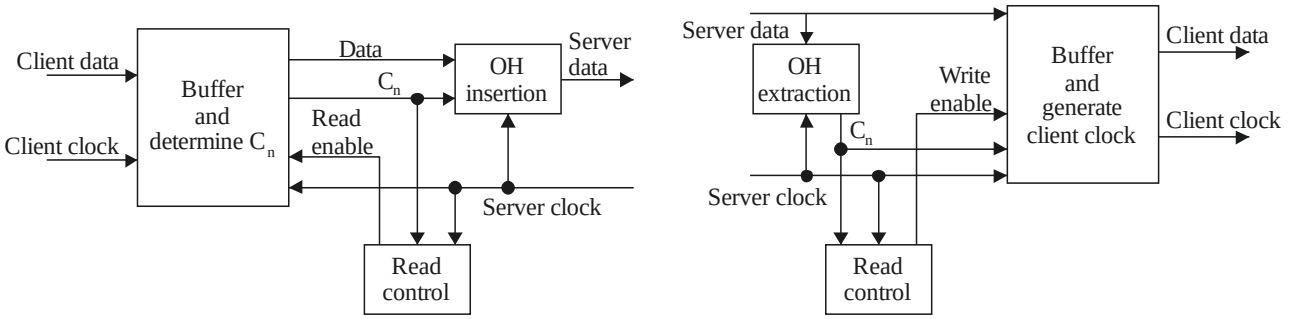


Figure D.1 – Generic functionality of a mapper/de-mapper circuit

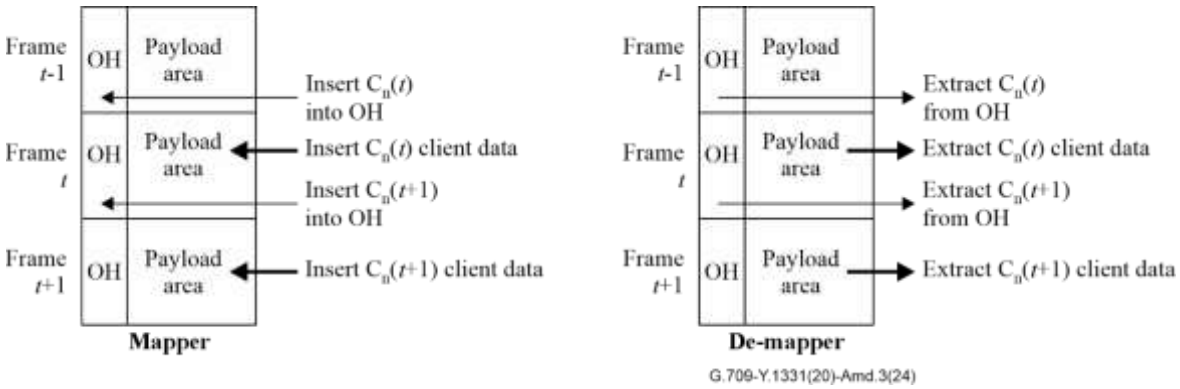


Figure D.2 – GMP processing flow and overhead generation and extraction

$C_n(t)$ client data entities are mapped into the payload area of the server frame using a sigma-delta data/stuff mapping distribution such that the client data entities are distributed over P_{server} locations as shown in Figure D.3. Payload field j (j = 1 .. P_{server}) carries:

– client data (D) if $(j \times C_n(t)) \bmod P_{server} < C_n(t)$ (D-12)

– stuff (S) if $(j \times C_n(t)) \bmod P_{server} \geq C_n(t)$. (D-13)

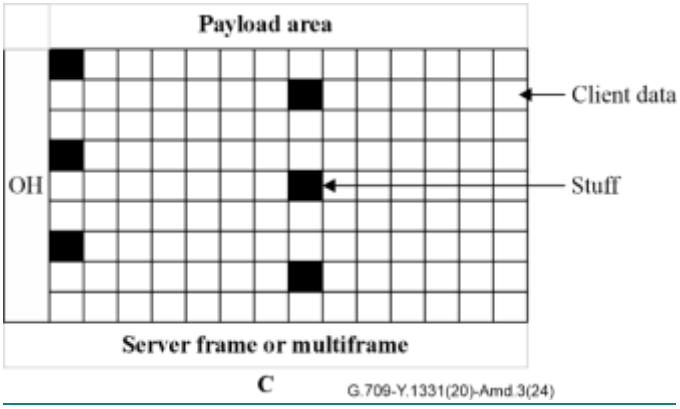


Figure D.3 – Sigma-delta based mapping

Specifically, successive client data entities are inserted with a spacing of $\frac{P_{server}}{C_n(t)}$. This is normally not an integer value, however it can be emulated by an integer calculation using the sigma-delta method based on an overflow accumulator as shown in Figure D.4.

The accumulator memory is reset to 0 at every frame start of the server frame. At every location of the payload area, $C_n(t)$ is added to the memory and the result is compared with P_{server}. If the result is lower than P_{server}, it is stored back into the memory and no client data is indicated for this payload position. If it is equal or greater than P_{server}, P_{server} is subtracted from the result and the new result is stored back in the memory. In addition, client data is indicated for the client position.

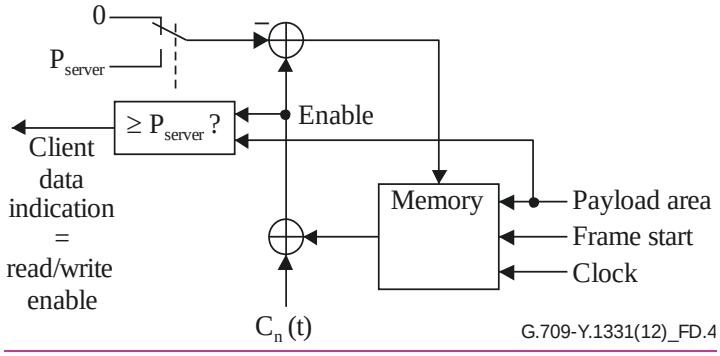


Figure D.4 – Sigma-delta accumulator

As the same start value and $C_n(t)$ are used at the mapper and de-mapper, the same results are obtained and interworking is achieved.

D.2 Practical application of GMP

Clause D.1 describes the GMP principle from the view of the client signal's *n*-bit data entities and their associated c_n and $C_n(t)$ parameters. This clause adjusts that description on the basis of a server frame, which is organized in *m*-bit data entities. Since it is the server frame's payload that needs to be filled with client data or stuff, it is more convenient to work with the *m*-bit data entities in a practical implementation, using the associated c_m and $C_m(t)$ parameters and an additional parameter ($\sum C_{nD}$) that represents the non-integer portion of c_m.

Clause D.2.1 describes the application principles of the *m*-bit environment. Clause D.2.2 describes the encoding and decoding of $C_m(t)$ into/from server frame overhead. Clause D.2.3 describes the encoding and decoding of the non-integer portion of $c_m(t)$ into/from server frame overhead.

In cases where the relationship between the client and server clocks can be represented by a fixed rational number ratio (i.e., the server clock is phase-locked to the client clock), it is possible to communicate c_n in terms of a deterministic integer sequence of $C_n(t)$ values. This special case of GMP, referred to as bit synchronous GMP (BGMP), is described in clause D.2.4.

D.2.1 Application principles

The payload of the server frame is divided into *m*-bit entities that contain either client data or stuff (see Figure D.3). The choice of *m* is specified to suit the requirements associated with the CBR client and server channel. The equations from clause D.1 can be used to derive the number of *m*-bit data entities per server frame, as follows:

$$c_m = \left(\frac{n \times c_n}{m}\right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{m}\right) \quad \text{(D-14)}$$

As only an integer number of m-bit data entities can be transported per server frame period, the integer value $C_m(t)$ of c_m must be used. $C_m(t)$ will therefore vary between the integer portion of c_m and the next largest integer (i.e., $C_m(t) = \{\lfloor c_m \rfloor, \lceil c_m \rceil\}$.

Since it is required that no client information is lost, the rounding process to the integer value $C_m(t)$ has to take care of the truncated part (i.e., the non-integer portion of c_m that is not transmitted during the server frame period). The conceptual process for avoiding client data loss during rounding was described in D.1 for c_n and $C_n(t)$. In this case, the read out from the buffer (see Figure D.1) is controlled by the value of $C_m(t)$ where $C_m(t)$ corresponds to the integer number of *m*-bit data entities that will be available in the buffer for transmission during server frame period *t*+1. See Figure D.5. Since the client data arrives at the buffer with rate c_n, it is typical to express the data accumulating in the buffer due to the non-integer portion of c_m as a function of c_n. Specifically, the difference between the client data arriving in the buffer and the transmitted data curing a given server frame period is expressed by:

$$c_{nD} = c_n - \left(\frac{m}{n} \times C_m(t)\right) \quad \text{(D-15)}$$

The amount of c_{nD} data accumulating in the buffer represents the running phase difference between the client signal and the client data being transmitted over the server channel. This phase difference has two parts, the fractional portion of c_n and the remaining *n*-bit entities in the conversion of $C_n(t)$ to *m*-bit data words. Accumulation of the fractional portion of c_n leads to a variance in $C_n(t)$ as given by D-1 and D-2. The remainder in the conversion of $C_n(t)$ to *m*-bit data words is accumulated in $\Sigma C_{nD}(t)$, which can be signalled in the GMP overhead along with $C_m(t)$ in order to provide the GMP sink a more accurate (finer resolution) view of c_n at the source. Specifically, the integer value $\Sigma C_{nD}(t)$ is:

$$\sum C_{nD}(t) = \left[\sum C_{nD}(t-1) + C_n(t) mod \left(\frac{m}{n}\right)\right] mod \left(\frac{m}{n}\right) \quad \text{(D-16)}$$

where $C_n(t) mod \left(\frac{m}{n}\right)$ represents the number of *n*-bit data entities remaining after grouping $C_n(t)$ into *m*-bit words. $C_m(t)$ is the number of *m*-bit data words that will be transmitted in server frame period *t*. The $C_m(t)$ value is:

$$C_m(t) = \left\lfloor \frac{C_n(t)}{(m/n)} \right\rfloor + \left\lfloor \frac{\sum C_{nD}(t-1) + C_n(t) mod \left(\frac{m}{n}\right)}{(m/n)} \right\rfloor \quad \text{(D-17)}$$

In other words, $C_m(t)$ is determined by the combination of the appropriately scaled integer portion $C_n(t)$ and the accumulated ΣC_{nD} from the previous server frame (i.e., frame *t*-1).

As the client data has to fit into the payload area of the server signal, the maximum value of C_m and as such the maximum client bit rate is limited by the size of the server payload area.

$$C_m(t) \leq P_{m,server} \quad \text{(D-18)}$$

$P_{m,server}$: maximum number of (m-bit) data entities in the server payload area

D.2.1.1 Impact of client or server frequency tolerances

If the client or server bit rate changes due to client or server frequency tolerances, c_n and $C_n(t)$ change accordingly. f_{client} and f_{server} have a time dependent value between a minimum and maximum value, determined by their nominal value ± their tolerance value. As such, c_n will vary between:

$$c_{m,min} = \left(\frac{f_{client,min}}{f_{server,max}} \times \frac{B_{server}}{m}\right) \text{ and } c_{m,max} = \left(\frac{f_{client,max}}{f_{server,mmin}} \times \frac{B_{server}}{m}\right) \leq P_{m,server} \quad \text{(D-19)}$$

And consequently, $C_m(t)$ will vary between

$$C_{m,min}(t) = \left\lfloor \frac{f_{client,min}}{f_{server,max}} \times \frac{B_{server}}{m} \right\rfloor \text{ and } C_{m,max}(t) = \left\lceil \frac{f_{client,max}}{f_{server,mmin}} \times \frac{B_{server}}{m} \right\rceil \quad \text{(D-20)}$$

with the maximum value of $C_m(t)$ and as such the maximum client bit rate is limited by the size of the server payload area given in equations D-11, D-18 and D-21.

$$Min\ Server\ payload\ capacity = f_{server,min} \times \left(\frac{P_{server}}{B_{server}}\right) \times m \quad \text{(D-21)}$$

NOTE 1 – A transparent mapping determines $C_n(t)$ once per server frame period. During a step change in the client bit rate (e.g., during start up, or when the client signal is replaced by its AIS signal, or the AIS signal is replaced by the client signal), the current value of $C_n(t)$ will typically not match the actual number of n-bit client data entities arriving at the mapper buffer. The $C_n(t)$ determination process has to adjust its value to the actual number of n-bit client data entities arriving. This adjustment method is implementation specific. In these scenarios, during the transient period the $C_m(t)$ may temporarily exceed the above steady state range of equation D-20.

NOTE 2 – Any given instance of client and server frequency tolerance will have a relatively constant c_m value (i.e. $c_{m,x}$), so $C_{m,x}(t)$ will vary in the range $int(c_{m,x}) \leq C_m(t) \leq int(c_{m,x}) + 1$. Except for the above noted transient period, a value outside this range indicates that there is a misalignment of the expected client bit rate and the actual client bit rate.

D.2.1.2 Synchronizing the mapper and demapper

As shown in Figure D.5, at the mapper, the $C_n(t)$ described in clause D.1.2 is encoded and transmitted in the overhead of server frame period t-1 as $C_m(t)$ and $\Sigma C_{nD}(t)$, with $C_m(t)$ controlling the read out of client data from the buffer and insertion into the payload of server frame period t. Note that server frame t carries in its overhead the determined $C_m(t+1)$ value, in addition to the $C_m(t)$ m-bit client data entities in its payload. $C_m(t+1)$ n-bit client data entities are carried in server frame $t+1$.

At the de-mapper, $C_m(t)$ and $\Sigma C_{nD}(t)$ are extracted from the overhead and used to compute $C_n(t)$. $C_m(t)$ controls the write enable signal for the buffer. The client clock is generated based on the server clock and the value of $C_n(t)$.

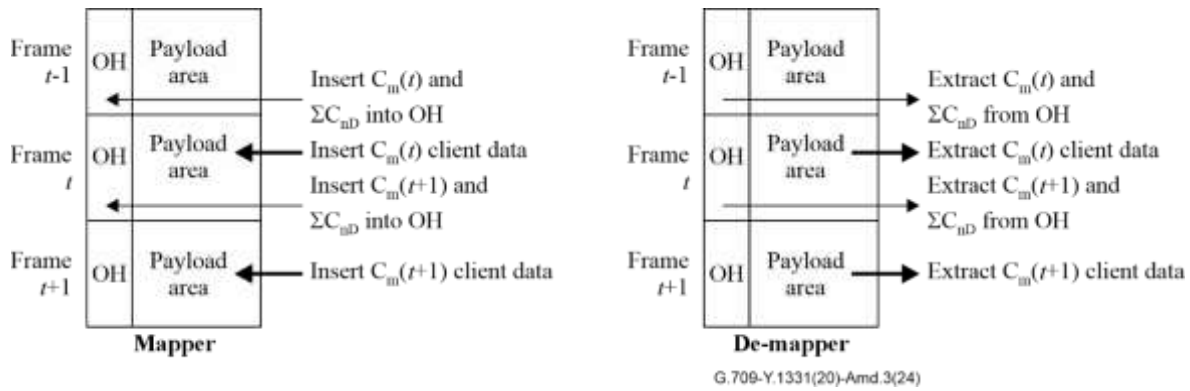


Figure D.5 – GMP processing flow

$C_m(t)$ client data entities are mapped into the payload area of the server frame using a sigma-delta data/stuff mapping distribution. It provides a distributed map as shown in Figure D.3. Payload field j (j = 1 .. P_{server}) carries:

– client data (D) if $(j \times C_m(t)) \bmod P_{server} < C_m(t)$ (D-22)

– stuff (S) if $(j \times C_m(t)) \bmod P_{server} \geq C_m(t)$. (D-23)

D.2.2 $C_m(t)$ encoding and decoding

D.2.2.1 Encoding

GMP uses a set of justification control overhead bits in the server frames to carry $C_m(t)$ and $\sum C_{nD}(t)$ information plus additional CRCs which protect $C_m(t)$ and $\sum C_{nD}(t)$ from the mapper to the demapper.

As described above, the server frame payload is divided into a number of *m*-bit entities, each of which can carry data or stuff. $C_m(t)$ is a count of the number of *m*-bit entities in the server frame payload that carry client data in the next frame; it has values between $\lfloor c_{m,\min} \rfloor$ and $\lceil c_{m,\max} \rceil$, which are client specific. Any $C_m(t)$ count value can be represented by an L-bit binary number, where $L = \lceil log_2(max.\#m\ bit\ entities\ in\ server\ frame\ payload) \rceil$ (i.e., L is the number of bits required to represent the maximum number of *m*-bit entities supported by the server payload area in a frame). Note that for overall robustness and convenience reasons explained below in clause D.2.2.2, L is always specified to be the smallest even number satisfying this equation. The L count bits (C[1:L]) are carried in the justification control overhead.

Because the value for any particular client/server combination varies in a narrow range, the robustness of the encoding of $C_m(t)$ can be improved by indicating the increment or decrement from the previous value $C_m(t\text{-}1)$ rather than the absolute value $C_m(t)$ except in cases where the value changes significantly (e.g., at start-up or due to a frequency jump in the client signal).

Specifically, in order to indicate that the $C_m(t)$ value it is sending has been incremented or decremented from the $C_m(t\text{-}1)$ value, the source sets the increment indicator (II) or decrement indicator (DI) bit and inverts a set of the L count bits (C[1:L]) that carry binary number representing $C_m(t\text{-}1)$. The specific count bit inversion patterns, which use orthogonal sets of bits to signal increment and decrement depend on the number of bits L and is server layer technology specific.

To protect against bit errors, these C[1:L], II and DI bits are to be carried in two parts in the server frame overhead with sufficient distance. The first part contains the $C\left[1{:}\left(\frac{L}{2}+1\right)\right]$ MSBs of the C[1:L] count bits. The second part contains the $C\left[\left(\frac{L}{2}+2\right){:}L\right]$ LSBs of C[1:L] count bits followed by the II and DI bits.

Further, the inversion pattern applied to the $C_m(t\text{-}1)$ bit values in the $C\left[\left(\frac{L}{2}+2\right){:}L\right]$ bits is applied to the $C_m(t\text{-}1)$ bit values in the $C\left[1{:}\left(\frac{L}{2}+1\right)\right]$ bits (i.e., the bits that occupy the same server byte bit positions have the same inversion pattern), and setting either the II or DI bit is mirrored by inverting the $C\left[\frac{L}{2}\right]$ or $C\left[\frac{L}{2}+1\right]$ bit, respectively.

To improve robustness, the bits in the first and second parts are protected by a $\left(\frac{L}{2}+1\right)$ bit CRC.

When $|C_m(t) - C_m(t\text{-}1)| \leq 2$, the bit inversion patterns are applied to the $C_m(t\text{-}1)$ value, prior to the increment or decrement operation, to generate the encoded value of $C_m(t)$ that will be signalled. The incremented or decremented $C_m(t)$ value becomes the base value for the next GMP overhead transmission. When no increment or decrement is performed (i.e., for $C_m(t) - C_m(t\text{-}1) = 0$), none of the bits of $C_m(t\text{-}1)$ are inverted. When the increment indicator (II) and decrement indicator (DI) bits are taken into account:

– When $0 < C_m(t) - C_m(t-1) \leq 2$, indicating an increment of +1 or +2, the appropriate inversion pattern is applied to the bits of the $C_m(t-1)$ value, the result is encoded, the II bit is set to 1 and the DI bit set to 0.

– When $0 > C_m(t) - C_m(t-1) \geq -2$, indicating a decrement of -1 or -2, the appropriate inversion pattern is applied to the bits of the $C_m(t-1)$ value, the result is encoded, the DI bit is set to 1 and the II bit set to 0.

– When the value of $C_m(t)$ is changed with a value larger than +2 or –2 from the value of $C_m(t-1)$, both the II and DI bits are set to 1 and new $C_m(t)$ value is encoded. The associated CRC in JC3 verifies whether the $C_m(t)$ value has been received correctly.

– When the value of $C_m(t)$ is unchanged from the value of $C_m(t-1)$, the $C_m(t)$ value is encoded and both the II and DI bits are set to 0.

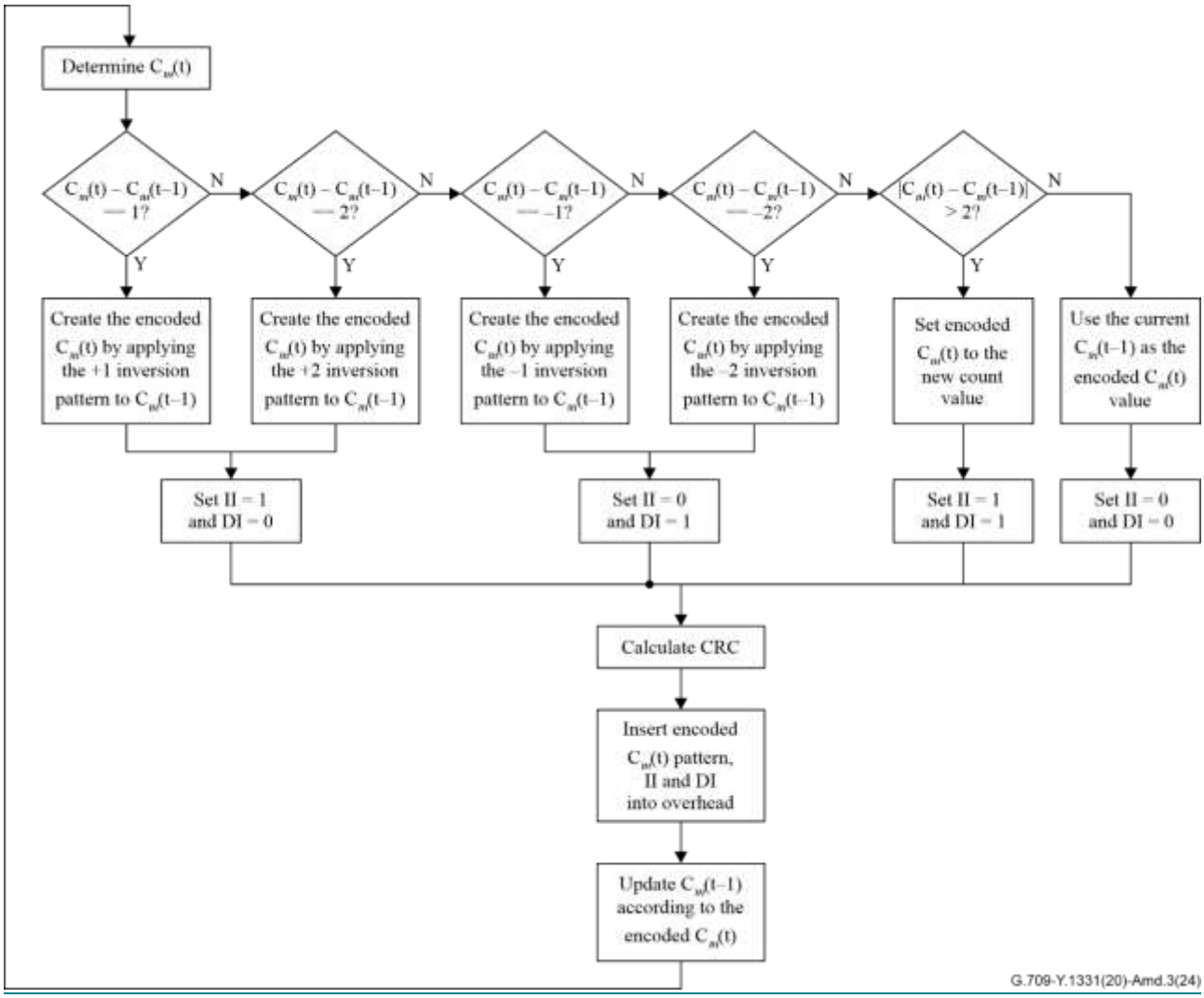


Figure D.6 – $C_m(t)$ encoding process

D.2.2.2 Decoding

GMP $C_m(t)$ alignment procedure

The GMP sink synchronizes its $C_m(t)$ value to the GMP source through the following process, which is illustrated in Figure D.7.

– When the GMP sink synchronization process is in the Start hunt state or in one of the Hunt states and receives justification control octets with a bad (invalid) CRC (i.e., xxxxB in Figure D.7), the GMP sink enters or remains in the Start hunt state.

– When the GMP sink synchronization process is in the Start hunt state and receives a frame with justification control bits containing II = DI and a valid CRC (i.e., x00G or x11G in Figure D.7), the GMP sink directly accepts the received C[1-L] bits as its $C_m(t)$ value for the next frame. At this point the GMP sink is synchronized to the GMP source.

– When the GMP sink synchronization process is in the Start hunt state and receives the frame *i* with justification control bits containing II ≠ DI with a valid (good) CRC (i.e., x10G or x01G in Figure D.7), the GMP sink must examine the received justification control overhead bits in the next frame (frame *i*+1) in order to obtain $C_m(t)$ synchronization, because the II and DI values do not indicate the magnitude of the increment or decrement that is being signalled. For this purpose, the GMP sink synchronization process enters the Hunt-A or Hunt-B state in Figure D.7.

The II ≠ DI in frame *i* indicates that the source is performing a count increment or decrement operation that will modify the $C_m(t)$ value it sends in frame *i*+1. Since this modification to the $C_m(t)$ will affect the count LSBs, the GMP sink uses the II, DI and count LSB in frame *i* to determine its synchronization hunt state when it receives frame *i*+1. Specifically, in Figure D.7, the Hunt state (A or B) is determined using C[L], II and DI.

- If II=DI with a valid CRC in frame *i*+1, $C_m(t)$ synchronization is achieved but directly accepting the received C[1-L] as the new $C_m(t)$.
- If II ≠ DI with a valid CRC in frame *i*+1, the sink uses the values of the new count LSB, II and DI to determine whether the source is communicating an increment or decrement operation and the magnitude of the increment/decrement step. This corresponds to the transition from the Hunt state row to the lower row of states (the "S" states) in Figure D.7. At this point, the GMP sink has identified the type of increment or decrement operation that is being signalled in frame *i*+1 and can determine the transmitted $C_m(t)$ value based on the corresponding inversion pattern.

Synchronization has now been achieved since the GMP sink has determined the current $C_m(t)$ and knows the expected $C_m(t)$ change in frame *i*+2.

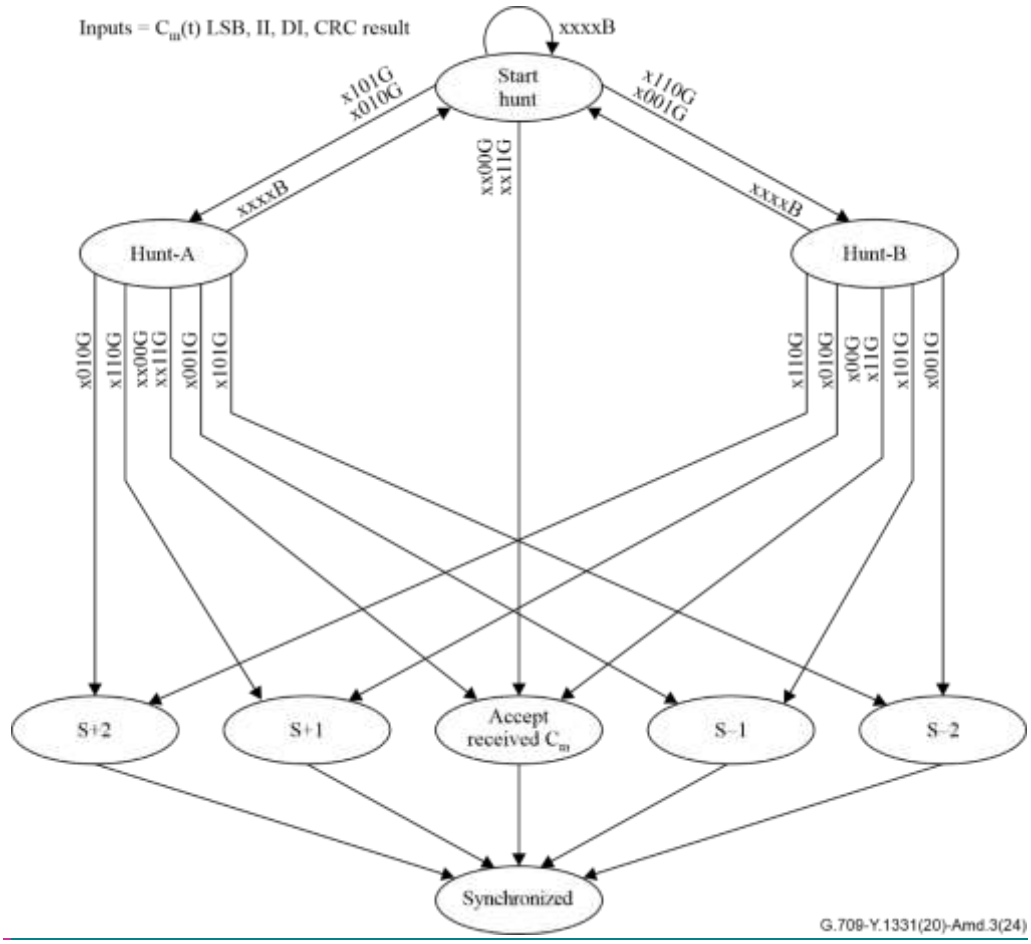


Figure D.7 – GMP sink count synchronization process diagram

Note that the state machine of Figure D.7 can also be used for off-line synchronization checking.

In-alignment GMP C_m(t) interpretation

When the GMP sink has synchronized its C_m(t) value to the GMP source, it interprets the received justification control overhead bits according to the following principles, making use of the first and second parts of the overhead described above in clause D.2.2.1:

– When the CRC is good and II = DI, the GMP sink accepts the received C_m(t) value.

– When the CRC is good and II ≠ DI, the GMP sink compares the received C_m(t) value to its expected C_m(t) value to determine the difference between these values. This difference is compared to the bit inversion patterns to determine the increment or decrement operation sent by the source and updates its C_m(t) accordingly. Since the CRC is good, the sink can use either the first part or second part of the justification control overhead bits for this comparison.

– When the CRC is bad, the GMP sink compares the received C_m(t) value to its current C_m(t) value. The sink then compares this difference between the received and current C_m(t) values to the valid received inversion patterns in the first part of the justification control overhead bits, and the bit inversion, II and DI pattern in the second part of the justification control overhead bits.

  • If both the first part and the second part of the justification control overhead bits contain valid patterns indicating the same no change, increment, or decrement operation, this indication is accepted, and the C_m(t) is updated accordingly.

- If one of the two parts of the justification control overhead bits contains a valid pattern and the other part does not, the operation indicated in the valid pattern is accepted, and the $C_m(t)$ is updated accordingly.
- If neither the first part nor the second part of the justification control overhead bits contain valid patterns, the sink shall keep its current count value and begin the search for synchronization.

NOTE 1 – If the first part and the second part of the justification control overhead bits each contain valid patterns that are different from each other, the receiver can either keep the current $C_m(t)$ value and begin a synchronization search, or it can use the CRC to determine whether the first part or the second part contains the correct pattern.

NOTE 2 – Choosing an even number for L has some robustness and complexity advantages. It guarantees that when II and DI are included, the same number of inversions exist in both the first part and the second part. Hence, both parts are equally strong (have the same Hamming distance) when trying to determine the correct value in the presence of errors. Using an odd number for L would add the potential complexity of expecting the receiver to give more weight to the part containing more of the inversion pattern.

The GMP sink uses the updated $C_m(t)$ value to extract the client data from the next frame or multiframe.

D.2.3 $\Sigma C_{nD}(t)$ encoding and decoding

The cumulative value of $C_{nD}(t)$ ($\Sigma C_{nD}(t)$) is encoded as a *k*-bit count value, where *k* is specified in order to achieve the desired GMP justification phase resolution. A *k*/2-bit CRC covering the *k*-bit field is calculated as follows, using a g(x) generator polynomial specified in the relevant server Recommendation.

The CRC-k/2 is calculated as follows over the D1-D*k* bits using the generator polynomial g(x) specified in the relevant server Recommendation:

1) The D[*i*] bits are taken in network transmission order, most significant bit first, to form a *k*-bit pattern representing the coefficients of a polynomial *M(x)* of degree (k-1).
2) *M(x)* is multiplied by x^k and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree (*k*/2 – 1) or less.
3) The coefficients of *R(x)* are considered to be a *k*/2-bit sequence, where $x^{k/2 - 1}$ is the most significant bit.
4) This *k*/2-bit sequence is the CRC-k/2 where the first bit of the CRC to be transmitted is the coefficient of $x^{k/2 - 1}$ and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits, resulting in *M(x)* having degree (1.5*k*-1). In the absence of bit errors, the remainder shall be all zeros.

The value of *k*, and the overhead bit positions occupied by *k*-bit field and the CRC are specified in the appropriate server layer Recommendation.

D.2.4 The bit synchronous GMP (BGMP) special case

The GMP overhead encoding method described in clauses D.2.1-D.2.3 could be used for any client and server pair, since it allows the client and server clocks to be independent. Another approach may be specified for the special case where the server clock is phase-locked to the client clock. In this case, the relationship between the client and server clocks involves a fixed ratio between two integers (i.e., the ratio is a rational number). Specifically, the fixed client-server clock ratio allows a bit-synchronous approach, with GMP used for distributing the client data and stuff information within the server payload frame. In other words, BGMP uses a systematically controlled *k*-bit sigma-delta

process to distribute the client payload and stuffing words within the server payload area, such that the server bit stream has a desired nominal bitrate and is phase-locked to the client stream.

The numerator and denominator of an integer ratio are typically designated as p and q, respectively. For example, consider a simple example where the ratio is 10.25. While this value could be considered to be 41/4, it is more conventional to regard it as Y + p/q where Y = 10 and p/q = 1/4. If 10.25 represents the value of c_m then it can be represented by transmitting a $C_m(t)$ integer sequence of 10,10,10,11, which averages to exactly 10.25.

We have the following sigma-delta process for generating and inserting $C_m(t)$:

– $C_m(t) = Y$ if $(j \times p) \bmod q < p$,

– $C_m(t) = Y+1$ if $(j \times p) \bmod q \geq p$

where the index "j" enumerates the number of frames in the frame sequence.

Since the fractional portion of c_m is presented by the deterministic $C_m(t)$ sequence, C_{nD} is not transmitted with BGMP.

The choice and method of determining Y and the p/q ratio depend on the server and the specific client. See clause 17.12 for an application of BGMP to OTN.

At the demapper, the received GMP overhead has the same meaning regardless of whether the mapper used dynamic GMP or BGMP. For the case of BGMP, the demapper can optionally make use of the known integer $C_m(t)$ sequence.

D.3 Applying GMP in an ODU

Clauses 17.7, 19.6 and 20.5 specify GMP as the asynchronous generic mapping method for the mapping of CBR client signals into OPUk, the mapping of ODUk signals into a server OPUk (via the ODTUk.ts) and the mapping of ODUk signals into an OPUCn (via ODTUCn.ts).

Asynchronous mappings in the OTN have a default 8-bit timing granularity. Such 8-bit timing granularity is supported in GMP by means of a c_n with n=8 (c_8). The jitter/wander requirements for some of the OTN client signals are such that for those signals an 8-bit timing granularity may not be sufficient. For such a case, a 1-bit timing granularity is supported in GMP by means of c_n with n=1 (c_1).

D.3.1 Mapping granularity

M-byte granularity mapping

Clauses 17.7 and 19.6 specify that the mapping of CBR client bits into the payload of an OPUk and the mapping of ODUj bits into the payload of an ODTUk.ts is performed with M-byte granularity, meaning the value m discussed in clause D.2 is $8 \times M$. The specific values of M for each server layer are described in Table D.1. The remaining $C_{nD}(t)$ data entities are signalled in the justification overhead as additional timing/phase information.

The corresponding c_m, $C_m(t)$, c_{nD} and $C_{nD}(t)$ values can be calculated using equations D-14 to D-17, with $m = 8 \times M$.

The OTN M-byte granularity uses the 14-bit $C_m(t)$ as described in clause D.3.2.1, with the corresponding $\Sigma C_{nD}(t)$ described in clause D.3.3.1.

16M-byte granularity mapping

Clause 20.5 specifies that the mapping of ODUk bits into the payload of an ODTUCn.ts is performed with 128 × M-bit (16M-byte) granularity, where M is the number of tributary slots occupied by the client signal, as shown in Table D.2. The value of m discussed in clause D.2 is $128 \times M$. The remaining $C_{nD}(t)$ data entities are signalled in the justification overhead as additional timing/phase information.

The corresponding c_m, $C_m(t)$, c_{nD} and $C_{nD}(t)$ values can be calculated using equations D-14 to D-17, with $m = 128 \times M$.

The OTN 16M-byte granularity uses the 10-bit $C_m(t)$ described in the example below in clause D.3.2.2, with the corresponding $C_{nD}(t)$ described in clause D.3.3.2.

Values of GMP parameters for OTN ODU servers

The values for n, m, M, f_{client}, f_{server}, T_{server}, B_{server}, and $P_{m,server}$ are specified in Table D.1.

Table D.1 – OPUk, ODTUk.ts and ODTUCn.ts GMP parameter values

GMP parameter	CBR client into OPUk	ODUj into OPUk (ODTUk.ts)	ODUk into OPUCn (ODTUCn.ts)
n	8 (default) 1 (client specific)	8	8
m	$m = 8 \times M$ OPU0: 8 × 1 = 8 OPU1: 8 × 2 = 16 OPU2: 8 × 8 = 64 OPU3: 8 × 32 = 256 OPU4: 8 × 80 = 640	$m = 8 \times M$ ODTU2.ts: 8 × ts ODTU3.ts: 8 × ts ODTU4.ts: 8 × ts ODTU25.ts: 8 × ts ODTU50.ts: 8 × ts	$m = 128 \times M$ ODTUCn.ts: 128 × ts
f_{client}	CBR client bit rate and tolerance	ODUj bit rate and tolerance (Table 7-2)	ODUk bit rate and tolerance (Table 7-2)
f_{server}	OPUk Payload bit rate and tolerance (Table 7-3)	ODTUk.ts Payload bit rate and tolerance (Table 7-7)	ODTUCn.ts Payload bit rate and tolerance (Table 7-7)
T_{server}	ODUk/OPUk frame period (Table 7-4)	OPUk multiframe period (Table 7-6)	OPUCn multiframe period (Table 7-6)
B_{server}	OPU0: 8 × 15232 OPU1: 8 × 15232 OPU2: 8 × 15232 OPU3: 8 × 15232 OPU4: 8 × 15200	ODTU2.ts: 8 × ts × 15232 ODTU3.ts: 8 × ts × 15232 ODTU4.ts: 8 × ts × 15200 ODTU25.ts: 8 × ts × 15232 ODTU50.ts: 8 × ts × 15232	ODTUCn.ts: 128 × ts × 952
$P_{m,server}$	OPU0: 15232 OPU1: 7616 OPU2: 1904 OPU3: 476 OPU4: 190	ODTU2.ts: 15232 ODTU3.ts: 15232 ODTU4.ts: 15200 ODTU25.ts: 15232 ODTU50.ts: 15232	ODTUCn.ts: 952
ΣC8D range	OPU0: N/A OPU1: 0 to +1 OPU2: 0 to +7 OPU3: 0 to +31 OPU4: 0 to +79	ODTUk.1: N/A ODTUk.2: 0 to +1 ODTUk.3: 0 to +2 ODTUk.4: 0 to +3 : ODTUk.8: 0 to +7 : ODTUk.32: 0 to +31 : ODTUk.79: 0 to +78 ODTUk.80: 0 to +79	ODTUCn.1: 0 to +15 ODTUCn.2: 0 to +31 ODTUCn.3: 0 to +47 ODTUCn.4: 0 to +59 : ODTUCn.20n-1: 0 to +320n-1 ODTUCn.20n: 0 to +320n
ΣC1D range (for selected clients)	OPU0: 0 to +7 OPU1: 0 to +15 OPU2: 0 to +63	Not applicable	Not applicable

Table D.1 – OPUk, ODTUk.ts and ODTUCn.ts GMP parameter values

GMP parameter	CBR client into OPUk	ODUj into OPUk (ODTUk.ts)	ODUk into OPUCn (ODTUCn.ts)
	OPU3: 0 to +255 OPU4: 0 to +639		

D.3.2 OTN $C_m(t)$ encoding and decoding

$C_m(t)$ is encoded in the OPUk and ODTUk.ts justification control bytes JC1, JC2 and JC3 specified in clause 19.4 for the 14-bit count and clause 20.4 for the 10-bit count field. Note that JC1 carries the first part, JC2 carries the second part and JC3 carries the CRC of the justification control overhead bits specified in D.2.2.1 and D.2.2.2.

D.3.2.1 OTN $C_m(t)$ encoding and decoding for OPUk

The encoding of $C_m(t)$into bytes JC1 and JC2 for OPUk is defined using the 14-bit count field of Table D.2. An "I" entry in the table indicates an inversion of that bit. A CRC-8 in JC3 verifies whether the transmitted $C_m(t)$ value has been received correctly.

Table D.2 – 14-bit $C_m(t)$ increment and decrement indicator patterns

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	II	DI	Change
U	U	U	U	U	U	U	U	U	U	U	U	U	U	0	0	0
I	U	I	U	I	U	I	U	I	U	I	U	I	U	1	0	+1
U	I	U	I	U	I	U	I	U	I	U	I	U	I	0	1	–1
U	I	I	U	U	I	I	U	U	I	I	U	U	I	1	0	+2
I	U	U	I	I	U	U	I	I	U	U	I	I	U	0	1	–2
binary value														1	1	More than +2/–2

NOTE:
– I indicates inverted Cj bit
– U indicates unchanged Cj bit

The CRC-8 located in JC3 is calculated over the JC1 and JC2 bits. The CRC-8 uses the $g(x) = x^8 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC1 and JC2 octets are taken in network octet order, most significant bit first, to form a 16-bit pattern representing the coefficients of a polynomial *M(x)* of degree 15.

2) *M(x)* is multiplied by x^8 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 7 or less.

3) The coefficients of *R(x)* are considered to be an 8-bit sequence, where x^7 is the most significant bit.

4) This 8-bit sequence is the CRC-8 where the first bit of the CRC-8 to be transmitted is the coefficient of x^7 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits of JC3, resulting in *M(x)* having degree 23. In the absence of bit errors, the remainder shall be 0000 0000.

A parallel logic implementation of the source CRC-8 associated with the 14-bit $C_m(t)$ encoding of Table D.2 is illustrated in Appendix VI.

Figure D.7 is applied to the GMP sink synchronization for the 14-bit $C_m(t)$. When II ≠ DI in frame *i*, the Hunt state is determined by the values of the II and DI bits and the count LSB (C14) in frame *i*. When frame *i*+1 is received, the Figure D.7 synchronization state machine "S" state interpretation associated with the 14-bit count field is specified in Table D.3.

Table D.3 – "S" state interpretation (see Figure D.7) with 14-bit C_m

S state	Interpretation/Action
S+2	Count = C1-C14 after inverting C2, C3, C6, C7, C10, C11, & C14; Increment +2 for the next frame
S+1	Count = C1-C14 after inverting C1, C3, C5, C7, C9, C11, & C13; Increment +1 for the next frame
S-1	Count = C1-C14 after inverting C2, C4, C6, C8, C10, C12, & C14; Decrement -1 for next frame
S-2	Count = C1-C14 after inverting C1, C4, C5, C8, C9, C12, C13; Decrement -2 for next frame

When synchronized, the GMP sink uses the updated $C_m(t)$ value to extract the client data from the next OPU frame or ODTUk.ts multiframe as described in clause D.2.2, interpreting the received JC octets according to the inversion patterns of Table D.2.

D.3.2.2 OTN $C_m(t)$ encoding and decoding for OPU*Cn*

The $C_m(t)$ encoding specific to the OTN 10-bit count field in bytes JC1 and JC2 used by OPU*Cn* is shown in Table D.4.

Table D.4 – 10-bit $C_m(t)$ increment and decrement indicator patterns

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	II	DI	Change
U	U	U	U	U	U	U	U	U	U	0	0	0
I	U	I	U	I	U	I	U	I	U	1	0	+1
I	U	U	I	U	I	I	U	U	I	0	1	–1
U	I	U	I	I	U	U	I	U	I	1	0	+2
U	I	I	U	U	I	U	I	I	U	0	1	–2
binary value										1	1	More than +2/–2

NOTE:
– I indicates inverted Cj bit
– U indicates unchanged Cj bit

The CRC-6 located in JC3 is calculated over bits 3-8 of JC1 and JC2 (i.e., the bits containing the GMP overhead value shown in Table D.4). The CRC-6 uses the $g(x) = x^6 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC1 and JC2 octets are taken in network octet order, most significant bit first, such that bit 3 of JC1 through bit 8 of JC2 form a 12-bit pattern representing the coefficients of a polynomial *M(x)* of degree 11.

2) *M(x)* is multiplied by x^6 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 5 or less.

3) The coefficients of *R(x)* are considered to be a 6-bit sequence, where x^5 is the most significant bit.

4) This 6-bit sequence is the CRC-6 where the first bit of the CRC-6 to be transmitted is the coefficient of x^5 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits of JC3, resulting in M(x) having degree 17. In the absence of bit errors, the remainder shall be 000000.

A parallel logic implementation of the source CRC-6 associated with the 10-bit $C_m(t)$ encoding of Table D.4 is illustrated in Appendix VI.

Figure D.7 is applied to the GMP sink synchronization for the 10-bit $C_m(t)$. When II ≠ DI in frame *i*, the Hunt state is determined by the values of the II and DI bits and the count LSB (C10) in frame *i*. When frame *i*+1 is received, the Figure D.7 synchronization state machine "S" state interpretation associated with the 10-bit count field is specified in Table D.5.

Table D.5 – "S"-state interpretation for 10-bit $C_m(t)$ (see Figure D.7)

S state	Interpretation/Action
S+2	Count = C1-C10 after inverting C2, C4, C5, C8 & C10; Increment +2 for the next frame
S+1	Count = C1-C10 after inverting C1, C3, C5, C7 & C9; Increment +1 for the next frame
S-1	Count = C1-C10 after inverting C1, C4, C6, C7 & C10; Decrement -1 for next frame
S-2	Count = C1-C10 after inverting C2, C3, C6, C8 & C9; Decrement -2 for next frame

When synchronized, the GMP sink uses the updated $C_m(t)$ value to extract the client data from the next OPU*Cn* frame as described in clause D.2.2, interpreting the received JC octets according to the inversion patterns of Table D.4.

D.3.3 OTN $\Sigma C_{nD}(t)$ encoding and decoding

D.3.3.1 $\Sigma C_{nD}(t)$ encoding and decoding for OPUk

The cumulative value of $C_{nD}(t)$ ($\Sigma C_{nD}(t)$) is encoded in bits 4-8 of the OPUk and ODTUk.ts justification control bytes JC4, JC5 and JC6. Bits D1 to D10 in JC4 and JC5 carry the value of $\Sigma C_{nD}(t)$. Bit D1 carries the most significant bit and bit D10 carries the least significant bit.

The CRC-5 located in JC6 is calculated over the D1-D10 bits in JC4 and JC5. The CRC-5 uses the $g(x) = x^5 + x + 1$ generator polynomial, and is calculated as follows:

1) The JC4 bits 4-8 and JC5 bits 4-8 octets are taken in network transmission order, most significant bit first, to form a 10-bit pattern representing the coefficients of a polynomial *M(x)* of degree 9.
2) *M(x)* is multiplied by x^5 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 4 or less.
3) The coefficients of *R(x)* are considered to be a 5-bit sequence, where x^4 is the most significant bit.
4) This 5-bit sequence is the CRC-5 where the first bit of the CRC-5 to be transmitted is the coefficient of x^4 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits of JC6, resulting in *M(x)* having degree 14. In the absence of bit errors, the remainder shall be 00000.

A parallel logic implementation of the source CRC-5 is illustrated in Appendix VI.

D.3.3.2 $\Sigma C_{nD}(t)$ encoding and decoding for OPU*Cn*

The cumulative value of $C_{nD}(t)$ ($\Sigma C_{nD}(t)$) is encoded in the ODTUCn.ts justification control bytes JC1, JC2, JC3, JC4, JC5 and JC6. Bits D1 to D18 carry the value of $\Sigma C_{nD}(t)$. Bit D1 carries the most significant bit and bit D18 carries the least significant bit. As shown in Figure 20-7, bits D1 to D7 are located in bits 2-8 of JC4, bits D8 to D9 are located in bits 1-2 of JC1, bits D10 to D16 are located in JC5, and bits D17 to D18 are located in bits 1-2 of JC2.

The CRC-9 located in bits 2-8 of JC6 and bits 1-2 of JC3 is calculated over the D1-D18 bits in JC4, JC1, JC5 and JC2. The CRC-9 uses the $g(x) = x^9 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC4 bits 2-8, JC1 bits 1-2, JC5 bits 2-8, and JC2 bits 1-2 are taken in network transmission order, most significant bit first, to form an 18-bit pattern representing the coefficients of a polynomial *M(x)* of degree 17.
2) *M(x)* is multiplied by x^9 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 8 or less.
3) The coefficients of *R(x)* are considered to be a 9-bit sequence, where x^8 is the most significant bit.
4) This 9-bit sequence is the CRC-9 where the first bit of the CRC-9 to be transmitted is the coefficient of x^8 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits of JC6 and JC3, resulting in *M(x)* having degree 26. In the absence of bit errors, the remainder shall be 000000000.

A parallel logic implementation of the source CRC-9 is illustrated in Appendix VI.

~~**D.1 Basic principle**~~

~~For any given CBR client signal, the number of n-bit (e.g., n = 1/8, 1, 8) data entities that arrive during one server frame or server multiframe period is defined by:~~

$$c_n = \left(\frac{f_{client}}{n} \times T_{server} \right) \quad \text{(D-1)}$$

~~f_{client}: client bit rate~~

~~T_{server}: frame period of the server frame or server multiframe~~

~~c_n: number of client n-bit data entities per server frame or server multiframe~~

~~As only an integer number of n-bit data entities can be transported per server frame or multiframe, the integer value $C_n(t)$ of c_n has to be used. Since it is required that no client information is lost, the rounding process to the integer value has to take care of the truncated part, e.g., a c_n with a value of 10.25 has to be represented by the integer sequence 10,10,10,11.~~

$$C_n(t) = \text{int}\left(\frac{f_{client}}{n} \times T_{server} \right) \quad \text{(D-2)}$$

~~$C_n(t)$:number of client n-bit data entities per server frame t or server multiframe t (integer)~~

~~For the case c_n is not an integer, $C_n(t)$ will vary between:~~

$$C_n(t) = floor\left(\frac{f_{client}}{n} \times T_{server} \right) \quad \text{(D-3)}$$

~~and~~

$$C_n(t) = ceiling\left(\frac{f_{client}}{n} \times T_{server}\right) = 1 + floor\left(\frac{f_{client}}{n} \times T_{server}\right) \quad \text{(D-4)}$$

The server frame or multiframe rate is defined by the server bit rate and the number of bits per server frame or multiframe:

$$T_{server} = \frac{B_{server}}{f_{server}} \quad \text{(D-5)}$$

f_{server}: server bit rate

B_{server}: bits per server frame or multiframe

Combining (D-5) with (D-1) and (D-2) results in:

$$c_n = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{n}\right) \quad \text{(D-6)}$$

and

$$C_n(t) = \text{int}\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{n}\right) \quad \text{(D-7)}$$

As the client data has to fit into the payload area of the server signal, the maximum value of C_n and as such the maximum client bit rate is limited by the size of the server payload area.

$$C_n(t) \leq P_{server} \quad \text{(D-8)}$$

$$f_{client} \leq f_{server} \times \frac{P_{server}}{B_{server}} \times n \quad \text{(D-9)}$$

P_{server}: maximum number of (n bits) data entities in the server payload area

The client and server bit rate are independent. This allows specifying the server bit rate independently from the client bit rates. Furthermore, client clock impairments are not seen at the server clock.

If the client or server bit rate changes due to client or server frequency tolerances, c_n and $C_n(t)$ change accordingly. A special procedure has to take care that $C_n(t)$ is changed fast enough to the correct value during start-up or during a step in the client bit rate (e.g., when the client signal is replaced by its AIS signal or the AIS signal is replaced by the client signal). This procedure may be designed to prevent buffer over-/underflow, or an additional buffer over-/underflow prevention method has to be deployed.

A transparent mapping has to determine $C_n(t)$ on a server (multi)frame per (multi)frame base.

In order to extract the correct number of client information entities at the de-mapper, $C_n(t)$ has to be transported in the overhead area of the server frame or multiframe from the mapper to the de-mapper.

Figure D.1 shows the generic functionality of the mapper and de-mapper circuit.

At the mapper, $C_n(t)$ is determined based on the client and server clocks. The client data is constantly written into the buffer memory. The read out is controlled by the value of $C_n(t)$.At the de-mapper, $C_n(t)$ is extracted from the overhead. $C_n(t)$ controls the write enable signal for the buffer. The client clock is generated based on the server clock and the value of $C_n(t)$.

$C_n(t)$ has to be determined first, then it has to be inserted into the overhead and afterwards $C_n(t)$ client data entities have to be inserted into the payload area of the server as shown in Figure D.2.

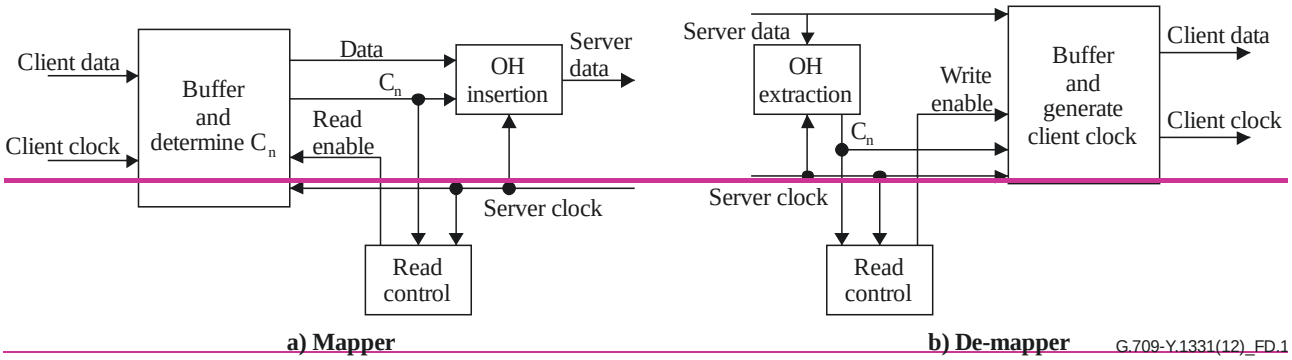


Figure D.1 – Generic functionality of a mapper/de-mapper circuit

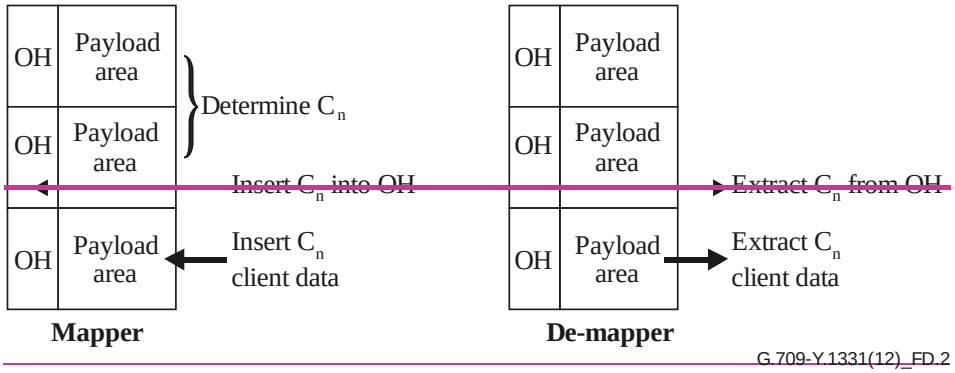


Figure D.2 – Processing flow

$C_n(t)$ client data entities are mapped into the payload area of the server frame or multiframe using a sigma-delta data/stuff mapping distribution. It provides a distributed mapping as shown in Figure D.3. Payload field j (j = 1 .. P_{server}) carries:

client data (D) if $(j \times C_n(t)) \bmod P_{server} < C_n(t)$ (D-10)

stuff (S) if $(j \times C_n(t)) \bmod P_{server} \geq C_n(t)$. (D-11)

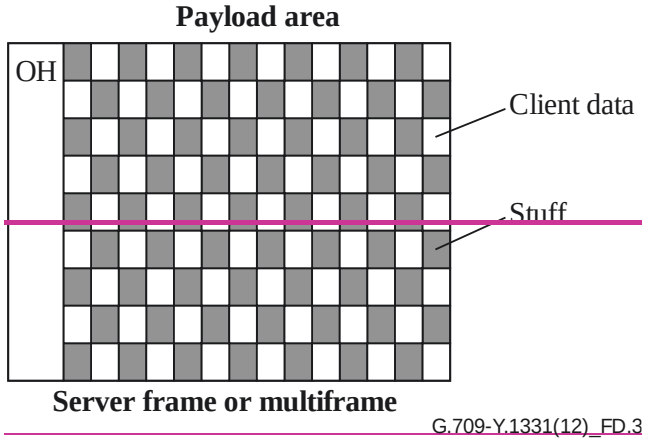


Figure D.3 – Sigma-delta based mapping

$C_n(t)$ client data entities have to be distributed over P_{server} locations. A client data entity has therefore to be inserted with a spacing of $\frac{P_{server}}{C_n(t)}$. This is normally not an integer value, however it can be

emulated by an integer calculation using the sigma-delta method based on an overflow accumulator as shown in Figure D.4.

The accumulator memory is reset to 0 at every frame start of the server frame. At every location of the payload area, $C_n(t)$ is added to the memory and the result is compared with P_{server}. If the result is lower than P_{server}, it is stored back into the memory and no client data is indicated for this payload position. If it is equal or greater than P_{server}, P_{server} is subtracted from the result and the new result is stored back in the memory. In addition, client data is indicated for the client position.

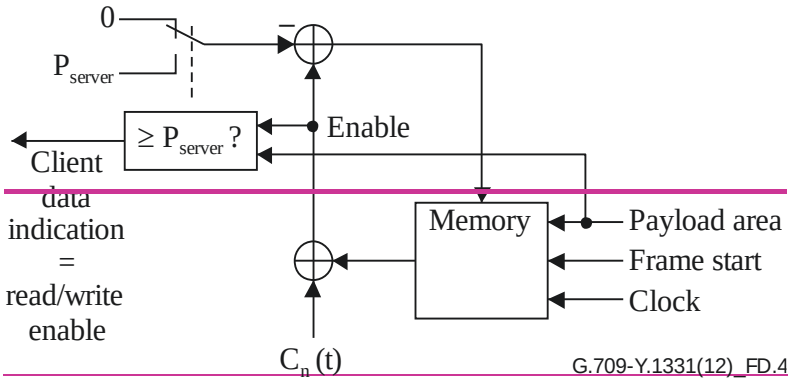


Figure D.4 – Sigma-delta accumulator

As the same start value and $C_n(t)$ are used at the mapper and de-mapper the same results are obtained and interworking is achieved.

D.2 Applying GMP in OTN

Clauses 17.7, 19.6 and 20.5 specify GMP as the asynchronous generic mapping method for the mapping of CBR client signals into OPUk, the mapping of ODUk signals into a server OPUk (via the ODTUk.ts) and the mapping of ODUk signals into an OPUCn (via ODTUCn.ts).

NOTE – GMP complements the traditional asynchronous client/server specific mapping method specified in clauses 17.6 and 19.5. GMP is intended to provide the justification of new CBR type client signals into OPUk.

Asynchronous mappings in the OTN have a default 8-bit timing granularity. Such 8-bit timing granularity is supported in GMP by means of a c_n with n=8 (c_8). The jitter/wander requirements for some of the OTN client signals are such that for those signals an 8-bit timing granularity may not be sufficient. For such a case, a 1-bit timing granularity is supported in GMP by means of c_n with n=1 (c_1).

M-byte granularity mapping

Clauses 17.7 and 19.6 specify that the mapping of CBR client bits into the payload of an OPUk and the mapping of ODUj bits into the payload of an ODTUk.ts is performed with 8 × M-bit (M-byte) granularity.

The insertion of CBR client data into the payload area of the OPUk frame and the insertion of ODUj data into the payload area of the ODTUk.ts multiframe at the mapper is performed in M-byte (or m-bit, m = 8 × M) data entities, denoted as $C_m(t)$. The remaining $C_{nD}(t)$ data entities are signalled in the justification overhead as additional timing/phase information.

$$c_m = \left(\frac{n \times c_n}{m} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{m} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{8 \times M} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/8}{M} \right) \quad \text{(D-12)}$$

As only an integer number of m-bit data entities can be transported per server frame or multiframe, the integer value $C_m(t)$ of c_m has to be used. Since it is required that no information is lost, the

rounding process to the integer value has to take care of the truncated part, e.g., a c_m with a value of 10.25 has to be represented by the integer sequence 10,10,10,11.

$$C_m(t) = \text{int}(c_m) = \text{int}\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/8}{M} \right) \quad \text{(D-13)}$$

For the case c_m is not an integer, $C_m(t)$ will vary between:

$$C_m(t) = floor\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/8}{M} \right) \text{and} C_m(t) = ceiling\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/8}{M} \right) \quad \text{(D-14)}$$

The remainder of c_n and $C_m(t)$ is:

$$c_{nD} = c_n - \left(\frac{8 \times M}{n} \times C_m(t) \right) \quad \text{(D-15)}$$

As only an integer number of c_{nD} n-bit data entities can be signalled per server frame or multiframe, the integer value $C_{nD}(t)$ of c_{nD} has to be used.

$$C_{nD}(t) = \text{int}(c_n) - \left(\frac{8 \times M}{n} \times C_m(t) \right) = C_n(t) - \left(\frac{8 \times M}{n} \times C_m(t) \right) \quad \text{(D-16)}$$

$C_{nD}(t)$ is a number between $1 - \frac{8 \times M}{n}$ and $\frac{8 \times M}{n} - 1$.

16M-byte granularity mapping

Clause 20.5 specifies that the mapping of ODUk bits into the payload of an ODTUCn.ts is performed with 128 × M-bit (16M-byte) granularity.

The insertion of ODUk data into the payload area of the ODTUCn.ts multiframe at the mapper is performed in 16M-byte (or m-bit, m = 128 × M) data entities, denoted as $C_m(t)$. The remaining $C_{nD}(t)$ data entities are signalled in the justification overhead as additional timing/phase information.

$$c_m = \left(\frac{n \times c_n}{m} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{m} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}}{128 \times M} \right) = \left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/128}{M} \right) \quad \text{(D-17)}$$

As only an integer number of m-bit data entities can be transported per server frame or multiframe, the integer value $C_m(t)$ of c_m has to be used. Since it is required that no information is lost, the rounding process to the integer value has to take care of the truncated part, e.g., a c_m with a value of 10.25 has to be represented by the integer sequence 10,10,10,11.

$$C_m(t) = \text{int}(c_m) = \text{int}\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/128}{M} \right) \quad \text{(D-18)}$$

For the case c_m is not an integer, $C_m(t)$ will vary between:

$$C_m(t) = floor\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/128}{M} \right) \text{and} C_m(t) = ceiling\left(\frac{f_{client}}{f_{server}} \times \frac{B_{server}/128}{M} \right) \quad \text{(D-19)}$$

The remainder of c_n and $C_m(t)$ is:

$$c_{nD} = c_n - \left(\frac{128 \times M}{n} \times C_m(t) \right) \quad \text{(D-20)}$$

As only an integer number of c_{nD} n-bit data entities can be signalled per server frame or multiframe, the integer value $C_{nD}(t)$ of c_{nD} has to be used.

$$C_{nD}(t) = \text{int}(c_n)\left(\frac{128\times M}{n}\times C_m(t)\right) = C_n(t)\left(\frac{128\times M}{n}\times C_m(t)\right) \quad \text{(D-21)}$$

$C_{nD}(t)$ is a number between $\frac{128\times M}{n} - 1$ and $\frac{128\times M}{n} - 1$.

As the client data has to fit into the payload area of the server signal, the maximum value of C_m and as such the maximum client bit rate is limited by the size of the server payload area.

$$C_m(t) \le P_{m,server} \quad \text{(D-22)}$$

$P_{m,server}$: maximum number of (m bits) data entities in the server payload area

In order to extract the correct number of client information entities at the de-mapper, $C_m(t)$ has to be transported in the overhead area of the server frame or multiframe from the mapper to the de-mapper.

At the mapper, $C_n(t)$ is determined based on the client and server clocks. The client data is constantly written into the buffer memory. The read-out is controlled by the value of $C_m(t)$.

At the de-mapper, $C_m(t)$ and $C_{nD}(t)$ are extracted from the overhead and used to compute $C_n(t)$. $C_m(t)$ controls the write enable signal for the buffer. The client clock is generated based on the server clock and the value of $C_n(t)$.

$C_n(t)$ has to be determined first, then it has to be inserted into the overhead as $C_m(t)$ and $\Sigma C_{nD}(t)$ and afterwards $C_m(t)$ client data entities have to be inserted into the payload area of the server as shown in Figure D.5.

The $C_n(t)$ value determines the $C_m(t)$ and $C_{nD}(t)$ values; $C_m(t) = \text{floor}(n/m \times C_n(t))$ and $C_{nD}(t) = C_n(t) - (m/n \times C_m(t))$. The values of $C_{nD}(t)$ are accumulated and if $\Sigma C_{nD}(t) \ge m/n$ then m/n is subtracted from $\Sigma C_{nD}(t)$ and $C_m(t)$ is incremented with +1. These latter two values are then encoded in the overhead bytes. This $C_m(t)$ value is applied as input to the sigma-delta process.

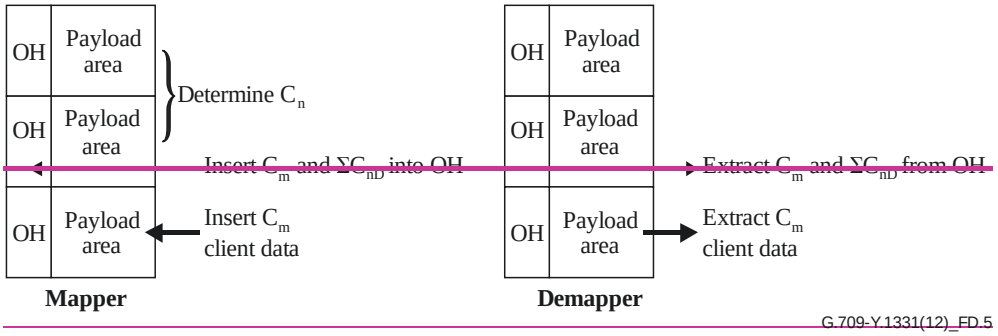


Figure D.5 – Processing flow for GMP in OTN

During start-up or during a step in the client bit rate, the value of $C_n(t)$ will not match the actual number of n-bit client data entities arriving at the mapper buffer and the $C_n(t)$ determination process has to adjust its value to the actual number of n-bit client data entities arriving. This adjustment method is implementation specific. During the mismatch period, the mapper buffer fill level may increase if more n-bit client data entities arrive per multiframe than there are transmitted, or decrease if less n-bit client data entities arrive per multiframe than there are transmitted.

To prevent overflow or underflow of the mapper buffer and thus data loss, the fill level of the mapper buffer has to be monitored. For the case where too many m-bit client data entities are in the buffer, it is necessary to insert temporarily more m-bit client data entities in the server (multi)frame(s) than required by $C_n(t)$. For the case too few m-bit client data entities are in the buffer, it is necessary to insert temporarily fewer m-bit client data entities in the server (multi)frame(s) than required by $C_n(t)$. This behaviour is similar to the behaviour of AMP under these conditions.

The OTN supports a number of client signal types for which transfer delay (latency) and transfer delay variation are critical parameters. Those client signal types require that the transfer delay introduced by the mapper plus de-mapper buffers is minimized and that the delay variation introduced by the mapper plus de-mapper buffers is minimized.

In steady state periods, $C_n(t)$ is a value in the range $C_{n,min}$ to $C_{n,max}$. A value outside this range indicates that there is a misalignment of the expected client bit rate and the actual client bit rate. During transient periods after e.g., a frequency step, $C_n(t)$ may be temporarily outside the range $C_{n,min}$ to $C_{n,max}$.

$C_m(t)$ client data entities are mapped into the payload area of the server frame or multiframe using a sigma-delta data/stuff mapping distribution. It provides a distributed mapping as shown in Figure D.3. Payload field j (j = 1 .. $P_{m,server}$) carries

client data (D) if $(j \times C_m(t)) \bmod P_{m,server} < C_m(t)$; (D-23)

stuff (S) if $(j \times C_m(t)) \bmod P_{m,server} \geq C_m(t)$. (D-24)

Values of n, m, M, f_{client}, f_{server}, T_{server}, B_{server}, and $P_{m,server}$ for OPUk and ODTUk.ts

The values for n, m, M, f_{client}, f_{server}, T_{server}, B_{server}, and $P_{m,server}$ are specified in Table D.1.

Table D.1 – OPUk, ODTUk.ts and ODTUCn.ts GMP parameter values

GMP parameter	CBR client into OPUk	ODUj into OPUk (ODTUk.ts)	ODUk into OPUCn (ODTUCn.ts)
n	8 (default) 1 (client specific)	8	8
m	$m = 8 \times M$ OPU0: 8 × 1 = 8 OPU1: 8 × 2 = 16 OPU2: 8 × 8 = 64 OPU3: 8 × 32 = 256 OPU4: 8 × 80 = 640	$m = 8 \times M$ ODTU2.ts: 8 × ts ODTU3.ts: 8 × ts ODTU4.ts: 8 × ts ODTU25.ts: 8 × ts ODTU50.ts: 8 × ts	$m = 128 \times M$ ODTUCn.ts: 128 × ts
f_{client}	CBR client bit rate and tolerance	ODUj bit rate and tolerance (Table 7-2)	ODUk bit rate and tolerance (Table 7-2)
f_{server}	OPUk Payload bit rate and tolerance (Table 7-3)	ODTUk.ts Payload bit rate and tolerance (Table 7-7)	ODTUCn.ts Payload bit rate and tolerance (Table 7-7)
T_{server}	ODUk/OPUk frame period (Table 7-4)	OPUk multiframe period (Table 7-6)	OPUCn multiframe period (Table 7-6)
B_{server}	OPU0: 8 × 15232 OPU1: 8 × 15232 OPU2: 8 × 15232 OPU3: 8 × 15232 OPU4: 8 × 15200	ODTU2.ts: 8 × ts × 15232 ODTU3.ts: 8 × ts × 15232 ODTU4.ts: 8 × ts × 15200 ODTU25.ts: 8 × ts × 15232 ODTU50.ts: 8 × ts × 15232	ODTUCn.ts: 128 × ts × 952
$P_{m,server}$	OPU0: 15232 OPU1: 7616 OPU2: 1904 OPU3: 476 OPU4: 190	ODTU2.ts: 15232 ODTU3.ts: 15232 ODTU4.ts: 15200 ODTU25.ts: 15232 ODTU50.ts: 15232	ODTUCn.ts: 952
ΣC_{8D} range	OPU0: N/A OPU1: 0 to +1 OPU2: 0 to +7	ODTUk.1: N/A ODTUk.2: 0 to +1 ODTUk.3: 0 to +2 ODTUk.4: 0 to +3	ODTUCn.1: 0 to +15 ODTUCn.2: 0 to +31 ODTUCn.3: 0 to +47 ODTUCn.4: 0 to +59

Table D.1 – OPUk, ODTUk.ts and ODTUCn.ts GMP parameter values

GMP parameter	CBR client into OPUk	ODUj into OPUk (ODTUk.ts)	ODUk into OPUCn (ODTUCn.ts)
	OPU3: 0 to +31 OPU4: 0 to +79	÷ ODTUk.8: 0 to +7 ÷ ODTUk.32: 0 to +31 ÷ ODTUk.79: 0 to +78 ODTUk.80: 0 to +79	÷ ODTUCn.20n-1: 0 to +320n-1 ODTUCn.20n: 0 to +320n
ΣC_{1D} range (for selected clients)	OPU0: 0 to +7 OPU1: 0 to +15 OPU2: 0 to +63 OPU3: 0 to +255 OPU4: 0 to +639	Not applicable	Not applicable

D.3 $C_m(t)$ encoding and decoding

$C_m(t)$ is encoded in the ODTUk.ts justification control bytes JC1, JC2 and JC3 specified in clause 19.4 for the 14-bit count and clause 20.4 for the 10-bit count field.

$C_m(t)$ is an L-bit binary count of the number of groups of m OPU payload bits that carry m client bits; it has values between Floor($C_{m,min}$) and Ceiling($C_{m,max}$), which are client specific. The Ci (i=1..L) bits that comprise $C_m(t)$ are used to indicate whether the $C_m(t)$ value is incremented or decremented from the value in the previous frame, that is indicated by $C_m(t-1)$. Tables D.2 and D.3 show the inversion patterns for the Ci bits of $C_m(t-1)$ that are inverted to indicate an increment or decrement of the $C_m(t)$ value. Table D.2 shows the inversion patterns for the 14-bit count and Table D.3 shows the inversion patterns for the 10-bit count. An "I" entry in the table indicates an inversion of that bit.

The bit inversion patterns apply to the $C_m(t-1)$ value, prior to the increment or decrement operation that is signalled by the inversion pattern when $|C_m(t) - C_m(t-1)| \leq 2$ (except $C_m(t) - C_m(t-1) = 0$). The incremented or decremented $C_m(t)$ value becomes the base value for the next GMP overhead transmission.

- When $0 < C_m(t) - C_m(t-1) \leq 2$, indicating an increment of +1 or +2, a subset of the Ci bits containing $C_m(t-1)$ is inverted as specified in Table D.2 or Table D.3 and the increment indicator (II) bit is set to 1.
- When $0 > C_m(t) - C_m(t-1) \geq -2$, indicating a decrement of -1 or -2, a subset of Ci bits containing $C_m(t-1)$ is inverted as specified in Table D.2 or Table D.3 and the decrement indicator (DI) bit is set to 1.
- When the value of $C_m(t)$ is changed with a value larger than +2 or -2 from the value of $C_m(t-1)$, both the II and DI bits are set to 1 and the Ci bits contain the new $C_m(t)$ value. The associated CRC in JC3 verifies whether the $C_m(t)$ value has been received correctly.
- When the value of $C_m(t)$ is unchanged from the value of $C_m(t-1)$, both the II and DI bits are set to 0.

The above encoding process is illustrated in Figure D.6.

Table D.2 – 14-bit $C_m(t)$ increment and decrement indicator patterns

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	II	DI	Change
U	U	U	U	U	U	U	U	U	U	U	U	U	U	0	0	0
I	U	I	U	I	U	I	U	I	U	I	U	I	U	1	0	+1
U	I	U	I	U	I	U	I	U	I	U	I	U	I	0	1	–1
U	I	I	U	U	I	I	U	U	I	I	U	U	I	1	0	+2
I	U	U	I	I	U	U	I	I	U	U	I	I	U	0	1	–2
binary value														1	1	More than +2/–2

NOTE
– I indicates inverted Ci bit
– U indicates unchanged Ci bit

The CRC-8 located in JC3 is calculated over the JC1 and JC2 bits. The CRC-8 uses the $g(x) = x^8 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC1 and JC2 octets are taken in network octet order, most significant bit first, to form a 16-bit pattern representing the coefficients of a polynomial $M(x)$ of degree 15.

2) $M(x)$ is multiplied by x^8 and divided (modulo 2) by $G(x)$, producing a remainder $R(x)$ of degree 7 or less.

3) The coefficients of $R(x)$ are considered to be an 8-bit sequence, where x^7 is the most significant bit.

4) This 8-bit sequence is the CRC-8 where the first bit of the CRC-8 to be transmitted is the coefficient of x^7 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the $M(x)$ polynomial of step 1 includes the CRC bits of JC3, resulting in $M(x)$ having degree 23. In the absence of bit errors, the remainder shall be 0000 0000.

Table D.3 – 10-bit $C_m(t)$ increment and decrement indicator patterns

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	II	DI	Change
U	U	U	U	U	U	U	U	U	U	0	0	0
I	U	I	U	I	U	I	U	I	U	1	0	+1
I	U	U	I	U	I	I	U	U	I	0	1	–1
U	I	U	I	I	U	U	I	U	I	1	0	+2
U	I	I	U	U	I	U	I	I	U	0	1	–2
binary value										1	1	More than +2/–2

NOTE
– I indicates inverted Ci bit
– U indicates unchanged Ci bit

~~The CRC-6 located in JC3 is calculated over bits 3-8 of JC1 and JC2 (i.e., the bits containing the GMP overhead value shown in Table D.3). The CRC-6 uses the $g(x) = x^6 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:~~

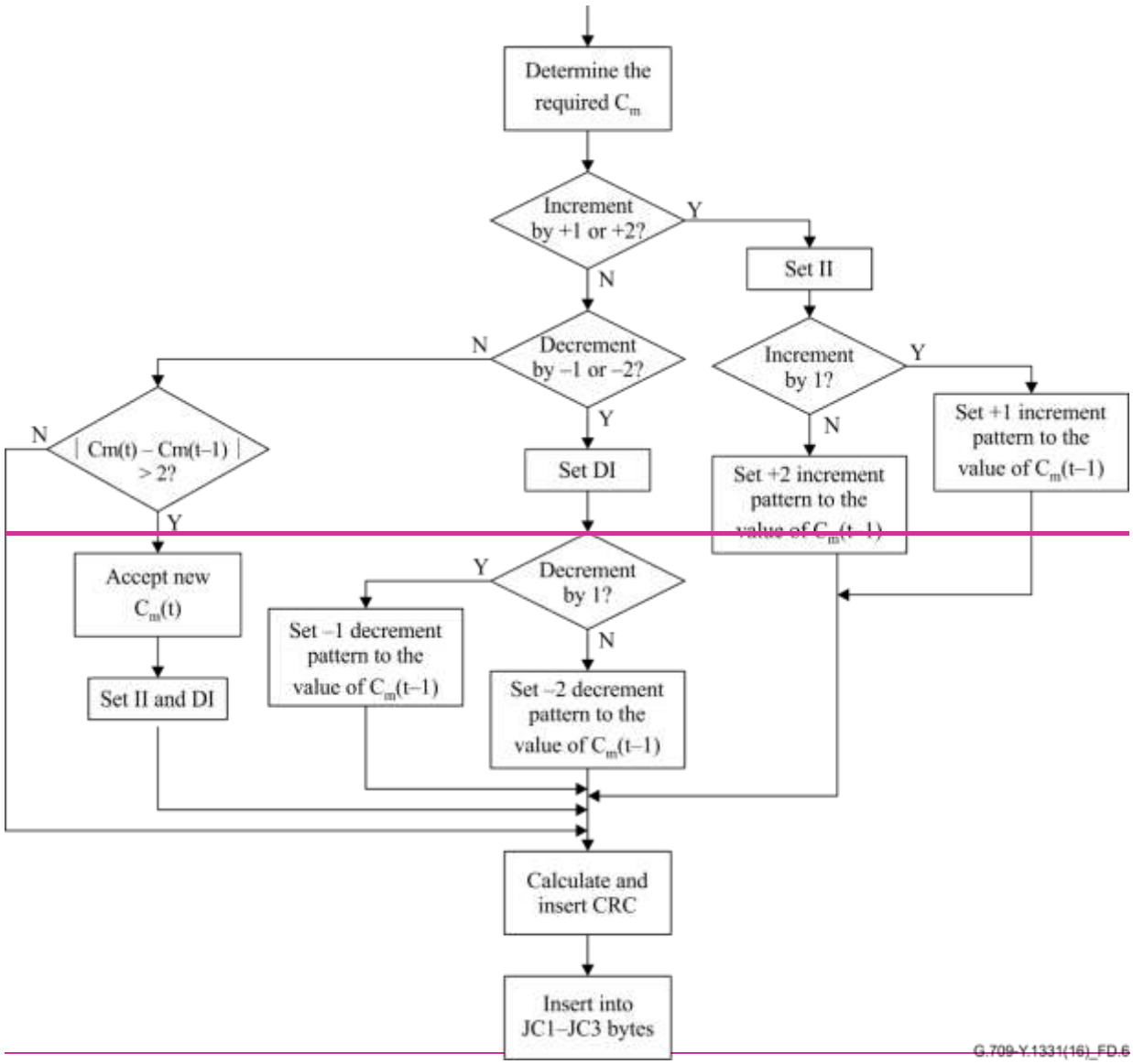
~~1) The JC1 and JC2 octets are taken in network octet order, most significant bit first, such that bit 3 of JC1 through bit 8 of JC2 form a 12-bit pattern representing the coefficients of a polynomial $M(x)$ of degree 11.~~

~~2) $M(x)$ is multiplied by x^6 and divided (modulo 2) by $G(x)$, producing a remainder $R(x)$ of degree 5 or less.~~

~~3) The coefficients of $R(x)$ are considered to be a 6-bit sequence, where x^5 is the most significant bit.~~

~~4) This 6-bit sequence is the CRC-6 where the first bit of the CRC-6 to be transmitted is the coefficient of x^5 and the last bit transmitted is the coefficient of x^0.~~

~~The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the $M(x)$ polynomial of step 1 includes the CRC bits of JC3, resulting in M(x) having degree 17. In the absence of bit errors, the remainder shall be 000000.~~

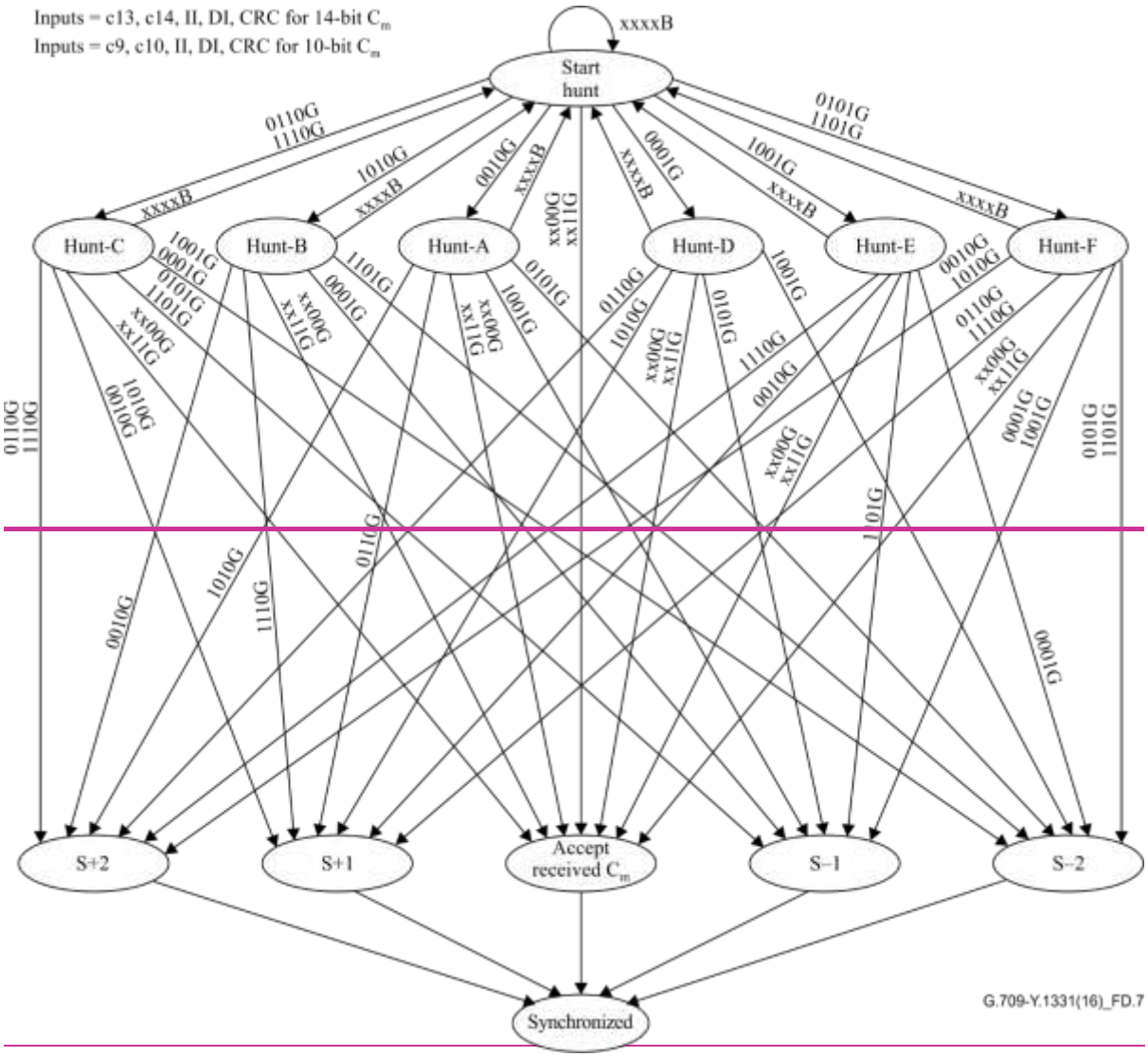


~~Figure D.6 – JC1, JC2 and JC3 generation~~

~~A parallel logic implementation of the source CRC-8 and CRC-6 are illustrated in Appendix VI.~~

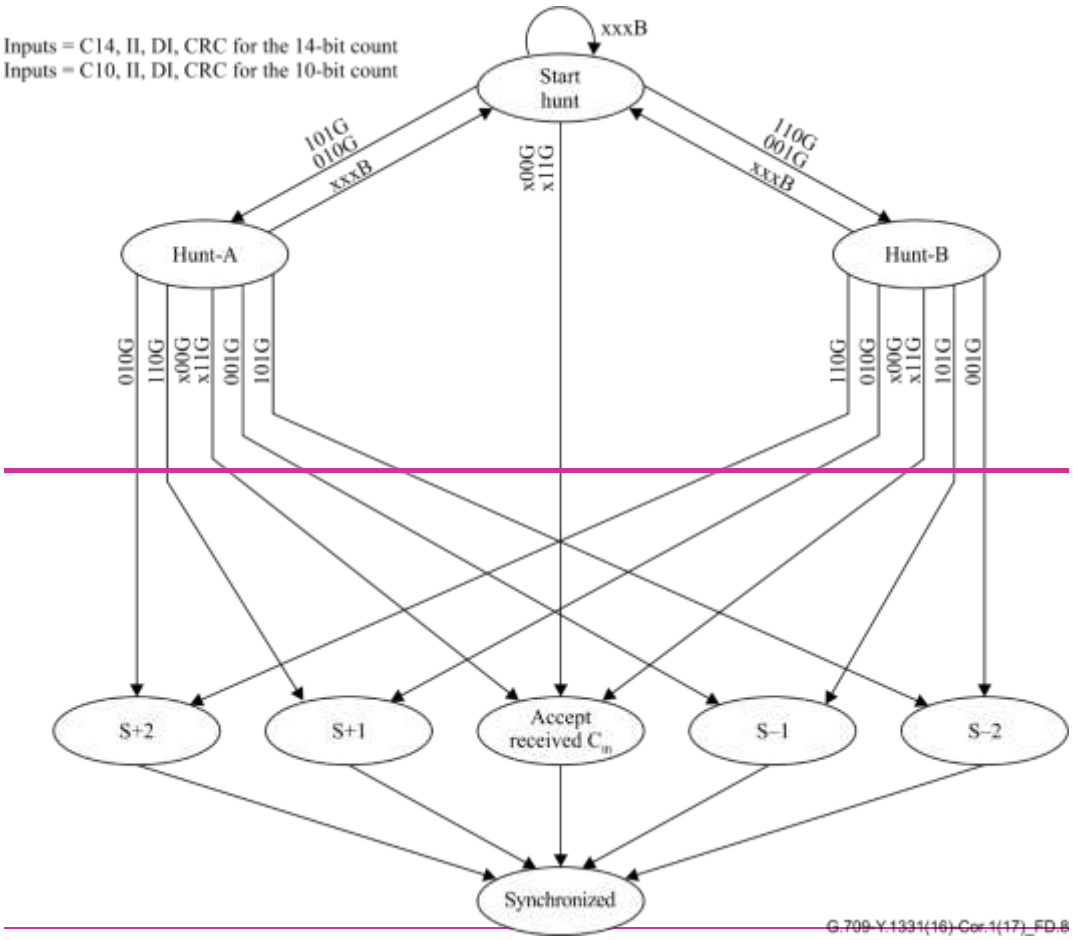
~~The GMP sink synchronizes its $C_m(t)$ value to the GMP source through the following process, which is illustrated in Figure D.7 and the equivalent process in Figure D.8.~~

~~When the received JC octets contain II = DI and a valid CRC-8, the GMP sink accepts the received C1-CLas its $C_m(t)$ value for the next frame. At this point the GMP sink is synchronized to the GMP source. When II ≠ DI with a valid CRC in the current received frame (frame *i*), the GMP sink must examine the received JC octets in the next frame (frame *i*+1) in order to obtain $C_m(t)$ synchronization. II ≠ DI in frame *i* indicates that the source is performing a count increment or decrement operation that will modify the $C_m(t)$ value it sends in frame *i*+1. Since this modification to the $C_m(t)$ will affect the count LSBs, the GMP sink uses the LSBs, II, and DI in frame *i* to determine its synchronization hunt state when it receives frame *i*+1. Specifically, in Figure D.7, the Hunt state (A-F) is determined using C13, C14, II and DI for the 14-bit count and C9, C10, II and DI for the 10-bit count. Equivalently, in Figure D.8, the Hunt state (A or B) is determined using C14, II, and DI for the 14-bit count and C10, II, and DI for the 10-bit count. If II = DI with a valid CRC in frame *i*+1, $C_m(t)$ synchronization is achieved by directly accepting the received C1-CL as the new $C_m(t)$. If II ≠ DI with a valid CRC in frame *i*+1, the sink uses the new LSBs, II and DI values to determine whether the source is communicating an increment or decrement operation and the magnitude of the increment/decrement step. This corresponds to the transition to the lower row of states (the "S" states) in Figure D.7 and Figure D.8. At this point, the GMP sink has identified the type of increment or decrement operation that is being signalled in frame *i*+1. As shown in Figure D.9, the sink then applies the appropriate bit inversion pattern from Table D.2 to the received C1-C14 field or the bit inversion pattern from Table D.3 to the received C1-C10 field to determine the transmitted $C_m(t)$ value. Synchronization has now been achieved since the GMP sink has determined the current $C_m(t)$ and knows the expected $C_m(t)$ change in frame *i*+2.~~



~~Figure D.7 – GMP sink count synchronization process representation using the two least significant bits of the count as inputs~~

~~Optimization has shown that Figure D.8 provides the same function. Either one can be chosen. For historical reasons both are kept.~~



~~Figure D.8 – GMP sink count synchronization process diagram~~

~~14-bit count~~	
~~S-state~~	~~Interpretation/Action~~
~~S+2~~	~~Count = C1-C14 after inverting C2, C3, C6, C7, C10, C11, & C14; Increment +2 for the next frame~~
~~S+1~~	~~Count = C1-C14 after inverting C1, C3, C5, C7, C9, C11, & C13; Increment +1 for the next frame~~
~~S-1~~	~~Count = C1-C14 after inverting C2, C4, C6, C8, C10, C12, & C14; Decrement -1 for next frame~~
~~S-2~~	~~Count = C1-C14 after inverting C1, C4, C5, C8, C9, C12, C13; Decrement -2 for next frame~~

~~10-bit count~~	
~~S-state~~	~~Interpretation/Action~~
~~S+2~~	~~Count = C1-C10 after inverting C2, C4, C5, C8 & C10; Increment +2 for the next frame~~
~~S+1~~	~~Count = C1-C10 after inverting C1, C3, C5, C7 & C9; Increment +1 for the next frame~~
~~S-1~~	~~Count = C1-C10 after inverting C1, C4, C6, C7 & C10; Decrement -1 for next frame~~
~~S-2~~	~~Count = C1-C10 after inverting C2, C3, C6, C8 & C9; Decrement -2 for next frame~~

~~Figure D.9 – "S"-state interpretation for Figures D.7 and D.8~~

~~Note that the state machine of Figure D.7 or Figure D.8 can also be used for off-line synchronization checking.~~

When the GMP sink has synchronized its $C_m(t)$ value to the GMP source, it interprets the received JC octets according to the following principles:

- When the CRC is good and II = DI, the GMP sink accepts the received $C_m(t)$ value.
- When the CRC is good and II ≠ DI, the GMP sink compares the received $C_m(t)$ value to its expected $C_m(t)$ value to determine the difference between these values. This difference is compared to the bit inversion patterns of Table D.2 or Table D.3 to determine the increment or decrement operation sent by the source and updates its $C_m(t)$ accordingly. Since the CRC is good, the sink can use either JC1 or JC2 for this comparison.
- When the CRC is bad, the GMP sink compares the received $C_m(t)$ value to its expected $C_m(t)$ value. The sink then compares the difference between these values, per Table D.2 or Table D.3, to the valid bit inversion patterns in JC1, and the bit inversion, II and DI pattern in JC2.
  - If JC1 contains a valid pattern and JC2 does not, the sink accepts the corresponding increment or decrement indication from JC1 and updates its $C_m(t)$ accordingly.
  - If JC2 contains a valid pattern and JC1 does not, the sink accepts the corresponding increment or decrement indication from JC2 and updates its $C_m(t)$ accordingly.
  - If both JC1 and JC2 contain valid patterns indicating the same increment or decrement operation, this indication is accepted and the sink updates its $C_m(t)$ accordingly.
  - If neither JC1 nor JC2 contain valid patterns, the sink shall keep its current count value and begin the search for synchronization.

  NOTE – If JC1 and JC2 each contain valid patterns that are different from each other, the receiver can either keep the current $C_m(t)$ value and begin a synchronization search, or it can use the CRC to determine whether JC1 or JC2 contains the correct pattern.

The GMP sink uses the updated $C_m(t)$ value to extract the client data from the next OPU frame or ODTUk.ts multiframe.

D.4 $\Sigma C_{nD}(t)$ encoding and decoding

D.4.1 $\Sigma C_{nD}(t)$ encoding and decoding for OPUk

The cumulative value of $C_{nD}(t)$ ($\Sigma C_{nD}(t)$) is encoded in bits 4-8 of the OPUk and ODTUk.ts justification control bytes JC4, JC5 and JC6. Bits D1 to D10 in JC4 and JC5 carry the value of $\Sigma C_{nD}(t)$. Bit D1 carries the most significant bit and bit D10 carries the least significant bit.

The CRC-5 located in JC6 is calculated over the D1-D10 bits in JC4 and JC5. The CRC-5 uses the $g(x) = x^5 + x + 1$ generator polynomial, and is calculated as follows:

1) The JC4 bits 4-8 and JC5 bits 4-8 octets are taken in network transmission order, most significant bit first, to form a 10-bit pattern representing the coefficients of a polynomial $M(x)$ of degree 9.
2) $M(x)$ is multiplied by x^5 and divided (modulo 2) by $G(x)$, producing a remainder $R(x)$ of degree 4 or less.
3) The coefficients of $R(x)$ are considered to be a 5-bit sequence, where x^4 is the most significant bit.
4) This 5-bit sequence is the CRC-5 where the first bit of the CRC-5 to be transmitted is the coefficient of x^4 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the $M(x)$ polynomial of step 1 includes the CRC bits of JC6, resulting in $M(x)$ having degree 14. In the absence of bit errors, the remainder shall be 00000.

A parallel logic implementation of the source CRC-5 is illustrated in Appendix VI.

D.4.2 $\Sigma C_{nD}(t)$ encoding and decoding for OPU*Cn*

The cumulative value of $C_{nD}(t)$ ($\Sigma C_{nD}(t)$) is encoded in the ODTUCn.ts justification control bytes JC1, JC2, JC3, JC4, JC5 and JC6. Bits D1 to D18 carry the value of $\Sigma C_{nD}(t)$. Bit D1 carries the most significant bit and bit D18 carries the least significant bit. As shown in Figure 20-7, bits D1 to D7 are located in bits 2-8 of JC4, bits D8 to D9 are located in bits 1-2 of JC1, bits D10 to D16 are located in JC5, and bits D17 to D18 are located in bits 1-2 of JC2.

The CRC-9 located in bits 2-8 of JC6 and bits 1-2 of JC3 is calculated over the D1-D18 bits in JC4, JC1, JC5 and JC2. The CRC-9 uses the $g(x) = x^9 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC4 bits 2-8, JC1 bits 1-2, JC5 bits 2-8, and JC2 bits 1-2 are taken in network transmission order, most significant bit first, to form an 18-bit pattern representing the coefficients of a polynomial $M(x)$ of degree 17.
2) $M(x)$ is multiplied by x^9 and divided (modulo 2) by $G(x)$, producing a remainder $R(x)$ of degree 8 or less.
3) The coefficients of $R(x)$ are considered to be a 9-bit sequence, where x^8 is the most significant bit.
4) This 9-bit sequence is the CRC-9 where the first bit of the CRC-9 to be transmitted is the coefficient of x^8 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the $M(x)$ polynomial of step 1 includes the CRC bits of JC6 and JC3, resulting in $M(x)$ having degree 26. In the absence of bit errors, the remainder shall be 000000000.

A parallel logic implementation of the source CRC-9 is illustrated in Appendix VI.

Annex E

Adaptation of parallel 64B/66B encoded clients

(This annex forms an integral part of this Recommendation.)

E.1 Introduction

IEEE 40GBASE-R and 100GBASE-R interfaces specified in [IEEE 802.3] are parallel interfaces intended for short-reach (up to 40 km) interconnection of Ethernet equipment. This annex describes the process of converting the parallel format of these interfaces into a serial bit stream to be carried over the OTN.

The order of transmission of information in all the diagrams in this annex is first from left to right and then from top to bottom.

E.2 Clients signal format

40GBASE-R and 100GBASE-R clients are initially parallel interfaces, but in the future they may be serial interfaces. Independent of whether these interfaces are parallel or serial, or what the parallel interface lane count is, 40GBASE-R signals are comprised of four PCS lanes, and 100GBASE-R signals are comprised of twenty PCS lanes. If the number of physical lanes on the interface is fewer than the number of PCS lanes, the appropriate number of PCS lanes is bit-multiplexed onto each physical lane of the interface. Each PCS lane consists of 64B/66B encoded data with a PCS lane alignment marker inserted on each lane once per 16384 66-bit blocks. The PCS lane alignment marker itself is a special format 66B codeword.

The use of this adaptation for 40GBASE-R into OPU3 also applies the transcoding method that appears in Annex B and the framing method of Annex F. The adaptation described in this annex alone can be used for the adaptation of 100GBASE-R into OPU4.

E.3 Client frame recovery

Client framing recovery consists of the following:

– bit-disinterleave the PCS lanes, if necessary. This is necessary whenever the number of PCS lanes and the number of physical lanes is not equal, and is not necessary when they are equal (e.g., a 4-lane 40GBASE-R interface);

– recover 64B/66B block lock as per the state diagram in Figure 82-10 of [IEEE 802.3];

– recover lane alignment marker framing on each PCS lane as per the state diagram in Figure 82-11 of [IEEE 802.3];

– reorder and de-skew the PCS lanes into a serialized stream of 66B blocks (including lane alignment markers). Figure E.1 illustrates the ordering of 66B blocks after the completion of this process for an interface with p PCS lanes.

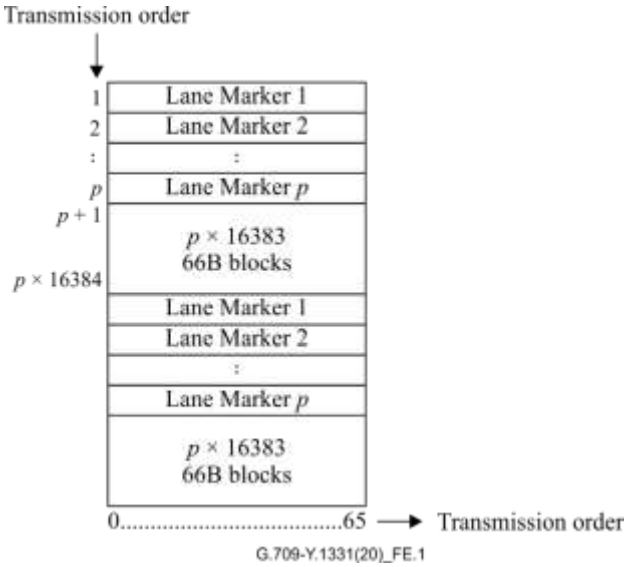


Figure E.1 – De-skewed/serialized stream of 66B blocks

Each 66B codeword is one of the following:

– a set of eight data bytes with a sync header of "01";

– a control block (possibly including seven or fewer data octets) beginning with a sync header of "10";

– a PCS lane alignment marker, also encoded with a sync header of "10". Of the 8 octets following the sync header, 6 octets have fixed values allowing the PCS lane alignment markers to be recognized (see Tables E.1 and E.2). The fourth octet following the sync header is a BIP-8 calculated over the data from one alignment marker to the next as defined in Table 82-4 of [IEEE 802.3]. The eighth octet is the complement of this BIP-8 value to maintain DC balance. Note that the intended operation is to pass these BIP-8 values transparently as they are used for monitoring the error ratio of the Ethernet link between Ethernet PCS sublayers. For the case of 100GBASE-R, the BIP-8 values are not manipulated by the mapping or de-mapping procedure. For the case of 40GBASE-R, a BIP-8 compensation is done as described in clause E.4.1.

For all-data blocks and control blocks, the 64 bits following the sync header are scrambled as a continuous bit-stream (skipping sync headers and PCS lane alignment markers) according to the polynomial $G(x) = 1 + x^{39} + x^{58}$.

After 64B/66B block lock recovery as per the state diagram in Figure 82-10 of [IEEE 802.3] to the single-lane received aggregate signal, these 66B blocks are re-distributed to PCS lanes at the egress interface. The 66B blocks (including PCS lane alignment markers) resulting from the decoding process are distributed round-robin to PCS lanes. If the number of PCS lanes is greater than the number of physical lanes of the egress interface, the appropriate numbers of PCS lanes are bit-multiplexed onto the physical lanes of the egress interface.

E.3.1 40GBASE-R client frame recovery

PCS lane alignment markers have the values shown in Table E.1 for 40GBASE-R signals which use PCS lane numbers 0-3.

Table E.1 – PCS lane alignment marker format for 40GBASE-R

Lane Number	SH	Encoding {M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7}
0	10	0x90, 0x76, 0x47, BIP_3, 0x6f, 0x89, 0xb8, BIP_7
1	10	0xf0, 0xc4, 0xe6, BIP_3, 0x0f, 0x3b, 0x19, BIP_7
2	10	0xc5, 0x65, 0x9b, BIP_3, 0x3a, 0x9a, 0x64, BIP_7
3	10	0xa2, 0x79, 0x3d, BIP_3, 0x5d, 0x86, 0xc2, BIP_7

Since a 40GBASE-R client signal must be transcoded into 1024B/1027B for rate reduction, the 64B/66B PCS receive process at the ingress interface further descrambles the bit-stream skipping sync headers and PCS lane alignment markers, and the 64B/66B PCS transmit process at the egress interface scrambles the bit-stream again skipping sync headers and PCS lane alignment markers, as shown in Figure E.1.

E.3.2 100GBASE-R client frame recovery

PCS lane alignment markers have the values shown in Table E.2 for 100GBASE-R signals which use PCS lane numbers 0-19.

The lane alignment markers transported over the OPU4 are distributed unchanged to the PCS lanes.

Table E.2 – PCS lane alignment marker format for 100GBASE-R

Lane Number	SH	Encoding {M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7}	Lane Number	SH	Encoding {M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7}
0	10	0xc1, 0x68, 0x21, BIP_3, 0x3e, 0x97, 0xde, BIP_7	10	10	0xfd, 0x6c, 0x99, BIP_3, 0x02, 0x93, 0x66, BIP_7
1	10	0x9d, 0x71, 0x8e, BIP_3, 0x62, 0x8e, 0x71, BIP_7	11	10	0xb9, 0x91, 0x55, BIP_3, 0x46, 0x6e, 0xaa, BIP_7
2	10	0x59, 0x4b, 0xe8, BIP_3, 0xa6, 0xb4, 0x17, BIP_7	12	10	0x5c, 0xb9, 0xb2, BIP_3, 0xa3, 0x46, 0x4d, BIP_7
3	10	0x4d, 0x95, 0x7b, BIP_3, 0xb2, 0x6a, 0x84, BIP_7	13	10	0x1a, 0xf8, 0xbd, BIP_3, 0xe5, 0x07, 0x42, BIP_7
4	10	0xf5, 0x 07, 0x09, BIP_3, 0x0a, 0xf8, 0xf6, BIP_7	14	10	0x83, 0xc7, 0xca, BIP_3, 0x7c, 0x38, 0x35, BIP_7
5	10	0xdd, 0x14, 0xc2, BIP_3, 0x22, 0xeb, 0x3d, BIP_7	15	10	0x35, 0x36, 0xcd, BIP_3, 0xca, 0xc9, 0x32, BIP_7
6	10	0x9a, 0x4a, 0x26, BIP_3, 0x65, 0xb5, 0xd9, BIP_7	16	10	0xc4, 0x31, 0x4c, BIP_3, 0x3b, 0xce, 0xb3, BIP_7
7	10	0x7b, 0x45, 0x66, BIP_3, 0x84, 0xba, 0x99, BIP_7	17	10	0xad, 0xd6, 0xb7, BIP_3, 0x52, 0x29, 0x48, BIP_7
8	10	0xa0, 0x24, 0x76, BIP_3, 0x5f, 0xdb, 0x89, BIP_7	18	10	0x5f, 0x66, 0x2a, BIP_3, 0xa0, 0x99, 0xd5, BIP_7
9	10	0x68, 0xc9, 0xfb, BIP_3, 0x97, 0x36, 0x04, BIP_7	19	10	0xc0, 0xf0, 0xe5, BIP_3, 0x3f, 0x0f, 0x1a, BIP_7

E.4 Additions to Annex B transcoding for parallel 40GBASE-R clients

Mapping 40GBASE-R clients into OPU3 requires the transcoding into 513B code blocks described in Annex B. This clause describes the additions to the Annex B transcoding process for transport of PCS lane alignment markers.

Ethernet path monitoring is the kind of behaviour that is desirable in the case where the Ethernet equipment and the OTN equipment are in different domains (e.g., customer and service provider) and from the standpoint of the Ethernet equipment. It is also the default behaviour which would result from the current mapping of 100GBASE-R where the 66B blocks would be mapped into the OPU4 container after management of skew. It may also be perceived as a transparency requirement that BIP-8 work end-to-end. Additional functionality as described below has to be built in to allow BIP-8 transparency for 40GBASE-R client signals.

PCS lane alignment markers are encoded together with 66B control blocks into the uppermost rows of the 513B code block shown in Figure B.3. The flag bit "F" of the 513B structure is 1 if the 513B structure contains at least one 66B control block or PCS lane alignment marker, and 0 if the 513B structure contains eight all-data 66B blocks.

The transcoding into 512B/513B must encode PCS lane alignment marker into a row of the structure shown in Figure B.3 as follows: The sync header of "10" is removed. The received M_0, M_1 and M_2 bytes of the PCS alignment marker encodings as shown in Table E.1 are used to forward the lane number information. The first byte of the row will contain the structure shown in Figure B.4, with a CB-TYPE field of "0100". The POS field will indicate the position where the PCS lane alignment marker was received among the group of eight 66B codewords being encoded into this 513B block. The flag continuation bit "FC" will indicate whether any other 66B control blocks or PCS lane alignment markers are encoded into rows below this one in the 513B block. Beyond this first byte, the next four bytes of the row are populated with the received M_0, M_1, M_2 and ingress BIP_3 bytes of the PCS alignment marker encodings at the encoder. At the decoder, a PCS lane alignment marker will be generated in the position indicated by the POS field among any 66B all-data blocks contained in this 513B block, the sync header of "10" is generated followed by the received M_0, M_1 and M_2 bytes, the egress BIP_3 byte, the bytes M_4, M_5 and M_6 which are the bit-wise inverted M_0, M_1 and M_2 bytes received at the decoder, and the egress BIP_7 byte which is the bit-wise inverted egress BIP_3 byte.

It will then be up to the Ethernet receiver to handle bit errors within the OTN section that might have altered the PCS alignment marker encodings (for details refer to clause 82.2.18.3 and Figure 82-11 in [IEEE 802.3]).

The egress BIP_3 and the egress BIP_7 bytes are calculated as described in clause E.4.1.

Figure E.2 below shows the transcoded lane marker format.

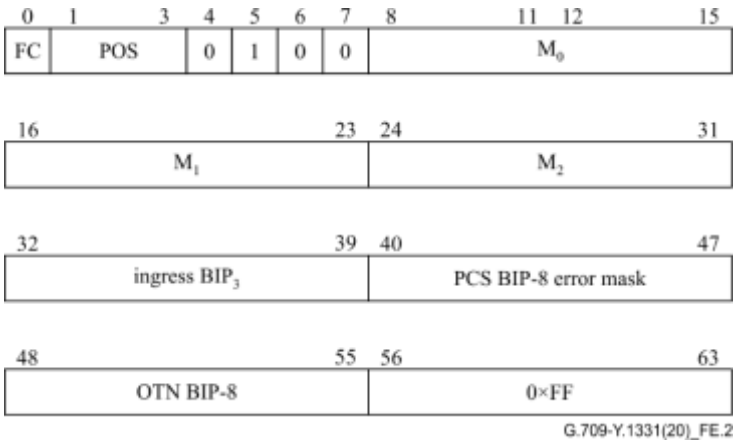


Figure E.2 – Transcoded lane marker format

E.4.1 40GBASE-R BIP-8 transparency

The transcoding method used for 40GBASE-R is timing and PCS codeword transparent. In normal operation, the only aspects of the PCS encoded bitstream that are not preserved given the mapping described in annexes B, E and F are for one the scrambling, since the scrambler does not begin with a known state and multiple different encoded bitstreams can represent the same PCS encoded content, and secondly the BIP-8 value in the Ethernet path or more precisely, the bit errors that occur between the Ethernet transmitter and the ingress point of the OTN domain and within the OTN domain. The BIP-8 values can be preserved with the scheme described below. As the scrambling itself does not contain any information that has to be preserved, no effort has been made to synchronize the scrambler states between OTN ingress and OTN egress.

Unfortunately, since the BIP-8 is calculated on the scrambled bitstream, a simple transport of the BIP-8 across the OTN domain in the transcoded lane marker will not result in a BIP-8 value that is meaningful for detecting errors in the received, descrambled, transcoded, trans-decoded, and then rescrambled bit stream.

To preserve the bit errors between the Ethernet transmitter and the egress side of the OTN domain, the bit-error handling is divided into two processes, one that takes place at the OTN ingress side, or encoder, and one on the OTN egress side, or decoder.

At the OTN ingress an 8-bit error mask is calculated by generating the expected BIP-8 for each PCS lane and XORing this value with the received BIP-8. This error mask will have a "1" for each bit of the BIP-8 which is wrong, and a "0" for each bit which is correct. This value is shown as a PCS BIP-8 error mask in Figure E.2.

In the event no errors are introduced across the OTN (as an FEC protected network can be an essentially zero error environment), the PCS BIP-8 error mask can be used to adjust the newly calculated PCS BIP-8 at the egress providing a reliable indication of the number of errors that are introduced across the full Ethernet path. If errors are introduced across the OTN, this particular BIP-8 calculation algorithm will not see these errors.

To overcome this situation, a new BIP-8 per lane for the OTN section is introduced. In the following this new BIP-8 will be identified as OTN BIP-8 in order to distinguish it from the PCS BIP-8.

It should be noted that the term OTN BIP-8 does not refer to and should not be confused with the BIP-8 defined in the OTUk overhead (byte SM[2]).

The OTN BIP-8 is calculated similarly to the PCS BIP-8 as described in clause 82.2.8 of [IEEE 802.3] with the exception that the calculation will be done over unscrambled PCS lane data, the original received lane alignment marker, after error control block insertion and before transcoding. Figure E.2 shows the byte location of the OTN BIP-8 in the transcoded lane marker.

The transcoded lane marker is transmitted together with the transcoded data blocks over the OTN section as defined in Annex B. At the OTN egress after trans-decoding and before scrambling, the ingress alignment marker is recreated using M_0, M_1, M_2 and ingress BIP_3 of the transcoded alignment marker followed by the bit-wise inversion of these bytes. This recreated alignment marker together with the trans-decoded and unscrambled data blocks is used to calculate the expected OTN BIP-8 for each PCS lane (refer to clause 82.2.8 of [IEEE 802.3]). The expected value will be XORed with the received OTN BIP-8. This error mask will have a "1" for each bit of the OTN BIP-8 which is wrong, and a "0" for each bit which is correct.

The egress BIP_3 for each PCS lane is calculated over the trans-decoded and scrambled data blocks including the trans-decoded alignment marker (refer to clause E.4) following the process depicted in clause 82.2.8 of [IEEE 802.3].

The egress BIP_3 is then adjusted for the errors that occurred up to the OTN egress by first XORing with the PCS BIP-8 error mask and then XORing with the OTN BIP-8 error mask. This combined error mask will be used to compute the number of BIP errors when used for non-intrusive monitoring.

The BIP_7 is created by bit-wise inversion of the adjusted BIP_3.

E.4.2 Errors detected by 40GBASE-R to OPU3 mapper

An invalid 66B block will be converted to an error control block before transcoding. An invalid 66B block is one which does not have a sync header of "01" or "10", or one which has a sync header of "10", is not a valid PCS lane alignment marker and has a control block type field which does not appear in Figure B.2 or has one of the values 0x2d, 0x33, 0x66, or 0x55 which are not used for 40GBASE-R. An error control block has sync bits of "10", a block type code of 0x1e, and 8 seven-bit/E/error control characters. This will prevent the Ethernet receiver from interpreting a sequence of bits containing this error as a valid packet.

E.4.3 Errors detected by 40GBASE-R to OPU3 de-mapper

Several mechanisms will be employed to reduce the probability that the 40GBASE-R to OPU3 de-mapper constructs erroneous parallel 64B/66B encoded data at the egress if bit errors have corrupted. Since detectable corruption normally means that the proper order of 66B blocks to construct at the de-mapper cannot be reliably determined, if any of these checks fail, the de-mapper will transmit eight 66B error control blocks (sync="10", control block type=0x1e, and eight 7-bit/E/control characters).

Mechanisms for improving the robustness and for 513B block lock including PCS lane alignment markers are discussed in Annex F.

Annex F

Improved robustness for mapping of 40GBASE-R into OPU3 using 1027B code blocks

(This annex forms an integral part of this Recommendation.)

F.1 Introduction

When a parallel 40GBASE-R signal is transcoded as per Annexes B and E and directly mapped into OPU3 without GFP framing, another method is needed to locate the start of 513B blocks and to provide protection to prevent bit errors creating an unacceptable increase in mean time to false packet acceptance (MTTFPA).

F.2 513B code block framing and flag bit protection

The mapping of 513B code blocks into OPU3 requires a mechanism for locating the start of the code blocks. A mechanism is also needed to protect the flag bit, whose corruption could cause data to be erroneously interpreted as control and vice versa.

Both of these requirements can be addressed by providing parity across the flag bits of two 513B blocks produced from the transcoding of Annex B.

Figure F.1 illustrates the flag parity bit across two 513B blocks. This creates a minimum two-bit Hamming distance between valid combinations of flag bits.

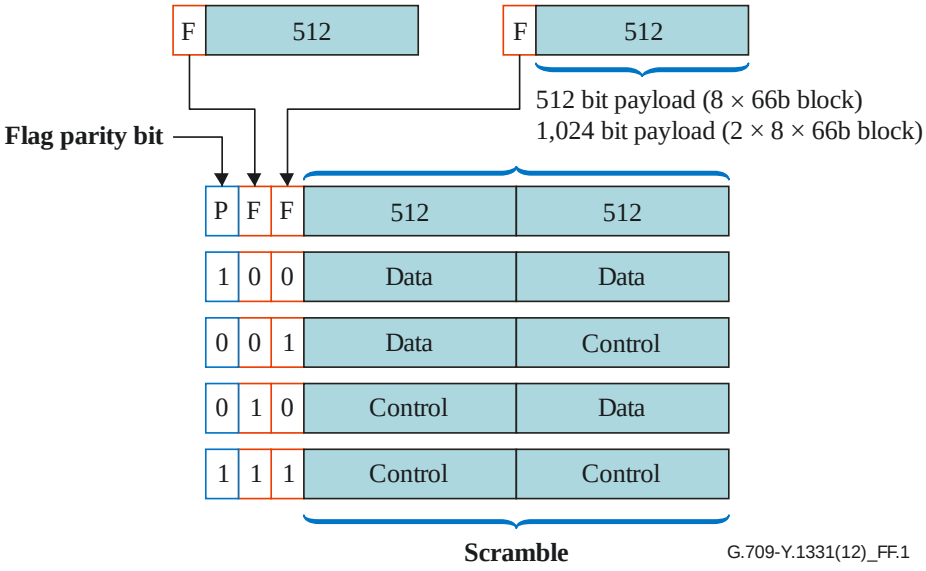


Figure F.1 – Flag parity bit on two 513B blocks (1027B code)

The flag parity bit creates a sequence that can be used for framing to locate the 513B blocks in a stream of bits. The state diagram of Figure 49-12 of [IEEE 802.3] is applied to locate a 3-bit pattern appearing once per 1027 bits (rather than a 2-bit pattern appearing once per 66 bits) where four out of eight 3-bit sequences (rather than two out of four two-bit values as used in IEEE 802.3) match the pattern. The additional step required is to scramble the non-flag bits so that the legal sequences of these bits are not systematically mimicked in the data itself. The scrambler to be used for this purpose is the Ethernet self-synchronous scrambler using the polynomial $G(x) = 1 + x^{39} + x^{58}$.

At the de-mapper, invalid flag bit parity will cause both of the 513B blocks across which the flag bit parity applies to be decoded as 8 × 2 66B error control blocks ("10" sync header, control block type 0x1e, followed by eight 7-bit/E/control characters).

F.3 66B block sequence check

Bit error corruption of the position or flag continuation bits could cause 66B blocks to be de-mapped from 513B code blocks in the incorrect order. Additional checks are performed to prevent that this results in incorrect packet delineation. Since detectable corruption normally means that the proper order of 66B blocks to construct at the decoder cannot be reliably determined, if any of these checks fail, the decoder will transmit eight 66B error control blocks (sync="10", control block type=0x1e, and eight 7-bit/E/control characters).

Other checks are performed to reduce the probability that invalid data is delivered at the egress in the event that bit errors have corrupted any of the POS fields or flag continuation bits "FC".

If flag bit "F" is 1 (i.e., the 513B block includes at least one 64B/66B control block), for the rows of the table up until the first one with a flag continuation bit of zero (the last one in the block), it is verified that no two 66B control blocks or lane alignment markers within that 513B block have the same value in the POS field, and further, that the POS field values for multiple control or lane alignment rows are in ascending order, which will always be the case for a properly constructed 513B block. If this check fails, the 513B block is decoded into eight 66B error control blocks.

The next check is to ensure that the block sequence corresponds to well-formed packets, which can be done according to the state diagram in Figures F.2 and F.3. This check will determine if 66B blocks are in an order that does not correspond to well-formed packets, e.g., if during an IPG an all-data 66B block is detected without first seeing a control block representing packet start, or if during a packet a control/idle block is detected without first seeing a control block representing packet termination, control blocks have likely been misordered by corruption of either the POS bits or a flag continuation bit. Failure of this check will cause the 513B block to be decoded as eight 66B error control blocks. Note that PCS lane alignment markers are accepted in either state and do not change state as shown in Figure F.3.

The sequence of PCS lane alignment markers is also checked at the decoder. For an interface with p PCS lanes, the PCS lane alignment markers for lanes 0 to p-1 will appear in a sequence, followed by $16383 \times p$ non-lane-marker 66B blocks, followed by another group of PCS lane alignment markers. A counter is maintained at the decoder to keep track of when the next group of lane alignment markers is expected. If, in the process of decoding lane alignment markers from a 513B block, a lane alignment marker is found in a position where it is not expected, or a lane alignment marker is missing in a position where it would have been expected, the entire 513B block is decoded as eight 66B error control blocks as shown in Figures F.2, F.3, and F.4.

F.3.1 State diagram conventions

The body of this clause is comprised of state diagrams, including the associated definitions of variables, constants, and functions. Should there be a discrepancy between a state diagram and descriptive text, the state diagram prevails.

The notation used in the state diagrams follows the conventions of clause 21.5 of [IEEE 802.3]. State diagram timers follow the conventions of clause 14.2.3.2 of [IEEE 802.3]. The notation ++ after a counter or integer variable indicates that its value is to be incremented.

F.3.2 State variables

F.3.2.1 Constants

EBLOCK_T<65:0>

66-bit vector to be sent to the PCS containing /E/ in all the eight character locations.

Mi<65:0>

66-bit vector containing the trans-decoded alignment marker of i-th PCS lane ($0 < i \leq p$). (p=4 for 40GBASE-R).

F.3.2.2 Variables

1027B_block_lock

Indicates the state of the block_lock variable when the state diagram of Figure 49-12 of [IEEE 802.3] is applied to locate a 3-bit pattern appearing once per 1027 bits (rather than a 2-bit pattern appearing once per 66 bits) as described in clause F.2. Set true when sixty-four contiguous 1027-bit blocks are received with valid 3-bit patterns, set false when sixteen 1027-bit blocks with invalid 3-bit patterns are received before sixty-four valid blocks.

1027B_high_ber

Indicates a Boolean variable when the state diagram of Figure 49-13 of [IEEE 802.3] is applied to count invalid 3-bit sync headers of 1027-bit blocks (rather than 2-bit sync headers of 66-bit blocks) within the current 250 μs (rather than 125 μs). Set true when the ber_cnt exceeds 8 (rather than 16) indicating a bit error ratio $>10^{-4}$.

Mseq_violation

Indicates a Boolean variable that is set and latched in each rx513_raw<527:0> PCS lane alignment marker cycle based on the PCS lane marker position and order. It is true if the unexpected marker sequence is detected and false if not.

POS_violation

A Boolean variable that is set in each rx513_raw<527:0> based on the POS field values for rx_tcd<65:0>. It is true if the two or more have the same POS values or if they are not in ascending order, and false if their POS values are in ascending order.

reset

A Boolean variable that controls the resetting of the PCS. It is true whenever a reset is necessary including when reset is initiated from the MDIO, during power on, and when the MDIO has put the PCS into low-power mode.

Rx513_coded<512:0>

A vector containing the input to the 512B/513B decoder.

rx513_raw<527:0>

A vector containing eight successive 66-bit vectors (tx_coded).

rx_tcd<65:0>

A 66-bit vector transcode-decoded from a 513-bit block following the rules shown in Figure B.5.

seq_violation

A Boolean variable that is set in each rx513_raw<527:0> based on the sequence check on an rx_tcd<65:0> stream. It is true if the unexpected sequence is detected and false if not.

F.3.2.3 Functions

DECODE(rx513_coded<512:0>)

Decodes the 513-bit vector returning rx513_raw<527 :0> which is sent to the client interface. The DECODE function shall decode the block as specified in Figure F.2.

R_BLOCK_TYPE = {C, S, T, D, E, M}

This function classifies each 66-bit rx_tcd vector as belonging to one of the six types depending on its contents.

Values: C, S, T, and D are defined in clause 49.2.13.2.3 of [IEEE 802.3].

M: the vector contains a sync header of 10 and is recognized as a valid PCS lane alignment marker by using the state machine shown in Figure F.3.

E: the vector does not meet the criteria for any other value.

R_TYPE(rx_tcd<65:0>)

Returns the R_BLOCK_TYPE of the rx_tcd<65:0> bit vector.

R_TYPE_NEXT

Prescient end of packet check function. It returns the R_BLOCK_TYPE of the rx_tcd vector immediately following the current rx_tcd vector.

F.3.2.4 Counters

cnt

Count up to a maximum of *p* of the number of PCS lanes.

F.3.3 State diagrams

The receive state machine for a series of 513-bit blocks shown in Figure F.2 determines whether the 513-bit block contains valid eight 66-bit blocks or not.

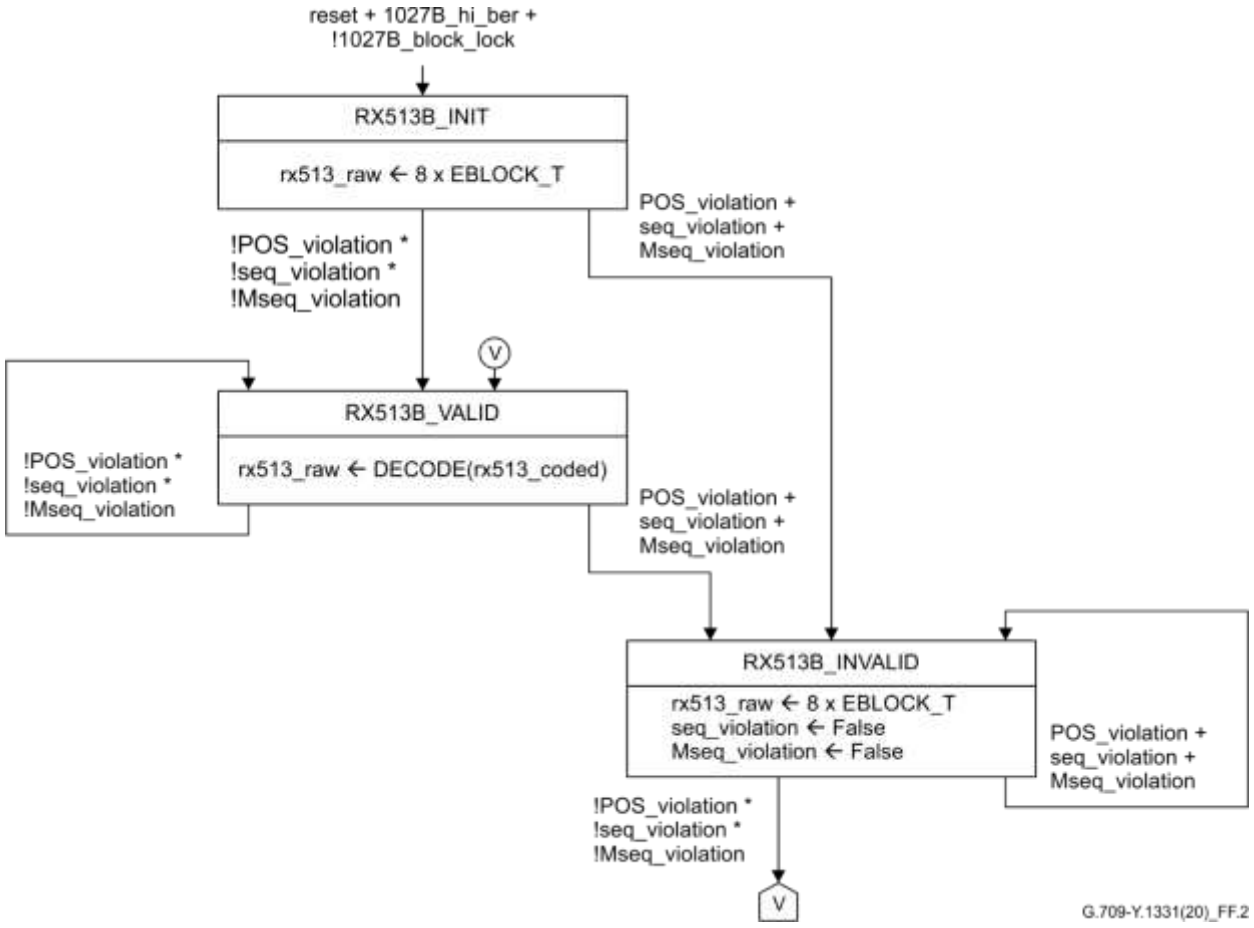


Figure F.2 – Receive state machine for the 512B/513B code blocks including lane alignment markers

The trans-decode state machine for a series of 66-bit blocks shown in Figure F.3 checks the block type sequence of recovered 66-bit blocks.

The PCS lane alignment marker state machine for a series of 66-bit blocks shown in Figure F.4 detects the alignment markers every $p \times 16384$ blocks and checks whether the marker is in ascendant order or not.

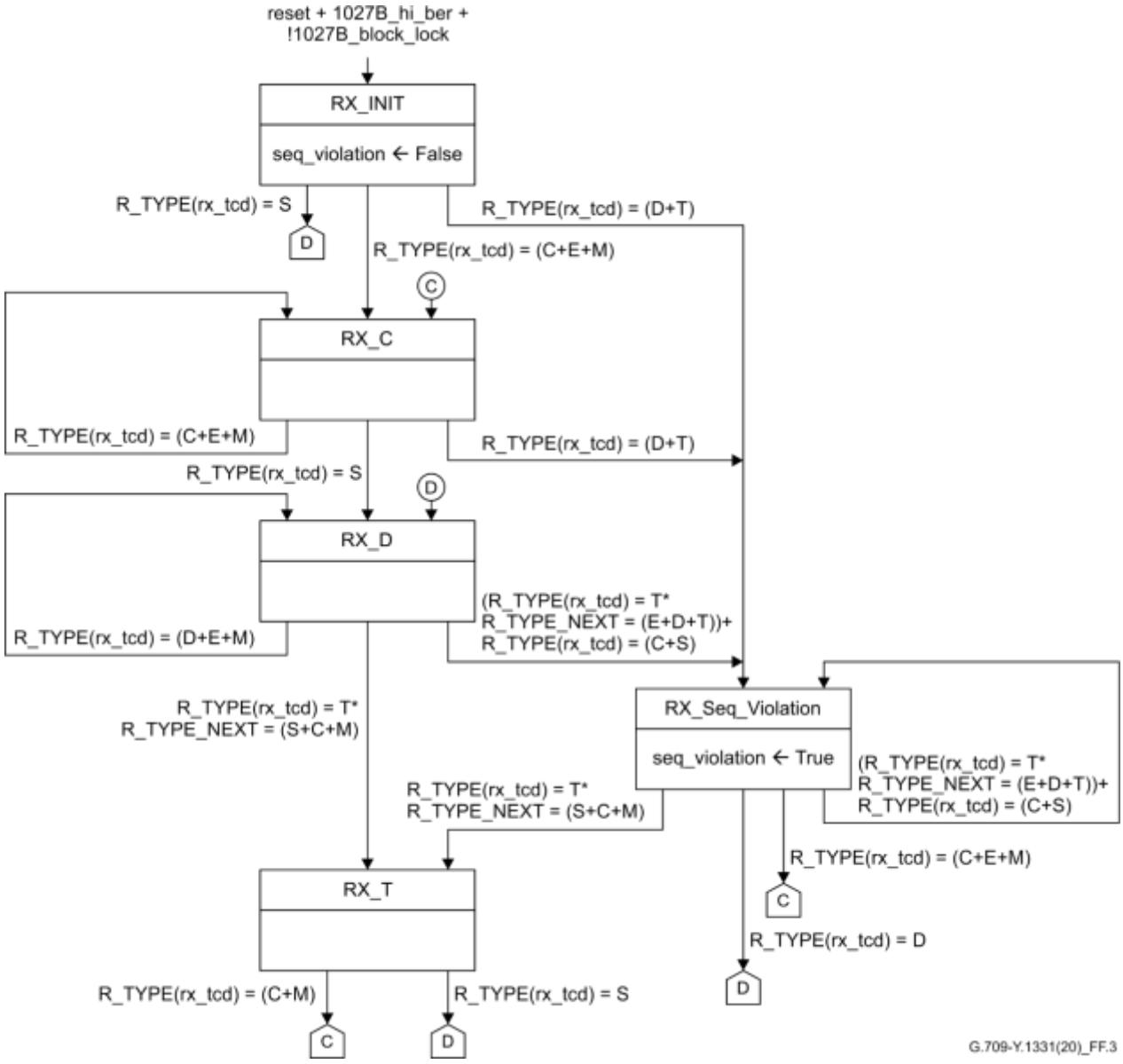


Figure F.3 – Trans-decode state machine for the 64B/66B code blocks including the lane alignment markers

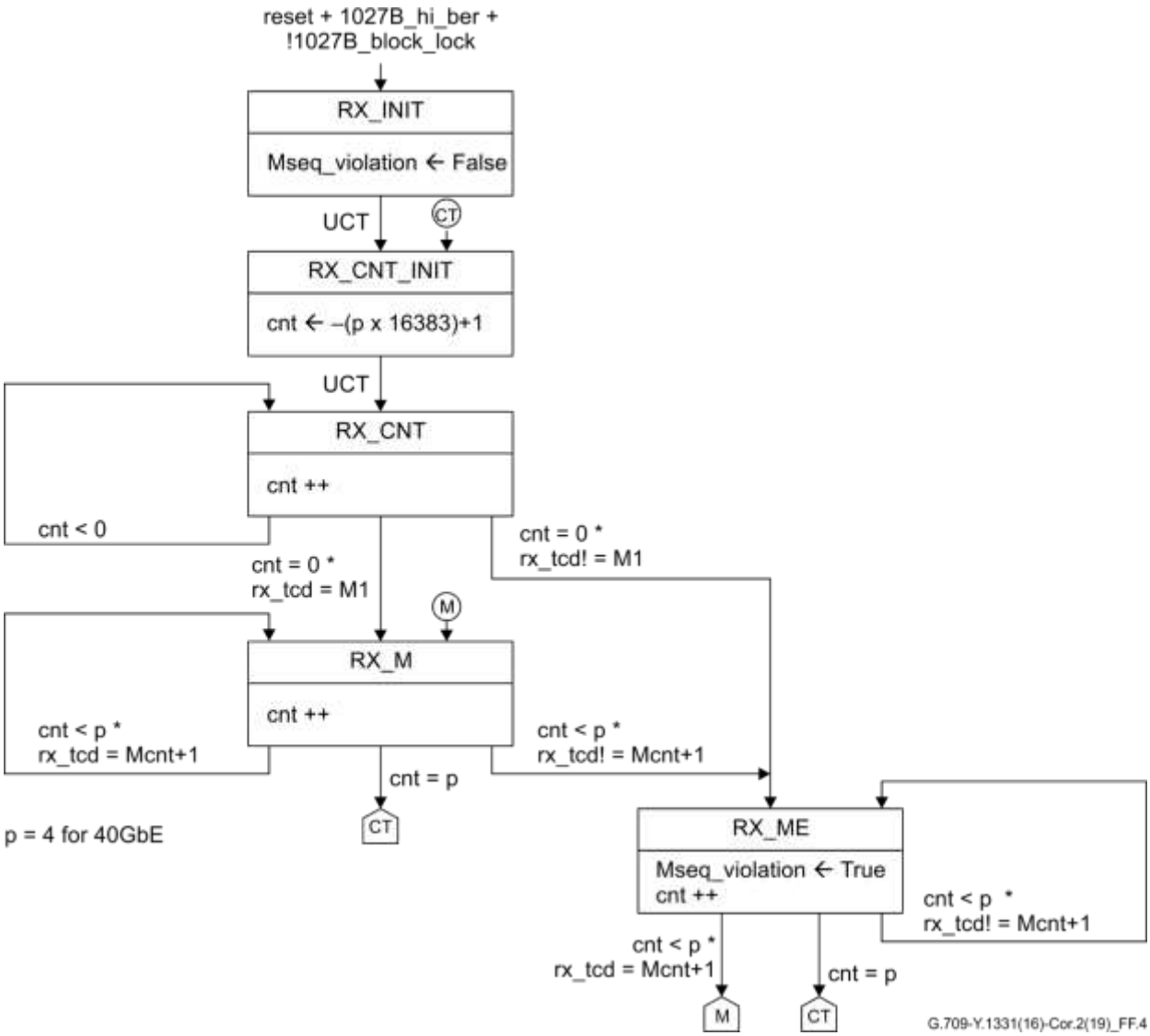


Figure F.4 – Receive state machine for the lane alignment markers

Annex G

Mapping ODU0 into a low latency OTU0 (OTU0LL)

(This annex forms an integral part of this Recommendation.)

G.1 Introduction

Within the optical transport network client signals with a bit rate up to 1.25 Gbit/s are transported within an ODU0 and the ODU0 is transported within a server ODUk and OTUk (k = 1,2,3,4). This annex specifies a low latency 1.25G OTU0 (OTU0LL) frame format in which one ODU0 is transported which carries a client (e.g., 1G Ethernet) signal using an [ITU-T G.698.3] application code. This OTU0LL may be used at the edge of the optical transport network.

G.2 Optical transport unit 0 low latency (OTU0LL)

The OTU0LL conditions the ODU0 for transport over a multi-vendor optical network interface at the edge of the optical transport network. The OTU0LL frame structure, including the OTU0LL FEC is completely standardized. The optical aspects of the multi-vendor optical network interface at the edge of the optical transport network are outside the scope of this Recommendation.

NOTE 1 – Transport of the OTU0LL over the interfaces specified in this Recommendation is not supported.

NOTE 2 – An ODU0 which is transported within an OTU0LL may be passed through the OTN and terminated at the far end edge of the OTN.

G.2.1 OTUk frame structure

The OTU0LL frame structure is based on the ODU0 frame structure and extends it with a distributed forward error correction (FEC) as shown in Figure G.1. Sixteen times 16 columns are added to the ODU0 frame for the FEC and the reserved overhead bytes in row 1, columns 8 to 14 of the ODU0 overhead are used for an OTU0LL specific overhead, resulting in an octet-based block frame structure with four rows and 4080 columns. The MSB in each octet is bit 1, the LSB is bit 8.

The OTU0LL overhead is the same as the OTUk overhead.

The bit rate of the OTU0LL signal is 255/239 × 1 244 160 kbit/s (~1 327 451.046 kbit/s). The frame period of the OTU0LL signal is approximately 98.354 μs.

The sixty-four 16-byte RS(255,239) FEC fields in the OTL0LL frame contain the Reed-Solomon RS(255,239) FEC code. Each RS(255,339) FEC is computed over the previous 239 OTU0LL bytes. Transmission of the OTU0LL FEC is mandatory.

NOTE – The distribution of the RS(255,239) FEC fields over the OTU0LL frame minimizes the transfer delay introduced by the processing of this FEC and the number of codecs to compute this FEC.

The RS(255,239) FEC code shall be computed as specified in Annex A with the notion that each FEC is computed over the previous 239 OTU0LL bytes instead of over a sub-row as described in this annex for the case of an OTUk (k=0,1,2,3,4).

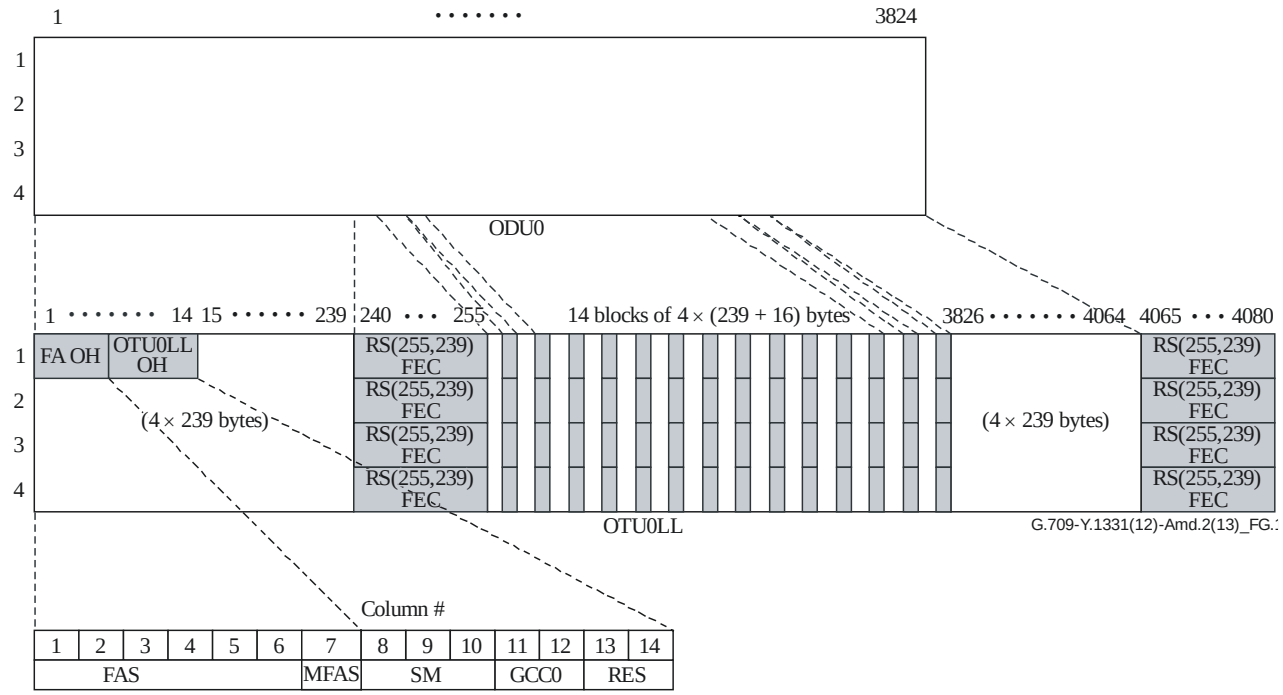


Figure G.1 – OTU0LL frame structure, overhead and ODU0 mapping

The transmission order of the bits in the OTU0LL frame is left to right, top to bottom, and MSB to LSB (see Figure G.2).

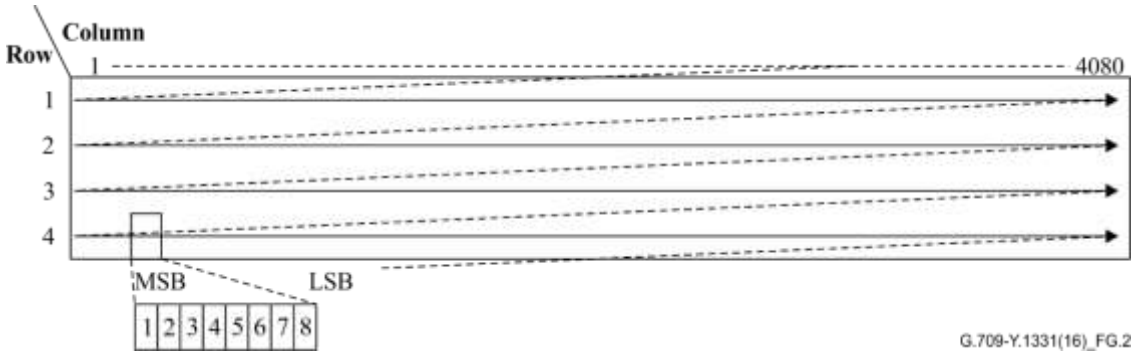


Figure G.2 – Transmission order of the OTU0LL frame bits

G.2.2 Scrambling

The OTU0LL signal must have sufficient bit timing content at the NNI. A suitable bit pattern, which prevents a long sequence of "1"s or "0"s, is provided by using a scrambler.

The operation of the scrambler shall be functionally identical to that of a frame synchronous scrambler of sequence length 65535 operating at the OTU0LL rate.

The generating polynomial shall be $1 + x + x^3 + x^{12} + x^{16}$. Figure G.3 shows a functional diagram of the frame synchronous scrambler.

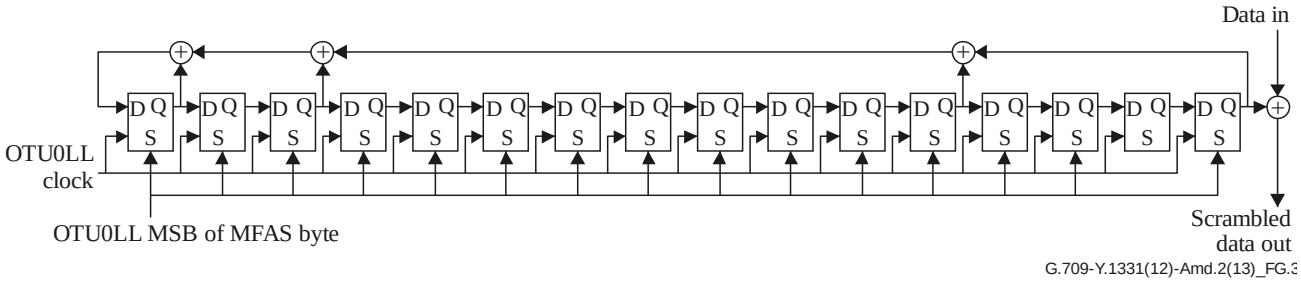


Figure G.3 – Frame synchronous scrambler

The scrambler shall be reset to "FFFF" (HEX) on the most significant bit of the byte following the last framing byte in the OTU0LL frame, i.e., the MSB of the MFAS byte. This bit, and all subsequent bits to be scrambled shall be added modulo 2 to the output from the x^{16} position of the scrambler. The scrambler shall run continuously throughout the complete OTU0LL frame. The framing bytes (FAS) of the OTU0LL overhead shall not be scrambled.

Scrambling is performed after FEC computation and insertion into the OTU0LL signal.

Annex H

OTUCn sub rates (OTUCn-M)

(This annex forms an integral part of this Recommendation.)

H.1 Introduction

This annex defines a subrate OTUCn, note that subrates are not defined for the OPUCn and ODUCn. Additional descriptive text, examples and applications are provided in Appendix XIII.

The OPUCn, ODUCn and OTUCn are defined in terms of n x 100 Gbit/s signals. For the OTN optical networking interface it may be advantageous to carry an OTUCn with a bit rate that is not constrained to be an integer multiple of 100 Gbit/s. This OTU is described as an OTUCn-M.

H.2 OTUCn-M frame format

The OTUCn-M frame is a type of OTUCn frame which contains n instances of OTUC, ODUC and OPUC overhead and M 5 Gbit/s OPUCn tributary slots. If a particular value of M is not indicated, the frame contains 20×n tributary slots. See Appendix XIII for examples.

An OTUCn-M framed signal can be carried over OTN optical networking interfaces. Values of M supported and the subset of the 20n 5 Gbit/s OPUCn tributary slots that are included in the OTUCn-M on such an OTN optical networking interface are vendor specific and outside the scope of this Recommendation.

The OPU tributary slots that are included in the OTUCn-M structure will have their availability bit in the OPUCn MSI overhead set to "available" (see 20.4.1.1). The OPU tributary slots which might not be carried transparently by the OTUCn-M structure will have their availability bit in the OPUCn MSI overhead set to "unavailable" (see 20.4.1.1). The data in tributary slots marked as unavailable must be set to "all 0".

During a signal fail condition of the OTUCn or ODUCn signal at the input port of the OTUCn-M section, OTUCn-AIS or ODUCn-AIS is applied at the OTUCn to OTUCn-M adaptation. Such AIS signals have their OPUCn MSI and OMFI overhead and data in their tributary slots set to "all 1". Presence of such condition can be detected via the ODUCn PM STAT field.

Annex I

This annex is intentionally left blank.

Annex J

Recovery of ~~64B~~xB/~~66B~~ yB encoded clients from parallel 256B/257B interfaces

(This annex forms an integral part of this Recommendation.)

Figure J.1 illustrates the processing performed on the 200 and 400 GbE signal between the physical Ethernet interface and the point where the Ethernet signal is connected with the ODUflex to 200/400GbE adaptation function. In the receive direction (Figure J.1, right) the 200 or 400 GbE signal presented at the PMA interface passes through the AM lock and deskew, lane reorder and de-interleave, FEC decoder, post-FEC interleave, AM removal, x^{58} descrambler and 256b/257b to 64B/66B transcode processes that are specified in clause 119 of [IEEE 802.3]. The 64B/66B encoded signal is then adapted into the ODUflex and mapped into the OPUCn.

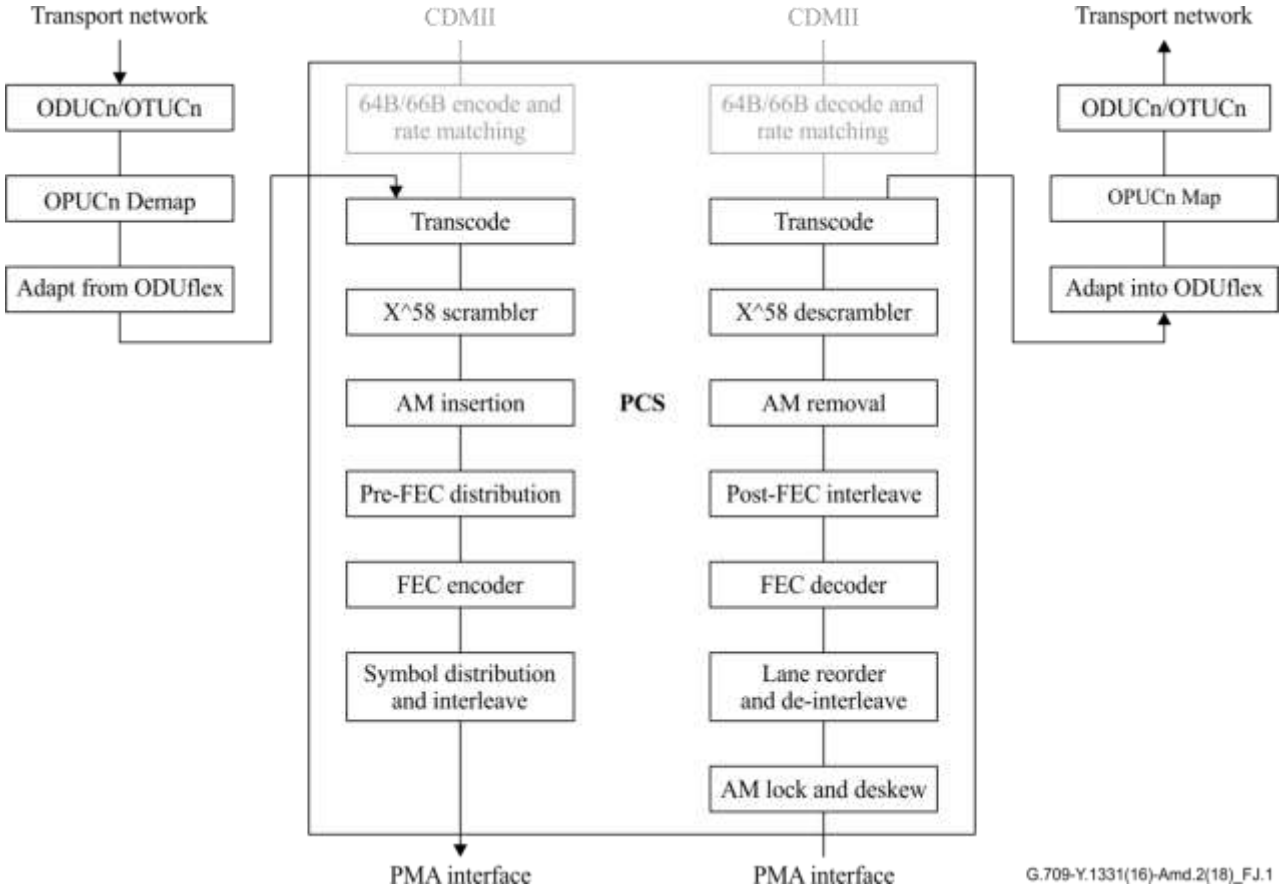


Figure J.1 – Processing of 200/400GbE signal

Figure J.2 illustrates the processing performed on the 50 GbE signal between the physical Ethernet interface and the point where the Ethernet signal is connected with the ODUflex to 50GbE adaptation function. In the receive direction (Figure J.2, right) the 50 GbE signal presented at the FEC or PMA interface passes through the Lane block sync, AM lock and deskew, lane reorder processes that are specified in clause 133 of [IEEE 802.3~~cd~~]. The 64B/66B encoded signal is then adapted into the ODUflex and mapped into the OPUk or OPUCn.

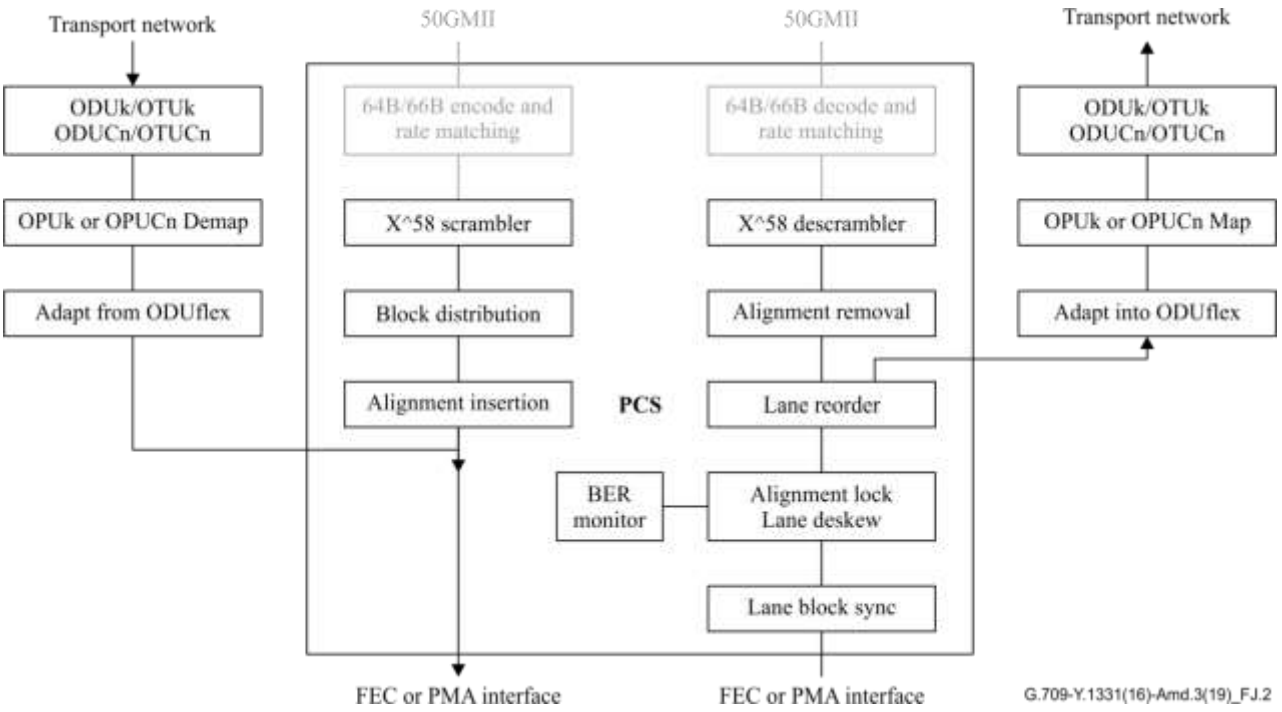


Figure J.2 – Processing of 50GbE signal

Figure J.3 illustrates the processing performed on the 800 GbE signal between the physical Ethernet interface and the point where the Ethernet signal (as two 400G streams of unscrambled 257-bit blocks: flow 0 and flow 1) is connected with the ODUflex to 800GbE adaptation function. In the receive direction (Figure J.3, right) the 800 GbE signal presented at the PMA interface passes through the AM lock and deskew, lane reorder and de-interleave as two aligned 400G streams (flow 0 and flow 1), and then for each flow through the FEC decoder, post-FEC interleave, AM removal, and x^{58} descrambling functions that are specified in clause 172 of [IEEE 802.3]. The two 256B/257B encoded streams (flow 0 and flow 1) are then adapted into the ODUflex and mapped into the OPUCn.

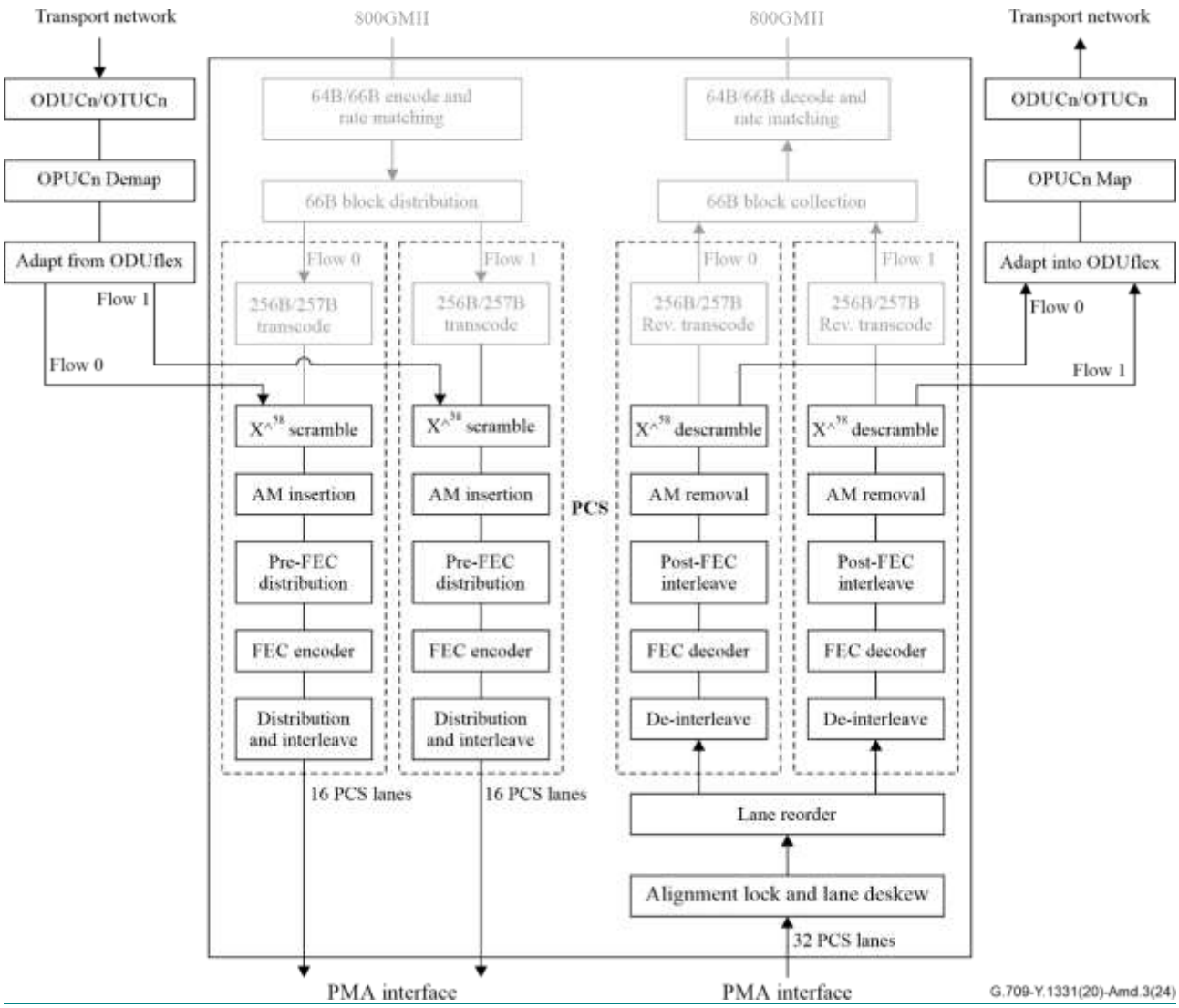


Figure J.3 – Processing of 800GbE signal

Annex K

Transporting 200GbE, ~~and~~ 400GbE and 800GbE am_sf<2:0> information through a single optical link between two Ethernet/OTN transponder entities in the OTN

(This annex forms an integral part of this Recommendation.)

K.1 Introduction

IP routers may support a "soft reroute" feature, which will direct IP traffic away from an optical link that experiences an increased number of FEC errors on that optical link. 200, ~~and~~ 400, and 800 Gbit/s Ethernet interfaces that are compliant with [IEEE 802.3] will be able to support this feature. Router vendors would like to see this feature also supported for cases in which two routers are interconnected via an OTN network. The Ethernet interfaces between router and OTN network could be equipped with (Router B in Figure K.1) or without (Router A) a clause 118 extender sublayer. In general such support is not feasible, because a 200G, ~~or~~ 400, or 800 Gbit/s Ethernet service can be transported over more than one optical link in the OTN passing through intermediate 3R regeneration functions and ODU cross connects. Nonetheless, support for this feature is possible for the case that such service is carried over a single optical link (supported by one OTSiA trail) in the OTN, with Ethernet to OTN transponders at the two end points as illustrated in Figures K.1 (for 200GbE and 400 GbE) and K.2 (for 800GbE). For such case the Ethernet to OTN mapper and demapper processes and the OTN FEC processes are within the same transponder and the FEC status information can be exchanged between these processes.

[IEEE 802.3] has specified three bits in the AM field (am_sf<2:0>) to carry link status information to support this feature. Bit am_sf<2> is defined as a Remote Degrade (RD) signal, bit am_sf<1> is defined as a Local Degrade (LD) signal and bit am_sf<0> is reserved. In the case of 800G Ethernet, the am_sf<2:0> bits are duplicated and present in the two 400G PCS streams (flow0 and flow1) alignment marker fields.

K.2 Client Degrade Indication (CDI) overhead

A 3-bit Client Defect Indication is defined in bits 2, 3 and 4 of the PSI[2] byte of the payload structure identifier to indicate the client degrade status of a 200, 400, or 800 Gbit/s Ethernet client signal which is transferred via one OTSiA trail. In the special cases described in Figures K.1 and K.2, the OTN transponders X and Y can forward the information from the am_sf field via the CDI overhead in the OTN frame.

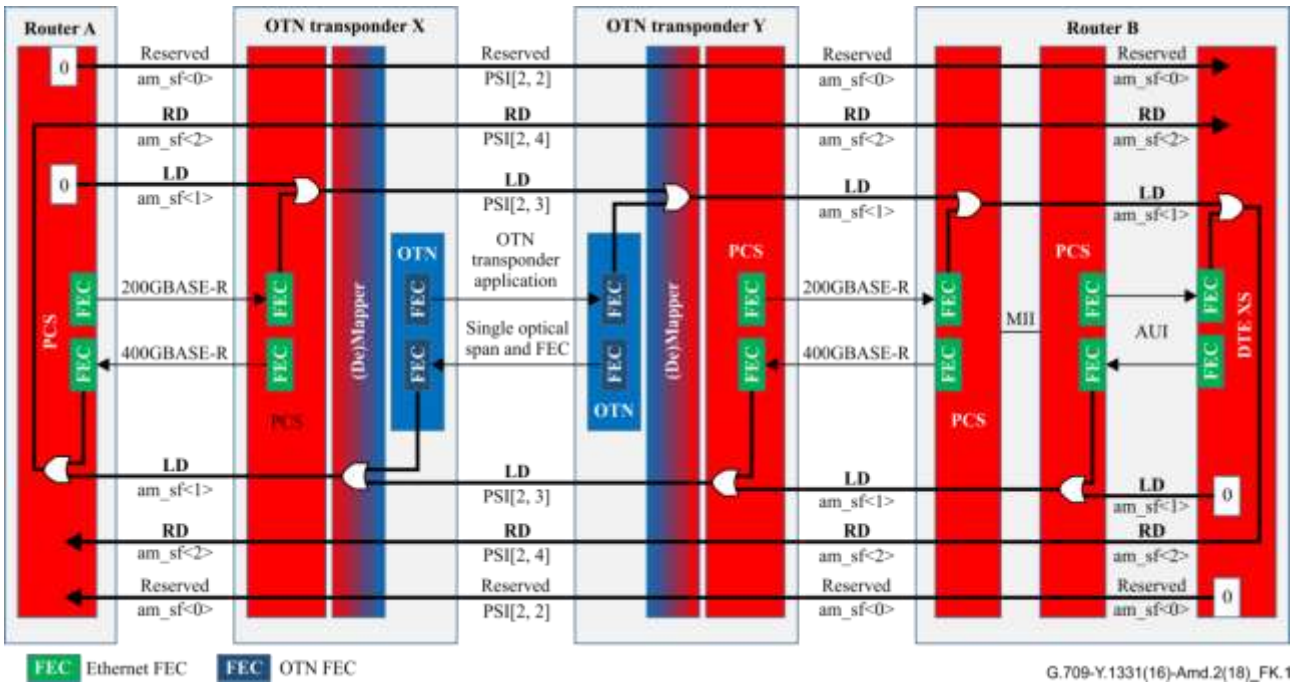


Figure K.1 – Local/Remote degrade interworking between router and OTN transponders

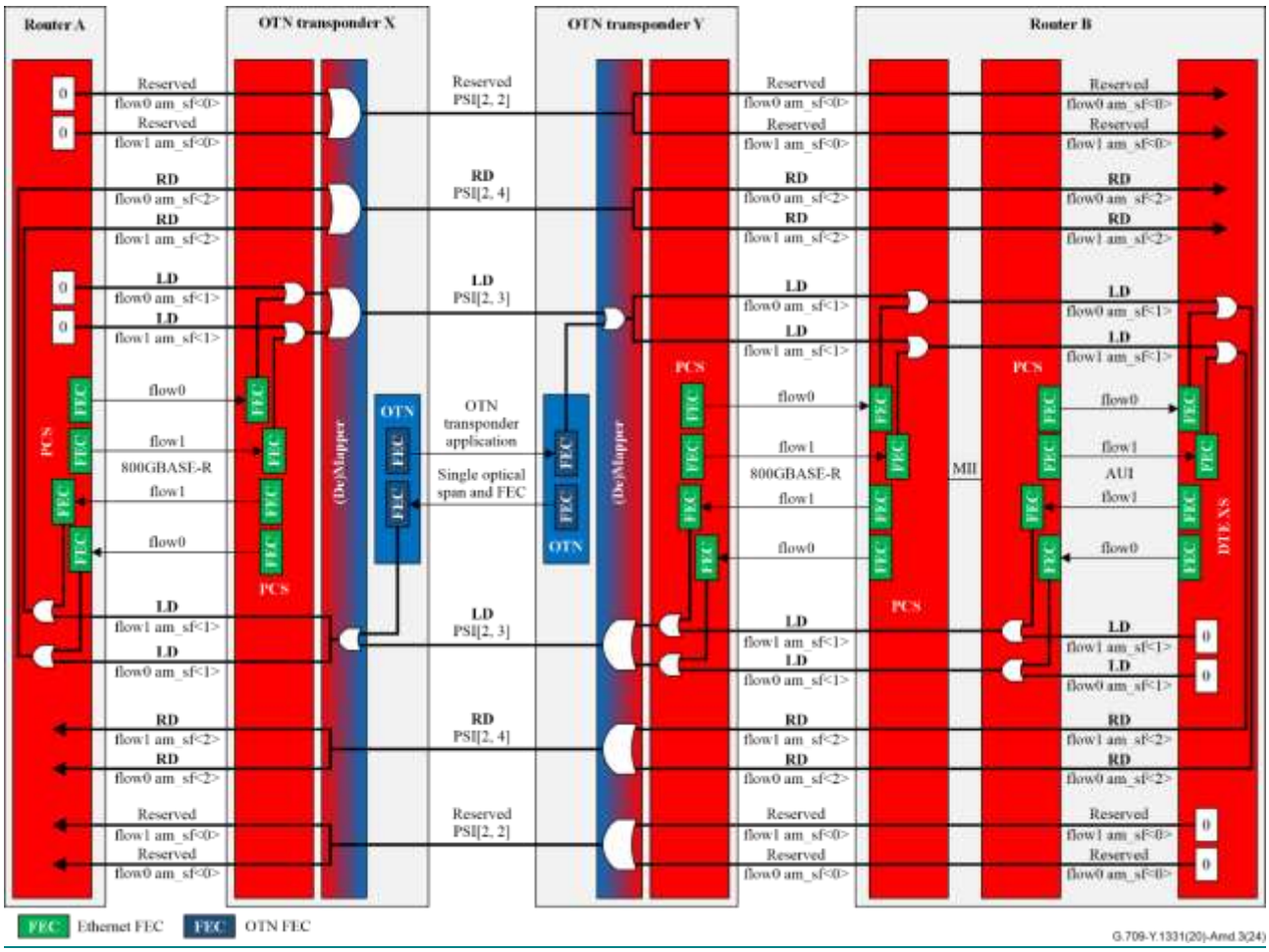


Figure K.2 – Local/Remote degrade interworking between router and OTN transponders for 800G Ethernet

The assignment of the various signals to the am_sf bits in the [IEEE 802.3] AM field and the CDI bits is shown in Table K.1.

Table K.1 – Link degrade signalling bit assignments

Signal	Location in AM field	Location in CDI overhead
Reserved	am_sf<0>	PSI byte 2, bit 2
LD	am_sf<1>	PSI byte 2, bit 3
RD	am_sf<2>	PSI byte 2, bit 4

For 200G and 400G clients, the following equations specify the translation between AM and CDI bits in the ingress transponder:

– PSI[2,2] ← am_sf<0>

– PSI[2,3] ← am_sf<1> OR Ethernet pre-FEC BER link degrade status

– PSI[2,4] ← am_sf<2>

For 800G clients, each 400G flows is processed independently and the logical OR of these intermediate results is encoded:

– PSI[2,2] ← $\text{am\_sf<0>}_{\text{flow0}}$ OR $\text{am\_sf<0>}_{\text{flow1}}$

– PSI[2,3] ← $\text{am\_sf<1>}_{\text{flow0}}$ OR Ethernet pre-FEC BER link degrade $\text{status}_{\text{flow0}}$ OR $\text{am\_sf<1>}_{\text{flow1}}$ OR Ethernet pre-FEC BER link degrade $\text{status}_{\text{flow1}}$

– PSI[2,4] ← $\text{am\_sf<2>}_{\text{flow0}}$ OR $\text{am\_sf<2>}_{\text{flow1}}$

In the egress transponder, the inverse operations are performed:

– am_sf<0> ← PSI[2,2]

– am_sf<1> ← PSI[2,3] OR OTN pre- FEC BER link degrade status

– am_sf<2> ← PSI[2,4]

For 800G clients, the information is replicated to the two flows.

For details refer to [ITU-T G.798].

~~For the special case described above (illustrated in Figures K.1 and K.2), the OTN transponders X and Y can forward the information in the Reserved (am_sf<0>) and RD (am_sf<2>) bits between ingress and egress transponder. The information in am_sf<0> can be carried in OPUflex PSI overhead byte 2, bit 2. The status information in am_sf<2>RD can be carried in OPUflex PSI overhead byte 2, bit 4. The status information in the LD (am_sf<1>) bit can be carried after some additional processing in OPUflex PSI overhead byte 2, bit 3 as illustrated in Figure K.1. The additional processing comprises the OR'ing of the LD status in the am_sf<1> bit of a 200G or 400G Ethernet signal with the local Ethernet FEC degrade status in the ingress transponder (X). In the egress transponder (Y), additional processing comprises the OR'ing of the LD status in the PSI[2,3] overhead bit of the OPUflex signal with the local OTN FEC degrade status in the transponder (details defined by [ITU-T G.798]). The resulting LD status is then carried in the am_sf<12:0> bit of the 200, or 400 Gbit/s Ethernet signal.~~

~~K.2 Client Degrade Indication (CDI) overhead~~

~~A 3-bit Client Defect Indication is defined in bits 2, 3 and 4 of the PSI[2] byte of the payload structure identifier to indicate the client degrade status of a 200, and 400 Gbit/s Ethernet client signal which is transferred via one OTSiA trail.~~

~~OPU PSI[2,2] is used to transfer the status information present in the 200 Gbit/s or 400 Gbit/s Ethernet client's am_sf<0> bit.~~

~~OPU PSI[2,3] is used to transfer the Local Degrade status information, which is determined by OR'ing the value present in the 200 or 400 Gbit/s Ethernet client's am_sf<1> bit with the 200 or 400 Gbit/s Ethernet FEC Degrade status as specified in [IEEE 802.3].~~
~~OPU PSI[2,4] is used to transfer the Remote Degrade status information present in the 200 Gbit/s or 400 Gbit/s Ethernet client''s am_sf<2> bit.~~

Annex L

OTU25u and OTU50u interfaces

(This annex forms an integral part of this Recommendation.)

L.1 Introduction

In network applications that do not require support of all client signals, the OTU25 and OTU50 interfaces with FEC as specified in [ITU-T G.709.4] may be operated at a lower rate that is within the ±100 ppm clock tolerance of the associated Ethernet 25GBASE-R and 50GBASE-R with RS-FEC rates. The bit rates of the corresponding "under-clocked" OTU25u and OTU50u are specified in this annex. All other OTU25 and OTU50 interface related specifications apply for the OTU25u and OTU50u.

Figure L.1 shows the relationship between various information structure elements and illustrates the multiplexing structure and mappings for the OTU25u and OTU50u. Figure L.1 shows that an ODUk (k = 0,1,2,2e,flex) signal is mapped into an OPU25u and an ODUk (k = 0,1,2,2e,3,flex) signal is mapped into an OPU50u. This OPU25u or OPU50u signal is mapped into the associated ODU25u or ODU50u. This ODU25u or ODU50u signal is mapped into the associated OTU25u or OTU50u signal.

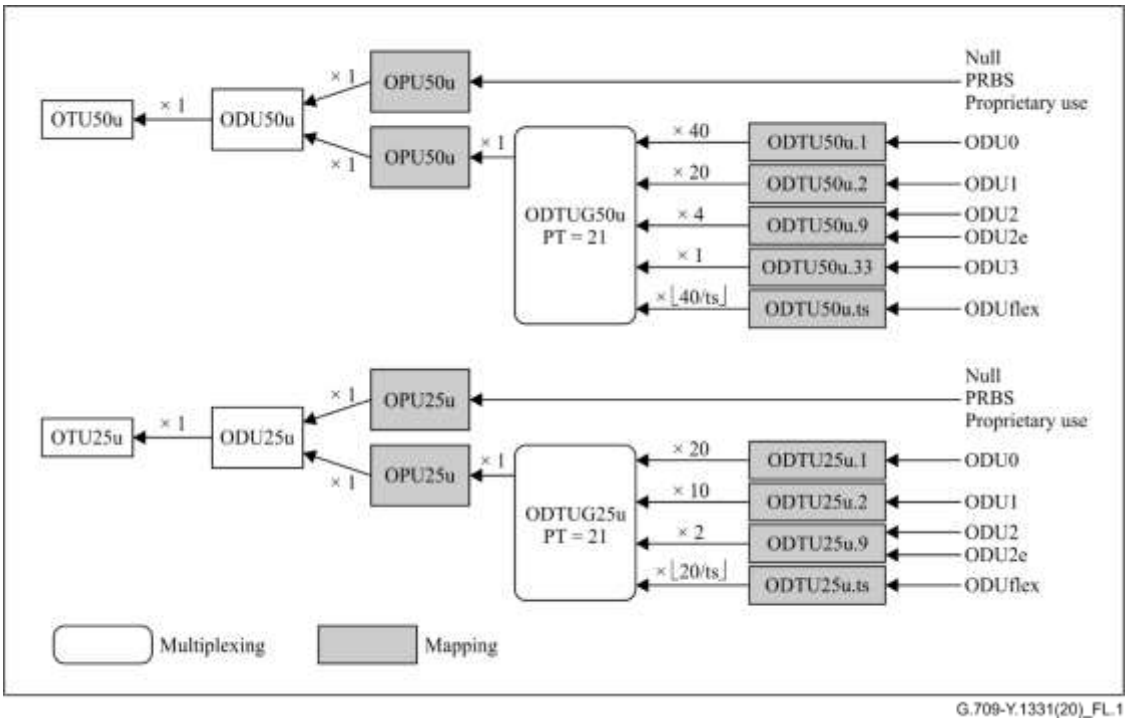


Figure L.1 – OTN multiplexing and mapping structures for OTU25u and OTU50u

L.2 Bit rates

The bit rates and tolerance of the OTU25u and OTU50u signals are defined in Table L.1.

The bit rates and tolerance of the ODU25u and ODU50u signals are defined in Table L.2.

The bit rates and tolerance of the OPU25u and OPU50u payload are defined in Table L.3.

The OTU25u, OTU50u, ODU25u, ODU50u, OPU25u and OPU50u frame periods are defined in Table L.4.

The 1.25G tributary slot related OPU25u and OPU50u multiframe periods are defined in Table L.5.

The ODTU payload area bandwidths are defined in Table L.6.

The number of OPU25u and OPU50u tributary slots required by a client ODU are summarized in Table L.7. Note that the value of n for ODUflex(GFP,n,2) is not restricted to the range $1 \leq n \leq 8$ as specified in Table 7-8.

The bit rates and tolerance of the OTU25u/ODU25u and OTU50u/ODU25u GCC signals are defined in Table L.8.

Table L.1 – OTU types and bit rates

OTU type	OTU nominal bit rate	OTU bit-rate tolerance
OTU25u	2937/2912 × 24 883 200 kbit/s	±20 ppm
OTU50u	2937/2912 × 49 766 400 kbit/s	
NOTE 1 – The nominal OTU rates are approximately: 25 096 826.374 kbit/s (OTU25u), 50 193 652.747 kbit/s (OTU50u). NOTE 2 – The OTU25u and OTU50u signal bit rates do not include a FEC overhead area. NOTE 3 – The OTU25u bit rate can be based on the OTU25u-RS bit rate as follows: 514/528 × 41118/41120 × OTU25u-RS bit rate = 514/528 × 41118/41120 × 660/637 × 24 883 200 kbit/s. NOTE 4 – The OTU50u bit rate can be based on the OTU50u-RS bit rate as follows: 514/544 × 41118/41120 × OTU50u-RS bit rate = 514/544 × 41118/41120 × 680/637 × 49 766 400 kbit/s.		

Table L.2 – ODU types and bit rates

ODU type	ODU nominal bit rate	ODU bit-rate tolerance
ODU25u	2937/2912 × 24 883 200 kbit/s	±20 ppm
ODU50u	2937/2912 × 49 766 400 kbit/s	
NOTE – The nominal ODU rates are approximately: 25 096 826.374 kbit/s (ODU25u), 50 193 652.747 kbit/s (ODU50u).		

Table L.3 – OPU types and bit rates

OPU type	OPU payload nominal bit rate	OPU payload bit-rate tolerance
OPU25u	49929/49712 × 24 883 200 kbit/s	±20 ppm
OPU50u	49929/49712 × 49 766 400 kbit/s	
NOTE 1 – The nominal OPU payload rates are approximately: 24 991 818.732 kbit/s (OPU25u Payload), 49 983 637.464 kbit/s (OPU50u Payload). NOTE 2 – The OPU25u bit rate can be based on the OTU25u-RS bit rate as follows: 238/239 × 514/528 × 41118/41120 × OTU25u-RS bit rate = 238/239 × 514/528 × 41118/41120 × 660/637 × 24 883 200 kbit/s. NOTE 3 – The OPU50u bit rate can be based on the OTU50u-RS bit rate as follows: 238/239 × 14/544 × 41118/41120 × OTU50u-RS bit rate = 238/239 × 514/544 × 41118/41120 × 680/637 × 49 766 400 kbit/s.		

Table L.4 – OTU/ODU/OPU frame periods

OTU/ODU/OPU type	Period (Note)
OTU25u/ODU25u/OPU25u	4.876 µs
OTU50u/ODU50u/OPU50u	2.483 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.	

Table L.5 – OPUk multiframe periods for 1.25G tributary slots

OPU type	1.25G tributary slot multiframe period (Note)
OPU25u	97.517 µs
OPU50u	97.517 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.	

Table L.6 – ODTU payload bandwidth (kbit/s)

ODTU type	ODTU payload nominal bandwidth		ODTU payload bit-rate tolerance
ODTU25u.ts	ts × 190.4/3824 × ODU25u bit rate		±20 ppm
ODTU50u.ts	ts × 95.2/3824 × ODU50u bit rate		
	Minimum	**Nominal**	**Maximum**
ODTU25u.ts, ODTU50u.ts	ts × 1 249 565.945	ts × 1 249 590.937	ts × 1 249 615.928
NOTE – The bandwidth is an approximated value, rounded to 3 decimal places.			

Table L.7 – Number of tributary slots required for ODUj into OPU25u and OPU50u

	# 1.25G tributary slots	
	OPU25u	**OPU50u**
ODU0	1	1
ODU1	2	2
ODU2	9	9
ODU2e	9	9
ODU3	–	33
ODUflex(CBR)		
– ODUflex(25GBase-R)	–	21
– ODUflex(50GBase-R)	–	–
ODUflex(GFP)	Note	Note
– ODUflex(GFP,8,2) for 10GBASE-R	8	8
– ODUflex(GFP,20,2) for 25GBASE-R	20	20
– ODUflex(GFP,40,2) for 50GBASE-R	40	40
– ODUflex(GFP,n,2), $1 \leq n \leq 20$ (OPU25u) and $1 \leq n \leq 40$ (OPU50u)	n	n
– ODUflex(GFP,n,3), $9 \leq n \leq 19$ (OPU25u) and $9 \leq n \leq 32$ (OPU50u)	n+1	n+1
– ODUflex(GFP,n,4), $33 \leq n \leq 38$ (OPU50u)	–	n+2
ODUflex(IMP)	Note	Note
– ODUflex(IMP,s)	Note	Note
NOTE – Refer to equations 19-1a and 19-1b in clauses 19.6.		

Table L.8 – GCC types and bit rates (kbit/s)

OTU/ODU type	Nominal bit rate GCC0	Nominal bit rate GCCi (i = 1,2)	Nominal bit rate GCC1+GCC2	Bit-rate tolerance
OTU25u/ODU25u	3 281.489	3 281.489	6 562.978	±20 ppm
OTU50u/ODU50u	6 562.978	6 562.978	13 125.955	
NOTE – The bandwidth is an approximated value, rounded to 3 decimal places.				

Annex M

Fine grain flexible ODU (fgODUflex) path layer network

(This annex forms an integral part of this Recommendation.)

This annex specifies a fine grain OTN path layer which complements the existing OTN to provide bandwidth efficient support for sub-1Gbit/s services. This fgODUflex layer network is a client layer network of the OTN ODUk/flex layer network.

The information structure for fgODUflex is represented by information containment relationships and flows. The principal information containment relationship is described in Figure M.1.

A client signal is mapped into the fgOPUflex payload area and the generated justification control information is inserted into fgOPUflex overhead area. It generates the fgODUflex path connection monitoring information which is inserted into fgODUflex PM overhead. Furthermore, it generates the fgODUflex tandem connection monitoring information which is inserted into fgODUflex TCM overhead. Up to two levels of fgODUflex tandem connection monitoring are supported. The generated fgODUflex signal is mapped into one or multiple fine grain tributary slots of the server OPU.

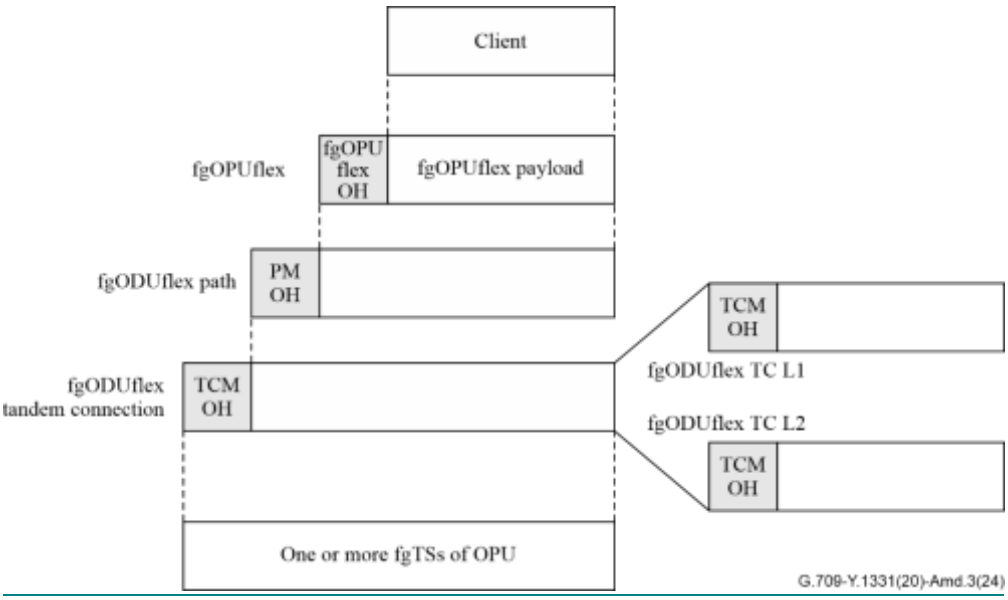


Figure M.1 – fgODUflex principal information containment relationship

M.1 fgODUflex frame structure

The fgODUflex frame structure is shown in Figure M.2. It is organized in an octet-based block frame structure with four rows and 3824 columns.

The two main areas of the fgODUflex frame are:

– fgODUflex overhead area

– fgOPUflex area

Columns 1 to 14 and 1905 to 1918 of the fgODUflex are dedicated to fgODUflex overhead area. It consists of frame alignment signals (FAS0 to FAS7), multiframe alignment signal (MFAS), PM, TCM1, TCM2, and phase difference accumulation (DA).

Columns 15 to 1904 and 1919 to 3824 of the fgODUflex are dedicated to fgOPUflex area. It consists of one fgOPUflex frame.

The two main areas of the fgOPUflex frame are:

– fgOPUflex overhead area;

– fgOPUflex payload area.

Columns 15 to 16 and 1919 to 1920 of the fgOPUflex are dedicated to fgOPUflex overhead area.

Columns 17 to 1904 and 1921 to 3824 of the fgOPUflex are dedicated to fgOPUflex payload area.

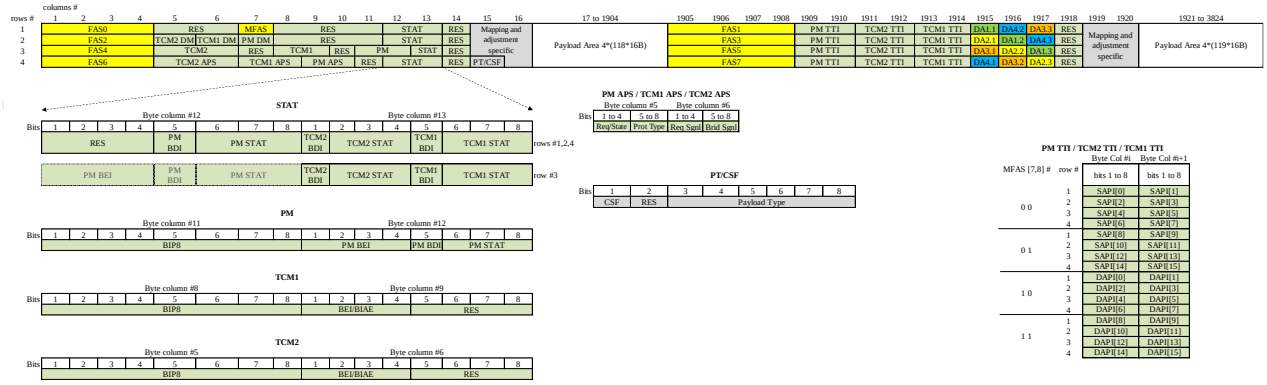


Figure M.2 – fgODUflex frame structure

M.2 fgODUflex frame alignment signal and overheads

M.2.1 fgODUflex frame alignment signal

Eight 4-byte FAS signals (FAS0 to FAS7) are defined in rows 1 to 4 and columns 1 to 4 and 1905 to 1908 of the fgODUflex overhead. The specific structure of FAS is described in Figure M.3. The fourth byte pattern is logically expressed as (0x28 XOR HRN) where the HRN is the half row number. The specific patterns of Byte 4 for FAS0 to FAS7, after XOR, are 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E and 0x2F, respectively.

	Byte 1	Byte 2	Byte 3	Byte 4	HRN
Bits	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8	
FAS0(Bytes #1 to 4 of FAS)	0xF6	0xF6	0xF6	0x28 XOR HRN	0 0 0
FAS1	0xF6	0xF6	0xF6	0x28 XOR HRN	0 0 1
FAS2	0xF6	0xF6	0xF6	0x28 XOR HRN	0 1 0
FAS3	0xF6	0xF6	0xF6	0x28 XOR HRN	0 1 1
FAS4	0xF6	0xF6	0xF6	0x28 XOR HRN	1 0 0
FAS5	0xF6	0xF6	0xF6	0x28 XOR HRN	1 0 1
FAS6	0xF6	0xF6	0xF6	0x28 XOR HRN	1 1 0
FAS7	0xF6	0xF6	0xF6	0x28 XOR HRN	1 1 1

G.709-Y.1331(20)-Amd.3(24)

Figure M.3 – fgODUflex FAS patterns

A multiframe alignment signal (MFAS) byte is defined in row 1, column 7 of the fgODUflex overhead. The value of the MFAS byte will be incremented at each fgODUflex frame and provides a 256-frame multiframe.

M.2.2 fgODUflex PM, TCM1 and TCM2 overheads

The fgODUflex OH information is added to the fgODUflex information payload to create an fgODUflex. It includes information for maintenance and operational functions to support fgODUflex

connections. The fgODUflex OH consists of fields dedicated to the end-to-end fgODUflex path and to two levels of tandem connection monitoring. The fgODUflex path OH is terminated where the fgODUflex is assembled and disassembled. The TC OH is added and terminated at the source and sink of the corresponding tandem connections, respectively. The specific OH format and coding are defined in clauses M.2.2.1 to M.2.2.7.

The PM/TCM1/TCM2 field contains the following subfields (see Figure M.2):

– trail trace identifier (TTI);

– bit interleaved parity (BIP-8);

– backward defect indication (BDI);

– backward error indication and backward incoming alignment error (BEI/BIAE);

– status bits indicating the presence of a maintenance signal (STAT);

– delay measurement (DM);

– automatic protection switch (APS).

NOTE – BIAE is only applicable to TCM1 or TCM2.

M.2.2.1 fgODUflex trail trace identifier (TTI)

A trail trace identifier (TTI) is defined as a 32-byte string with the structure shown in Figure M.2.

For PM, an 8-byte TTI field is defined in rows 1 to 4, columns 1909 and 1910; For TCM1, an 8-byte TTI field is defined in rows 1 to 4, columns 1913 and 1914; For TCM2, an 8-byte TTI field is defined in rows 1 to 4, columns 1911 and 1912. Each four frames transmit once complete 32-byte trail trace identifier information for PM, TCM1 or TCM2. The four 8-byte sets in the trail trace identifier are indexed by MFAS[7,8].

M.2.2.2 fgODUflex error detection code (BIP-8)

A bit interleaved parity-8 (BIP-8) code is defined in the fgODUflex overhead for error detection. Each fgODUflex BIP-8 is computed over the bits in the fgOPUflex area (columns 15 to 1904 and 1919 to 3824) of fgODUflex frame #i, and is inserted in the fgODUflex BIP-8 overhead location in the fgODUflex frame #i+2 (see Figure M.4).

For PM, BIP-8 is located in row 3 column 11; For TCM1, BIP-8 is located in row 3 column 8; For TCM2, BIP-8 is located in row 3 column 5.

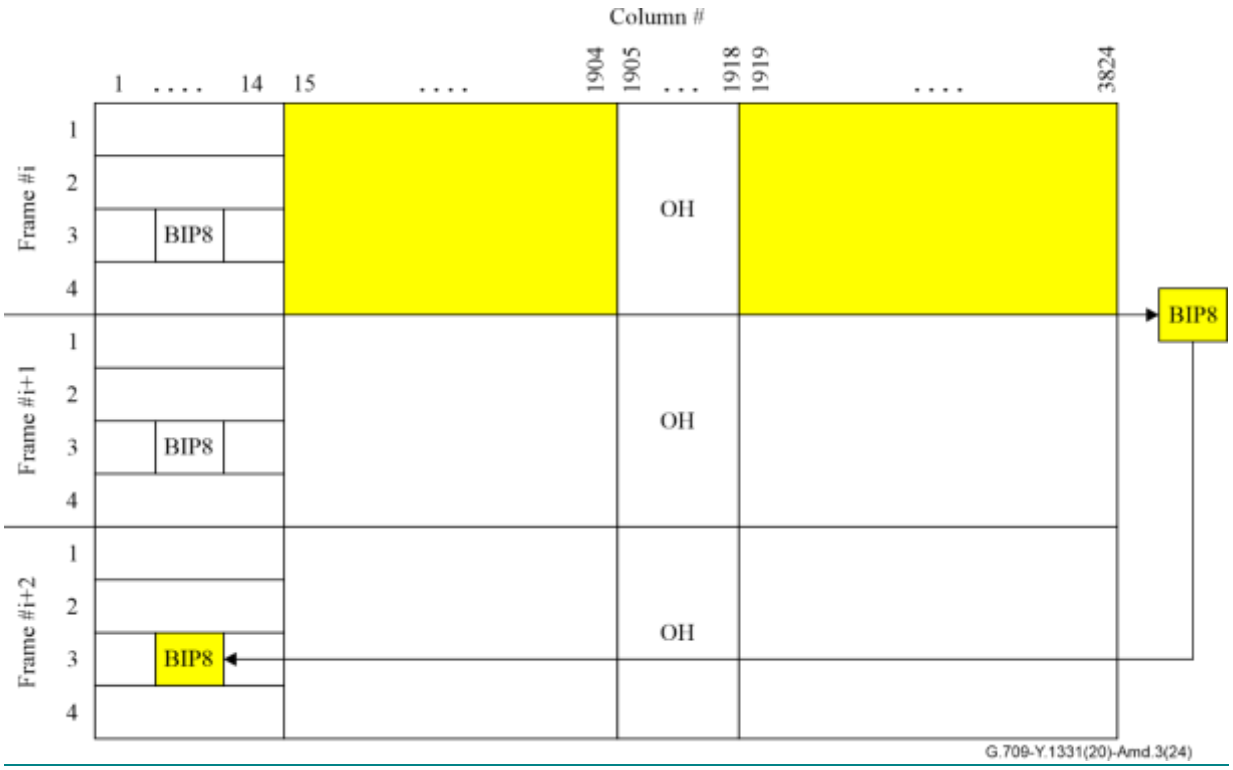


Figure M.4 – fgODUflex BIP-8 computation

M.2.2.3 fgODUflex backward defect indication (BDI)

A single-bit backward defect indication (BDI) signal is defined to convey the signal fail status detected in a path or tandem connection termination sink function in the upstream direction.

BDI is set to "1" to indicate an fgODUflex backward defect indication, otherwise it is set to "0".

For PM, BDI is located in bit 5 of column 12 of each row; For TCM1, BDI is located in bit 5 of column 13 of each row; For TCM2, BDI is located in bit 1 of column 13 of each row.

M.2.2.4 fgODUflex backward error indication (BEI) and backward incoming alignment error (BIAE)

For PM, a 4-bit backward error indication (BEI) signal is defined in bits 1 to 4 of row 3 column 12. It keeps the consistent function with ODUk PM BEI as described in clause 15.8.2.1.4. The PM BEI interpretation is same as described in Tables 15-4.

For TCM1 and TCM2, a 4-bit backward error indication (BEI) and backward incoming alignment error (BIAE) signal is defined separately in bits 1 to 4 of row 3 column 9 and bits 1 to 4 of row 3 column 6. It keeps the consistent function with ODUk TCM BEI/BIAE as described in clause 15.8.2.2.4. The TCM BEI/BIAE interpretation is same as described in Tables 15-6.

M.2.2.5 fgODUflex status (STAT)

For path monitoring, three bits are defined as status bits (PM STAT). They indicate the presence of a maintenance signal. A PM status field is located in bits 6 to 8 of column 12 of each row. The status interpretation of bits 6 7 8 of PM STAT follows Table 15-5, with ODU-LCK, ODU-OCI and ODU-AIS, replaced by fgODUflex-LCK, fgODUflex-OCI and fgODUflex-AIS, respectively.

For each tandem connection monitoring field, three bits are defined as status bits (TCM1 STAT and TSM2 STAT). They indicate the presence of a maintenance signal, if there is an incoming alignment error at the source TC-CMEP, or if there is no source TC-CMEP active. A TCM1 STAT is located in bits 6 to 8 of column 13 of each row. A TCM2 STAT is located in bits 2 to 4 of column 13 of each row. The status interpretation of bits 6 7 8 of TCM1 STAT and bits 2 3 4 of TCM2 STAT follows

Table 15-7, with ODU-LCK, ODU-OCI and ODU-AIS, replaced by fgODUflex-LCK, fgODUflex-OCI and fgODUflex-AIS, respectively.

M.2.2.6 fgODUflex delay measurement (DM)

A 1-byte PM DM is defined in row 2 and column 7; A 1-byte TCM1 DM is defined in row 2 and column 6; A 1-byte TCM2 DM is defined in row 2 and column 5. The delay measurement includes three DM messages as shown in Figure M.5. Each DM message is transmitted over 32 fgODUflex frames and is aligned to MFAS[4:8]. The timing reference point of a DM message is at the first bit of the fgODUflex frame (MFAS[4:8]=0) carrying the first byte of the DM message. I.e., the reference point is Bit 1 of Row 1, Column 1 of the fgODUflex frame carrying the DM_MSG_TYPE byte. The 32-bit nanoseconds field of each DM message corresponds to the 32-bit nanoseconds field of the [IEEE 1588] format (e.g., bit 1 of Tx-f-TS[0] carrying the MSB and bit 8 of Tx-f-TS[3] carrying the LSB). The 32-bit seconds field of each DM message corresponds to the least significant 32 bits of the 48-bit seconds field of the IEEE 1588 format (e.g., bit 1 of Tx-f-TS[4] carrying the MSB and bit 8 of Tx-f-TS[7] carrying the LSB). The CRC-12 only protects the content in the yellow area as shown in Figure M.5. The polynomial of the CRC-12 is $G(x) = x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$.

In the case of one-way delay measurement, the source node sends 1DM message with the forward transmitting timestamp in Tx-f-TS bytes. The sink node compares the forward transmitting timestamp in the 1DM message with the forward receiving timestamp when it received the 1DM message to compute the one-way delay (it equals to *Rx-f-TS – Tx-f-TS*).

In the case of two-way delay measurement, the source node sends 2DMM message with the forward transmitting timestamp in Tx-f-TS bytes. The sink node responds with the 2DMR message with the backward transmitting timestamp in Tx-b-TS bytes, the forward receiving timestamp of the received 2DMM message in Rx-f-TS bytes, and the received forward transmitting timestamp Tx-f-TS of 2DMM in Tx-f-TS bytes. The source node that receives the 2DMR uses the three timestamps in the 2DMR message, plus the backward receiving timestamp (Rx-b-TS) of the received 2DMR message, to compute the two-way delay (it equals to (*Rx-b-TS – Tx-f-Ts*) – (*Tx-b-TS – Rx-f-Ts*)).

NOTE – The CRC-12 field is generated using the mathematical steps specified in clause 9.3.3.1.1 of [b-ITU-T G.8312].

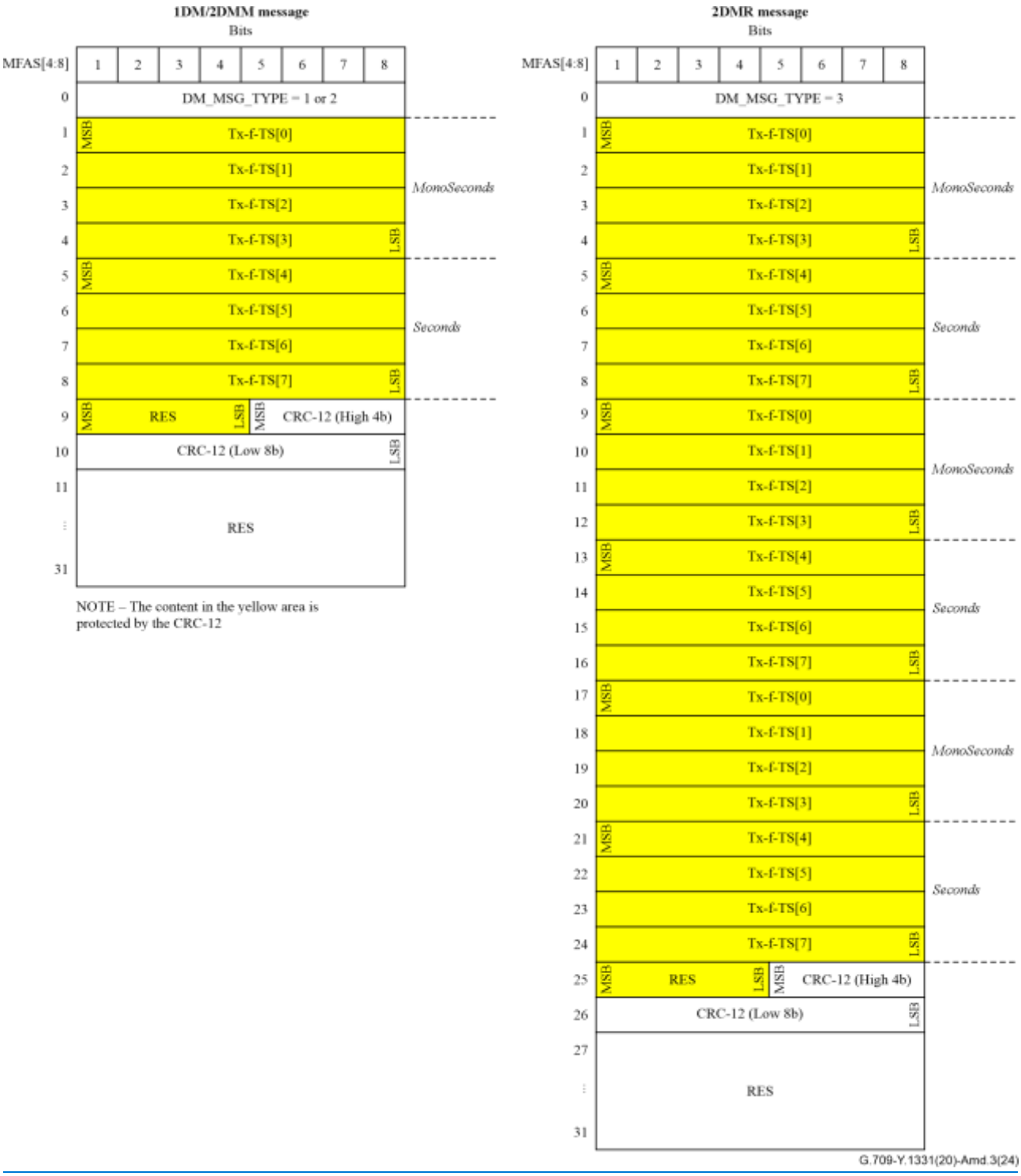


Figure M.5 – Delay measurement messages

M.2.2.7 fgODUflex automatic protection switch (APS)

A 2-byte PM APS signal is defined in row 4, columns 9 and 10; a 2-byte TCM1 APS signal is defined in row 4, columns 7 and 8; a 2-byte TCM2 APS signal is defined in row 4, columns 5 and 6.

The 2-byte APS specific encode format is specified in Figure M.6.

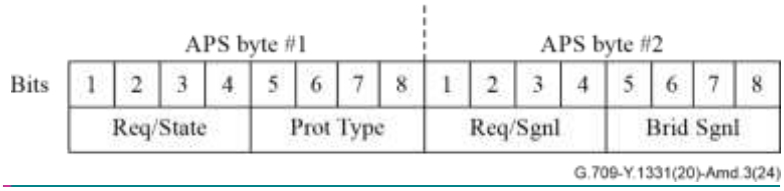


Figure M.6 – fgODUflex APS format

M.2.3 fgODUflex clock difference accumulation overhead

The clock of fgODUflex signals are not required to be low-pass filtered at intermediate fgODUflex switching nodes. A scheme accumulating the relative clock offset between adjacent fgODUflex processing nodes is provided in the fgODUflex OH to enable regeneration of the CBR client clock.

At the Source NE, the local reference clock (REFCLKsrc) is used to measure the bit rate of the CBR client. The result is encoded as GMP C_m:C_nD and inserted into the fgODUflex OPU overhead region. The fgODUflex server signal is multiplexed into an ODUk(fgTS) or an ODUflex(fgTS,n) multiplexing server signal. The fgODUflex and the ODUk(fgTS)/ODUflex(fgTS, n) are phase locked to REFCLKsrc. At an intermediate fgODUflex switching NE, fgODUflex signals are demultiplexed from ingress ODUk(fgTS)/ODUflex(fgTS, n) signals and re-multiplexed into egress ODUk(fgTS)/ODUflex(fgTS, n) signals. The egress ODUk(fgTS)/ODUflex(fgTS, n) signals are generated from and phase locked to the local reference clock (REFCLKsw) of the switching NE. At the Sink NE, the clock of the CBR client is re-generated as a function of the received GMP C_m:C_nD and the relative clock offset between the local reference clock (REFCLKsk) and the reference clock at the source NE (REFCLKsrc).

At each fgODUflex processing NE (Intermediate fgODUflex switch NE and Sink NE), the clock of the ingress ODUk(fgTS)/ODUflex(fgTS,n) multiplexing server signal is recovered and scaled to form a Normalized Multiplexing Server clock (NMSCk) that has a nominal frequency defined to be 311.04 MHz, common to all NEs. The relationship between NMSCk and the ingress ODUk(fgTS)/ODUflex(fgTS,n) for various values of k and n is shown in Table M.1.

Table M.1 – Relationship between NMSCk and ODUk(fgTS)/ODUflex(fgTS,n) clock

ODUk(fgTS) ODUflex(fgTS,n)	Bit rate	NMSCk Bit rate (Nominally 311.04 MHz)
ODU0	1 244 160 kbit/s	(1/4) × ODU0 Bit Rate
ODU1	239/238 × 2 488 320 kbit/s	(1/8) × (238/239) × ODU1 Bit rate
ODUflex(fgTS,n=3)	3 × 1 244 160 kbit/s	(1/12) × ODUflex(fgTS,3) Bit rate
ODUflex(fgTS,n=4)	4 × 1 244 160 kbit/s	(1/16) × ODUflex(fgTS,4) Bit rate
ODUflex(fgTS,n=5)	5 × 1 244 160 kbit/s	(1/20) × ODUflex(fgTS,5) Bit rate
ODUflex(fgTS,n=6)	6 × 1 244 160 kbit/s	(1/24) × ODUflex(fgTS,6) Bit rate
ODUflex(fgTS,n=7)	7 × 1 244 160 kbit/s	(1/28) × ODUflex(fgTS,7) Bit rate
ODU2	239/237 × 9 953 280 kbit/s	(1/32) × (237/239) × ODU2 Bit rate

The number of NMSCk cycles is measured every sampling period (Tsamp), which is set to be greater than or equal to the DAi overhead spacing (corresponding to one row period of the fgODUflex(p)). Tsamp is generated from the local reference clock (REFCLKsw or REFCLKsk), or Tsamp instants may be triggered by external events, such as the consecutive arrivals or N^{th} arrivals of DAi overheads in the fgODUflex(p) client, in compliance with the requirement of Tsamp ≤ 3 ms.

NOTE 1 – Examples of Tsamp generation are:

- Example 1: Tsamp may be set at Tsamp = 3 ms for all fgODUflex clients or set to Tsamp = 3 ms/p, where p is the rate class of the fgODUflex(p) client.
- Example 2: Tsamp may be set to the interval between consecutive arrivals or N^{th} arrivals of DAi overheads in the fgODUflex(p) client.

The difference between the measured NMSCk cycles and the expected value of (311.04 MHz × Tsamp) yields the relative clock offset between the reference clock at the NE sourcing the ingress ODUk(fgTS)/ODUflex(fgTS,n) multiplexing server signal and the local reference clock in units of NMSCk UI.

PD = measured NMSCk cycles – (311.04 MHz × Tsamp) (M-1)

The phase difference (PD) is generated at each Tsamp period and is applicable to all the fgODUflex signals sharing the same Tsamp instant and period, that are multiplexed inside the corresponding ODUk(fgTS)/ODUflex(fgTS,n) multiplexing server signal. Each fgODUflex accumulates the PD encountered at each fgODUflex processing NE in the path from the Source NE to the Sink NE in the DA overhead fields located at the fgODUflex overhead area of the fgODUflex signal.

A set of four DA overheads (DAi; i=1, 2, 3 and 4) are defined in rows 1 to 4, columns 1915 to 1917 of the fgODUflex overhead. Each DAi consists of 3 bytes (DAi.1, DAi.2 and DAi.3) as shown in Figure M.2. The PD is summed into the existing signed value already in the DAi overhead. DAi.2 and DAi.3 serve as replicated values to the preceding DAi.1, for error protection. At the fgODUflex sink node, the received DAi.1, and its successive DAi.2 and DAi.3 support majority voting (2 out of 3) to recover the DAi value in the presence of bit errors. The summing of PD into DAi must not cause the DAi to exceed its legal range of –127 to +127. Actions taken to stay within the legal range must be lossless. The 8-bit signed value in DAi.1, DAi.2 or DAi.3 is encoded in two's complement format, with bit 1 carrying the MSB and bit 8 carrying the LSB.

NOTE 2 – the 8-bit signed value in DAi.1, DAi.2 or DAi.3 is encoded in two's-complement format. For example, a value of 0 is encoded as "0000 0000", a value of +31 is encoded as "0001 1111" and a value of -31 is encoded as "1110 0001".

M.2.4 fgOPUflex overhead

A 16-byte fgOPUflex overhead is defined in rows 1 to 4, columns 15, 16, 1919 and 1920 of the fgOPUflex overhead. It consists of payload type (PT), the client signal fail (CSF), and the specific overhead associated with the mapping of client signals into the fgOPUflex payload. The specific overhead associated with the mapping of client signals into the fgOPUflex payload is described in clause M.5.

A 6-bit payload type signal is defined in bits 3 to 8 of row 4 and column 15 of the fgOPUflex overhead to indicate the composition of the fgOPUflex signal. The code points are defined in Table M.2.

Table M.2 – fgOPUflex payload type (PT) code points

Bit 3 4 5 6 7 8	Hex code (Note 1)	Interpretation
0 0 0 0 0 1	01	Experimental mapping (Note 3)
0 0 0 0 1 0	02	Packet client mapping into fgOPUflex, see clause M.5.2
0 0 0 0 1 1	03	CBR mapping into fgOPUflex, see clause M.5.3
0 0 0 1 0 0	04	Not available (Note 2)
0 0 0 1 0 1	05	v_1×VC-12 mapping into fgOPUflex, see clause M.5.4.1
0 0 0 1 1 0	06	Not available (Note 2)
0 0 0 1 1 1	07	v_2×VC-3 mapping into fgOPUflex, see clause M.5.4.1

Table M.2 – fgOPUflex payload type (PT) code points

Bit 3 4 5 6 7 8	Hex code (Note 1)	Interpretation
0 0 1 0 0 0	08	VC-4 mapping into fgOPUflex, see clause M.5.4.1
0 0 1 0 0 1	09	E1 mapping into fgOPUflex, see clause M.5.4.2
0 1 0 1 0 1	15	Not available (Note 2)
1 0 0 1 1 0	26	Not available (Note 2)
1 1 0 x x x	30-37	Reserved codes for proprietary use (Note 4)
1 1 1 1 1 0	3E	PRBS test signal mapping, see clause M.5.1
1 1 1 1 1 1	3F	Not available (Note 2)

NOTE 1 – There are 43 spare codes left for future international standardization. Refer to Annex A of [ITU-T G.806] for the procedure to obtain one of these codes for a new payload type.

NOTE 2 – These values are excluded from the set of available code points. These bit patterns are present in fgODUflex maintenance signals.

NOTE 3 – Value "01" is only to be used for experimental activities in cases where a mapping code is not defined in this table. Refer to Annex A of [ITU-T G.806] for more information on the use of this code.

NOTE 4 – These 8 code values will not be subject to further standardization. Refer to Annex A of [ITU-T G.806] for more information on the use of these codes.

A single-bit fgOPUflex client signal fail (CSF) indicator is defined to convey the fail status of the client signal mapped into an fgOPUflex at the ingress of the OTN, to the egress of the OTN. The fgOPUflex CSF is located in bit 1 of row 4 and column 15 of the fgOPUflex overhead. The fgOPUflex CSF is set to "1" to indicate a client signal fail indication, otherwise it is set to "0".

M.3 fgODUflex maintenance signals

Three fgODUflex maintenance signals are defined below:

– fgODUflex alarm indication signal (fgODUflex-AIS);

– fgODUflex open connection indication (fgODUflex-OCI);

– fgODUflex locked (fgODUflex-LCK).

The fgODUflex-AIS is specified as all "1"s in the entire fgODUflex signal, excluding the frame alignment signals (FAS0 to FAS7) and multiframe alignment signal (MFAS) (see Figure M.7).

The fgODUflex-OCI is specified as a repeating "0110 0110" pattern in the entire fgODUflex signal, excluding the frame alignment signals (FAS0 to FAS7) and multiframe alignment signal (MFAS) (see Figure M.7).

The fgODUflex-LCK is specified as a repeating "0101 0101" pattern in the entire fgODUflex signal, excluding the frame alignment signals (FAS0 to FAS7) and multiframe alignment signal (MFAS) (see Figure M.7).

In addition, the fgODUflex-AIS, fgODUflex-OCI, or fgODUflex-LCK signal may be extended with one or two levels of fgODUflex tandem connection and/or APS overhead before it is presented at the OTN interface. This is dependent on the functionality between the fgODUflex-AIS, fgODUflex-OCI, or fgODUflex-LCK insertion point and the fgODUflex interface.

The presence of the fgODUflex-AIS, fgODUflex-OCI, or fgODUflex-LCK is detected by monitoring the fgODUflex STAT bits in the PM, TCM1 and TCM2 overhead fields.

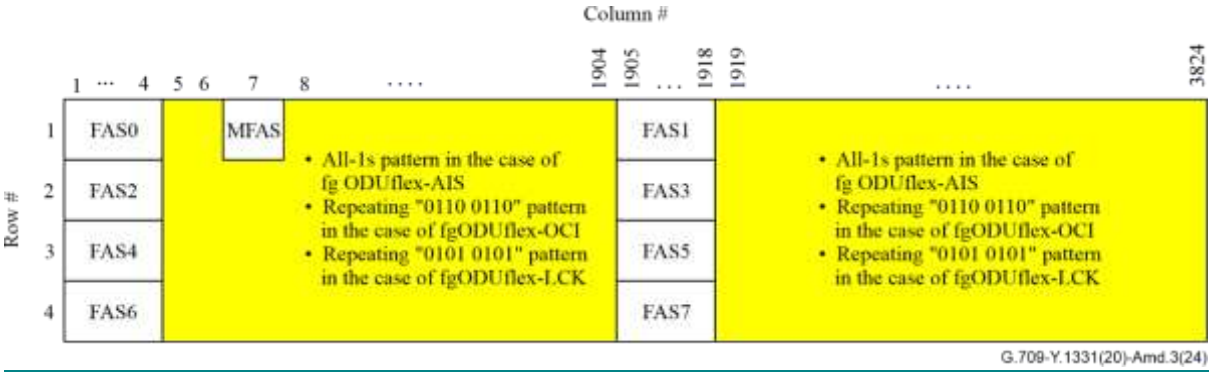


Figure M.7 – fgODUflex-AIS, fgODUflex-OCI or fgODUflex-LCK

M.4 fgODUflex bit rates and bit-rate tolerances

The bit rates and tolerance of the fgODUflex signals are defined in Table M.3A. The fgODUflex signals may be generated using a local clock. The fgODUflex frequency and frequency tolerance are locked to the local clock's frequency and frequency tolerance. The local clock frequency tolerance for the fgODUflex is specified to be ±20 ppm.

The fgODUflex maintenance signals (AIS, OCI, LCK) are generated using a local clock. In a number of cases this local clock may be the clock of a server ODU signal over which the fgODUflex signal is transported between equipment or through equipment (in one or more of the fine grain tributary slots). For these cases, the nominal justification ratio should be deployed to comply with the fgODUflex's bit-rate tolerance specification.

Table M.3A – fgODUflex/fgOPUflex payload bit rates

fgODUflex type	fgODUflex nominal bit rate	fgOPUflex payload nominal bit rate	bit-rate tolerance
fgODUflex(p) (p = 1 to 119)	(p/119.525) × 1 244 160 kbit/s	(p/119.525) × (237/239) × 1 244 160 kbit/s	±20 ppm
NOTE – The nominal ODU rates are approximately 10 409.203 kbit/s (fgODUflex(1)), and p × 10 409.203 kbit/s (fgODUflex(p)). The nominal OPU payload rates are approximately 10 322.097 kbit/s (fgOPUflex(1) Payload), and p × 10 322.097 kbit/s (fgOPUflex(p) Payload).			

The fgODUflex and fgOPUflex frame periods are defined in Table M.3B.

Table M.3B – fgODUflex/fgOPUflex frame periods

fgODUflex/fgOPUflex type	Frame period
fgODUflex(p)/fgOPUflex(p) (p = 1 to 119)	11.756/p ms
NOTE – The frame period is an approximated value, rounded to 3 decimal places.	

The bit rates and tolerance of the server ODU and fgODTUk.M are defined in Table M.4. The ODU and fgODTUk.M signals may be generated using a local clock. The ODU and fgODTUk.M frequency and frequency tolerance are locked to the local clock's frequency and frequency tolerance. The local clock frequency tolerance for the ODU and fgODTUk.M are specified to be ±20 ppm.

Table M.4 – Bit rates of Server ODU and fgODTUk.M carrying fgODUflex signals

ODU type	ODU nominal bit rate	fgODTUk.M payload nominal bit rate	bit-rate tolerance
ODU0(fgTS)	1 244 160 kbit/s	(M/119) × (238/239) × 1 244 160 kbit/s	±20 ppm
ODU1(fgTS)	239/238 × 2 488 320 kbit/s	(M/238) × 2 488 320 kbit/s	
ODU2(fgTS)	239/237 × 9 953 280 kbit/s	(M/952) × (238/237) × 9 953 280 kbit/s	
ODUflex(fgTS,n) (n = 3 to 7)	n ×1 244 160 kbit/s	(M/119) × (238/239) × 1 244 160 kbit/s	

NOTE – The fgODTU0.M nominal bit rate is approximately M×10 411 kbit/s; the fgODTU1.M nominal bit rate is approximately M×10 455 kbit/s; the fgODTU2.M nominal bit rate is approximately M×10 499 kbit/s; the ODUflex(fgTS,n) denotes n×1.25G ODUflex which is n times of ODU0 bit rate and the value of n is from 3 to 7; the fgODTUflex.M nominal bit rate of ODUflex(fgTS,n) is approximately M×10 411 kbit/s. The value of M is from 1 to 119.

M.5 Mapping of client signals into fine grain flexible OPU (fgOPUflex)

M.5.1 Mapping of PRBS test signal into fgOPUflex

The fgOPUflex end to end or segment connections may be tested using a PRBS test signal as the client signal instead of the actual client signal. A 2 147 483 647-bit pseudo-random test sequence ($2^{31} – 1$) as specified in clause 5.8 of [ITU-T O.150] can be mapped into the fgOPUflex payload. Groups of eight successive bits of the 2 147 483 647-bit pseudo-random test sequence signal are mapped into 8 data bits (8D) (i.e., one byte) of the fgOPUflex payload (see Figure M.8).

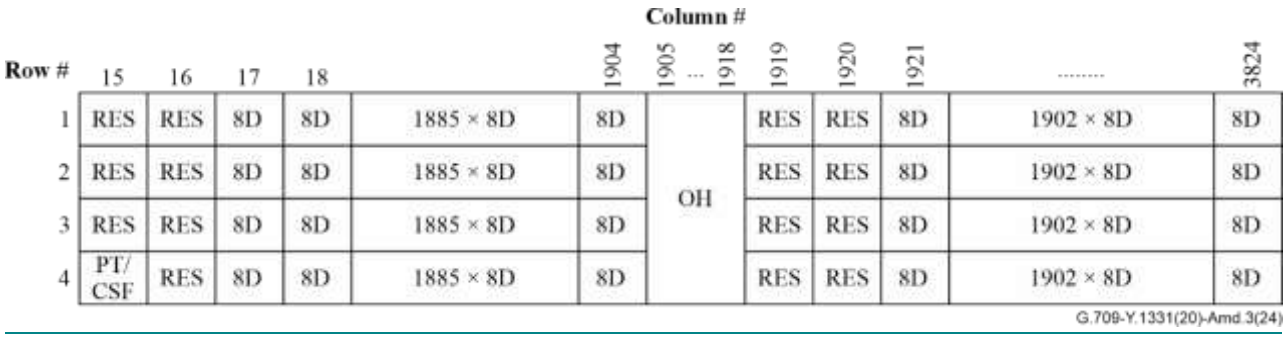


Figure M.8 – Mapping of 2 147 483 647-bit pseudo-random test sequence into fgOPUflex

The fgOPUflex overhead for the PRBS mapping consists of only a payload type (PT=0x3E). The fgOPUflex payload for the PRBS mapping consists of 4 × 3792 bytes.

M.5.2 Mapping of a packet client signal into fgOPUflex

A set of packet client signals with an aggregated bandwidth of less than or equal to 1 Gbit/s may be presented as a n x 10 Mbit/s stream of client packets and inter-packet gaps and then 64B/66B encoded as specified in [IEEE 802.3] Figure 82-5. The /S/ block is initially set to the Preamble value and the Start Frame Delimiter (SFD) as specified in clause 81.2.2 of [IEEE 802.3].

This 64B/66B encoded packet client signal is mapped into an fgOPUflex(p). It uses 64B/66B idle control block insert/delete to complete the rate adaptation between the 64B/66B encoded packet client and the fgOPUflex(p) payload capacity. After the rate adaptation and before mapping into fgOPUflex(p), a Fine Grain OTN Frame Check Sequence (fgOFCS) is appended for error marking. The polynomial of the fgOFCS CRC is the same polynomial as Ethernet ($G(x) = x^{32} + x^{26} + x^{23} + x^{22}$

+ x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1 as specified in clause 3.2.9 of [IEEE 802.3]), but in reverse order of transmission of the bits (from MSB to LSB) in each octet when calculating the fgOFCS, as shown in Figure M.9 below.

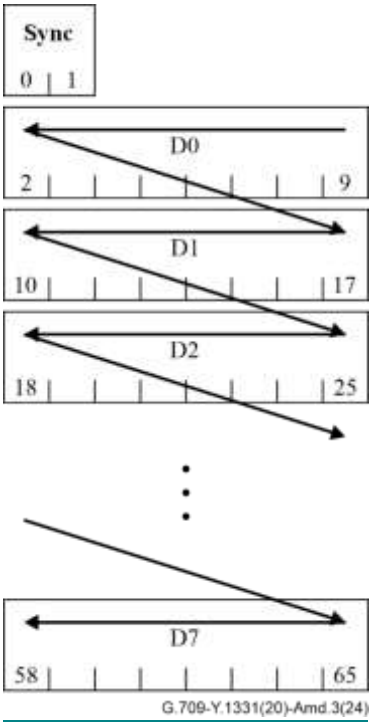


Figure M.9 – Order of 64B/66B block bits in fgOFCS calculation

The fgOFCS covers all the bits in the /D/ and /T/ blocks, excluding the Sync header. The fgOFCS checksum bits x^{31} to x^{0} are placed in bit positions 10 to 41, respectively, of the /I/ or /S/ 64B/66B block immediately following the /T/ block as illustrated in Figure M.10.

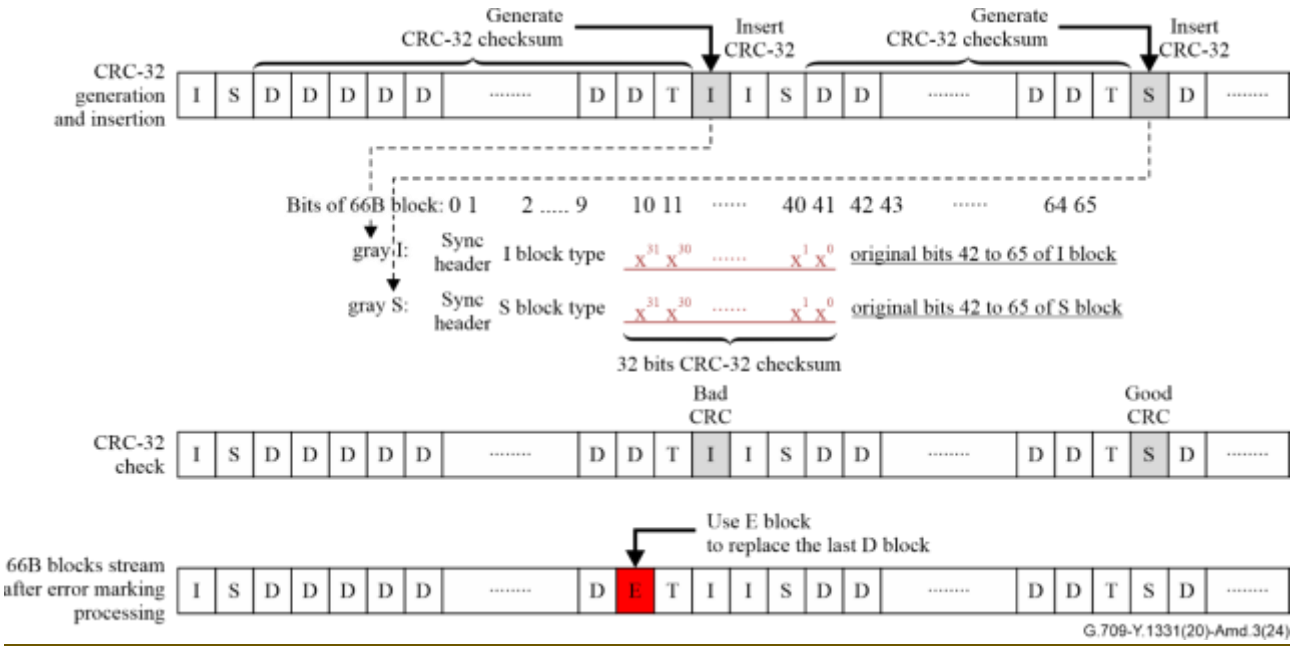


Figure M.10 – Illustration of error marking process

The 64B/66B blocks of the encoded packet client signal are aligned to the fgOPUflex payload area. The payload area of every 11 fgOPUflex frames carries 20224 64B/66B blocks as shown in Figure M.11. Thus, the 64B/66B blocks are aligned with fgODUflex over an 11-frame multiframe. The first bit of 64B/66B block #1 is mapped to the first payload bit of fgOPUflex frame #1 while the last bit of 64B/66B block #20224 is mapped to the last bit payload bit of fgOPUflex frame #11. The eleven fgOPUflex frames are identified by OMFI.

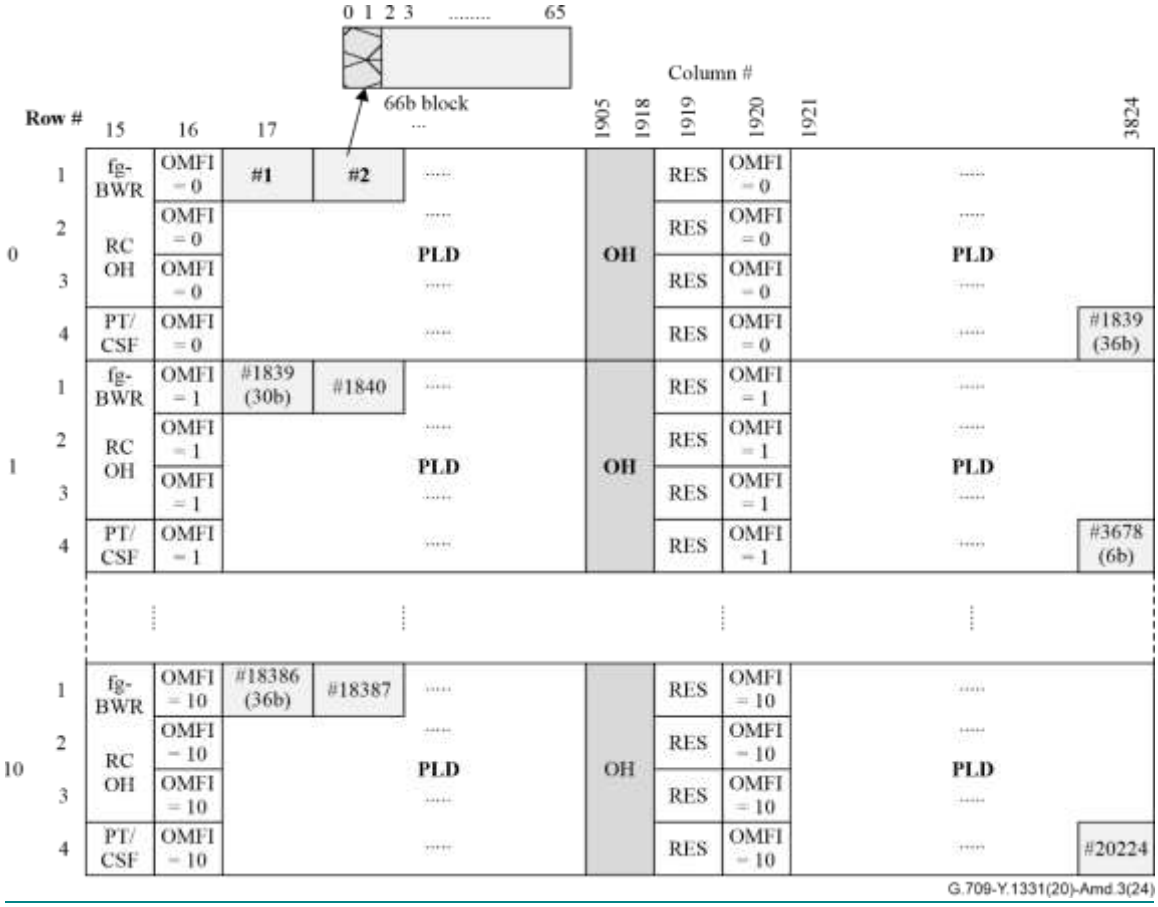


Figure M.11 – fgOPUflex frame structure for the mapping of a packet client signal

The bit position of the first 64B/66B block at the beginning of each of the eleven fgOPUflex frames is indicated by OMFI as illustrated in Table M.5.

Table M.5 – OMFI interpretation

OMFI	Bit position of the first 64B/66B block at the beginning of each fgOPUflex frame
0	0
1	36
2	6
3	42
4	12
5	48
6	18
7	54

Table M.5 – OMFI interpretation

OMFI	Bit position of the first 64B/66B block at the beginning of each fgOPUflex frame
8	24
9	60
10	30

The fgOFCS error checking, error marking and restoration of /I/ and /S/ are performed on the 64B/66B encoded packet client signal demapped from an fgOPUflex, before rate adaptation. When fgOFCS detects an error, the packet is marked as errored by replacing the final data (/D/) 64B66B block before the /T/ block with a 64B/66B Error Control block (/E/) as illustrated in Figure M.10. Bit positions 10 to 41 of the /I/ or /S/ 64B/66B block holding the fgOFCS checksum are restored to the idle control codes of an /I/ block or to the Preamble of an /S/ block, respectively.

The fgOPUflex(PKT) overhead for this mapping consists of a:

– payload type (PT=0x02) as specified in Table M.2;

– the client signal fail (CSF);

– the OPU multiframe identifier (OMFI) which is defined in rows 1 to 4 and columns 16 and 1920, and the value (0 to 10) of each OMFI byte will be incremented at each fgOPUflex frame and provide a 11-frame multiframe;

– fgBWR resizing control overhead (RCOH) which is used for the fgODUflex hitless bandwidth resizing.

During a signal fail condition of the incoming packet client signal (e.g., loss of input signal), the failed incoming client signal is replaced by a stream of 66B blocks, with each block carrying one local fault sequence ordered set (as specified in [IEEE 802.3]). This replacement signal is then mapped into the fgOPUflex.

During a signal fail condition of the incoming fgODUflex signal (e.g., fgODUflex-AIS, fgODUflex-LCK, or fgODUflex-OCI condition), a stream of 66B blocks, with each block carrying one local fault sequence ordered sets (as specified in [IEEE 802.3]) is generated as a replacement signal for the lost client signal.

Table M.6 shows some example ETH MAC frame clients and the payload capacity of the associated fgODUflex(p).

Table M.6 – sub1G MAC clients into fgODUflex(p)

ETH MAC frame client rate (kbit/s)	64B/66B encoded packet client rate (kbit/s)	64B/66B encoded packet client bit-rate tolerance (ppm)	fgODUflex(p) payload bit rate (kbit/s)
10 000	10 312.5	±100	10 322.097 (p=1)
50 000	51 562.5	±100	51 610.484 (p=5)
100 000	103 125	±100	103 220.968 (p=10)
1000 000	1031 250	±100	1032 209.679 (p=100)

M.5.3 Mapping of constant bit rate client signals into fgOPUflex

Mapping of CBR client signals into fgOPUflex(p) is performed by the generic mapping procedure (GMP) as specified in Annex D. Each CBR client is mapping into a corresponding fgODUflex signal.

Rows 1 and 2, and rows 3 and 4 of the fgOPUflex payload form two independent payload containers, respectively. A GMP C_m:C_{nD} rate report is provided for each 2-row payload container. A 2-row payload container consists of 2 × 3 792 bytes. Groups of 16 bytes in the container are numbered 1 to 474 as illustrated in Figure M.12. Thus, GMP P_{server} = 474. In the first row of the 2-row payload container the first 16-byte will be labelled 1, the next 16-bytes will be labelled 2, etc., which correspond to GMP index 'j'.

Groups of 128 successive bits of the CBR client signal are mapped into a group of 16 successive bytes of the 2-row payload container under control of the GMP data/stuff control mechanism. Each group of 16 bytes in the 2-row payload container may either carry 128 client bits, or 128 stuff bits. The stuff bits are set to zero.

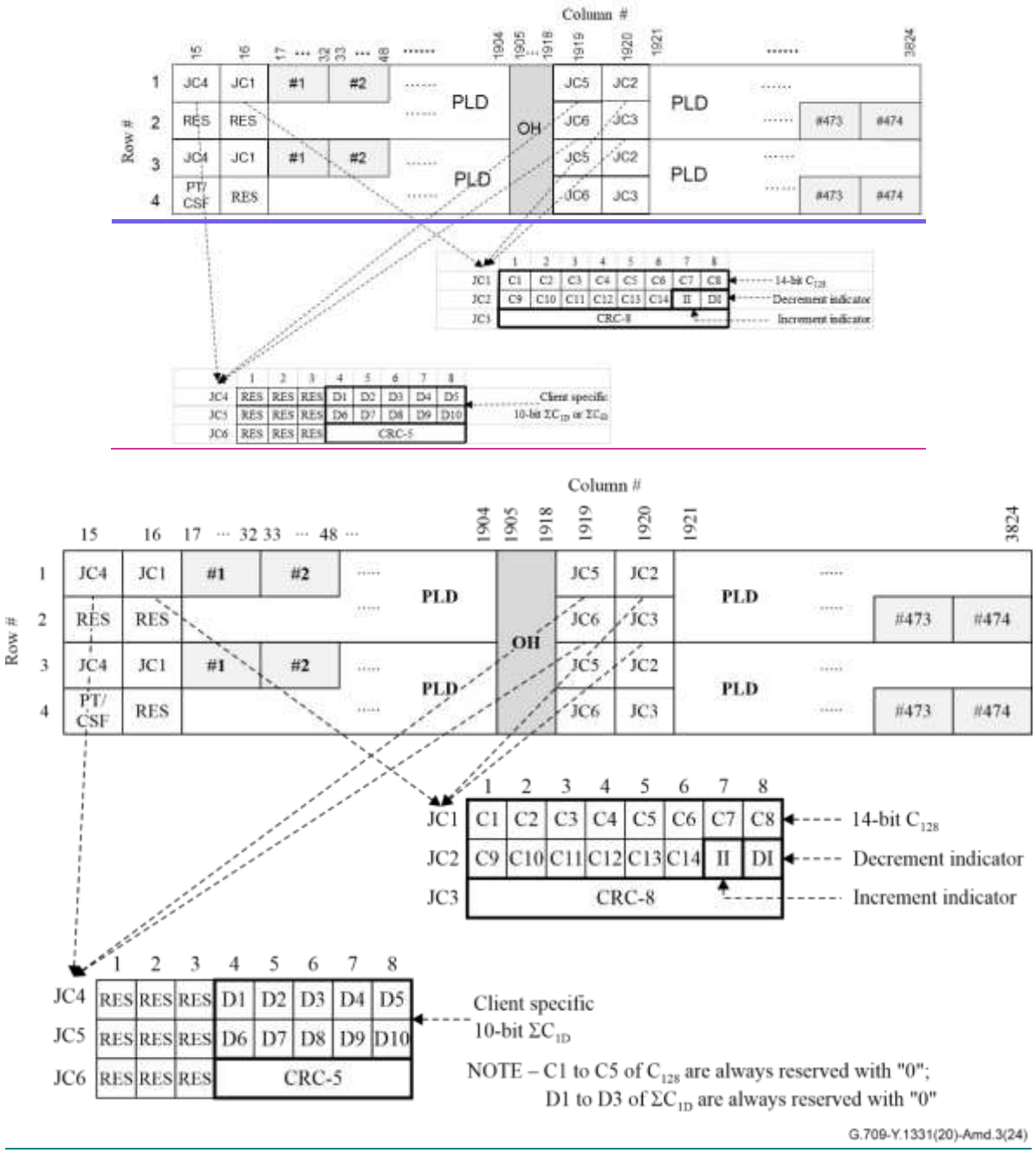


Figure M.12 – fgOPUflex frame structure for the mapping of a constant bit rate client signal

The fgOPUflex(CBR) overhead for this mapping consists of a:

– payload type (PT=0x03) as specified in Table M.2;

– the client signal fail (CSF);

– two groups of JC1 to JC6 bytes for C_m and C_{nD} as specified in Figure M.12, where each group of JC1 to JC6 bytes carries once C_m and C_nD, and the JC1, JC2 and JC3 are used for Cm overhead information, and the bits 4 to 8 of JC4, JC5 and JC6 are used for C_nD overhead information.

Table M.7 specifies parameters m, and n in GMP C_m:C_nD overhead for CBR clients STM-1 and STM-4. Other CBR clients may be defined in the future. In all cases, the granularity (m) of the GMP Stuff and Data words will be 128 bits (16 bytes). The granularity (n) of C_nD is client dependent. Table M.8 specifies the replacement signals for the STM-1 and STM-4 clients. Table M.9 specifies the fgOPUflex(p) payload capacity and their $C_{128,min}$ and $C_{128,max}$ values for these clients.

During a signal fail condition of the incoming CBR client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in Table M.8.

During a signal fail condition of the incoming fgODUflex signal (e.g., in the case of an fgODUflex-AIS, fgODUflex-LCK, fgODUflex-OCI condition), the failed client signal is replaced by the appropriate replacement signal as defined in Table M.8.

Table M.7 – GMP parameters m, n and C_{nD} for CBR clients into fgOPUflex

Client signal	Nominal bit rate (kbit/s)	Bit rate tolerance (ppm)	m	n	C_{nD} Active
STM-1	155 520	±20	128	1	Yes
STM-4	622 080	±20	128	1	Yes

Table M.8 – Replacement signal for CBR clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
STM-1	Generic-AIS	±20
STM-4	Generic-AIS	±20

Table M.9 – $C_{128,min}$ and $C_{128,max}$ for CBR clients into fgOPUflex(p)

CBR client signal	fgOPUflex(p) payload bit rate (kbit/s)	$C_{128,min}$	$C_{128,max}$
STM-1	165 153.549 (p=16)	446	447
STM-4	629 647.904 (p=61)	468	469

M.5.4 Mapping of packet oriented and PDH clients into fgOPUflex using SDH Structures

Packet and PDH clients can be mapped into fgOPUflex using SDH VC-n structures.

M.5.4.1 Mapping of VC-n client signals into fgOPUflex

The mapping of VC-n clients into fgOPUflex is based on the source and sink NE of the VC-n client share a common synchronization source. Note that in case the VC-n is retimed to a TU[G]-n using this common synchronization clock at the source/sink NE or for VC-n carrying packet clients, the

jitter and wander performance of the mapping process and the end-to-end transport process is not tightly constrained.

The VC-n client signals include v_1×VC-12, v_2×VC-3 and VC-4, where v_1=1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 or 60, and v_2=1 or 2. At the ingress STM-N client interfaces of the mapping NE, the VC-n clients do not need to be extracted from the terminated STM-N. The TU[G]-n and AU-n is extracted from the STM-N and then mapped into a fgOPUflex, without SDH pointer processing. The v_1×TU-12, or v_2×TUG-3 are byte interleaved to form a single merged signal, which is then mapped into an fgOPUflex using generic mapping procedure (GMP) as specified in Annex D. The mapping process as specified in clause M.5.3, except C_{nD}, is not required. The v_1×TU-12 signals or v_2×TUG-3 signals are extracted from a single HO-VC, as a result the v_1×TU-12 signals or v_2×TUG-3 signals be multiframe and frame aligned to each other.

At the de-mapping NE, the v_1×TU-12, or v_2×TUG-3, or AU-4 are extracted from the fgOPUflex. The v_1×VC-12, or v_2×VC-3, or VC-4 are then extracted from v_1×TU-12, or v_2×TUG-3, or AU-4, and then the v_1×VC-12, or v_2×VC-3, or VC-4 are mapped to new v_1×TU-12, or v_2×TUG-3, or AU-4 which are multiplexed into new egress STM-N client interfaces. The new TU-12/TUG-3/AU-4 and egress STM-N are generated from the local reference clock of de-mapping NE which is connected to the common synchronization source as the mapping NE, as mentioned above.

– A 1×TU-12 frame, carrying a single VC-12 client, is aligned to the 16-byte block boundary of a fgODUflex(1) payload frame;

– A v_1×TU-12 byte-interleaved merged frame (v_1>1), carrying v_1×VC-12 clients, is aligned to the 16-byte block boundary of a fgODUflex(p) payload frame, where the specific values of p are described in Table M.12;

– A multiframe of 8 consecutive TUG-3 frames carrying a single VC-3 client, is aligned to the 16-byte block boundary of a fgODUflex(5) payload frame;

– A multiframe of 4 consecutive 2×TUG-3 byte-interleaved merged frames carrying 2×VC-3 clients, is aligned to the 16-byte block boundary of a fgODUflex(10) payload frame;

– A multiframe of 8 consecutive AU-4 frames carrying a single VC-4 client is aligned to the 16-byte block boundary of the fgODUflex(16) payload frame.

NOTE 1 – For VC-4 clients, fgOPUflex(15) is defined to only support a single VC-4. VC-4-xc support is for further study.

The fgOPUflex(VC-n) overhead for this mapping consists of a:

– payload type (PT=0x05, 0x07, and 0x08) as specified in Table M.2;

– the client signal fail (CSF);

– two groups of JC1 to JC6 bytes for GMP C_m and C_{nD} are specified in Figure M.13, where each group of JC1 to JC6 bytes carries one GMP C_m and C_{nD}. The JC1, JC2 and JC3 bytes are used for C_m overhead information, while the bits 4 to 8 of the JC4, JC5 and JC6 bytes are reserved.

– two 9-bit client frame start (CFS) overheads are defined in row 1 columns 14 (bits 3 to 8) and 15 (bits 1 to 3) and in row 3 columns 14 (bits 3 to 8) and 15 (bits 1 to 3), respectively. The CFS in row 1 indicates the number of Data blocks starting at Block #1 in Row 1 to the block ahead of the first new client frame start in the associated 2-row payload container. The CFS in row 3 indicates the number of Data blocks starting at Block #1 in Row 3 to the block ahead of the first new client frame start. As illustrated in Figure M.13, CFS indicates the number of Data blocks from Block #1 to Block #M-1. CFS is a binary number with the range of 0 to C_m-1. CFS=0 indicates that there is no Data block ahead of the first new client frame start in the current 2-row payload container. The first new client frame start is located at the first Data block. CFS= C_m-1 indicates that the first new client frame start is located at the last Data

block of the current 2-row payload container. If there is no client frame start in the associated 2-rows payload container, CFS is set to 0x1FF.

NOTE 2 – the client frame is TU-12 frame, v_1×TU-12 byte-interleaved frame, consecutive 8 TUG-3 frame multiframe, consecutive 4 2×TUG-3 byte-interleaved multiframe, or consecutive 8 AU-4 frame multiframe. The client frame start is the first H1 byte for TUG-3, AU-4, or the first V1 byte for TU-12.

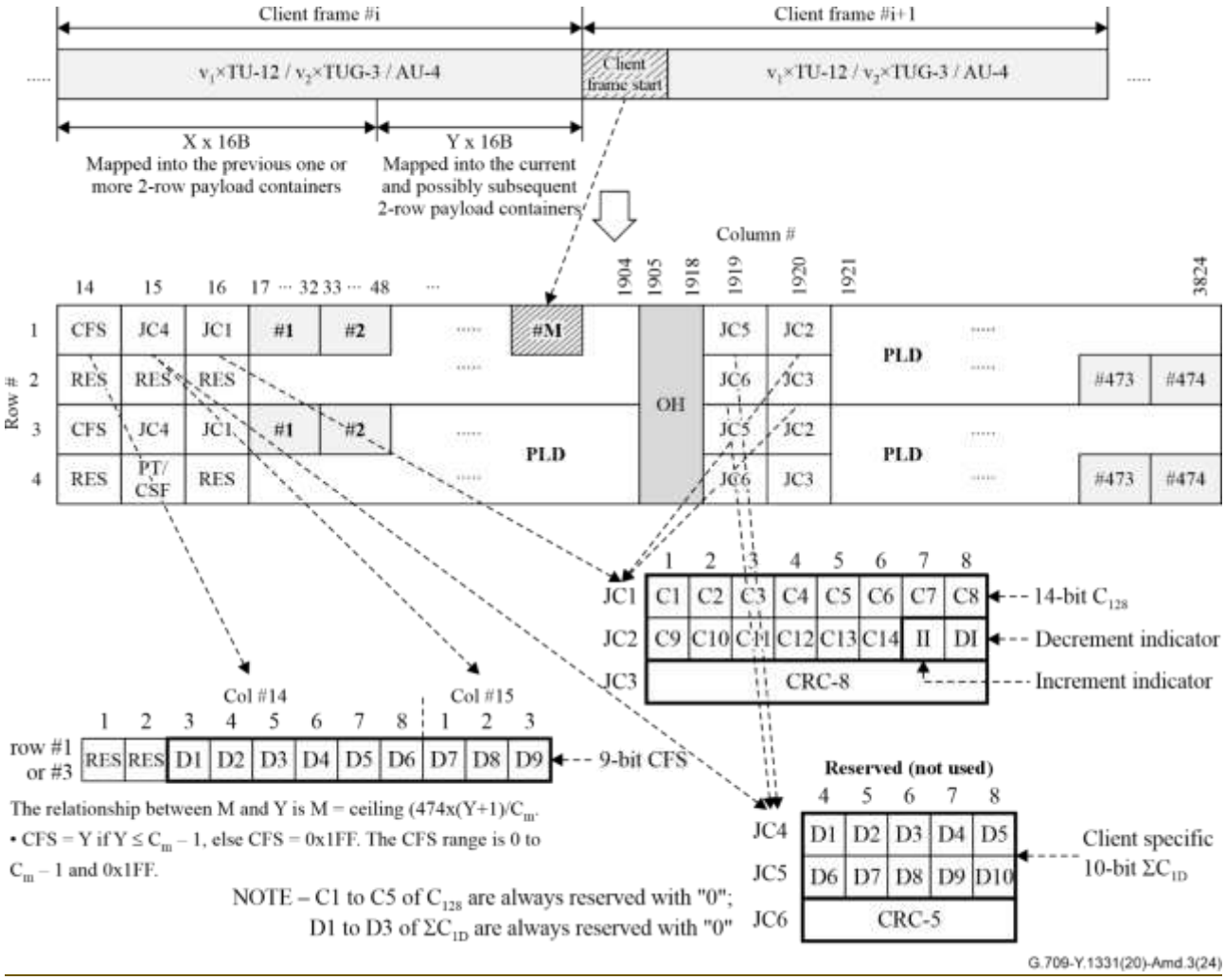


Figure M.13 – fgOPUflex frame structure mapping VC-n client signals

Table M.10 specifies parameters m and n in GMP C_m:C_{nD} overhead for SDH Virtual Container (VC) clients VC-12, VC-3 and VC-4. Other SDH VC clients may be defined in the future. In all cases, the granularity (m) of the GMP Stuff and Data words will be 128 bits (16 bytes). Table M.11 specifies the replacement signals for the SDH VC clients. Table M.12 specifies the fgOPUflex(p) payload capacity and their $C_{128,min}$ and $C_{128,max}$ values for these clients.

During a signal fail condition of the incoming SDH VC client signal (e.g., in the case of a loss of input signal), this failed incoming signal is replaced by the appropriate replacement signal as defined in Table M.11.

During a signal fail condition of the incoming fgODUflex signal (e.g., in the case of an fgODUflex-AIS, fgODUflex-LCK, fgODUflex-OCI condition), the affected SDH VC client signal is replaced by the appropriate replacement signal as defined in Table M.11.

Table M.10 – m, n and C_{nD} for SDH VC-n clients into fgOPUflex

Client signal	Nominal bit rate (kbit/s)	Bit rate tolerance (ppm)	m	n	C_{nD} Active
TU-12 (VC-12/E1)	2 304	±20	128	N/A	No
TUG-3 (VC-3)	49 536	±20	128	N/A	No
AU-4 (VC-4)	150 912	±20	128	N/A	No

Table M.11 – Replacement signal for SDH VC-n clients

Client signal	Replacement signal	Bit-rate tolerance (ppm)
VC-12 (TU-12)	TU-AIS	±20
VC-3 (TUG-3)	TU-AIS	±20
VC-4 (AU-4)	AU-AIS	±20
E1 (VC-12/TU-12)	TU-AIS	±20

Table M.12 – $C_{128,min}$ and $C_{128,max}$ for SDH VC-n clients into fgOPUflex(p)

client signals	fgOPUflex(p) payload bit rate (kbit/s)	$C_{128,min}$	$C_{128,max}$
VC-12 (TU-12)	10 322.097 (p=1)	105	106
5×VC-12 (5×TU-12)	20 644.194 (p=2)	264	265
10×VC-12 (10×TU-12)	30 966.29 (p=3)	352	353
15×VC-12 (15×TU-12)	41 288.387 (p=4)	396	397
20×VC-12 (20×TU-12)	51 610.484 (p=5)	423	424
25×VC-12 (25×TU-12)	61 932.581 (p=6)	440	441
30×VC-12 (30×TU-12)	72 254.678 (p=7)	453	454
35×VC-12 (35×TU-12)	82 576.774 (p=8)	462	463
40×VC-12 (40×TU-12)	92 898.871 (p=9)	470	471
45×VC-12 (45×TU-12)	113 543.065 (p=11)	432	433
50×VC-12 (50×TU-12)	123 865.162 (p=12)	440	441
55×VC-12 (55×TU-12)	134 187.258 (p=13)	447	448
60×VC-12 (60×TU-12)	144 509.355 (p=14)	453	454
VC-3 (TUG-3)	51 610.484 (p=5)	454	455
2×VC-3 (2×TUG-3)	103 220.968 (p=10)	454	455
VC-4 (AU-4)	154 831.452 (p=15)	461	463
E1 (VC-12/TU-12)	10 322.097 (p=1)	105	106

M.5.4.2 Mapping of PDH E1 client signals into fgOPUflex

In the case of an E1 signal delivered via an STM-N client interface of an fgOTN mapping and de-mapping NE, the mapping process follows that of mapping/demapping VC-12 specified in clause M.5.4.1. The TU-12 is extracted from the STM-N and GMP mapped into an fgOPUflex without active C_{nD}. The mapping process in the fgOTN NE does not distinguish the VC-12 client carrying an E1 from packet oriented VC-12 clients.

In the case of E1 client interface directly connected to the mapping and de-mapping NEs, the E1 client signal is asynchronously mapped into VC-12 as specified in [ITU-T G.707], and then the resulting VC-12 is synchronously mapped into TU-12. This TU-12 is synchronously mapped to fgOPUflex(1) via GMP as described in clause M.5.4.1. The VC-12, TU-12 and fgOPUflex(1) are generated from and are phase locked to local reference clock of the NE. The ratio of the bit rate of the fgODUflex(1) to that of the TU-12 is given by:

Bit rate fgODUflex(1) : Bit Rate TU-12 (M-2)
= 1.24416Gb/s / 119.525 : 144 × 8 × 2000
= 540 : 119.525

The number of 16-byte blocks of the 2-row payload container of an fgODUflex(1) occupied by a phase locked TU-12 can be expressed precisely by:

16-byte fgODUflex(1) blocks occupied (M-3)
= Frame Period ODUflex(1) / 2 × (144 × 8 × 2000) / 128
= 239 × 119.525 / 270
= 105.80176

As the precise occupancy of the fgODUflex payload is known, GMP C_{nD} is not needed.

The fgOPUflex(E1) overhead for this mapping consists of a:

– payload type (PT=0x09) as specified in Table M.2;

– the client signal fail (CSF);

– two groups of JC1 to JC6 bytes for C_m and C_{nD} as specified in Figure M.13 which keep the same specification as specified in clause M.5.4.1;

– two 9-bit client frame start (CFS) overheads separately defined in row 1 columns 14 (bits 3 to 8) and 15 (bits 1 to 3) and in row 3 columns 14 (bits 3 to 8) and 15 (bits 1 to 3) which keep the same specification as specified in clause M.5.4.1.

Annex N

Mapping fgODUflex signals into the fgODTU signal and the fgODTU into the OPU fine grain tributary slots

(This annex forms an integral part of this Recommendation.)

This annex specifies the multiplexing of fgODUflex signals into OPU. The server ODU includes ODU0(fgTS), ODU1(fgTS), ODU2(fgTS), and ODUflex(fgTS, n); n = 3 to 7. The multiplexing is performed in two steps:

1) Asynchronous mapping of fgODUflex into a fine grain optical data tributary unit (fgODTU) using fgGMP;

2) 16-byte-synchronous mapping of an fgODTU into one or more OPU fine grain tributary slots (fgTS).

N.1 Fine grain tributary slot number and multiplex structure of OPU

The OPU0(fgTS) is divided into 119 10M fine grain tributary slots (fgTS); the OPU1(fgTS) is divided into 2×119 10M fine grain tributary slots; the OPU2(fgTS) is divided into 8×119 10M fine grain tributary slots; the OPUflex(fgTS, n) is divided into n×119 10M fine grain tributary slots. The fine grain tributary slots are 16-byte interleaved within the OPU payload. A fine grain tributary slot includes a part of the OPU OH area and a part of the OPU payload area. The fgODUflex frame is mapped in 16-byte blocks into the fgODTU payload area and the fgODTU bytes are mapped into one or more OPU fine grain tributary slots. The bytes of the fgODTU justification overhead are mapped into the OPU OH area.

Figure N.1 presents the fine grain tributary slot allocation in an OPU0(fgTS). The OPUk(fgTS)/OPUflex(fgTS, n) is arranged into a fgTSOH multiframe. The frames within the fgTSOH multiframe are grouped in sets of 32 frames. The fgTSOH multiframe in an OPU0 consists of 1 set × 32 OPU0 frames and is divided into 119 fine grain tributary slots numbered 1 to 119. The 119 fgTSs are 16-byte interleaved in the OPU0 payload area. An OPU0 fine grain tributary slot consists of 256 16-byte blocks plus a 12-bit fine grain tributary slot overhead (fgTSOH). OPU0 frames #1-#29 of the set carry 4 fgTSOHs, which are associated with consecutive fgTSs. OPU0 frame #30 carries the remaining 3 fgTSOHs. OPU0 frame #31-32 do not carry any fgTSOH. The set of 32 OPU0 frames in fgTSOH multiframe carries a total of 119 fgTSOHs, each associated with a consecutive fgTS. For example, fgTSOH1 to fgTSOH4 are located in OPU0 Frame #1 and are associated with fgTS #1 to fgTS #4, respectively. Similarly, fgTSOH5 to fgTSOH8 are located in OPU0 Frame #2 and are associated with fgTS #5 to fgTS #8, respectively. Finally, fgTSOH117 to fgTSOH119 are located in OPU0 Frame #30 and are associated with fgTS #117 to fgTS #119, respectively. The fgTSOH of each fine grain tributary slot is available once per fgTSOH multiframe (32 OPU0 frames). The period of an fgTSOH multiframe in an OPU0 is 32 × 98.354 µs = 3.147 ms (Table 7-4). The 32 OPU0 frames in the fgTSOH multiframe structure is indexed by MFAS[4:8] as shown in Figure N.1.

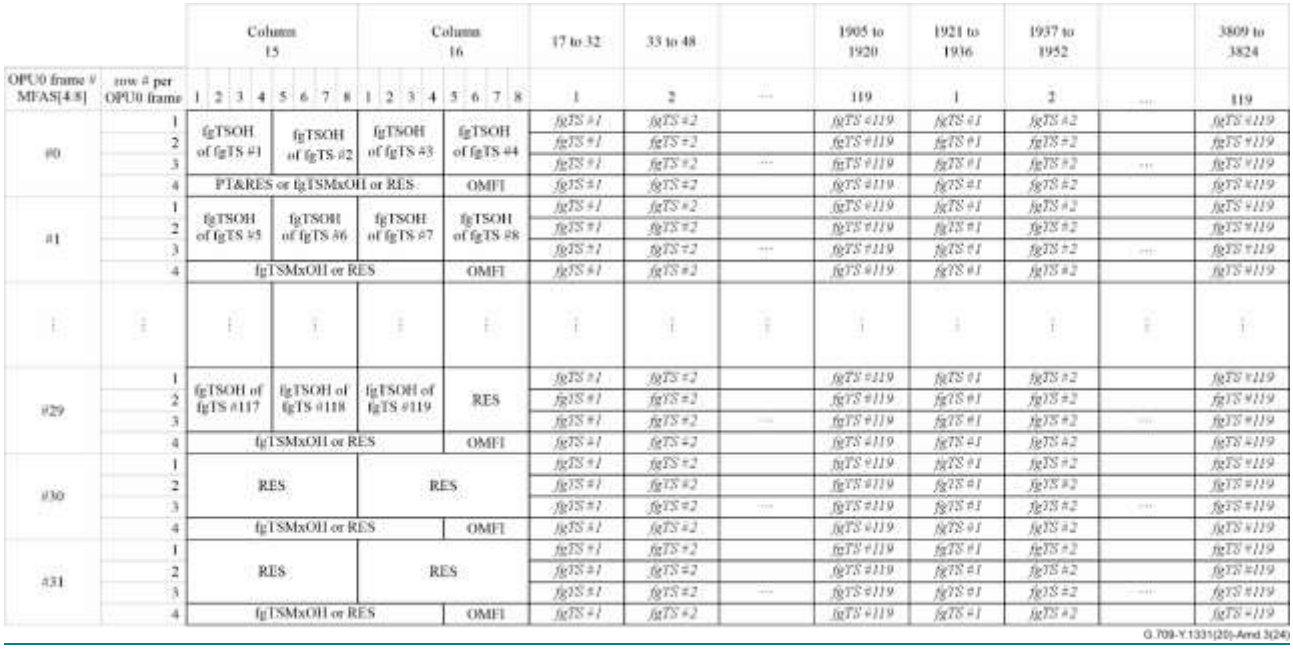


Figure N.1 – OPU0 fine grain tributary slot allocation

The allocation fine grain tributary slots and structure of the fgTSOH multiframe for OPU1(fgTS) is very similar to that of OPU0(fgTS). The fgTSOH multiframe in an OPU1 consists of 2 sets × 32 OPU1 frames and is divided into 2×119 fine grain tributary slots numbered 1 to 238. The 238 fgTSs are 16-byte interleaved in the OPU1 payload area. An OPU1 fine grain tributary slot consists of 256 16-byte blocks plus a 12-bit fine grain tributary slot overhead (fgTSOH). Within each set of 32 OPU1 frames in the fgTSOH multiframe, OPU1 frames #1-#29 carry 4 fgTSOHs, which are associated with consecutive fgTSs. OPU1 frame #30 carries the remaining 3 fgTSOHs. OPU1 frame #31-32 do not carry any fgTSOH. Each set of 32 OPU1 frames in fgTSOH multiframe carries a total of 119 fgTSOHs, each associated with a consecutive fgTS. There are 2 sets of 32 OPU1 frames in an OPU1 fgTSOH multiframe. The fgTSOH of each fine grain tributary slot is available once per fgTSOH multiframe (2×32=64 OPU1 frames). The period of an fgTSOH multiframe in an OPU1 is 64 × 48.971 µs = 3.134 ms (Table 7-4). The 64 OPU1 frames in the fgTSOH multiframe structure is indexed by MFAS[3:8].

The allocation fine grain tributary slots and structure of the fgTSOH multiframe for OPU2(fgTS) is very similar to that of OPU0(fgTS). The fgTSOH multiframe in an OPU2 consists of 8 sets × 32 OPU2 frames and is divided into 8×119 fine grain tributary slots numbered 1 to 952. The 952 fgTSs are 16-byte interleaved in the OPU2 payload area. An OPU2 fine grain tributary slot consists of 256 16-byte blocks plus a 12-bit fine grain tributary slot overhead (fgTSOH). Within each set of 32 OPU2 frames in the fgTSOH multiframe, OPU2 frames #1-#29 carry 4 fgTSOHs, which are associated with consecutive fgTSs. OPU2 frame #30 carries the remaining 3 fgTSOHs. OPU2 frames #31-32 do not carry any fgTSOH. Each set of 32 OPU2 frames in fgTSOH multiframe carries a total of 119 fgTSOHs, each associated with a consecutive fgTS. There are 8 sets of 32 OPU2 frames in an OPU2 fgTSOH multiframe. The fgTSOH of each fine grain tributary slot is available once per fgTSOH multiframe (8×32=256 OPU2 frames). The period of an fgTSOH multiframe in an OPU2 is 256 × 12.191µs= 3.121 ms (Table 7-4). The 256 OPU2 frames in the fgTSOH multiframe structure is indexed by MFAS [1:8].

Figure N.2 presents the fine grain tributary slot allocation in an OPUflex(fgTS, n). The fgTSOH multiframe in an OPUflex(fgTS,n) consists of n sets × 32 OPUflex frames and is divided into n×119 fine grain tributary slots numbered 1 to n×119. The n×119 fgTSs are 16-byte interleaved in the OPUflex payload area. An OPUflex fine grain tributary slot consists of 256 16-byte blocks plus a 12-bit fine grain tributary slot overhead (fgTSOH). Within each set of 32 OPUflex frames in the fgTSOH

multiframe, OPUflex frames #1-#29 of the set carry 4 fgTSOHs, which are associated with consecutive fgTSs. OPUflex frame #30 carries the remaining 3 fgTSOHs. OPUflex frames #31-32 do not carry any fgTSOH. Each set of 32 OPUflex frames in fgTSOH multiframe carries a total of 119 fgTSOHs, each associated with a consecutive fgTS. There are n sets of 32 OPU2 frames in an OPUflex(fgTS,n) fgTSOH multiframe. The fgTSOH of each fine grain tributary slot is available once per fgTSOH multiframe (n×32 OPUflex(fgTS,n) frames). The period of an fgTSOH multiframe in an OPUflex(fgTS,n) is 32 × n × (98.354 μs / n) = 3.147 ms (Table 7-4). The n×32 OPUflex frames in the fgTSOH multiframe structure is indexed by MFAS[4:8] and OMFI. The specification of OMFI is described in clause N.3. As shown in Table N.1, the OPUflex frame number is given by:

$$\text{OPUflex Frame \#} = ((8\times \text{OMFI} + \text{MFAS}[1:3]) \bmod n) \times 32 + \text{MFAS}[4:8] \qquad \text{(N-1)}$$

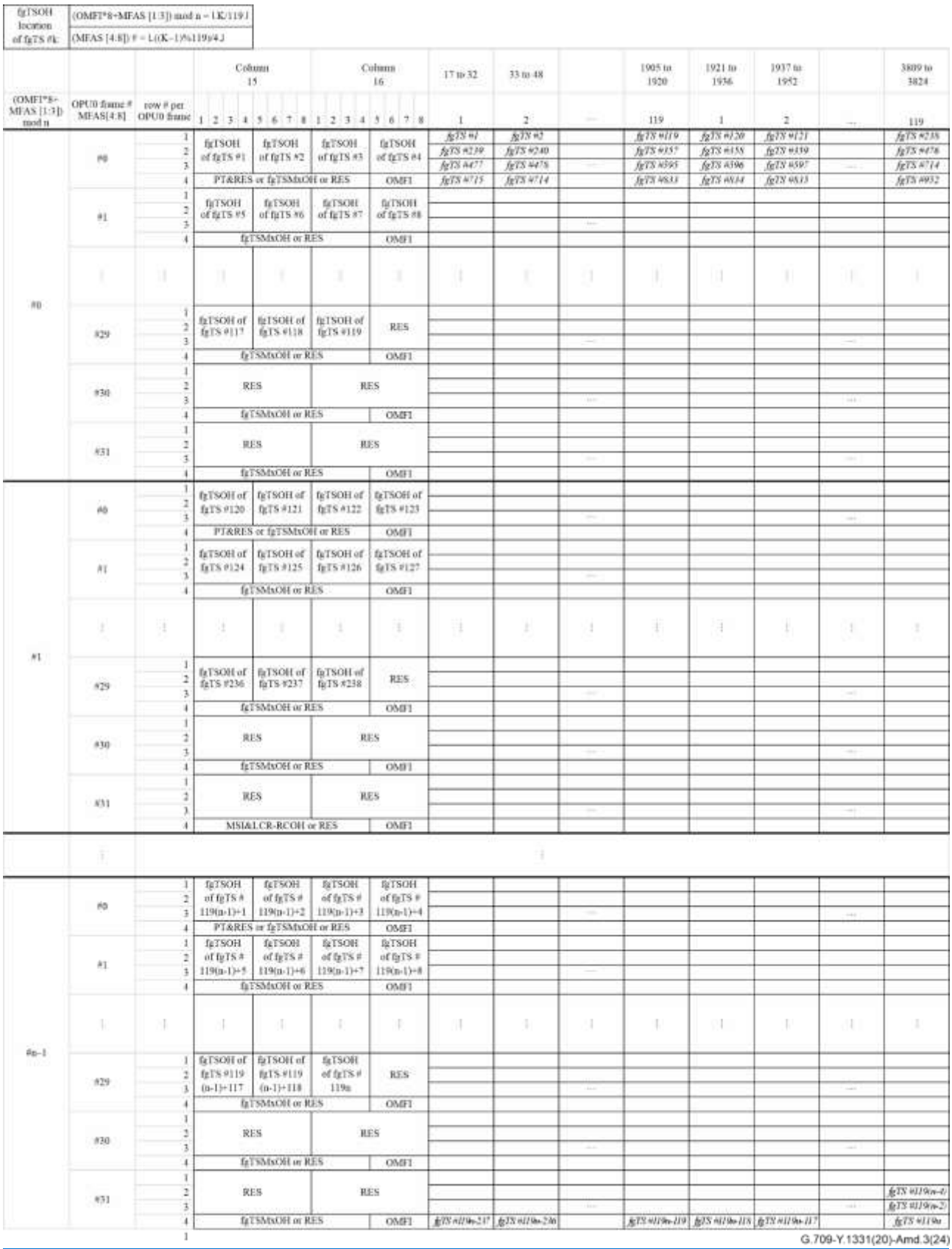


Figure N.2 – OPUflex(fgTS, n) fine grain tributary slot allocation

Table N.1 – OPUflex(fgTS, n) fine grain tributary slot OH allocation

(8*OMFI + MFAS[1:3]) mod n	MFAS bits 4 to 8	fgTSOH of fgTS #
0	0	1, 2, 3, 4
	1	5, 6, 7, 8
	……	……
	28	113, 114, 115, 116
	29	117, 118, 119, RES
	30	RES
	31	RES
1	0	120, 121, 122, 123
	1	124, 125, 126, 127
	……	……
	28	232, 233, 234, 235
	29	236, 237, 238, RES
	30	RES
	31	RES
……	……	……
n-1	0	119(n-1)+1, 119(n-1)+2, 119(n-1)+3, 119(n-1)+4
	1	119(n-1)+5, 119(n-1)+6, 119(n-1)+7, 119(n-1)+8
	……	……
	28	119(n-1)+113, 119(n-1)+114, 119(n-1)+115, 119(n-1)+116
	29	119(n-1)+117, 119(n-1)+118, 119n, RES
	30	RES
	31	RES

N.2 fgODTU.M structure and multiplexing fgODTU.M signals into the OPU

fgODTU.M structure

The fine grain optical data tributary unit M (fgODTU.M) is a structure which consists of an fgODTU.M payload area and an fgODTU.M overhead area (see Figure N.3). The fgODTU.M payload area has 128 rows and 2×M 16-byte columns and consists of 256×M 16-byte payload blocks. The 256×M 16-byte payload blocks in the fgODTU.M payload area are numbered from 1 to 256M. The fgODTU.M overhead area has 12 bits. The fgODTU.M is carried in "M" fine grain tributary slots of an OPUk(fgTS)/OPUflex(fgTS,n).

The location of the fgODTU.M overhead depends on the OPU fine grain tributary slot used when multiplexing the fgODTU.M in the OPU. The fgODTU.M overhead is located in the OPU fgTSOH of the last OPU fine grain tributary slot allocated to the fgODTU.M.

The fgODTU.M overhead carries the fgGMP justification overhead (C_mT) as specified in clause N.3.

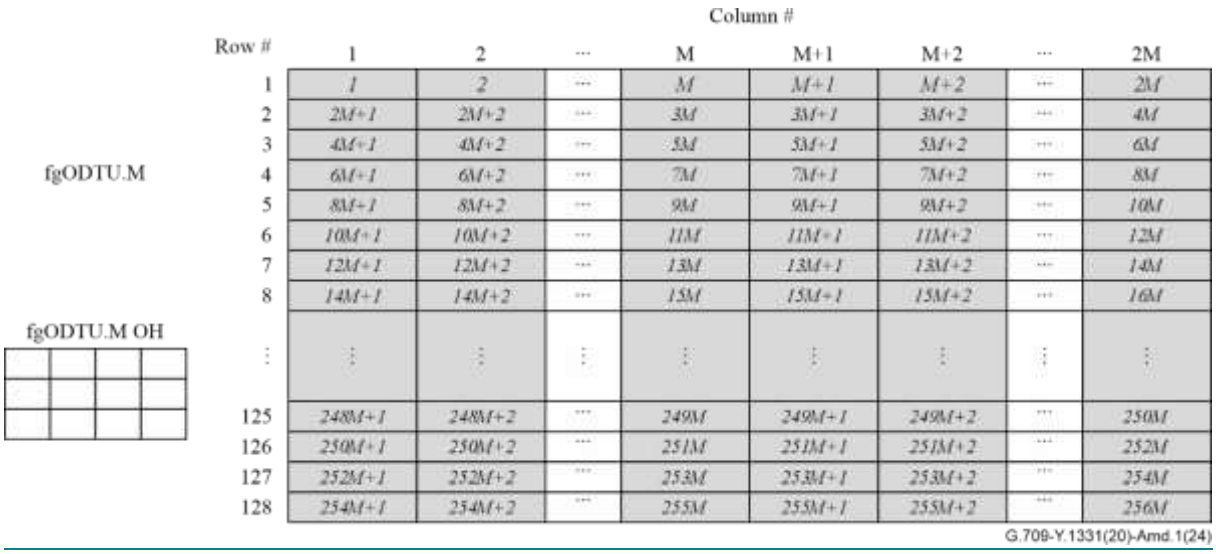


Figure N.3 – fgODTU.M frame format

Multiplexing fgODTU.M signals into the OPU

A 16-byte block of the fgODTU.M payload signal is mapped into a 16-byte of an OPU fgTS #i (i = $fgTS_1$, ..., $fgTS_M$) payload area. The 256×M 16-byte payload blocks of the fgODTU.M payload signal are mapped into 256×M 16-byte of M fgTSs (e.g., $fgTS_1$, ..., $fgTS_M$) of n×32-frame OPU multiframe as indicated in Figure N.2.

The 12 bits overhead of the fgODTU.M is mapped into the OPU fgTSOH of the last OPU fine grain tributary slot allocated to the fgODTU.M.

N.3 OPU multiplexing overhead and fgODTU justification overhead

The OPU fine grain Tributary Slot multiplexing overhead (fgTSMxOH) consists of the multiplexing structure identifier (MSI) and the fine grain link connection resizing control overhead (fgLCR RCOH) and is carried in a fgTSMxOH multiframe. Each fgTSMxOH is associated with a single fgTS and is transmitted over four OPUk(fgTS)/OPUflex(fgTS,n) frames. Two fgTSMxOH are present simultaneously in Row 4, Columns 15 and 16 of the OPUk(fgTS)/OPUflex(fgTS,n). The association of fgTSMxOH to fgTS is indexed by MFAS and OMFI.

Fine Grain Tributary Slot Multiplexing Overhead (fgTSMxOH)

The format of the fgTSMxOH is shown in Figure N.4 below. The fgTSMxOH for two fgTS (fgTS #k and fgTS #k+1) are presented simultaneously. The fgTSMxOH for fgTS #k is located at Row 4, Column 15, Bits 1-6. The fgTSMxOH for fgTS #k+1 is located at Row 4, Column 15, Bits 7-8 and Column 16, Bits 1-4. The fgTS occupation bit (OCCU) indicates if the fine grain tributary slot is allocated or unallocated. The multiplexing structure identifier (MSI) overhead indicates the port number of the fgODTU.ts that is being transported in this fgTS; a flexible assignment of tributary port to fine grain tributary slots is possible. The fgODTU.ts tributary ports are numbered 1 to 952. The MSI is set to all-1s when the occupation bit has the value 0 (fine grain tributary slot is unallocated). The MSI also serves as the TPID OH in the fgLCR RCOH for hitless resizing to identify fgTS to be added or removed from the specified Port Identifier. The polynomial of the CRC-6 is $g(x) = x^6 + x^3 + x^2 + 1$ (see Annex D) and is computed over fgTSMxOH[1:18]. The remaining overhead fields in the fgLCR overhead are defined in Annex O.

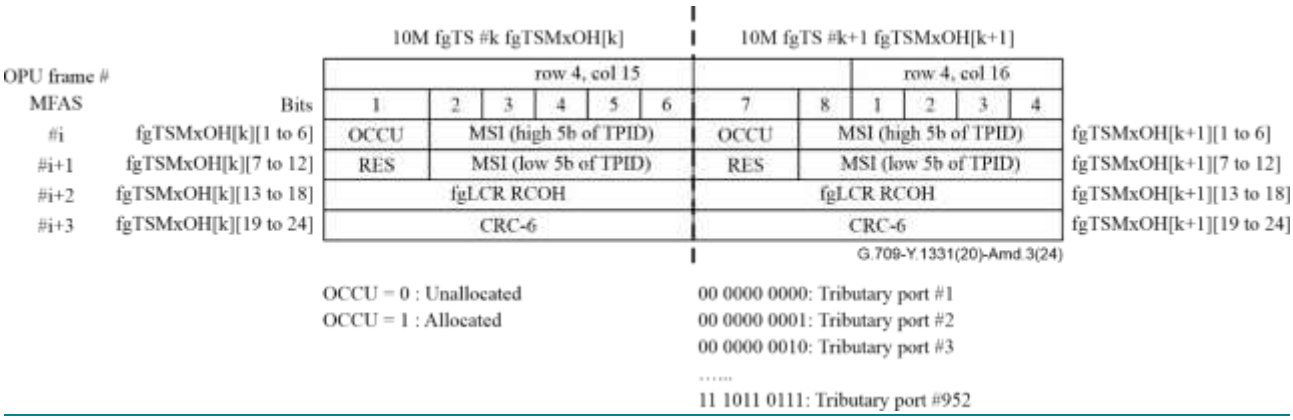


Figure N.4 – Fine grain tributary slot multiplexing overhead (fgTSMxOH)

Figure N.5 presents the organization of the fgTSMxOH in an OPU0(fgTS). The OPUk(fgTS)/OPUflex(fgTS, n) is arranged into a fgTSMxOH multiframe. The frames within the fgMUX multiframe are grouped in sets of 256 frames. The fgTSMxOH multiframe in an OPU0 consists of 1 set × 256 OPU0 frames and is divided into 119 fine grain tributary slot multiplexing overhead (fgTSMxOH) numbered 1 to 119. Each of the 119 fgTSMxOH is associated with a corresponding fgTS. An fgTSMxOH occupies 4 OPU frames. Two fgTSMxOH are presented in parallel in each OPU0 frame. OPU0 frames #4 to #7 of the set carry fgTSMxOHs for fgTS #1 and #2. OPU0 frame #240 to #243 carry the fgTSMxOHs for fgTS #119. OPU0 frames #0 to #3 and #244 to #255 do not carry any fgTSMxOH. The set of 256 OPU0 frames in the fgTSMxOH multiframe carries a total of 119 fgTSMxOHs, each associated with a consecutive fgTS. The fgTSMxOH of each fine grain tributary slot is available once per fgTSMxOH multiframe (256 OPU0 frames). The 256 OPU0 frames in the fgTSMxOH multiframe structure is indexed by MFAS as shown in Figure N.5. OMFI is not used to associate fgTSMxOH to fgTS.

MSI location of fgTS #k :	MFAS #i to #i+3 where i = 4*⌊(k%119+1)/2⌋		
OPU frame # MFAS	row 4, col 15		row 4, col 16
0	PT		OMFI (4b)
1	RES		OMFI (4b)
2			OMFI (4b)
3			OMFI (4b)
4	fgTS #1 fgTSMxOH	fgTS #2 fgTSMxOH	OMFI (4b)
5			OMFI (4b)
6			OMFI (4b)
7	CRC-6	CRC-6	OMFI (4b)
8	fgTS #3 fgTSMxOH	fgTS #4 fgTSMxOH	OMFI (4b)
9			OMFI (4b)
10			OMFI (4b)
11	CRC-6	CRC-6	OMFI (4b)
⋮	⋮	⋮	⋮
236	fgTS #117 fgTSMxOH	fgTS #118 fgTSMxOH	OMFI (4b)
237			OMFI (4b)
238			OMFI (4b)
239	CRC-6	CRC-6	OMFI (4b)
240	fgTS #119 fgTSMxOH	RES	OMFI (4b)
241			OMFI (4b)
242			OMFI (4b)
243	CRC-6		OMFI (4b)
244	RES		OMFI (4b)
245			OMFI (4b)
⋮			⋮
255			OMFI (4b)

G.709-Y.1331(20)-Amd.3(24)

Figure N.5 – Fine grain tributary slot multiplexing overhead (fgTSMxOH) in an OPU0(fgTS)

Figure N.6 presents the organization of the fgTSMxOH~~)~~ in an OPUflex(fgTS, n). The OPUflex(fgTS, n) is arranged into a fgTSMxOH multiframe. The frames within the fgTSMxOH multiframe are grouped in sets of 256 OPUflex frames. The sets (1 to n) are indexed by OMFI+1. The fgTSMxOH multiframe in an OPUflex(fgTS, n) consists of n × 256 OPUflex frames and is divided into n × 119 fgTSMxOH numbered 1 to n × 119. Each of the n × 119 fgTSMxOH is associated with a corresponding fgTS. An fgTSMxOH occupies 4 OPUflex frames. Two fgTSMxOH are presented in parallel in each OPUflex frame. OPUflex frames #4 to #7 of each set carry fgTSMxOHs for fgTS #(OMFI×119+1) and #(OMFI×119+2). OPUflex frame #240 to #243 of each set carry the fgTSMxOHs for fgTS #(OMFI×119+119). OPUflex frames #0 to #3 and #244 to #255 in each set do not carry any fgTSMxOH. The n × 256 OPUflex frames in the fgTSMxOH multiframe carries a total of n × 119 fgTSMxOHs, each associated with a consecutive fgTS. The fgTSMxOH of each fine grain tributary slot is available once per fgTSMxOH multiframe (n × 256 OPUflex frames). The n × 256 OPUflex frames in the fgTSMxOH multiframe structure are indexed by MFAS and OMFI as shown in Figure N.6.

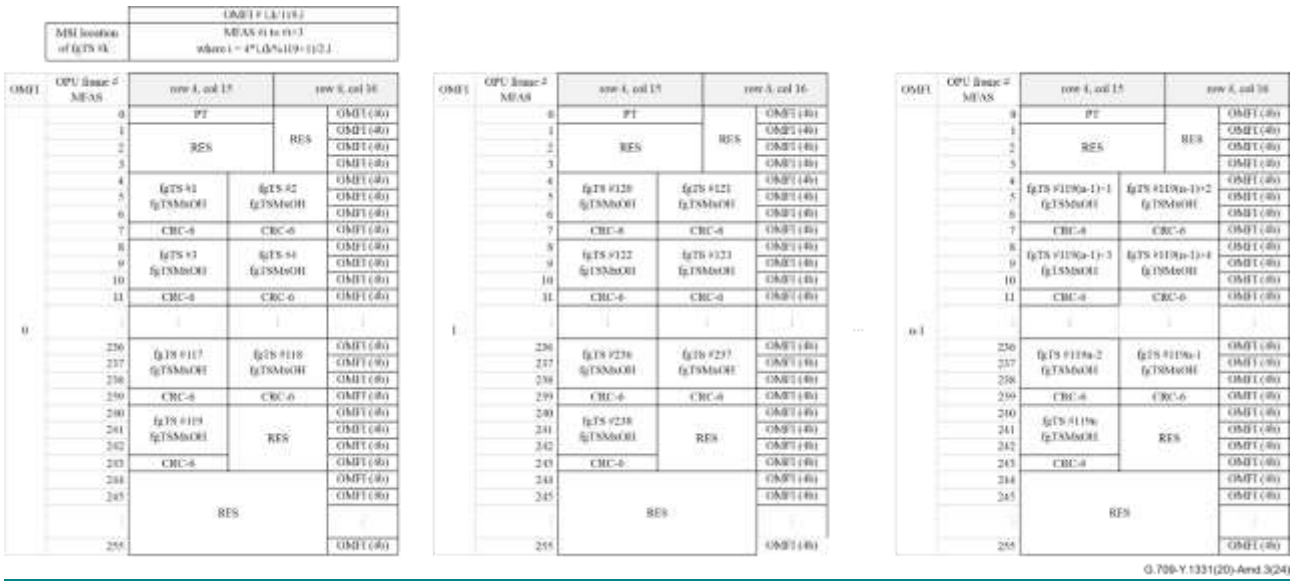


Figure N.6 – fine grain tributary slot multiplexing overhead (fgTSMxOH) in an OPUflex(fgTS, n)

The organization of the fgTSMxOH in an OPU1(fgTS) is very similar to that of an OPUflex(fgTS, n) with n set to n=2. The OPU1(fgTS) is arranged into a fgTSMxOH multiframe. The frames within the fgTSMxOH multiframe are grouped in sets of 256 OPU1 frames. The 2 sets are indexed by OMFI+1. The fgTSMxOH multiframe in an OPU1(fgTS) consists of 2 × 256 OPU1 frames and is divided into 2 × 119 fgTSMxOH numbered 1 to 2 × 119. The each of the 2 × 119 fgTSMxOH is associated with a corresponding fgTS. An fgTSMxOH occupies 4 OPU1 frames. Two fgTSMxOH are presented in parallel in each OPU1 frame. OPU1 frames #4 to #7 of each set carry fgTSMxOHs for fgTS #(OMFI×119+1) and #(OMFI×119+2). OPU1 frame #240 to #243 of each set carry the fgTSMxOHs for fgTS #(OMFI×119+119). OPU1 frames #0 to #3 and #244 to #255 in each set do not carry any fgTSMxOH. The 2 × 256 OPU1 frames in the fgTSMxOH multiframe carries a total of 2 × 119 fgTSMxOHs, each associated with a consecutive fgTS. The fgTSMxOH of each fine grain tributary slot is available once per fgTSMxOH multiframe (2 × 256 OPU1 frames). The 2 × 256 OPU1 frames in the fgTSMxOH multiframe structure is indexed by MFAS and OMFI as shown in Figure N.6 with n set to 2.

The organization of the fgTSMxOH in an OPU2(fgTS) is very similar to that of an OPUflex(fgTS, n) with n set to n=8. The OPU2(fgTS) is arranged into a fgTSMxOH multiframe. The frames within the fgTSMxOH multiframe are grouped in sets of 256 OPU2 frames. The 8 sets are indexed by OMFI+1. The fgTSMxOH multiframe in an OPU2(fgTS) consists of 8 × 256 OPU2 frames and is divided into 8 × 119 fgTSMxOH numbered 1 to 8 × 119. Each of the 8 × 119 fgTSMxOH is associated with a corresponding fgTS. An fgTSMxOH occupies 4 OPU2 frames. Two fgTSMxOH are presented in parallel in each OPU2 frame. OPU2 frames #4 to #7 of each set carry fgTSMxOHs for fgTS #(OMFI×119+1) and #(OMFI×119+2). OPU2 frame #240 to #243 of each set carry the fgTSMxOHs for fgTS #(OMFI×119+119). OPU2 frames #0 to #3 and #244 to #255 in each set do not carry any fgTSMxOH. The 8 × 256 OPU2 frames in the fgTSMxOH multiframe carries a total of 8 × 119 fgTSMxOHs, each associated with a consecutive fgTS. The fgTSMxOH of each fine grain tributary slot is available once per dgTSMxOH multiframe (8 × 256 OPU2 frames). The 8 × 256 OPU2 frames in the fgTSMxOH multiframe structure is indexed by MFAS and OMFI as shown in Figure N.6 with n set to 8.

OPU multiframe identifier (OMFI)

A 4-bit OPUflex(fgTS, n) multiframe identifier (OMFI) is defined in bits 5 to 8 of row 4 and column 16 as the description in Figure N.6. The value of the OMFI will be incremented every 256 OPUflex

frames. The range of OMFI value is 0 to n-1, which provides n × 256-frame fgTSMxOH multiframe. The OMFI for OPU0 is held constant at 0. The value of the OMFI will be incremented every 256 OPU1 and OPU2 frames. The range of OMFI value is 0 to 1 for an OPU1, which provides 2 × 256-frame fgTSMxOH multiframe. The range of OMFI value is 0 to 7 for an OPU2, which provides 8 × 256-frame fgTSMxOH multiframe.

fgODTU justification overhead

The fgODTU justification overhead includes 12 bits which consist of a 6-bit C_mT field (bits C1, C2, ..., C6), a 1-bit increment indicator (II) field, a 1-bit decrement indicator (DI) field and a 4-bit CRC-4 field which contains an error check code over the 12-bit field as indicated in Figure N.7.

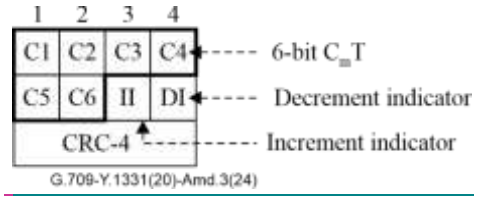


Figure N.7 – fgODTU justification overhead for fgGMP

Table N.2 shows the inversion patterns for the 6-bit count of C_mT.

Table N.2 – 6-bit $C_mT(t)$ increment and decrement indicator patterns

C1	C2	C3	C4	C5	C6	II	DI	Change
U	U	U	U	U	U	0	0	0
I	U	I	U	I	U	1	0	+1
U	I	U	I	U	I	0	1	–1
binary value						1	1	More than +1/–1

NOTE
– I indicates inverted Ci bit
– U indicates unchanged Ci bit

The CRC-4 is calculated over bits C1 to C6, II and DI. The CRC-4 uses the $g(x) = x^4 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) Bits C1 to C6, II and DI form an 8-bit pattern representing the coefficients of a polynomial *M(x)* of degree 7.

2) *M(x)* is multiplied by x^4 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 3 or less.

3) The coefficients of *R(x)* are considered to be a 4-bit sequence, where x^3 is the most significant bit.

4) This 4-bit sequence is the CRC-4 where the first bit of the CRC-4 to be transmitted is the coefficient of x^3 and the last bit transmitted is the coefficient of x^0.

N.4 Mapping fgODUflex into fgODTU.M

The mapping of the fgODUflex signal (with up to ±20 ppm bit-rate tolerance) into the fgODTU.M signal is performed by means of a simplified generic mapping procedure specific for fine grain optical transport network (fgGMP). It does not require the passing and processing of the C_{nD} information, but only requires the passing and processing of a simplified C_m information denoted by the C_mT value. The C_mT value is equalling to $C_m - C_mB$ where C_mB is a base value.

The OPU and therefore the fgODTU.M signals are created from a locally generated clock (within the limits specified in Table M.4), which is independent of the fgODUflex client signal.

The fgODUflex signal is adapted to the locally generated OPU/fgODTU.M clock by means of a simplified generic mapping procedure (fgGMP). The value of M is the number of fine grain tributary slots occupied by the fgODUflex. The value of m of C_m is 128. It uses the 128-bit as the data/stuff adaptation granularity.

A group of 16 successive bytes of the fgODUflex signal are mapped into a group of 16 successive bytes of the fgODTU.M payload area under control of the fgGMP data/stuff control mechanism. Each group of 16 bytes in the fgODTU.M payload area may either carry 16 fgODUflex bytes, or carry 16 stuff bytes. The value of the stuff bytes is set to all-0s.

The groups of 16 bytes in the fgODTU.M payload area are numbered from 1 to 256M as specified in Figure N.3. In row 1 of the fgODTU.M multiframe the first 16-byte block will be labelled 1, the next 16-byte block will be labelled 2, etc.

NOTE 1 – The 16-byte block alignment of the fgODUflex is preserved through the mapping procedure; e.g., the position of the first 16 OH bytes of the fgODUflex is always located after an integer number of 16-byte blocks from the start of the fgODTU.M structure.

The generic mapping process generates for the fgODUflex signal once per fgODTU.M multiframe the $C_m(t)$ information and then generates the corresponding $C_mT(t)$ value. The $C_mT(t)$ information is encoded in the fgODTU.M justification control overhead. The de-mapping process decodes $C_mT(t)$ information from fgODTU.M justification control overhead to obtain $C_m(t)$, and then de-maps the fgODUflex signal according to $C_m(t)$.

During a signal fail condition of the incoming fgODUflex signal, this failed incoming signal will contain the fgODUflex-AIS signal as specified in clause M.3. This fgODUflex-AIS is then mapped into the fgODTU.M.

For the case when the fgODUflex is received from the output of a fabric (fgODUflex connection function), the incoming signal may contain (in the case of an open matrix connection) the fgODUflex-OCI signal as specified in clause M.3. This fgODUflex-OCI signal is then mapped into the fgODTU.M.

During a signal fail condition of the incoming ODU/OPU signal (e.g., in the case of an ODU-AIS, ODU-LCK, ODU-OCI condition), the fgODUflex-AIS pattern as specified in clause M.3 is generated as a replacement signal for the lost fgODUflex signal.

The values of M, $C_{m,nom}$, $C_{m,min}$, $C_{m,max}$, C_mB, $C_mT_{,min}$ and $C_mT_{,max}$ for fgODUflex(p) into fgODTU.M are as follows:

$$M = p \text{ for fgODUflex(p)} \quad \text{(N-2)}$$

$$C_{m,\,nom} = \frac{R_{fgODU(p)}}{R_{fgODTU.M}} \times P_{fgODTU.M} \quad \text{(N-3)}$$

$$C_{m,\,min} = floor\left[\frac{(1-\delta R_{fgODU(p)})}{(1+\delta R_{fgODTU.M})} \times C_{m,\,nom}\right] \quad \text{(N-4)}$$

$$C_{m,\,max} = ceiling\left[\frac{(1+\delta R_{fgODU(p)})}{(1-\delta R_{fgODTU.M})} \times C_{m,\,nom}\right] \quad \text{(N-5)}$$

$$C_mB = floor[C_{m,\,nom}] - 16 \quad \text{(N-6)}$$

$$C_mT_{,min} = C_{m,\,min} - C_mB \quad \text{(N-7)}$$

$$C_mT_{,max} = C_{m,\,max} - C_mB \quad \text{(N-8)}$$

where $R_{fgODU(p)}$, $R_{fgODTU.M}$, $\delta R_{fgODU(p)}$, $\delta R_{fgODTU.M}$ and $P_{fgODTU.M}$ are the nominal rate of fgODUflex(p), the nominal rate of fgODTU.M, the bit rate tolerance of fgODUflex(p), the bit rate

tolerance of fgODTU.M, and the number of m-bits in the server fgODTU.M payload area respectively.

Tables N.3, N.4, and N.5 show the specific values of $C_{128,\ min}$, $C_{128,\ max}$, $C_{128}B$, $C_{128}T_{,\ min}$ and $C_{128}T_{,\ max}$ for some bit rate fgODUflex(p) signals over fgODTU.M of OPU0/flex, OPU1 and OPU2 separately.

NOTE 2 – For normal operation, C_mT is expected to fall within the range of $[C_mT_{,min}, C_mT_{,max}]$; during transient conditions, C_mT may exceed its normal range, e.g., using a range of $[\max(C_mT_{,min}\text{-X},\ 0),\ \min(C_mT_{,max}\text{+X},\ P_{fgODTU.M}]$, where X=1.

Table N.3 – $C_{128}B$ and $C_{128}T$ values for mapping of fgODUflex into fgODTU.M of OPU0/flex

p	fgODUflex(p) bit rate (kbit/s)	fgODTU.M Pserver size (16-byte)	Blocks per fgTSOH multiframe	$C_{128,\ nom}$	$C_{128,\ min}$	$C_{128,\ max}$	$C_{128}B$	$C_{128}T_{,\ min}$	$C_{128}T_{,\ max}$
1	10 409.203 ± 20 ppm	256	255.947 ± 0.0102	255	255	256	239	16	17
10	104 092.031 ± 20 ppm	2560	2559.47 ± 0.1024	2559	2559	2560	2543	16	17
50	520 460.155 ± 20 ppm	12800	12797.3 ± 0.5119	12797	12796	12798	12781	15	17
100	1 040 920.31 ± 20 ppm	25600	25594.7 ± 1.024	25594	25593	25596	25578	15	18
119	1 238 695.168 ± 20 ppm	30464	30457.7 ± 1.218	30457	30456	30459	30441	15	18
p	p × 10 409.203 ± 20 ppm	p × 256	p × 255.947 ± p × 0.01024	p × 255.947	See equation N-4	See equation N-5	*Floor* (p × 255.947) – 16	See equation N-7	See equation N-8

NOTE 1 – The fgTSOH multiframe period is 3.147328 ms.

NOTE 2 – The fgODTU.M carrying fgODUflex(p) consists of M fgTSs of OPU and the M value is equaling to the p value of the fgODUflex(p).

NOTE 3 – Variances in blocks per fgTSOH Multiframe are due to ±20 ppm in the fgODUflex bit rate and the ±20 ppm in the fgTSOH multiframe period.

Table N.4 – $C_{128}B$ and $C_{128}T$ values for mapping of fgODUflex into fgODTU.M of OPU1

p	fgODUflex(p) bit rate (kbit/s)	fgODTU.M Pserver size (16-byte)	Blocks per fgTSOH multiframe	$C_{128, nom}$	$C_{128, min}$	$C_{128, max}$	$C_{128}B$	$C_{128}T_{, min}$	$C_{128}T_{, max}$
1	10 409.203 ± 20 ppm	256	254.875 ± 0.0102	254	254	255	238	16	17
10	104 092.031 ± 20 ppm	2560	2548.75 ± 0.1019	2548	2548	2549	2532	16	17
50	520 460.155 ± 20 ppm	12800	12743.7 ± 0.5097	12743	12743	12745	12727	16	18
100	1 040 920.31 ± 20 ppm	25600	25487.5 ± 1.0219	25487	25486	25489	25471	15	18
119	1 238 695.168 ± 20 ppm	30464	30330.1 ± 1.213	30330	30328	30332	30314	14	18
p	p × 10 409.203 ± 20 ppm	p × 256	p × 254.875 ± p × 0.01019	p × 254.875	See equation N-4	See equation N-5	*Floor* (p × 254.875) – 16	See equation N-7	See equation N-8

NOTE 1 – The fgTSOH multiframe period is 3.134144 ms.

NOTE 2 – The fgODTU.M carrying fgODUflex(p) consists of M fgTSs of OPU and the M value is equaling to the p value of the fgODUflex(p).

NOTE 3 – Variances in blocks per fgTSOH Multiframe are due to ±20 ppm in the fgODUflex bit rate and the ±20 ppm in the fgTSOH multiframe period.

Table N.5 – C128B and C128T values for mapping of fgODUflex into fgODTU.M of OPU2

p	fgODUflex(p) bit rate (kbit/s)	fgODTU.M Pserver size (16-byte)	Blocks per fgTSOH Multiframe	$C_{128, nom}$	$C_{128, min}$	$C_{128, max}$	$C_{128}B$	$C_{128}T, min$	$C_{128}T, max$
1	10 409.203 ± 20 ppm	256	253.797 ± 0.0102	253	253	254	237	16	17
10	104 092.031 ± 20 ppm	2560	2537.97 ± 0.1015	2538	2537	2539	2522	15	17
50	520 460.155 ± 20 ppm	12800	12689.9 ± 0.5075	12690	12689	12691	12674	15	17
100	1 040 920.31 ± 20 ppm	25600	25379.7 ± 1.0152	25380	25379	25382	25364	15	18
119	1 238 695.168 ± 20 ppm	30464	30201.9 ± 1.208	30202	30201	30204	30186	15	18
p	p × 10 409.203 ± 20 ppm	p × 256	p × 253.797 ± p × 0.01015	p × 253.797	See equation N-4	See equation N-5	*Floor* (p × 253.797) – 16	See equation N-7	See equation N-8

NOTE 1 – The fgTSOH multiframe period is 3.120896 ms.

NOTE 2 – The fgODTU.M carrying fgODUflex(p) consists of M fgTSs of OPU and the M value is equaling to the p value of the fgODUflex(p).

NOTE 3 – Variances in blocks per fgTSOH Multiframe are due to ±20 ppm in the fgODUflex bit rate and the ±20 ppm in the fgTSOH multiframe period.

Annex O

Hitless bandwidth adjustment of fgODUflex

(This annex forms an integral part of this Recommendation.)

This annex specifies the hitless bandwidth adjustment of fgODUflex which consists of the OPUk(fgTS) /OPUflex(fgTS,n) multiplex section link connection resizing function (fgLCR) and the fgODUflex bandwidth resizing function (fgBWR). It specifies resizing control overheads, resizing procedure and resizing parameters.

O.1 Resizing control overheads

The resizing control overheads consist of fgLCR RCOH and fgBWR RCOH.

Fine grain link connection resizing control overhead (fgLCR RCOH)

Each fgTS has an associated fgLCR RCOH which is part of the fgTSMxOH. Each fgLCR RCOH is transmitted over four OPUk(fgTS)/OPUflex(fgTS,n) frames. Two fgLCR RCOH are present simultaneously in Row 4, Columns 15 and 16 of the OPUk(fgTS)/OPUflex(fgTS,n) as shown in Figure O.1A. The association of fgLCR RCOH to fgTS is indexed by MFAS and OMFI.

The format of the fgLCR RCOH consists of RP (1 bit), TSCC (1 bit), CTRL (2 bits), and TSGS (2 bits) as shown in Figure O.1A. The fgLCR RCOH for two fgTSs (fgTS #k and fgTS #k+1) are presented in Figure O.1A. The fgLCR RCOH for fgTS #k is located at bits 13 to 18 of fgTSMxOH[k]. The fgLCR RCOH for fgTS #k+1 is located at bits 13 to 18 of fgTSMxOH[k+1].

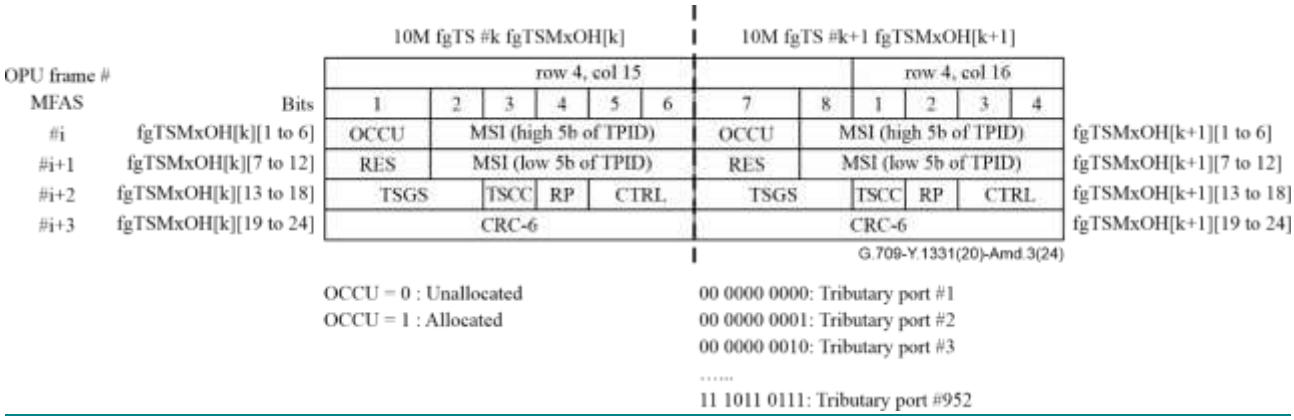


Figure O.1A – fgLCR RCOH in OPUk(fgTS)/OPUflex(fgTS,n) overhead area

The specific functions of the RP, TSCC, CTRL, and TSGS in fgLCR RCOH are specified in Table O.1A.

Table O.1A – fgLCR RCOH functions

RCOH	Value	Command	Remarks
CTRL	00	IDLE	Default value. Indicates that there is no fgLCR request on the associated fgTS.
	01	ADD	Indication from ODUkP/fgODUflex_A_So to ODUkP/fgODUflex_A_Sk requesting that the associated fgTS is to be added to the fgODUflex connection named in TPID.
	10	REMOVE	Indication from ODUkP/fgODUflex_A_So to ODUkP/fgODUflex_A_Sk that the associated fgTS is to be removed from the fgODUflex connection named in TPID.
	11	RES	Reserved
TSGS	00	IDLE	Default value. Indicates that there is no acknowledgement response.
	01	ACCEPT	Accept response indication that the sink accepts the ADD/REMOVE request of the fgTS from the source on its CTRL. The ACCEPT response is sent in the reverse direction by the ODUkP/fgODUflex_A_So co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A adaptation function.~~.~~
	10	REJECT	Reject response indication that the sink rejects the ADD/REMOVE request of the fgTS from the source on its CTRL. The reject response is sent in the reverse direction by the ODUkP/fgODUflex_A_So co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A adaptation function
	11	RES	Reserved
TSCC	1	Enable	Enable indication that the TSCC bit is used to check the connectivity of the OPUk(fgTS)/OPUflex(fgTS,n) multiplex section link connection and fgODUflex connection.
	0	Disable	Default value~~ and disable indication~~. Indicates there is no active connectivity check for the associated fgTS.
RP	1	Enable	Enable indication of the RP bit is used to indicate that resizing protocol in active on the associated fgTS.
	0	Disable	Default value. Used to indicate that resizing protocol is inactive or to end the resizing protocol.
NOTE – The MSI field of fgTSMxOH also serves as the TPID OH in the fgLCR RCOH for hitless resizing to identify fgTS to be added or removed from the specified port identifier.			

Fine grain bandwidth resize control overhead (fgBWR RCOH)

The fgBWR RCOH is located in rows 1 to 3 and column 15 of fgOPUflex overhead area as shown in Figure O.1B. It consists of BWR_IND (1 bit), NCS (1 bit), and CRC-3 (3 bits).

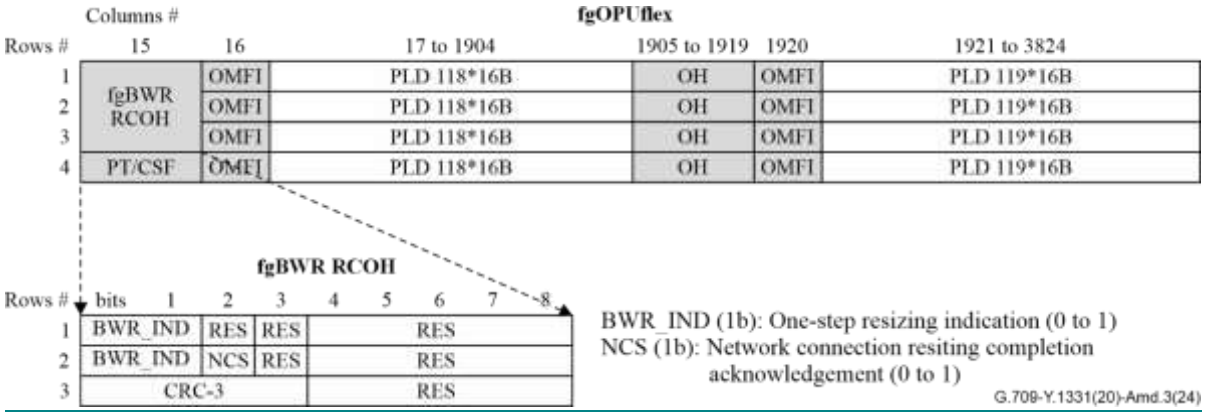


Figure O.1B – fgBWR RCOH in fgOPUflex overhead area

The specific functions of the BWR_IND, NCS, and CRC-3 in fgBWR RCOH are specified in Table O.1B.

Table O.1B – fgBWR RCOH functions

RCOH	Value	Command	Remarks
BWR_IND	0−>1	One-step rate adjustment indication	Indication for simultaneous: – one-step rate jump of fgODUflex bit rate – format change of the server fgODTU container switch from M fgTSs to M+N/M-N fgTSs, and – change in allocation of the fgODTU container from the original set of M fgTSs to the new set of M+N/M−N fgTSs at the designated position.
	1−>0	Resizing end	Indication for the completion of resizing process.
NCS	0−>1	Resizing completion acknowledgement	Network connection resizing completion acknowledgement indication from fgODUflex sink to fgODUflex source that the sink has completed the resizing.
	1−>0	Resizing process end	Indication for acknowledging the end of the resizing process.
NOTE – The 3 bits CRC-3 provides an error check code over bits 1 to 3 of rows 1 and 2 of fgODUflex frame. Its polynomial and calculation reuse CRC-3 as defined in clause 6.2 of [ITU-T G.7044].			

O.2 Resizing procedure

O.2.1 Bandwidth increase

The increase resizing procedure consists of the following steps (with increasing from M×10M to (M+N)×10M assumed):

Step 1 The fgODUflex source node starts the resizing protocol after receiving the increase command. It enables [RP=1] and starts a timeout timer. It triggers link by link connection resizing from the first multiplex section to the last multiplex section. In each multiplex section link connection, the ODUkP/fgODUflex_A source signals [RP=1, TSCC=0][CTRL=ADD, TPID=#a] to the ODUkP/fgODUflex_A sink via the fgLCR overhead of fgTSs to be added, to request the ODUkP/fgODUflex_A sink to reserve the corresponding bandwidth resource. When all fgTSs to be added have signalled [CTRL=ADD, TPID=#a], the ODUkP/fgODUflex_A source will signal [TSCC=1] on all the added fgTS to the ODUkP/fgODUflex_A sink. The [RP=1, TSCC=0], [CTRL=ADD,

TPID=#a] and [TSCC=1] are forwarded link by link from the fgODUflex source node to the fgODUflex sink node.

Step 2 When the fgODUflex sink node receives [RP=1, TSCC=1] on all the added fgTS, and it is ready to accept the requested bandwidth increase, it acknowledges with [TSGS=ACCEPT] in the reverse direction on all the added fgTS by the ODUkP/fgODUflex_A_S co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirecetional ODUkP/fgODUflex_A adaptation function towards the fgODUflex source node.

Step 3 When the source node receives [TSGS=ACCEPT] on all the added fgTS, it triggers fgODUflex bit rate resizing. The source node signals [BWR_IND=1] via the fgODUflex fgBWR overhead and then completes one-step rate jump at the designated position (after bit 8 of row 3 and column 1904 of fgODUflex and before bit 1 of row 3 and column 1905 of fgODUflex; see clause O.3.1). This [BWR_IND=1] is transparently transported to all downstream nodes.

Step 4 In each multiplex section link connection, when the ODUkP/fgODUflex_A source detects the incoming BWR_IND transitioning from 0 to 1, it changes the fgODTU from M fgTSs to M+N fgTSs immediately after the designated position Y (see clause O.3.2 and O.3.3). Simultaneously, after the designated position Y, the fgODUflex jumps to the new bit rate of (M+N)×10M and is mapped into the fgODTU.M+N.

Step 5 When the sink node finishes the resizing (after reaching designated position Y), it acknowledges with [NCS=1] to the source node in fgODUflex fgBWR overhead.

Step 6 After the source node completes one-step rate jump of fgODUflex and receives [NCS=1] from the fgODUflex sink node, it signals [BWR_IND=0] to all downstream nodes to exit the resizing procedure. It also sends [RP=0, TSCC=0] and stops the timeout timer. When each node receives [RP=0], it sends [RP=0, TSCC=0][CTRL=IDLE, TPID=#0] and exits the resizing procedure.

NOTE 1 – During step 1, if the fgLCR operate fails in one of the multiplex section link connections, the follow-up fgLCR operation is stopped and acknowledges [TSGS=REJECT] to the upstream nodes via link by link. All the upstream nodes return back the initial state.

NOTE 2 – When the timer in the source node times out, the source node sends [RP=0, TSCC=0] and exits the resizing procedure. When each node detects the upstream RP from 1 to 0, it sends [RP=0, TSCC=0][CTRL=IDLE, TPID=#0] and exits the resizing procedure.

NOTE 3 – The fgODUflex path bandwidth (including protection path bandwidth, if applicable) is increased before fine grain client bandwidth is increased.

Figure O.2 shows an example for bandwidth increase procedure (NE #A to NE #C), where the first multiplex section (#A1 to #B1) assumes fgTSs #5 and #9 to be added, and the second multiplex section (#B2 to #C1) assumes fgTSs #11 and #12 to be added.

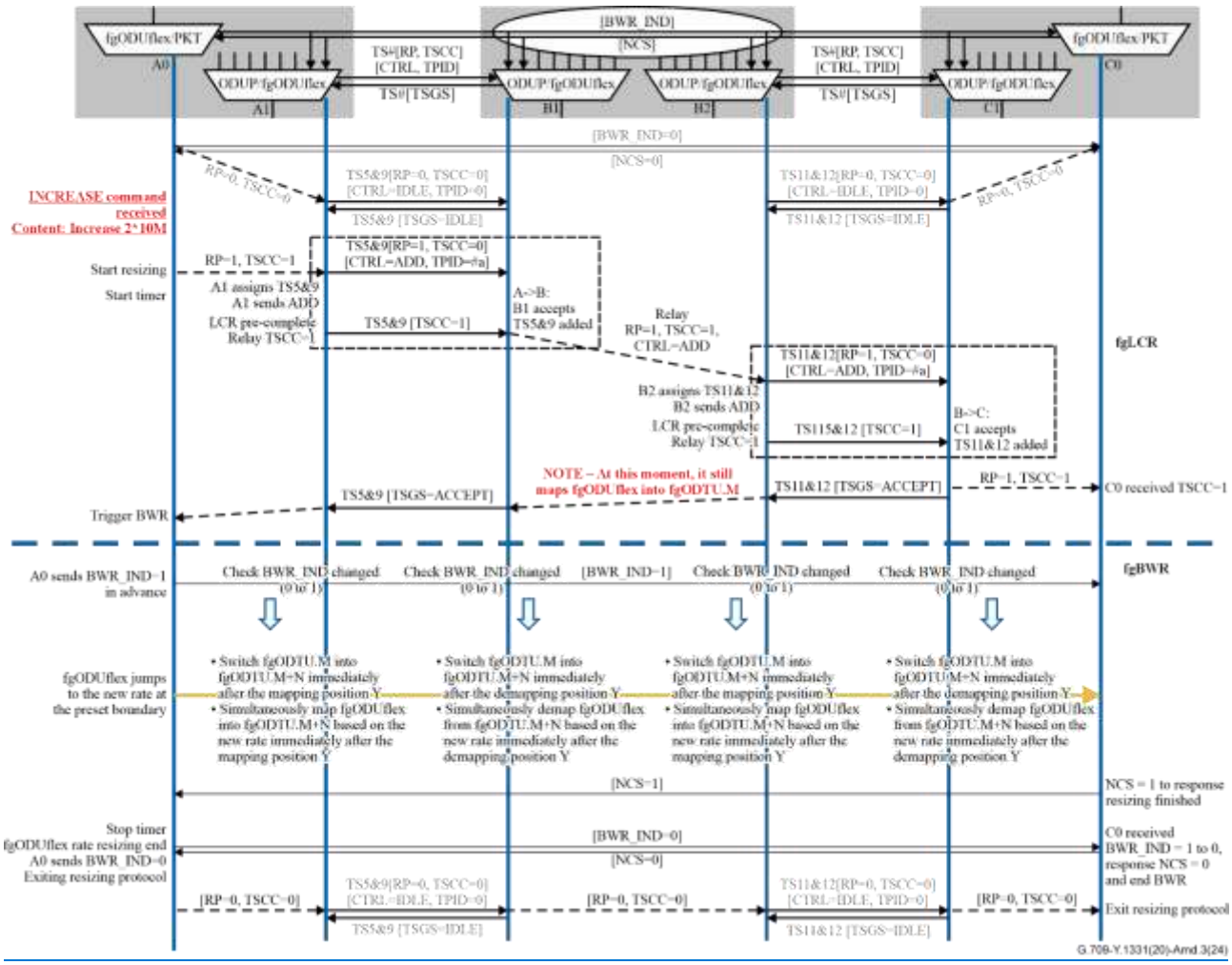


Figure O.2 – Bandwidth increase resizing procedure

O.2.2 Bandwidth decrease

The decrease resizing procedure consists of the following steps (with decreasing from M×10M to (M−N)×10M assumed):

Step 1 The fgODUflex source node starts the resizing protocol after receiving the decrease command. It enables [RP=1] and starts a timeout timer. It triggers link by link connection resizing from the first multiplex section to the last multiplex section. In each multiplex section link connection, the ODUkP/fgODUflex_A source signals [RP=1, TSCC=0][CTRL=REMOVE, TPID=#a] to the ODUkP/fgODUflex_A sink via the fgLCR overhead of fgTSs to be removed and to request ODUkP/fgODUflex_A sink to mark the corresponding bandwidth resource for future removal. When all fgTSs to be removed have signalled [CTRL=REMOVE, TPID=#a], the ODUkP/fgODUflex_A source will signal [TSCC=1] on all the removed fgTS to the ODUkP/fgODUflex_A sink. The [RP=1, TSCC=0], [CTRL=REMOVE, TPID=#a] and [TCC=1] are forwarded link by link from the fgODUflex source node to the fgODUflex sink node.

Step 2 When the fgODUflex sink node receives [RP=1, TSCC=1] on all the removed fgTS, and it is ready to accept the requested bandwidth decrease, it acknowledges with [TSGS=ACCEPT] in the reverse direction on all the removed fgTS by the ODUkP/fgODUflex_A. So co-located with the ODUkP/fgODUflex_A Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A adaptation function towards the fgODUflex source node.

Step 3 When the source node receives [TSGS=ACCEPT] on all the removed fgTS, it triggers fgODUflex bit rate resizing. The source node signals [BWR_IND=1] via the fgODUflex

fgBWR overhead and then completes one-step rate jump at the designated position (after bit 8 of row 3 and column 1904 of fgODUflex and before ~~at~~ bit 1 of row 3 and column 1905 of fgODUflex; see clause O.3.1). This [BWR_IND=1] is transparently transported to all downstream nodes.

Step 4 In each multiplex section link connection, when the ODUkP/fgODUflex_A source detects the incoming BWR_IND transitioning from 0 to 1, it changes the fgODTU from M fgTSs to M-N fgTSs immediately after the designated position Y (see clause O.3.2 and O.3.3). ~~After~~Simultaneously, after the designated position Y, the fgODUflex jumps to the new bit rate of (M−N)×10M and is mapped in to the fgODTU.M−N.

Step 5 When the sink node finishes the resizing (after reaching designated position Y), it acknowledges with [NCS=1] to the source node in fgODUflex fgBWR overhead.

Step 6 After the source node completes one-step rate jump of fgODUflex and receives [NCS=1] from the fgODUflex sink node, it signals [BWR_IND=0] to all downstream nodes to exit the resizing procedure. It also sends [RP=0, TSCC=0] and stops the timeout timer. When each node receives [RP=0], it sends [RP=0, TSCC=0][CTRL=IDLE, TPID=#0] and exits the resizing procedure.

NOTE 1 – During step 1, if the fgLCR operate fails in one of the multiplex section link connections, the follow-up fgLCR operation is stopped and acknowledges [TSGS=REJECT] to the upstream nodes via link by link. All the upstream nodes return back the initial state.

NOTE 2 – When the timer in the source node times out, the source node sends [RP=0, TSCC=0] and exits the resizing procedure. When each node detects the upstream RP from 1 to 0, it sends [RP=0, TSCC=0][CTRL=IDLE, TPID=#0] and exits the resizing procedure.

NOTE 3 – The fine grain client bandwidth is decreased before the fgODUflex path bandwidth (including protection path bandwidth, if applicable) is decreased.

Figure O.3 shows an example for bandwidth decrease procedure (NE #A to NE #C), where the first multiplex section (#A1 to #B1) assumes fgTSs #5 and #9 to be removed, and the second multiplex section (#B2 to #C1) assumes fgTSs #11 and #12 to be removed.

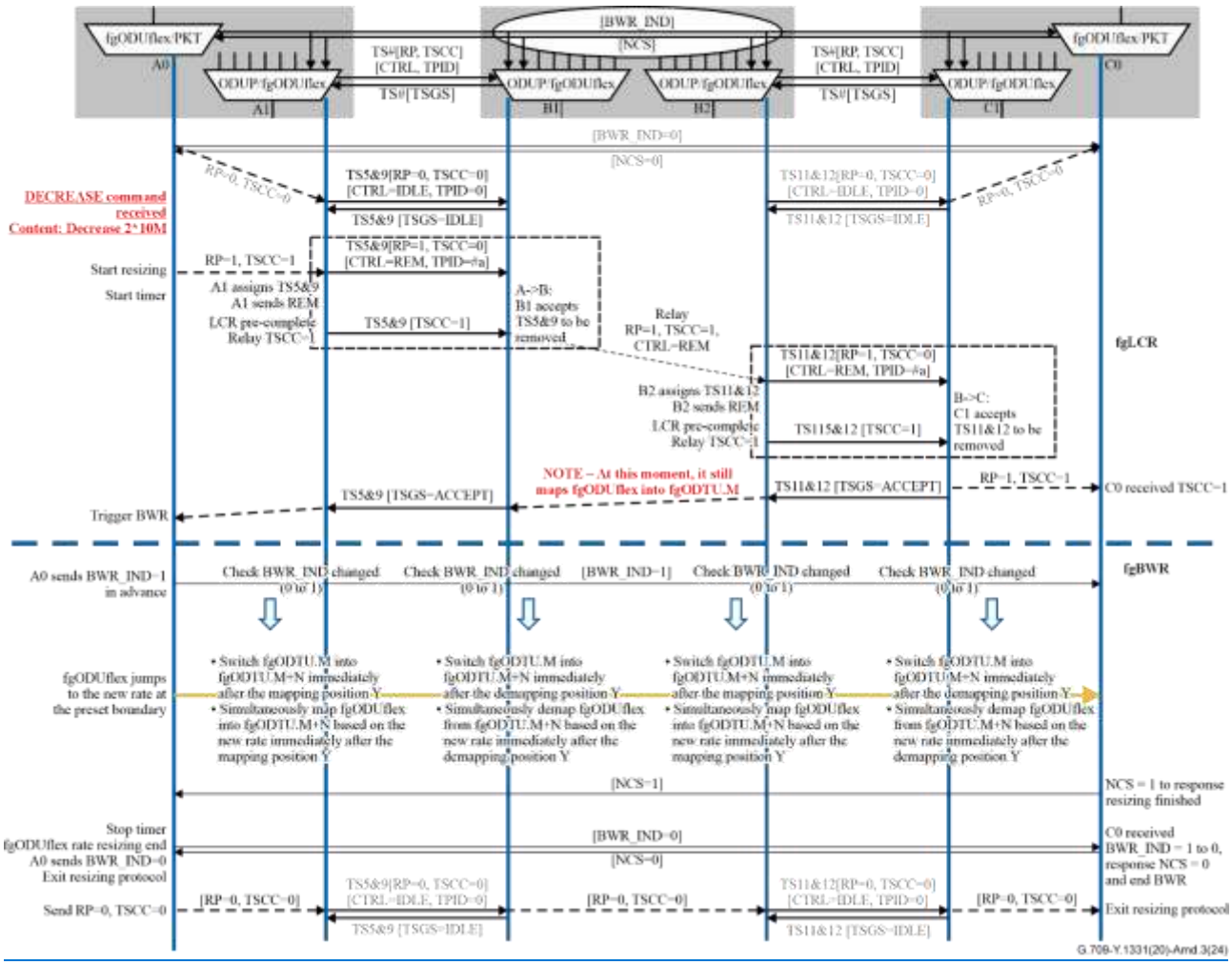


Figure O.3 – Bandwidth decrease resizing procedure

O.2.3 Hop-by-hop fgLCR

Fine grain hitless resizing procedure uses a hop-by-hop scheme to set up the resized path from the fgODUflex source node to the fgODUflex sink node. A network-wide management system is not required, but is not precluded. The decision to accept or reject a resizing operation may be made by individual NE in the path based on local configuration parameters and/or local conditions.

In fgLCR, the set of fgTSs to be added or removed are signalled by [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications, respectively, at each added/removed fgTS. The time taken by the ODUkP/fgODUflex_A source to send the full set of add/remove indications may span one or more fine grain tributary slot multiplexing overhead (fgTSMxOH) multiframes. After all the [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications have been sent, the ODUkP/fgODUflex_A source sends [TSCC=1] on all the added/removed fgTS to signal completion. The time taken by the ODUkP/fgODUflex_A source to send the full set of TSCC indications may span one or more fgTSMxOH multiframes. The ODUkP/fgODUflex_A sink interprets all the add/remove requests from the ODUkP/fgODUflex_A source are completed when it receives the [TSCC=1] indication on any of the fgTS with [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications.

An intermediate fgODUflex switching node has an ODUkP/fgODUflex_A sink on its ingress side and a corresponding ODUkP/fgODUflex_A source on its egress side, in the forward direction. [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications received by the ODUkP/fgODUflex_A sink are forwarded to the corresponding ODUkP/fgODUflex_A source. The switching node waits for [TSCC=1] on any of the [CTRL=ADD, TPID=#a] or [CTRL=REMOVE,

TPID=#a] indications to determine that the full fgLCR request has been completed before forwarding the [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications.

The fgODUflex sink node only has an ODUkP/fgODUflex_A sink in the forward direction. When the sink node has determined that the full fgLCR request has been completed by the reception of [TSCC=1] indication and [CTRL=ADD, TPID=#a] or [CTRL=REMOVE, TPID=#a] indications on all the added/removed fgTS, and it is ready to support the fgBWR process from the fgODUflex source node, it will signal [TSGS=ACCEPT] in the reverse direction on all the added/removed fgTS by the ODUkP/fgODUflex_A_So co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A adaptation function. The time taken by the ODUkP/fgODUflex_A sink to send the full set of TSGS indications may span one or more fgTSMxOH Multiframes. In the case where the fgODUflex sink node wishes to reject the fgLCR request, a [TSGS=REJECT] indication will be sent in the reverse direction on all the added/removed fgTS by the ODUkP/fgODUflex_A_So co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A.

In the reverse direction, an intermediate fgODUflex switching node also has an ODUkP/fgODUflex_A sink on its ingress side and a corresponding ODUkP/fgODUflex_A source on its egress side. [TSGS=ACCEPT] or [TSGS=REJECT] indications received by the ODUkP/fgODUflex_A sink are forwarded to the corresponding ODUkP/fgODUflex_A source. [TSGS=REJECT] indications can be forwarded without delay. However, [TSGS=ACCEPT] indications can be delayed and only forwarded when the switching node is ready to support the fgBWR process from the fgODUflex source node. The time taken by the ODUkP/fgODUflex_A source to send the full set of TSGS indications may span one or more fgTSMxOH multiframes.

Based on its configuration parameters and local conditions, an intermediate fgODUflex switching node may reject an incoming fgLCR request. The ODUkP/fgODUflex_A_So in the reverse direction co-located with the ODUkP/fgODUflex_A_Sk receiving the CTRL request in the bidirectional ODUkP/fgODUflex_A adaptation function will send [TSGS=REJECT] for all fgTS making add/remove requests on that TPID.

O.3 Resizing parameters

O.3.1 Rate change position in fgODUflex frame during resizing

Figure O.4 illustrates the rate change at the designated position in the fgODUflex frame during resizing via two examples (increase resizing from 20M to 40M, and decrease resizing from 20M to 10M). The green block is the 16-byte located in row 3 columns 1 to 16 which carries the last 8 bits of the fgBWR RCOH overhead. The blue block is the 16-byte block located in row 3 columns 1889 to 1904 which denotes the last 16-byte data block before fgBWR takes effect. The yellow block is the 16-byte block located in row 3 and column 1905 to 1920 which denotes the first 16-byte data block after fgBWR takes effect.

After the source node signals [BWR_IND=1] via fgODUflex fgBWR overhead, it will complete the one-step rate jump of fgODUflex immediately after the last bit of the blue block and before the first bit of the yellow block.

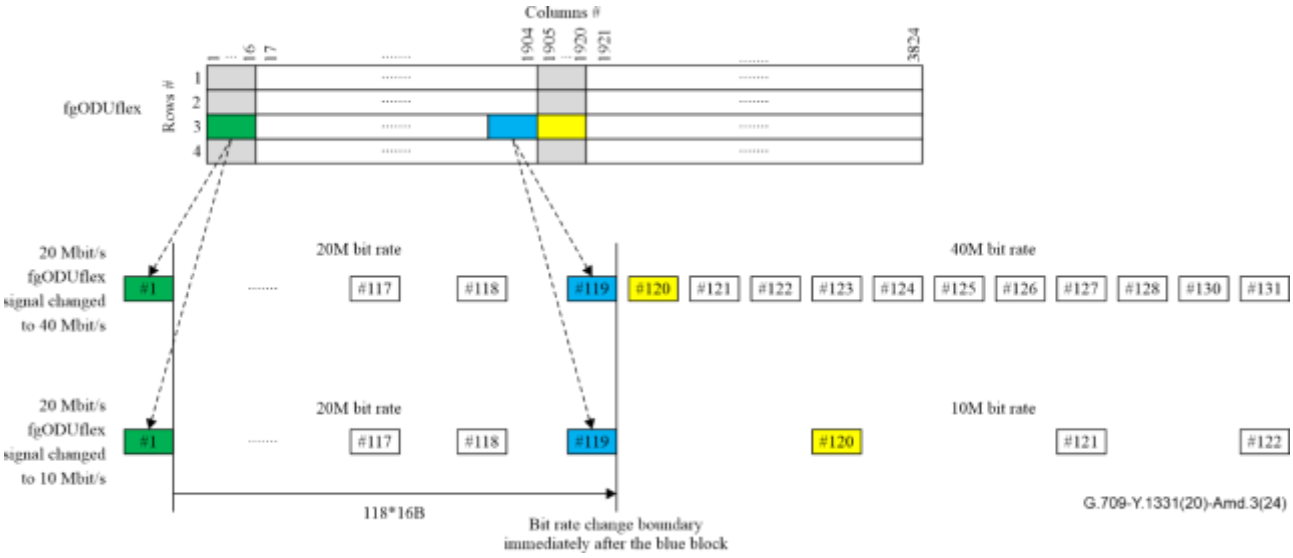


Figure O.4 – Examples of rate change position in fgODUflex

O.3.2 Cm values and mapping overhead location update during resizing

Figure O.5 illustrates the Cm values update during resizing. There are two cases for the distribution of the green and blue blocks of fgODUflex in fgODTU before the fgODUflex bit rate changes. Case 1 shows the green and blue blocks located in the same fine grain tributary slot overhead (fgTSOH) multiframe (e.g. #i). Case 2 shows the green block and the blue block located separately in the two consecutive fgTSOH multiframes (e.g., #i-1 and #i).

- In fgTSOH multiframe #i-1, the GMP overhead carries the old C_mT in the fgTSOH of the last fgTS occupied by the fgODTU.Mo.
- In fgTSOH multiframe #i, the GMP overhead carries the old C_mT in the fgTSOH of the last fgTS occupied by the fgODTU.Mo. Up to and including the blue block (the designated position Y) in the fgODTU.Mo, the fgODUflex is mapped into the fgODTU.Mo payload according to the old C_m value. Immediately after the blue block of the fgODTU.Mn, a new intermediate fgODTU.Mi is started with index j initialized to j=1. The size and fgTS allocation of the fgODTU.Mi is the same as the resized new fgODTU.Mn. Starting at the designated position of the fgODUflex (i.e., the yellow block in Figure O.4), the fgODUflex is mapped into the intermediate fgODTU.Mi payload according to the intermediate C_m value (for intermediate Cm values near Pserver, the fgODTU.Mi block at index j=1 will be a Stuff block. The Yellow block of the fgODUflex will be located at index j=2.). The intermediate fgODTU.Mi terminates at the end of ~~RMF~~fgTSOH multiframe #i.
- In fgTSOH multiframe #i+1, the GMP overhead carries the intermediate C_mT with [II=1, DI=1] in the fgTSOH of the last fgTS occupied by the fgODTU.Mn, and maps the fgODUflex into the fgODTU.Mn payload according to the intermediate Cm value. The fgODTU.Mn begins at the start of fgTSOH multiframe #i+1 with index j initialized to j=1.
- In fgTSOH multiframe #i+2, the GMP overhead begins to carry the new generated C_mT in the fgTSOH of the last fgTS occupied by the fgODTU.Mn, and continues to map the fgODUflex into the fgODTU.Mn payload according to the intermediate C_m value.
- In fgTSOH multiframe #i+3, the GMP overhead carries the new generated C_mT in the fgTSOH of the last fgTS occupied by the fgODTU.Mn, and maps the fgODUflex into the fgODTU.Mn payload according to the new generated C_m value in fgTSOH multiframe #i+2.

NOTE 1 – the intermediate C_m value is $C_{m,nom}$.

NOTE 2 – the fgODTU.Mo denotes the old server container to carry fgODUflex before fgODTU changes, and it is fgODTU.M which occupies M fgTSs of the server OPU. The fgODTU.Mn denotes the new server container to carry fgODUflex after fgODTU changes. In the case of increase, it is fgODTU.M+N which

occupies M+N fgTSs of the server OPU; In the case of decrease, it is fgODTU.M-N which occupies M-N fgTSs of the server OPU.

NOTE 3 – The old C_mT is sent up to and including the fgTSOH multiframe containing the blue block, while the intermediate C_mT is sent in next fgTSOH multiframe. The new C_mT is sent in the second fgTSOH multiframe after the fgTSOH multiframe containing the blue block.

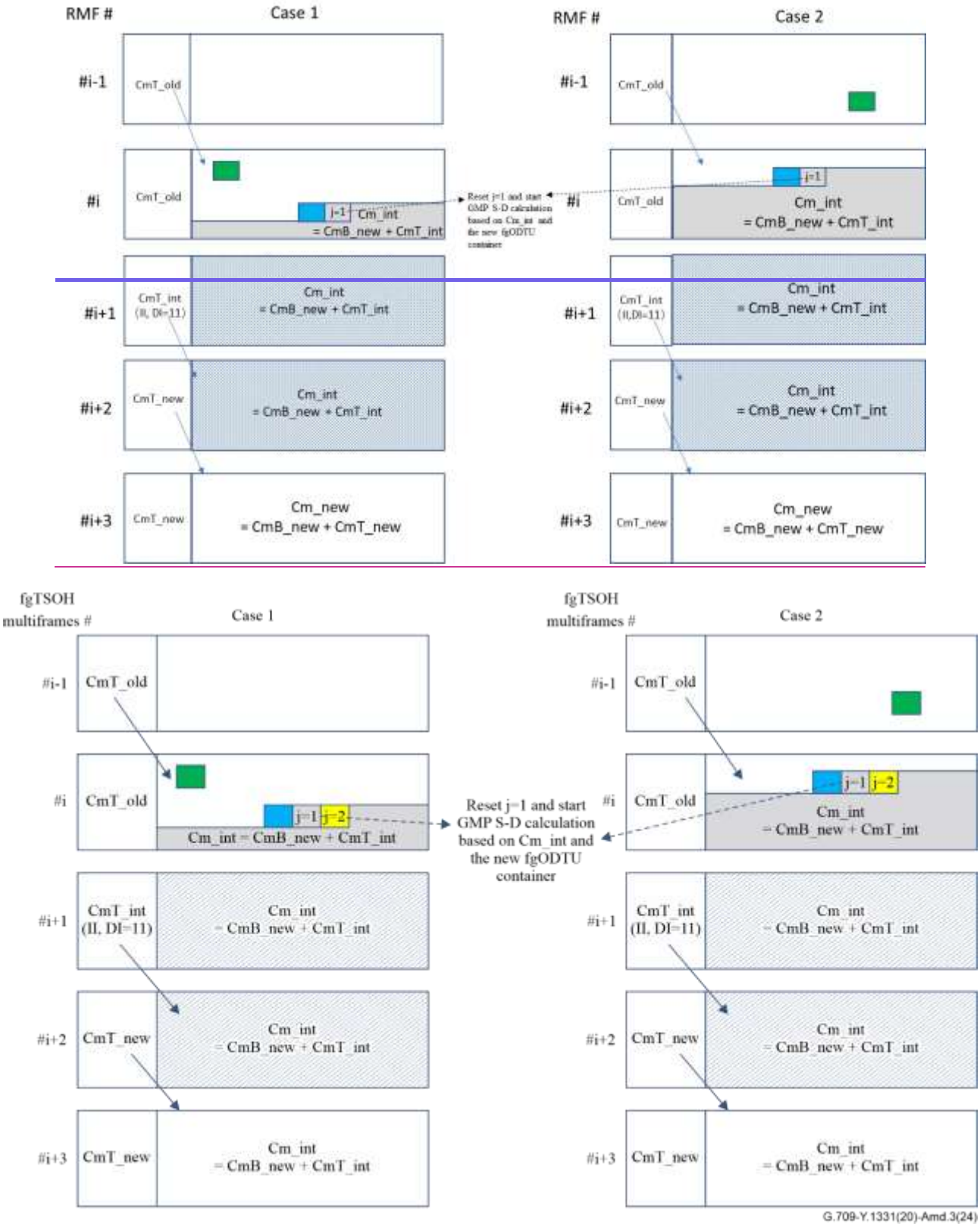


Figure O.5 – C_mT values update during resizing

Table O.2 further provides some examples to explain fgODTU.Mi alignment under hitless ADD and REMOVE operations. Initially, the old fgODTU.Mo is mapped into the old fgTS set {fgTS}o. The fgODTU.Mo is aligned to the start of the fgTSOH multiframe. fgTS members in {fgTS}o are arranged in ascending order by fgTS number. fgODTU.Mo includes and terminates at the Blue block. The intermediate fgODTU.Mi starts immediately after the fgODTU.Mo terminates and is mapped into the intermediate fgTS set {fgTS}i. fgTS members in {fgTS}i are also arranged in ascending order by fgTS number. The fgODTU.Mi is aligned to the first member of {fgTS}i with a fgTS number higher than (including wrap-around at the end of the set) the last fgTS occupied by fgODTU.Mo. In examples of Table O.2, it assumes the old fgTS set {fgTS}o, fgTS added and fgTS removed as the following description.

- {fgTS}o = fgTS #: 10, 20, 30, 40, 50, 60, 70, 80
- fgTS added = fgTS #: 1, 2, 3, 11, 12, 13, 41, 42, 43
- fgTS removed = fgTS #: 20, 40

Table O.2 – Examples of fgODTU.Mi alignment

Hitless operation	Last fgTS of fgODTU.Mo	fgODTU.Mi alignment	Comment
ADD	fgTS #20	fgTS #30	No new fgTS added to {fgTS}i between the last fgTS assigned to fgODTU.Mo and the next fgTS in {fgTS}o. fgODTU.Mi aligned the next fgTS already in {fgTS}o.
ADD	fgTS #10	fgTS #11	New fgTS added to {fgTS}i between the last fgTS assigned to fgODTU.Mo and the next fgTS in {fgTS}o. fgODTU.Mi aligned to next added fgTS in {fgTS}i.
ADD	fgTS #80	fgTS #1	The last fgODTU.Mo assigned to the last fgTS of {fgTS}o and {fgTS}i. New fgTS added ahead of the first fgTS in {fgTS}o. fgODTU.Mi aligned to the first fgTS in {fgTS}i.
REMOVE	fgTS #50	fgTS #60	No fgTS removed from {fgTS}o between the last fgTS assigned to fgODTU.Mo and the next fgTS in {fgTS}i. fgODTU.Mi aligned the next fgTS already in {fgTS}o.
REMOVE	fgTS #10	fgTS #30	fgTS removed from {fgTS}o between the last fgTS assigned to fgODTU.Mo and the next fgTS in {fgTS}i. fgODTU.Mi aligned the next fgTS in {fgTS}i.
REMOVE	fgTS #20	fgTS #30	Last fgTS occupied by fgODTU.Mo in {fgTS}o is removed. Next fgTS remains in {fgTS}i. fgODTU.Mi aligned the next fgTS in {fgTS}i.

O.3.3 fgODTU container switch position during resizing

When mapping fgODUflex into fgODTU with one-step rate jump of the fgODUflex, the fgODTU container begins to switch from M fgTSs to M+N fgTSs (bandwidth increase) or M-N fgTSs (bandwidth decrease) immediately after the blue block in the fgODUflex. The mapping position of the blue block of fgODUflex in fgODTU can be predicted by the following equations after it detects BWR_IND changed from 0 to 1. Starting at the first block immediately after the blue block in the fgODTU.Mn, the fgODUflex is mapped into the intermediate fgODTU.Mi payload according to the intermediate C_m value.

If $floor\left(\frac{X \times C_m(i)}{P_{server}}\right) + 118 \leq C_m(i)$,

$$Y = celling\left(\frac{floor\left(\frac{X \times C_m(i)}{P_{server}}\right)+118}{C_m(i)} \times P_{server}\right) \quad \text{(for case 1)}$$

else $$Y = celling\left(\frac{floor\left(\frac{X \times C_m(i)}{P_{server}}\right)+118-C_m(i)}{C_m(i+1)} \times P_{server}\right) \quad \text{(for case 2)}$$

where,

P_{server}: it denotes the size of fgODTU payload, and it is 256M 16-byte blocks.

$C_m(i)$: it denotes the used C_m value in fgODTU #i (for case 1) or #i-1 (for case 2) as described in Figure O.5, and it is corresponding to M×10M before fgODUflex bit rate changes.

$C_m(i+1)$: it denotes the used C_m value in fgODTU #i (for case 2) as described in Figure O.5, and it is corresponding to M×10M before fgODUflex bit rate changes.

X: it denotes the green block mapping position in fgODTU #i (for case 1) or #i-1 (for case 2) as described in Figure O.5, $1 \leq X \leq P_{server}$

Y: it denotes the blue block mapping position in fgODTU #i as described in Figure O.5, $1 \leq Y \leq P_{server}$

Appendix I

Range of stuff ratios for asynchronous mappings of CBR2G5, CBR10G, and CBR40G clients with ±20 ppm bit-rate tolerance into OPUk, and for asynchronous multiplexing of ODUj into ODUk (k > j)

(This appendix does not form an integral part of this Recommendation.)

Clause 17.2 describes asynchronous and bit-synchronous mappings of CBR2G5, CBR10G, and CBR40G clients with ±20 ppm bit-rate tolerance into ODU1, 2, and 3, respectively. Clause 19 describes asynchronous mapping (multiplexing) of ODUj into ODUk (k > j). For asynchronous CBR client mappings, any frequency difference between the client and local OPUk server clocks is accommodated by the +1/0/–1 justification scheme. For asynchronous multiplexing of ODUj into ODUk (k > j), any frequency difference between the client ODUj and local OPUk server clocks is accommodated by the +2/+1/0/–1 justification scheme. The OPUk payload, ODUk, and OTUk bit rates and tolerances are given in clause 7.3. The ODU1, ODU2, and ODU3 rates are 239/238, 239/237, and 239/236 times 2 488 320 kbit/s, 9 953 280 kbit/s, and 39 813 120 kbit/s, respectively. The ODUk bit-rate tolerances are ±20 ppm. This appendix shows that each justification scheme can accommodate these bit rates and tolerances for the respective mappings, and also derives the range of justification (stuff) ratio for each mapping.

The +1/0/–1 mapping in clause 17.2 provides for one positive justification opportunity (PJO) and one negative justification opportunity (NJO) in each ODUk frame. The +2/+1/0/–1 mapping in clause 19 provides for 2 PJOs and one NJO in each ODUk frame. For the case of ODU multiplexing (i.e., the latter case), the ODUj being mapped will get only a fraction of the full payload capacity of the ODUk. There can be, in general, a number of fixed stuff bytes per ODUj or CBR client. Note that in both mapping cases, there is one stuff opportunity in every ODUk frame. For mapping of a CBR client into ODUk, the CBR client is allowed to use all the stuff opportunities (because only one CBR client signal is mapped into an ODUk). However, for mapping ODUj into ODUk (k > j), the ODUj can only use 1/2 (ODU0 into ODU1), 1/4 (ODU1 into ODU2 or ODU2 into ODU3) or 1/16 (ODU1 into ODU3) of the stuff opportunities. The other stuff opportunities are needed for the other clients being multiplexed into the ODUk.

Traditionally, the justification ratio (stuff ratio) for purely positive justification schemes is defined as the long-run average fraction of justification opportunities for which a justification is done (i.e., for a very large number of frames, the ratio of the number of justifications to the total number of justification opportunities). In the +1/0/–1 scheme, positive and negative justifications must be distinguished. This is done by using different algebraic signs for positive and negative justifications. With this convention, the justification ratio can vary at most (for sufficiently large frequency offsets) from –1 to +1 (in contrast to a purely positive justification scheme, where the justification ratio can vary at most from 0 to 1). In the case of ODUk multiplexing, the justification ratio is defined relative to the stuff opportunities available for the client in question. Then, the justification ratio can vary (for sufficiently large frequency offsets) from –1 to +2. (If the justification ratio were defined relative to all the stuff opportunities for all the clients, the range would be –1/2 to +1 for multiplexing ODU0 into ODU1, –1/4 to +1/2 for multiplexing ODU1 into ODU2 and ODU2 into ODU3, and –1/16 to +1/8 for multiplexing ODU1 into ODU3.)

Let α represent the justification ratio ($-1 \le \alpha \le 1$ for CBR client into ODUk mapping; $-2 \le \alpha \le 1$ for ODUj into ODUk mapping (k > j)), and use the further convention that positive α will correspond to negative justification and negative α to positive justification (the reason for this convention is explained below).

Define the following notation (the index j refers to the possible ODUj client being mapped, and the index k refers to the ODUk server layer into which the ODUj or CBR client is mapped):

N = number of fixed stuff bytes in the OPUk payload area associated with the client in question (note that this is not the total number of fixed stuff bytes if multiple clients are being multiplexed)

S = nominal STM-N or ODUj client rate (bytes/s)

T = nominal ODUk frame period(s)

y_c = client frequency offset (fraction)

y_s = server frequency offset (fraction)

p = fraction of OPUk payload area available to this client

N_f = average number of client bytes mapped into an ODUk frame, for the particular frequency offsets (averaged over a large number of frames)

Then N_f is given by:

$$N_f = ST\frac{1+y_c}{1+y_s} \tag{I-1}$$

For frequency offsets small compared to 1, this may be approximated:

$$N_f = ST(1+y_c-y_s) \equiv ST\beta \tag{I-2}$$

The quantity β–1 is the net frequency offset due to client and server frequency offset.

Now, the average number of client bytes mapped into an ODUk frame is also equal to the total number of bytes in the payload area available to this client (which is (4)(3808)p = 15232p), minus the number of fixed stuff bytes for this client (N), plus the average number of bytes stuffed for this client over a very large number of frames. The latter is equal to the justification ratio α multiplied by the fraction of frames p corresponding to justification opportunities for this client. Combining this with equation I-1 produces:

$$ST\beta + \alpha p + 15232p - N \tag{I-3}$$

In equation I-3, a positive α corresponds to more client bytes mapped into the ODUk, on average. As indicated above, this corresponds to negative justification. This sign convention is used so that α enters in equation I-3 with a positive sign (for convenience).

Equation I-3 is the main result. For mapping STM-N (CBR clients) into ODUk, the quantity p is 1.

The range of stuff ratio may now be determined for mapping STM-N or ODUj clients into ODUk, using equation I-3. In what follows, let R_{16} be the STM-16 rate, i.e., 2.48832 Gbit/s = 3.1104×10^8 bytes/s.

Asynchronous mapping of CBR2G5 (2 488 320 kbit/s) signal into ODU1

The nominal client rate is $S = R_{16}$. The nominal ODU1 rate is (239/238)S (see clause 7.3). But the nominal ODU1 rate is also equal to (4)(3824)/T. Then:

$$\mathrm{ST} = (4)(3824)\frac{238}{239} = 15232 \tag{I-4}$$

Inserting this into equation I-3, and using the fact that N = 0 (no fixed stuff bytes) for this mapping produces:

$$\alpha = 15232(\beta - 1) \tag{I-5}$$

Since the ODUk and client frequency tolerances are each ±20 ppm, β ranges from 0.99996 to 1.00004. Using this in equation I-5 gives as the range of α:

$$-0.60928 \le \alpha \le +0.60928 \tag{I-6}$$

Asynchronous mapping of CBR10G (9 953 280 kbit/s) signal into ODU2

The nominal client rate is $S = 4R_{16}$. The nominal ODU2 rate is (239/237)*S* (see clause 7.3). But the nominal ODU2 rate is also equal to (4)(3824)/*T*. Then:

$$ST = (4)(3824)\frac{237}{239} = 15168 \tag{I-7}$$

Inserting this into equation I-3, and using the fact that $N = 64$ (number of fixed stuff bytes) for this mapping produces:

$$\alpha = 15168\beta + 64 - 15232 = 15168(\beta - 1) \tag{I-8}$$

As before, the ODUk and client frequency tolerances are ±20 ppm, and β ranges from 0.99996 to 1.00004. Using this in equation I-8 gives as the range of α:

$$-0.60672 \le \alpha \le +0.60672 \tag{I-9}$$

Asynchronous mapping of CBR40G (39 813 120 kbit/s) signal into ODU3

The nominal client rate is $S = 16R_{16}$. The nominal ODU3 rate is (239/236)*S* (see clause 7.3). But the nominal ODU3 rate is also equal to (4)(3824)/*T*. Then:

$$\mathrm{ST} = (4)(3824)\frac{236}{239} = 15104 \tag{I-10}$$

Inserting this into equation I-3, and using the fact that $N = 128$ (number of fixed stuff bytes) for this mapping produces:

$$\alpha = 15104\beta + 128 - 15232 = 15104(\beta - 1) \tag{I-11}$$

As before, the ODUk and client frequency tolerances are ±20 ppm, and β ranges from 0.99996 to 1.00004. Using this in equation I-11 gives as the range of α:

$$-0.60416 \le \alpha \le +0.60416 \tag{I-12}$$

ODU1 into ODU2 multiplexing

The ODU1 nominal client rate is (see clause 7.3):

$$\mathrm{S} = \frac{239}{238}\mathrm{R}_{16} \tag{I-13}$$

The ODU2 nominal frame time is:

$$\mathrm{T} = \frac{(3824)(4)}{\frac{239}{237}(4\mathrm{R}_{16})} \tag{I-14}$$

The fraction *p* is 0.25. Inserting into equation I-3 produces:

$$\frac{239}{238}\mathrm{R}_{16}\frac{(3824)(4)}{\frac{239}{237}(4\mathrm{R}_{16})}\beta = \frac{\alpha}{4} + 3808 - \mathrm{N} \tag{I-15}$$

Simplifying and solving for α produces:

$$\alpha = \frac{237}{238}(15296)\beta + 4N - 15232 \tag{I-16}$$

Now let β = 1 + *y*, where *y* is the net frequency offset (and is very nearly equal to $y_c - y_s$ for client and server frequency offset small compared to 1). Then:

$$\begin{aligned}\alpha &= \frac{237}{238}(15296) - 15232 + 4\mathrm{N} + \frac{237}{238}(15296)\mathrm{y} \\ &= 4\mathrm{N} - 0.2689076 + 15231.731092\mathrm{y}\end{aligned} \tag{I-17}$$

The number of fixed stuff bytes *N* is zero, as given in clause 19.5.1. The client and mapper frequency offsets are in the range ±20 ppm, as given in clause 7.3. Then, the net frequency offset *y* is in the range ±40 ppm. Inserting these values into equation I-17 gives for the range for α:

$$\begin{aligned}\alpha &= 0.340362 \quad \text{for} \quad y = +40\,\text{ppm} \\ \alpha &= -0.268908 \quad \text{for} \quad y = 0\,\text{ppm} \\ \alpha &= -0.878177 \quad \text{for} \quad y = -40\,\text{ppm}\end{aligned} \tag{I-18}$$

In addition, stuff ratios of −2 and +1 are obtained for frequency offsets of −113.65 ppm and 83.30 ppm, respectively. The range of frequency offset that can be accommodated is approximately 197 ppm. This is 50% larger than the range that can be accommodated by a +1/0/−1 justification scheme (see above), and is due to the additional positive stuff byte.

ODU2 into ODU3 multiplexing

The ODU2 nominal client rate is (see clause 7.3):

$$S = \frac{239}{237}(4R_{16}) \tag{I-19}$$

The ODU3 nominal frame time is:

$$T = \frac{(3824)(4)}{\frac{239}{236}(16R_{16})} \tag{I-20}$$

The fraction *p* is 0.25. Inserting into equation I-3 produces:

$$\frac{239}{237}4R_{16}\frac{(3824)(4)}{\frac{239}{236}(16R_{16})}\beta = \frac{\alpha}{4} + 3808 - N \tag{I-21}$$

Simplifying and solving for α produces:

$$\alpha = \frac{236}{237}(15296)\beta + 4N - 15232 \tag{I-22}$$

As before, let β = 1 + *y*, where *y* is the net frequency offset (and is very nearly equal to $y_c - y_s$ for client and server frequency offset small compared to 1). Then:

$$\begin{aligned}\alpha &= \frac{236}{237}(15296) - 15232 + 4N + \frac{236}{237}(15296)y \\ &= 4N - 0.5400844 + 15231.459916y\end{aligned} \tag{I-23}$$

The number of fixed stuff bytes N is zero, as given in clause 19.5.3. The client and mapper frequency offsets are in the range ±20 ppm, as given in clause 7.3. Then, the net frequency offset y is in the range ±40 ppm. Inserting these values into equation I-23 gives for the range for α:

$$\begin{aligned} \alpha &= 0.0691740 \quad \text{for} \quad y = +40\,\text{ppm} \\ \alpha &= -0.5400844 \quad \text{for} \quad y = 0\,\text{ppm} \\ \alpha &= -1.149343 \quad \text{for} \quad y = -40\,\text{ppm} \end{aligned} \tag{I-24}$$

In addition, stuff ratios of −2 and +1 are obtained for frequency offsets of −95.85 ppm and 101.11 ppm, respectively. As above, the range of frequency offset that can be accommodated is approximately 197 ppm, which is 50% larger than the range that can be accommodated by a +1/0/−1 justification scheme (see above) due to the additional positive stuff byte.

ODU1 into ODU3 multiplexing

The ODU1 nominal client rate is (see clause 7.3):

$$S = \frac{239}{238}(R_{16}) \tag{I-25}$$

The ODU3 nominal frame time is:

$$T = \frac{(3824)(4)}{\frac{239}{236}(16R_{16})} \tag{I-26}$$

The fraction p is 0.0625. Inserting into equation I-3 produces:

$$\frac{239}{238}R_{16}\frac{(3824)(4)}{\frac{239}{236}(16R_{16})}\beta = \frac{\alpha}{16} + 952 - N \tag{I-27}$$

Simplifying and solving for α produces:

$$\alpha = \frac{236}{238}(15296)\beta + 16N - 15232 \tag{I-28}$$

As before, let β = 1 + y, where y is the net frequency offset (and is very nearly equal to $y_c - y_s$ for client and server frequency offset small compared to 1). Then:

$$\begin{aligned} \alpha &= \frac{236}{238}(15296) - 15232 + 16N + \frac{236}{238}(15296)y \\ &= 16N - 64.5378151 + 15167.462185y \end{aligned} \tag{I-29}$$

The total number of fixed stuff bytes in the ODU3 payload is 64, as given in clause 19.5.2; the number for one ODU1 client, N, is therefore 4. The client and mapper frequency offsets are in the range ±20 ppm, as given in clause 7.3. Then, the net frequency offset y is in the range ±40 ppm. Inserting these values into equation I-29 gives for the range for α:

$$\begin{aligned} \alpha &= 0.0688834 \quad \text{for} \quad y = +40\,\text{ppm} \\ \alpha &= -0.5378151 \quad \text{for} \quad y = 0\,\text{ppm} \\ \alpha &= -1.144514 \quad \text{for} \quad y = -40\,\text{ppm} \end{aligned} \tag{I-30}$$

In addition, stuff ratios of −2 and +1 are obtained for frequency offsets of −96.40 ppm and 101.39 ppm, respectively. As above, the range of frequency offset that can be accommodated is approximately 197 ppm, which is 50% larger than the range that can be accommodated by a +1/0/−1 justification scheme (see above) due to the additional positive stuff byte.

ODU0 into ODU1 multiplexing

The ODU0 nominal client rate is (see clause 7.3):

$$S = \frac{1}{2}(R_{16}) \tag{I-31}$$

The ODU1 nominal frame time is:

$$T = \frac{(3824)(4)}{\frac{239}{238}(R_{16})} \tag{I-32}$$

The fraction p is 0.5. Inserting into equation I-3 produces:

$$\frac{1}{2}R_{16}\frac{(3824)(4)}{\frac{239}{238}(R_{16})}\beta = \frac{\alpha}{2} + 7616 - N \tag{I-33}$$

Simplifying and solving for α produces:

$$\alpha = \frac{238}{239}(15296)\beta + 2N - 15232 \tag{I-34}$$

As before, let $\beta = 1 + y$, where y is the net frequency offset (and is very nearly equal to $y_c - y_s$ for client and server frequency offset small compared to 1). Then:

$$\begin{aligned}\alpha &= \frac{238}{239}(15296) - 15232 + 2N + \frac{238}{239}(15296)y \\ &= 2N + 15232y\end{aligned} \tag{I-35}$$

The total number of fixed stuff bytes N is zero, as given in clause 19.5.4. The client and mapper frequency offsets are in the range ±20 ppm, as given in clause 7.3. Then, the net frequency offset y is in the range ±40 ppm. Inserting these values into equation I-35 gives for the range for α:

$$\begin{aligned}&\alpha = 0.6092800 && \text{for} \quad y = +40\,\text{ppm} \\ &\alpha = 0.0000000 && \text{for} \quad y = 0\,\text{ppm} \\ &\alpha = -0.6092800 && \text{for} \quad y = -40\,\text{ppm}\end{aligned} \tag{I-36}$$

In addition, stuff ratios of −2 and +1 are obtained for frequency offsets of −130 ppm and 65 ppm, respectively. As above, the range of frequency offset that can be accommodated is approximately 195 ppm.

Appendix II

Examples of functionally standardized OTU frame structures

(This appendix does not form an integral part of this Recommendation.)

This appendix provides examples of functionally standardized OTU frame structures. These examples are for illustrative purposes and by no means impy a definition of such structures. The completely standardized OTUk frame structure as defined in this Recommendation is shown in Figure II.1. Functionally standardized OTUkV frame structures will be needed to support, e.g., alternative FEC. Examples of OTUkV frame structures are:

– OTUkV with the same overhead byte allocation as the OTUk, but use of an alternative FEC as shown in Figure II.2;

– OTUkV with the same overhead byte allocation as the OTUk, but use of a smaller, alternative FEC code and the remainder of the OTUkV FEC overhead area filled with fixed stuff as shown in Figure II.3;

– OTUkV with a larger FEC overhead byte allocation as the OTUk, and use of an alternative FEC as shown in Figure II.4;

– OTUkV with no overhead byte allocation for FEC as shown in Figure II.5;

– OTUkV with a different frame structure than the OTUk frame structure, supporting a different OTU overhead (OTUkV overhead and OTUkV FEC) as shown in Figure II.6;

– OTUkV with a different frame structure than the OTUk frame structure, supporting a different OTU overhead (OTUkV overhead) and with no overhead byte allocation for FEC as shown in Figure II.7.

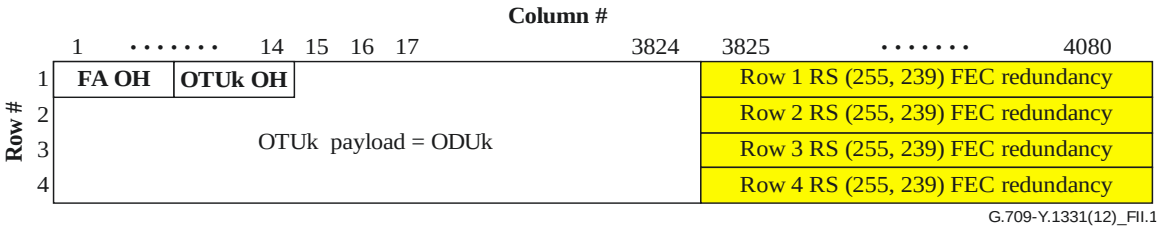


Figure II.1 – OTUk (with RS(255,239) FEC)

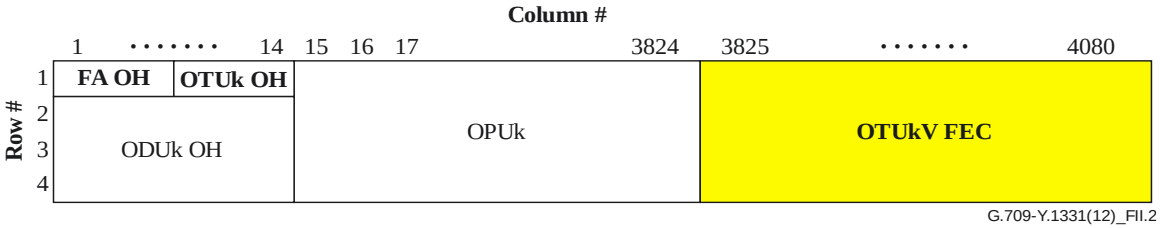


Figure II.2 – OTUk with alternative OTUkV FEC (OTUk-v)

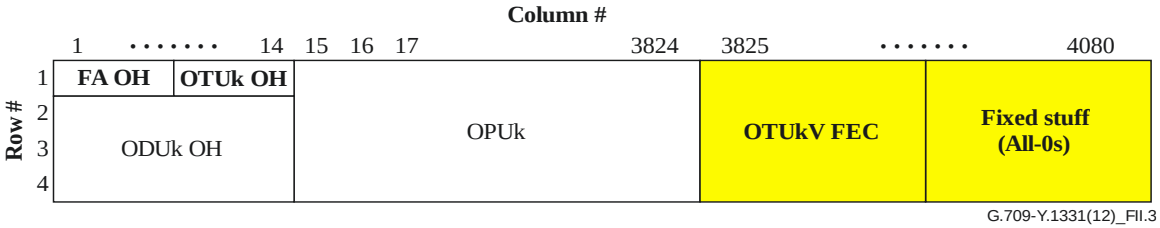


Figure II.3 – OTUk with a smaller OTUkV FEC and the remainder of an FEC area filled with fixed stuff

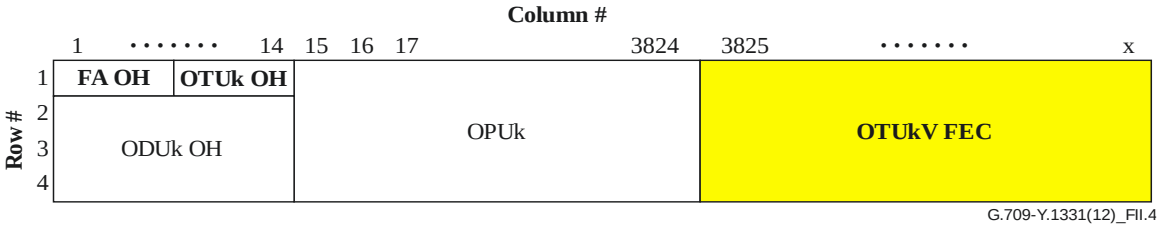


Figure II.4 – OTUk with a larger OTUkV FEC

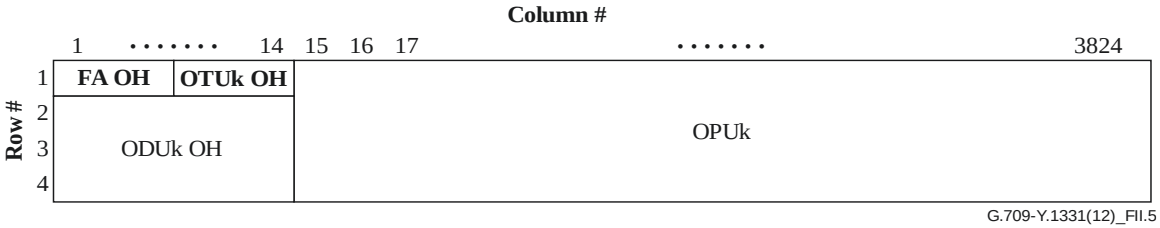


Figure II.5 – OTUk without an OTUkV FEC area

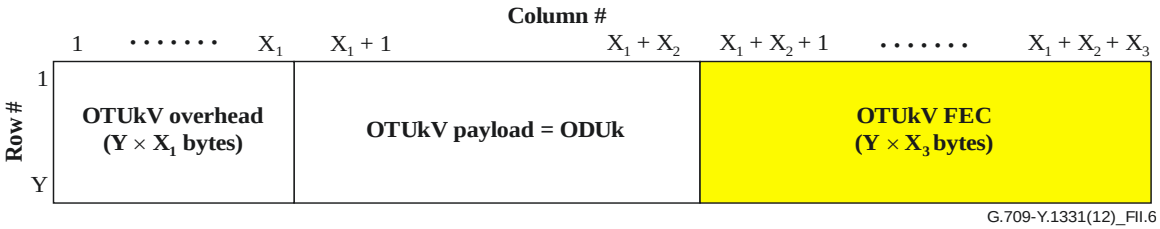


Figure II.6 – OTUkV with a different frame structure

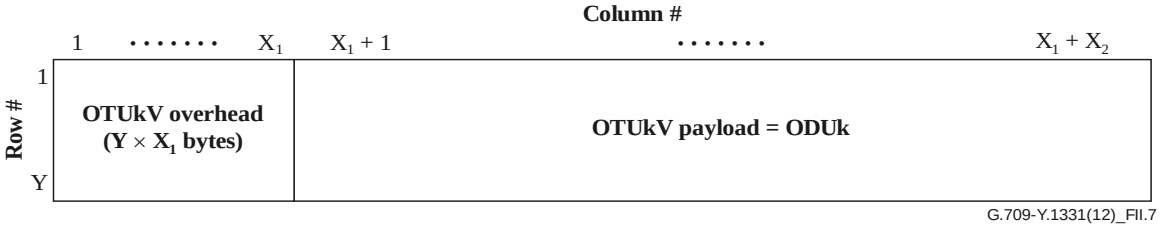


Figure II.7 – OTUkV with a different frame structure and without FEC area

For the case of Figures II.6 and II.7, the mapping of the ODUk signal can be either asynchronous, bit-synchronous, or frame synchronous.

For the case of asynchronous mapping, the ODUk and OTUkV bit rates can be asynchronous. The ODUk signal is mapped as a bit stream into the OTUkV payload area using a stuffing technique.

For the case of bit-synchronous mapping, the ODUk and OTUkV bit rates are synchronous. The ODUk signal is mapped into the OTUkV payload area without stuffing. The ODUk frame is not related to the OTUkV frame.

For the case of a frame synchronous mapping, the ODUk and OTUkV bit rates are synchronous and the frame structures are aligned. The ODUk signal is mapped into the OTUkV payload area without stuffing and with a fixed position of the ODUk frame within the OTUkV frame. (See Figure II.8.)

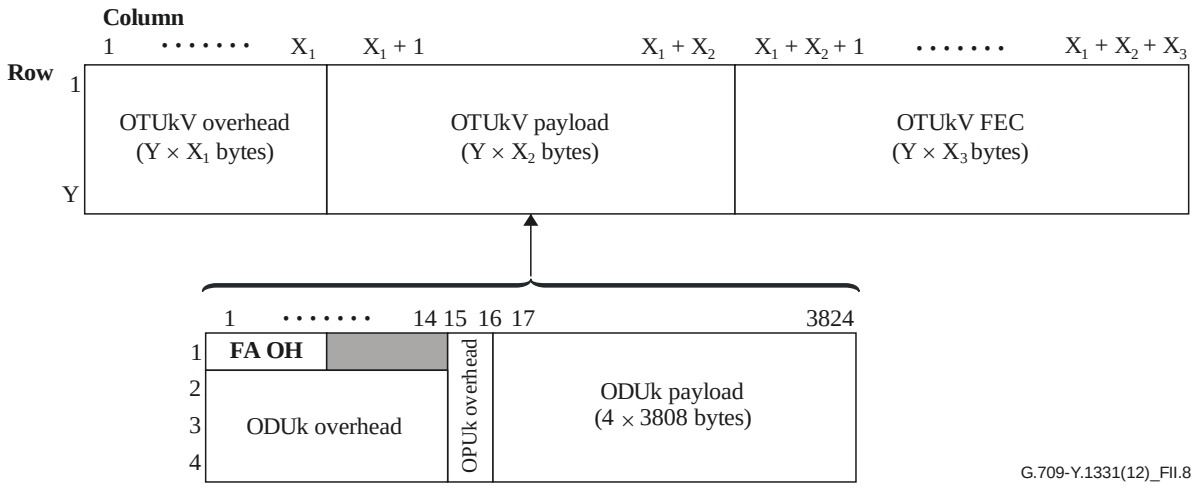


Figure II.8 – Asynchronous (or bit-synchronous) mapping of ODUk into OTUkV

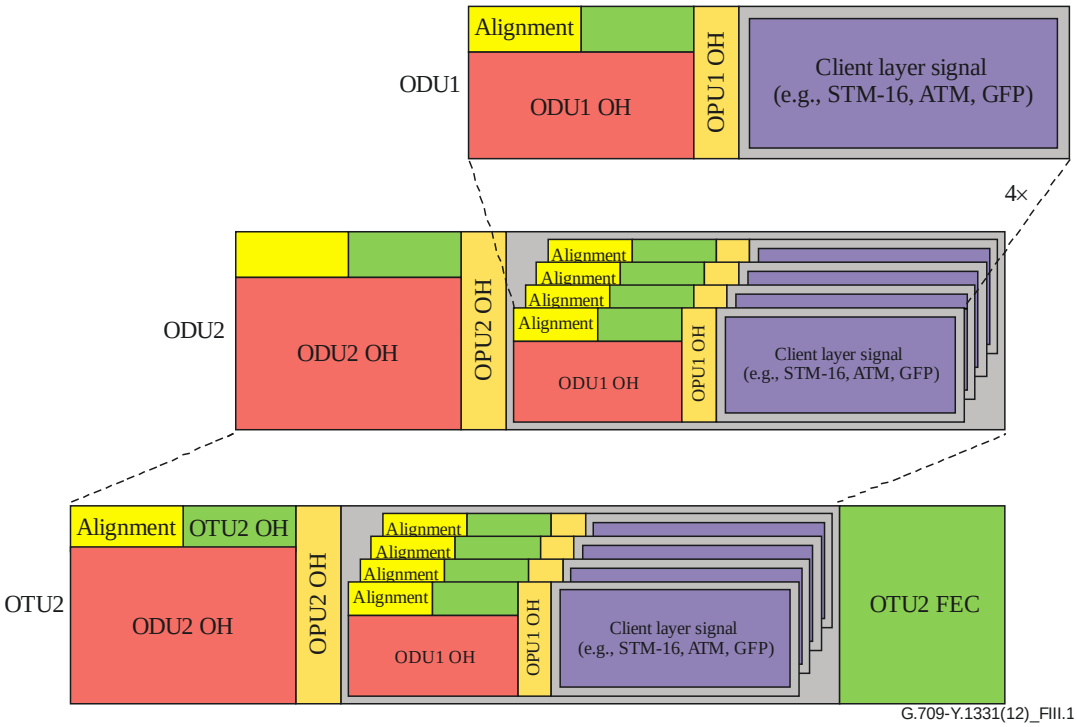
Appendix III

Example of ODUk multiplexing

(This appendix does not form an integral part of this Recommendation.)

Figure III.1 illustrates the multiplexing of four ODU1 signals into an ODU2. The ODU1 signals including the frame alignment overhead and an all-0s pattern in the OTUk overhead locations are adapted to the ODU2 clock via justification (asynchronous mapping). These adapted ODU1 signals are byte interleaved into the OPU2 payload area, and their justification control and opportunity signals (JC, NJO) are frame interleaved into the OPU2 overhead area.

ODU2 overhead is added after which the ODU2 is mapped into the OTU2 [or OTU2V]. OTU2 [or OTU2V] overhead and frame alignment overhead are added to complete the signal for transport via an OTM signal.



NOTE – The ODU1 floats in a quarter of the OPU2 payload area. An ODU1 frame will cross multiple ODU2 frame boundaries. A complete ODU1 frame (15296 bytes) requires the bandwidth of (15296/3808) 4.017 ODU2 frames. This is not illustrated.

Figure III.1 – Example of multiplexing 4 ODU1 signals into an ODU2

Figure III.2 illustrates the multiplexing of two ODU0 signals into an ODU1. The ODU0 signals including the frame alignment overhead and an all-0s pattern in the OTUk overhead locations are adapted to the ODU1 clock via justification (asynchronous mapping). These adapted ODU0 signals are byte interleaved into the OPU1 payload area, and their justification control and opportunity signals (JC, NJO) are frame interleaved into the OPU1 overhead area and ODU1 overhead is added.

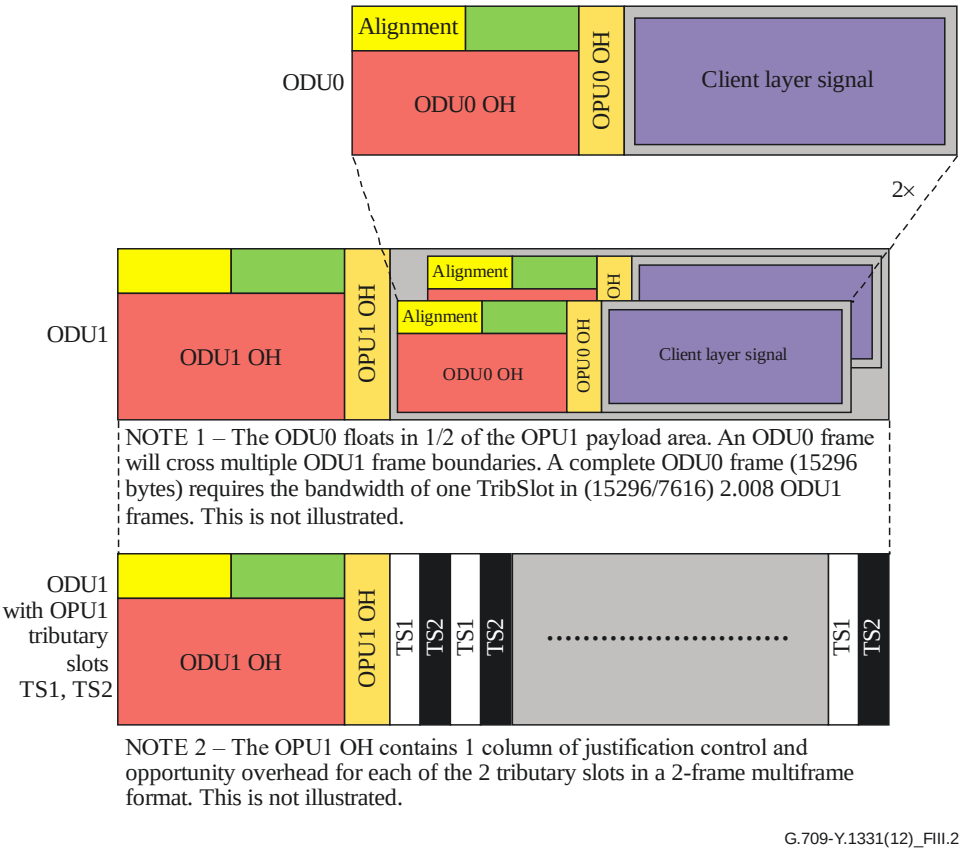


Figure III.2 – Example of multiplexing 2 ODU0 signals into an ODU1

Appendix IV

Blank appendix

This appendix is intentionally left blank.

Appendix V

ODUk multiplex structure identifier (MSI) examples

(This appendix does not form an integral part of this Recommendation.)

The following figures present four examples of ODU1 and ODU2 carriage within an OPU3 and the associated MSI encoding.

	1	2	3	4	5	6	7	8	
PSI[2]	00		000000						*TS*1
PSI[3]	00		000001						*TS*2
PSI[4]	00		000010						*TS*3
PSI[5]	00		000011						*TS*4
PSI[6]	00		000100						*TS*5
PSI[7]	00		000101						*TS*6
PSI[8]	00		000110						*TS*7
PSI[9]	00		000111						*TS*8
PSI[10]	00		001000						*TS*9
PSI[11]	00		001001						*TS*10
PSI[12]	00		001010						*TS*11
PSI[13]	00		001011						*TS*12
PSI[14]	00		001100						*TS*13
PSI[15]	00		001101						*TS*14
PSI[16]	00		001110						*TS*15
PSI[17]	00		001111						*TS*16

Figure V.1 – OPU3-MSI coding for the case of 16 ODU1s into OPU3

	1	2	3	4	5	6	7	8	
PSI[2]	01		000000						TS1
PSI[3]	01		000001						TS2
PSI[4]	01		000010						TS3
PSI[5]	01		000011						TS4
PSI[6]	01		000000						TS5
PSI[7]	01		000001						TS6
PSI[8]	01		000010						TS7
PSI[9]	01		000011						TS8
PSI[10]	01		000000						TS9
PSI[11]	01		000001						TS10
PSI[12]	01		000010						TS11
PSI[13]	01		000011						TS12
PSI[14]	01		000000						TS13
PSI[15]	01		000001						TS14
PSI[16]	01		000010						TS15
PSI[17]	01		000011						TS16

Figure V.2 – OPU3-MSI coding for the case of 4 ODU2s into OPU3 TS# (1, 5, 9, 13), (2, 6, 10, 14), (3, 7, 11, 15) and (4, 8, 12, 16)

	1	2	3	4	5	6	7	8	
PSI[2]	01		000000						*TS*1
PSI[3]	01		000001						*TS*2
PSI[4]	01		000001						*TS*3
PSI[5]	01		000010						*TS*4
PSI[6]	01		000000						*TS*5
PSI[7]	01		000011						*TS*6
PSI[8]	01		000011						*TS*7
PSI[9]	01		000011						*TS*8
PSI[10]	01		000000						*TS*9
PSI[11]	01		000000						*TS*10
PSI[12]	01		000001						*TS*11
PSI[13]	01		000001						*TS*12
PSI[14]	01		000011						*TS*13
PSI[15]	01		000010						*TS*14
PSI[16]	01		000010						*TS*15
PSI[17]	01		000010						*TS*16

Figure V.3 – OPU3-MSI coding for the case of 4 ODU2s into OPU3 TS# (1, 5, 9, 10), (2, 3, 11, 12), (4, 14, 15, 16) and (6, 7, 8, 13)

	1 2	3 4 5 6 7 8	
PSI[2]	01	000000	*TS*1
PSI[3]	00	000001	*TS*2
PSI[4]	00	000010	*TS*3
PSI[5]	01	000001	*TS*4
PSI[6]	01	000000	*TS*5
PSI[7]	00	000101	*TS*6
PSI[8]	00	000110	*TS*7
PSI[9]	01	000001	*TS*8
PSI[10]	01	000000	*TS*9
PSI[11]	01	000001	*TS*10
PSI[12]	00	001010	*TS*11
PSI[13]	00	001011	*TS*12
PSI[14]	01	000000	*TS*13
PSI[15]	00	001101	*TS*14
PSI[16]	00	001110	*TS*15
PSI[17]	01	000001	*TS*16

Figure V.4 – OPU3-MSI coding for the case of 5 ODU1s and 2 ODU2s into OPU3 TS# (2), (6), (11), (12), (14), (1, 5, 9, 13) and (4, 8, 10, 16) and OPU3 TS# 3, 7, 15 unallocated (default to ODU1)

Appendix VI

Parallel logic implementation of the CRC-9, CRC-8, CRC-5 and CRC-6

(This appendix does not form an integral part of this Recommendation.)

CRC-9

Table VI.1 illustrates example logic equations for a parallel implementation of the CRC-9 using the $g(x) = x^9 + x^3 + x^2 + 1$ polynomial over the ΣC_{nD} fields of JC1, JC2, JC4 and JC5. An "X" in a column of the table indicates that the message bit of that row is an input to the Exclusive-OR equation for calculating the CRC bit of that row. JC4.D1 corresponds to bit 2 of the JC4 mapping overhead octet, JC4.D2 corresponds to bit 3 of the JC4 octet, etc. (See Figure 20-7.) After computation, CRC bits crc1 to crc9 are inserted into the JC6 and JC3 octets with crc1 occupying bit 2 of the JC6 octet and crc9 occupying bit 2 of the JC3 octet.

Table VI.1 – Parallel logic equations for the CRC-9 implementation

Mapping overhead bits	CRC checksum bits								
	crc1	crc2	crc3	crc4	crc5	crc6	crc7	crc8	crc9
JC4.D1		X	X			X			
JC4.D2			X	X			X		
JC4.D3				X	X			X	
JC4.D4					X	X			X
JC4.D5	X					X		X	
JC4.D6		X					X		X
JC4.D7	X		X				X		
JC1.D8		X		X				X	
JC1.D9			X		X				X
JC5.D10	X			X		X	X	X	
JC5.D11		X			X		X	X	X
JC5.D12	X		X			X	X		X
JC5.D13	X	X		X					
JC5.D14		X	X		X				
JC5.D15			X	X		X			
JC5.D16				X	X		X		
JC2.D17					X	X		X	
JC2.D18						X	X		X

CRC-8

Table VI.2 illustrates example logic equations for a parallel implementation of the CRC-8 using the $g(x) = x^8 + x^3 + x^2 + 1$ polynomial over the JC1-JC2. An "X" in a column of the table indicates that the message bit of that row is an input to the Exclusive-OR equation for calculating the CRC bit of that row. JC1.C1 corresponds to the first bit (MSB) of the first mapping overhead octet (JC1), JC1.C2 corresponds to bit 2 of the first mapping overhead octet, etc. After computation, CRC bits crc1 to crc8 are inserted into the JC3 octet with crc1 occupying MSB and crc8 the LSB of the octet.

Table VI.2 – Parallel logic equations for the CRC-8 implementation

Mapping overhead bits	CRC checksum bits							
	crc1	crc2	crc3	crc4	crc5	crc6	crc7	crc8
JC1.C1		X				X		X
JC1.C2	X		X			X		
JC1.C3		X		X			X	
JC1.C4			X		X			X
JC1.C5	X			X			X	
JC1.C6		X			X			X
JC1.C7	X		X				X	
JC1.C8		X		X				X
JC2.C9	X		X		X	X	X	
JC2.C10		X		X		X	X	X
JC2.C11	X		X		X	X		X
JC2.C12	X	X		X				
JC2.C13		X	X		X			
JC2.C14			X	X		X		
JC2.II				X	X		X	
JC2.DI					X	X		X

CRC-5

Table VI.3 illustrates example logic equations for a parallel implementation of the CRC-5 using the $g(x) = x^5 + x + 1$ polynomial over the JC4-JC5 C_nD fields. An "X" in a column of the table indicates that the message bit of that row is an input to the Exclusive-OR equation for calculating the CRC bit of that row. JC4.D1 corresponds to the first bit (MSB) of the first mapping overhead octet (JC1), JC4.D2 corresponds to bit 2 of the first mapping overhead octet, etc. After computation, CRC bits crc1 to crc5 are inserted into the JC6 octet with crc1 occupying JC6 bit 4 and crc5 the JC6 bit 8.

Table VI.3 – Parallel logic equations for the CRC-5 implementation

Mapping overhead bits	CRC checksum bits				
	crc1	crc2	crc3	crc4	crc5
JC4.D1	X		X	X	
JC4.D2		X		X	X
JC4.D3	X		X		
JC4.D4		X		X	
JC4.D5			X		X
JC5.D6	X			X	X
JC5.D7	X	X			
JC5.D8		X	X		
JC5.D9			X	X	
JC5.D10				X	X

CRC-6

Table VI.4 illustrates example logic equations for a parallel implementation of the CRC-6 using the $g(x) = x^6 + x^3 + x^2 + 1$ polynomial over the JC1-JC2. An "X" in a column of the table indicates that the message bit of that row is an input to the Exclusive-OR equation for calculating the CRC bit of that row. JC1.C1 corresponds to bit 3 of the first mapping overhead octet (JC1), JC1.C2 corresponds to bit 4 of the first mapping overhead octet, etc. After computation, CRC bits crc1 to crc6 are inserted into the JC3 octet with crc1 occupying bit 3 and crc6 the LSB of the octet. (See Figure 20-7)

Table VI.4 – Parallel logic equations for the CRC-6 implementation

Mapping overhead bits	**CRC checksum bits**					
	crc1	**crc2**	**crc3**	**crc4**	**crc5**	**crc6**
JC1.C1			X		X	X
JC1.C2	X				X	X
JC1.C3	X	X		X	X	X
JC1.C4	X	X	X	X		X
JC1.C5	X	X	X			
JC1.C6		X	X	X		
JC2.C7			X	X	X	
JC2.C8				X	X	X
JC2.C9	X			X		X
JC2.C10	X	X		X		
JC2.II		X	X		X	
JC2.DI			X	X		X

Appendix VII

OTL4.10 structure

(This appendix does not form an integral part of this Recommendation.)

The information of this appendix is included in [b-ITU-T G-Sup.58].

Appendix VIII

CPRI into ODU mapping

(This appendix does not form an integral part of this Recommendation.)

The information of this appendix is included in [b-ITU-T G-Sup.56].

Appendix IX

Overview of CBR clients into OPU mapping types

(This appendix does not form an integral part of this Recommendation.)

As there are many different constant bit rate client signals and multiple mapping procedures, Table IX.1 provides an overview of the mapping procedure that is specified for each client.

Table IX.1 – Overview of CBR client into OPUk mapping types

	OPU0	OPU1	OPU2	OPU2e	OPU3	OPU4	OPUflex
STM-1	GMP with C_{1D}	–	–	–	–	–	–
STM-4	GMP with C_{1D}	–	–	–	–	–	–
STM-16	–	AMP, BMP	–	–	–	–	–
STM-64	–	–	AMP, BMP	–	–	–	–
STM-256	–	–	–	–	AMP, BMP	–	–
1000BASE-X	TTT+GMP no C_{nD}	–	–	–	–	–	–
10GBASE-R	–	–	–	16FS+BMP	–	–	–
25GBASE-R	–	–	–	–	–	–	BMP
40GBASE-R	–	–	–	–	TTT+GMP with C_{8D}	–	–
50GBASE-R	–	–	–	–	–	–	BMP
100GBASE-R	–	–	–	–	–	GMP with C_{8D}	–
200GBASE-R	–	–	–	–	–	–	BMP
400GBASE-R	–	–	–	–	–	–	BMP
800GBASE-R	–	–	–	–	–	–	BMP (Note 2)
FC-100	GMP no C_{nD}	–	–	–	–	–	–
FC-200	–	GMP with C_{8D}	–	–	–	–	–
FC-400	–	–	–	–	–	–	BMP
FC-800	–	–	–	–	–	–	BMP
FC-1200	–	–	–	TTT+16FS+ BMP (Note 1)	–	–	–
FC-1600	–	–	–	–	–	–	BMP
FC-3200	–	–	–	–	–	–	BMP
CM_GPON	–	AMP	–	–	–	–	–

Table IX.1 – Overview of CBR client into OPUk mapping types

	OPU0	OPU1	OPU2	OPU2e	OPU3	OPU4	OPUflex
CM_XGPON	–	–	AMP	–	–	–	–
IB SDR	–	–	–	–	–	–	BMP
IB DDR	–	–	–	–	–	–	BMP
IB QDR	–	–	–	–	–	–	BMP
SBCON/ESCON	GMP no C_{nD}	–	–	–	–	–	–
DVB_ASI	GMP no C_{nD}	–	–	–	–	–	–
SDI	GMP TBD C_{nD}	–	–	–	–	–	–
1.5G SDI	–	GMP TBD C_{nD}	–	–	–	–	–
3G SDI	–	–	–	–	–	–	BMP
FlexE Client	–	–	–	–	–	–	IMP
FlexE-aware	–	–	–	–	–	–	BGMP
NOTE 1 – For this specific case the mapping used is byte synchronous. NOTE 2 – This mapping includes padding and deterministic stuffing.							

Appendix X

Overview of ODUj into OPUk mapping types

(This appendix does not form an integral part of this Recommendation.)

As there are many different ODUj bit rate signals and multiple mapping procedures, Table X.1 provides an overview of the mapping procedure that is specified for each ODUj.

Table X.1 – Overview of ODUj client into OPUk mapping types

	2.5G tributary slots		**1.25G tributary slots**					
	OPU2	**OPU3**	**OPU1**	**OPU2**	**OPU3**	**OPU4**	**OPU25**	**OPU50**
ODU0	–	–	ODTU01 AMP (PT=20)	ODTU2.1 GMP (PT=21)	ODTU3.1 GMP (PT=21)	ODTU4.1 GMP (PT=21)	ODTU25.1 GMP (PT=21)	ODTU50.1 GMP (PT=21)
ODU1	ODTU12 AMP (PT=20)	ODTU13 AMP (PT=20)	–	ODTU12 AMP (PT=21)	ODTU13 AMP (PT=21)	ODTU4.2 GMP (PT=21)	ODTU25.2 GMP (PT=21)	ODTU50.2 GMP (PT=21)
ODU2	–	ODTU23 AMP (PT=20)	–	–	ODTU23 AMP (PT=21)	ODTU4.8 GMP (PT=21)	ODTU25.8 GMP (PT=21)	ODTU50.8 GMP (PT=21)
ODU2e	–	–	–	–	ODTU3.9 GMP (PT=21)	ODTU4.8 GMP (PT=21)	ODTU25.8 GMP (PT=21)	ODTU50.8 GMP (PT=21)
ODU3	–	–	–	–	–	ODTU4.31 GMP (PT=21)	–	ODTU50.31 GMP (PT=21)
ODUflex	–	–	–	ODTU2.ts GMP (PT=21)	ODTU3.ts GMP (PT=21)	ODTU4.ts GMP (PT=21)	ODTU25.ts GMP (PT=21)	ODTU50.ts GMP (PT=21)
ODUflex(IB SDR)	–	–	–	ODTU2.3 GMP (PT=21)	ODTU3.3 GMP (PT=21)	ODTU4.2 GMP (PT=21)	ODTU25.2 GMP (PT=21)	ODTU50.2 GMP (PT=21)
ODUflex(IB DDR)	–	–	–	ODTU2.5 GMP (PT=21)	ODTU3.5 GMP (PT=21)	ODTU4.4 GMP (PT=21)	ODTU25.4 GMP (PT=21)	ODTU50.4 GMP (PT=21)
ODUflex(IB QDR)	–	–	–	–	ODTU3.9 GMP (PT=21)	ODTU4.8 GMP (PT=21)	ODTU25.8 GMP (PT=21)	ODTU50.8 GMP (PT=21)
ODUflex(FC-400)	–	–	–	ODTU2.4 GMP (PT=21)	ODTU3.4 GMP (PT=21)	ODTU4.4 GMP (PT=21)	ODTU25.4 GMP (PT=21)	ODTU50.4 GMP (PT=21)
ODUflex(FC-800)	–	–	–	ODTU2.7 GMP (PT=21)	ODTU3.7 GMP (PT=21)	ODTU4.7 GMP (PT=21)	ODTU25.7 GMP (PT=21)	ODTU50.7 GMP (PT=21)
ODUflex(FC-1600)	–	–	–	–	ODTU3.12 GMP (PT=21)	ODTU4.11 GMP (PT=21)	ODTU25.11 GMP (PT=21)	ODTU50.11 GMP (PT=21)
ODUflex(FC-3200)	–	–	–	–	ODTU3.23 GMP (PT=21)	ODTU4.22 GMP (PT=21)	–	ODTU50.22 GMP (PT=21)

Table X.1 – Overview of ODUj client into OPUk mapping types

	2.5G tributary slots		1.25G tributary slots					
	OPU2	**OPU3**	**OPU1**	**OPU2**	**OPU3**	**OPU4**	**OPU25**	**OPU50**
ODUflex(25G BASE-R)	–	–	–	–	ODTU3.21 GMP (PT=21)	ODTU4.20 GMP (PT=21)	ODTU25.20 GMP (PT=21)	ODTU50.20 GMP (PT=21)
ODUflex(50G BASE-R)	–	–	–	–	–	ODTU4.40 GMP (PT=21)	–	ODTU50.40 GMP (PT=21)
ODUflex(GFP)	–	–	–	ODTU2.ts (GMP) (PT=21)	ODTU3.ts (GMP) (PT=21)	ODTU4.ts (GMP) (PT=21)	ODTU25.ts (GMP) (PT=21)	ODTU50.ts (GMP) (PT=21)
ODUflex(GFP ,n,2), n=1, … ,8 (ts=n)	–	–	–	ODTU2.ts (GMP) (PT=21)	ODTU3.ts (GMP) (PT=21)	ODTU4.ts (GMP) (PT=21)	ODTU25.ts (GMP) (PT=21)	ODTU50.ts (GMP) (PT=21)
ODUflex(GFP ,n,3), n=9, … ,20 (ts=n)	–	–	–	–	ODTU3.ts (GMP) (PT=21)	ODTU4.ts (GMP) (PT=21)	ODTU25.ts (GMP) (PT=21)	ODTU50.ts (GMP) (PT=21)
ODUflex(GFP ,n,3), n=21, … ,32 (ts=n)	–	–	–	–	ODTU3.ts (GMP) (PT=21)	ODTU4.ts (GMP) (PT=21)	–	ODTU50.ts (GMP) (PT=21)
ODUflex(GFP), n=33, … ,40 (ts=n)	–	–	–	–	–	ODTU4.ts (GMP) (PT=21)	–	ODTU50.ts (GMP) (PT=21)
ODUflex(GFP ,n,4), n=41, … ,80 (ts=n)	–	–	–	–	–	ODTU4.ts (GMP) (PT=21)	–	–
ODUflex(IMP)	–	–	–	ODTU2.ts (GMP) (PT=21)	ODTU3.ts (GMP) (PT=21)	ODTU4.ts (GMP) (PT=21)	ODTU25.ts (GMP) (PT=21)	ODTU50.ts (GMP) (PT=21)
ODUflex (IMP,s), s=2	–	–	–	–	ODTU3.9 (GMP) (PT=21)	ODTU4.8 (GMP) (PT=21)	ODTU25.8 (GMP) (PT=21)	ODTU50.8 (GMP) (PT=21)
ODUflex(IMP, s), s=5	–	–	–	–	ODTU3.21 (GMP) (PT=21)	ODTU4.20 (GMP) (PT=21)	ODTU25.20 (GMP) (PT=21)	ODTU50.20 (GMP) (PT=21)
ODUflex(IMP, s), s=8	–	–	–	–	–	ODTU4.32 (GMP) (PT=21)	–	ODTU50.32 (GMP) (PT=21)
ODUflex(IMP, s), s=10	–	–	–	–	–	ODTU4.40 (GMP) (PT=21)	–	ODTU50.40 (GMP) (PT=21)
ODUflex(IMP, s), s=15,20 (ts=4×s)	–	–	–	–	–	ODTU4.ts (GMP) (PT=21)	–	–

Appendix XI

Derivation of recommended ODUflex(GFP) bit-rates based on n × ODUk.ts clock and examples of ODUflex(GFP,n,k) clock generation

(This appendix does not form an integral part of this Recommendation.)

XI.1 Introduction

The recommended bit-rates for ODUflex(GFP,n,k) are provided in Table 7-8. While in principle an ODUflex(GFP) may be of any bit-rate, there were at the time of the introduction of the ODUflex a variety of reasons for recommending particular rates for the ODUflex(GFP,n,k):

– To encourage a common set of bit-rates which can be expected to be supported by multiple manufacturers.

– To provide the largest amount of bandwidth possible within a given amount of resource (number of tributary slots) independent of the ODUk over which the ODUflex(GFP,n,k) may be routed.

– To maintain the number of tributary slots required if the ODUflex(GFP,n,k) must be rerouted, e.g., during a restoration.

– To satisfy a protocol requirement for ODUflex hitless resizing that a resizable ODUflex(GFP,n,k) must occupy the same number of tributary slots on every ODUk path over which it is carried, and that a resize operation must always add or remove at least one tributary slot.

XI.2 Tributary slot sizes

ODUflex(GFP,n,k) is mapped via GMP into a certain number of 1.25G tributary slots of a HO OPU2, OPU3, or OPU4 . Each of these have different tributary slot sizes:

$$OPU2\_TS = \frac{238}{237} \times 4 \times STM16 \times \frac{476 columns}{3808 columns} = 1249409.620 \text{ kbit/s} \pm 20\text{ppm}$$

$$OPU3\_TS = \frac{238}{236} \times 16 \times STM16 \times \frac{119 columns}{3808 columns} = 1254703.729 \text{ kbit/s} \pm 20\text{ppm}$$

$$OPU4\_TS = \frac{238}{227} \times 40 \times STM16 \times \frac{47.5 columns}{3808 columns} = 1301709.251 \text{ kbit/s} \pm 20\text{ppm}$$

An ODUflex(GFP,n,2) that occupies 8 or fewer tributary slots may be routed over OPU2, OPU3, or OPU4. The smallest tributary slot that may be encountered along the route of the ODUflex(GFP,n,2) is that of OPU2. Even if the initially selected route does not chose a link of OPU2, the ODUflex(GFP,n,2) should be sized to a multiple of the OPU2 tributary slot size to preserve the possibility to restore the ODUflex(GFP,n,2) over a route that includes OPU2 without changing the size of the ODUflex or the number of tributary slots it occupies.

An ODUflex(GFP,n,3) that occupies at least 9, but no more than 32 tributary slots may be routed over OPU3 or OPU4. It does not fit over OPU2. Therefore such an ODUflex(GFP,n,3) may be sized to a multiple of the OPU3 tributary slot size. Even if the initially selected route does not chose a link of OPU3, the ODUflex(GFP,n,3) should be sized to a multiple of the OPU3 tributary slot size to preserve the possibility to restore the ODUflex(GFP,n,3) over a route that includes OPU3 without changing the size of the ODUflex or the number of tributary slots it occupies.

An ODUflex(GFP,n,4) that occupies at least 33, but no more than 80 tributary slots may only be carried via OPU4, and may therefore take advantage of the full size of the OPU4 tributary slot size.

A small margin must be left between the ODUflex(GFP,n,k) size and the integral multiple of the tributary slot size to accommodate possible clock variation along a sequence of OPUk links without overflowing the range of C_m in the GMP mapper.

Physical layers for data interfaces such as Ethernet and Fibre Channel have historically used a clock tolerance of ±100 ppm. This range is sufficiently wide that specifying this as the clock tolerance for ODUflex(GFP,n,k) can accommodate a variety of mechanisms for generating an ODUflex(GFP,n,k) clock and remain within the clock tolerance range.

ODUk.ts as shown in Table 7-8 is an increment of bandwidth, which, when multiplied by a number of 1.25G tributary slots, gives the recommended size of an ODUflex(GFP,n,k) optimized to occupy a given number of tributary slots of a server OPUk. These values are chosen to allow sufficient margin that allows the OPUk and the ODUflex(GFP,n,k) to independently vary over their full clock tolerance range without exceeding the capacity of the allocated tributary slots.

The nominal values for ODUk.ts are chosen to be 186 ppm below the bandwidth of a single 1.25G tributary slot of a server OPUk. This allows the ODUflex(GFP,n,k) clock to be as much as 100 ppm above its nominal rate and the server OPUk to be as much as 20 ppm below its nominal clock rate, allowing approximately 66 ppm of margin to accommodate jitter and to ensure that the largest average C_m value even in the worst-case situation of the OPUk at -20 ppm from its nominal value and the ODUflex(GFP,n,k) at +100 ppm from its nominal value will be one less than the maximum value (i.e., the maximum average C_m is no more than 15231 out of 15232 for ODUflex(GFP,n,k) carried over OPU2 or OPU3, and no more than 15199 out of 15200 for ODUflex(GFP,n,k) carried over OPU4).

Table XI.1 – Generation of ODUflex(GFP,n,k) clock from server OPUk clock using fixed C_m

OPUk	Nominal payload bit rate	Nominal 1.25G TS bit rate		ODU2.ts							
				C_m out of	Bit-rate per TS						
OPU2	9'995'276.962	1'249'409.620		15230	1'249'245.570						
				15232			ODU3.ts				
			+20 ppm	1'249'270.555		C_m out of	Bit-rate per TS				
			–20 ppm	1'249'220.585							
OPU3	40'150'519.322	1'254'703.729		15165	1'249'184.746	15230	1'254'538.983				
				15232		15232			ODU4.ts		
			+20 ppm	1'249'209.729			1'254'564.074	C_m out of	Bit-rate per TS		
			–20 ppm	1'249'159.762			1'254'513.892				
OPU4	104'355'975.330	1'301'709.251		14587	1'249'212.687	14649	1'254'522.291	15198	1'301'537.974		
				15200		15200		15200			
			+20 ppm	1'249'237.671			1'254'547.381		1'301'564.004		
			–20 ppm	1'249'187.703			1'254'497.200		1'301'511.943		
		ODUk.ts	nominal	1'249'177.230			1'254'470.354		1'301'467.133		
			+100 ppm	1'249'302.148			1'254'595.801		1'301'597.280		
			–100 ppm	1'249'052.312			1'254'344.907		1'301'336.986		

XI.3 Example methods for ODUflex(GFP,n,k) clock generation

XI.3.1 Generating ODUflex(GFP,n,k) clock from OPUk clock

The clock for an ODUflex(GFP,n,k) may be generated from the initial server OPUk over which the ODUflex(GFP,n,k) is carried by setting the value of C_m to a fixed value on the initial segment. Normal GMP processing on subsequent segments avoids the need to couple the server OPUk clocks along the path.

Table XI-1 illustrates how a clock for an ODUflex(GFP,n,k) occupying n × ODUk.ts can be derived from the server OPUk clock using a fixed value of C_m in the initial segment of the path.

For example, for an ODUflex(GFP,n,k) occupying up to 8 tributary slots should be based on ODU2.ts, and therefore have a clock frequency of n × 1'249'177.230 kbit/s ± 100 ppm. This allows the ODUflex(GFP,n,k) to have a frequency of between n × 1'249'052.312 kbit/s and n × 1'249'302.148 kbit/s.

– If the initial segment over which the ODUflex(GFP,n,k) is carried is an OPU2, a clock in this range can be generated by fixing the value of C_m on the initial segment to 15230, which will result in the ODUflex having a clock of n × 1'249'245.570 kbit/s ± 20 ppm. While the center frequency of this range differs from the nominal value of ODU2.ts, the clock tolerance is narrower, being locked to the OPU2, so the possible clock range is fully within the ±100 ppm range allowed.
– If the initial segment is an OPU3, the ODUflex(GFP,n,k) of a multiple of ODU2.ts can be generated using a fixed value of C_m=15165 on the initial ODU3 segment, which will result in the ODUflex having a clock of n × 1'249'184.746 kbit/s ± 20 ppm,
– If the initial segment is an OPU4, the ODUflex(GFP,n,k) of a multiple of ODU2.ts can be generated using a fixed value of C_m=14587 on the initial OPU4 segment, which will result in the ODUflex having a clock of n × 1'249'212.687 kbit/s ± 20 ppm.

The center frequencies of all of these ODUflex(GFP,n,k) are slightly different, but the resulting ranges for the clocks all fall within the ±100 ppm window (see Figure XI.1). Fixed C_m for generating ODU3.ts and ODU4.ts from the initial OPUk can similarly be found from this table.

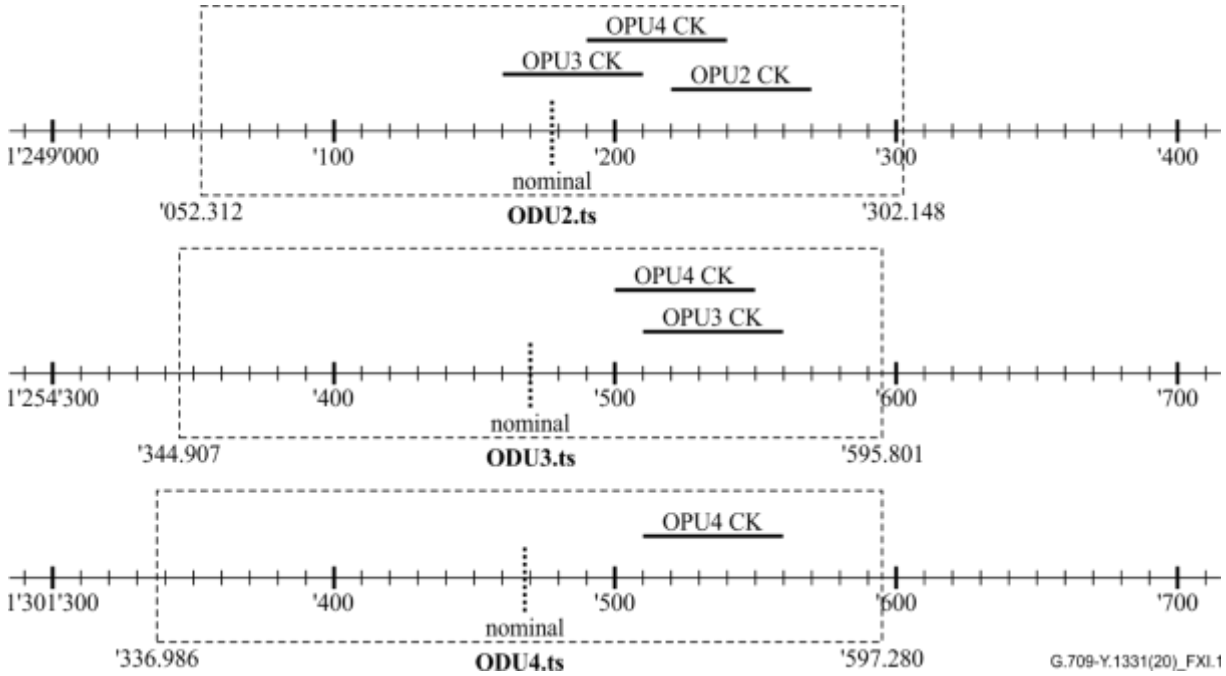


Figure XI.1 – Graphical representation of frequency ranges in Table XI.1

XI.3.2 Generating ODUflex(GFP,n,k) clock from system clock

The clock for an ODUflex(GFP,n,k) may be generated using a multiplier from the internal system clock. Normally the internal system clock will have an accuracy of at least ±20 ppm, perhaps even ±4.6 ppm for a network element that supports both SDH and OTN interfaces. The exact multiplier to be used is implementation specific, and be chosen so that the range of the generated clock falls within the specified ±100 ppm window around the nominal value of n × ODUk.ts. Refer to Appendix XV for further information.

Appendix XII

Terminology changes between ITU-T G.709 Edition 4, Edition 5 and Edition 6

(This appendix does not form an integral part of this Recommendation.)

In Edition 5 of this Recommendation a number of terms used in Editions 1 to 4 have been modified. In Edition 6 of this Recommendation a number of terms used in Edition 5 have been modified. Table XII.1 provides the mapping between these terms in Editions 1 to 4, Edition 5, and Edition 6.

Table XII.1 – Terminology mapping

ITU-T G.709 Edition 1 to 4	ITU-T G.709 Edition 5	ITU-T G.709 Edition 6
OTM-0.m	SOTU	OTN point-to-point interface, type I
OTM-0.mvn	(multi-lane) SOTU	OTN point-to-point interface, type I
OTM-nr.m	MOTU	OTN point-to-point interface, type II
OTM-1.m	SOTUm	OTN optical networking interface, type II
OTM-n.m	MOTUm	OTN optical networking interface, type I
Optical Channel (OCh)	OCh	OCh
Optical Channel with reduced functionality (OChr)	OCh-P	OTSiG
Optical Channel Payload (OCh-P)	OCh-P	OTSiG
Optical Channel Overhead (OCh-OH)	OCh-O	OCh-O
Optical channel Transport Unit (OTU)	Optical Transport Unit (OTU)	Optical Transport Unit (OTU)
Optical channel Data Unit (ODU)	Optical Data Unit (ODU)	Optical Data Unit (ODU)
Optical channel Payload Unit (OPU)	Optical Payload Unit (OPU)	Optical Payload Unit (OPU)
Optical channel Data Tributary Unit (ODTU)	Optical Data Tributary Unit (ODTU)	Optical Data Tributary Unit (ODTU)
Optical channel Data Tributary Unit Group (ODTUG)	Optical Data Tributary Unit Group (ODTUG)	Optical Data Tributary Unit Group (ODTUG)
Optical Multiplex Unit (OMU)	–	–
Optical Channel Carrier (OCC)	–	–
Optical Channel Carrier – overhead (OCCo)	–	–
Optical Channel Carrier – payload (OCCp)	–	–
Optical Channel Carrier with reduced functionality (OCCr)	–	–
Optical Carrier Group (OCG)	–	–
Optical Carrier Group with reduced functionality (OCGr)	–	–
Optical Carrier Group of order n (OCG)	–	–

Table XII.1 – Terminology mapping

ITU-T G.709 Edition 1 to 4	ITU-T G.709 Edition 5	ITU-T G.709 Edition 6
OTM Overhead Signal (OOS)	Optical Supervisory Channel (OSC)	Optical Supervisory Channel (OSC)
Optical Transport Lane (OTLk.n)	OTLk.n lane	OTLk.n lane
Optical Transport Lane Carrier Group (OTLCG)	–	–
Optical Transport Lane Carrier (OTLC)	–	–
OPS0	OPS	–
OPSn	OPS	–
OPSMnk	OPS	–
OTS-OH	OTS-O	OTS-O
OMS-OH	OMS-O	OMS-O
ODUflex(GFP)	ODUflex(GFP)	ODUflex(GFP,n,k)
–	ODUflex(IMP)	ODUflex(IMP,s)

In Edition 5.0 of this Recommendation the following new terms are introduced:

Table XII.2 – New terms in Edition 5

Optical Tributary Signal (OTSi)
Optical Tributary Signal Group (OTSiG)
Optical Tributary Signal Group – Overhead (OTSiG-O)
Optical Tributary Signal Assembly (OTSiA)

The relationship between these new terms in and the OCh, OChr, OPS0, OPSn and OPSM terms is as follows:

Table XII.3 – Relationship between OCh, OChr, OPS0, OPSn, OPSM and OTSi terminology

OCh-P	OTSiG with one or more OTSi
OCh-O	Reduced version of OTSiG-O: – FDI-P, FDI-O and OCI: supported – TTI, BDI-P, BDI-O, TSI: not supported
OCh	Reduced version of OTSiA: – OTSiG with one or more OTSi – FDI-P, FDI-O and OCI: supported – TTI, BDI-P, BDI-O, TSI: not supported
OChr	OTSiG with one OTSi; no OTSiG-O
OPS0	OTSiG with one OTSi; no OTSiG-O
OPSMnk	OTSiG with m OTSi; no OTSiG-O
OPSn	n OTSiG and no OTSiG-O; each OTSiG with one OTSi

Appendix XIII

OTUCn sub rates (OTUCn-M) Applications

(This appendix does not form an integral part of this Recommendation.)

XIII.1 Introduction

This appendix describes OTUCn-M application scenarios for the OTUCn-M signal defined in Annex H. It is assumed that the OTUCn-M processing is performed in the PHY module so that the interface between the module and framer (MFI) is not changed.

Figures XIII.1 and XIII.2 illustrate two scenarios for subrating applications. The first scenario deploys subrating between two line ports connecting two ODU cross connects (XC). The second scenario deploys subrating between transponders which are in a different domain B, which are separated from the ODU XCs in domain A and/or C. The transponders could either relay the OTUCn_CI or the ODUCn_CI. In the former case, the OTUCn between domain B and domain A or C continue in domain B as OTUCn-M. In the latter case, the OTUCn and OTUCn-M are terminated in the two transponders and the ODUCn is forwarded.

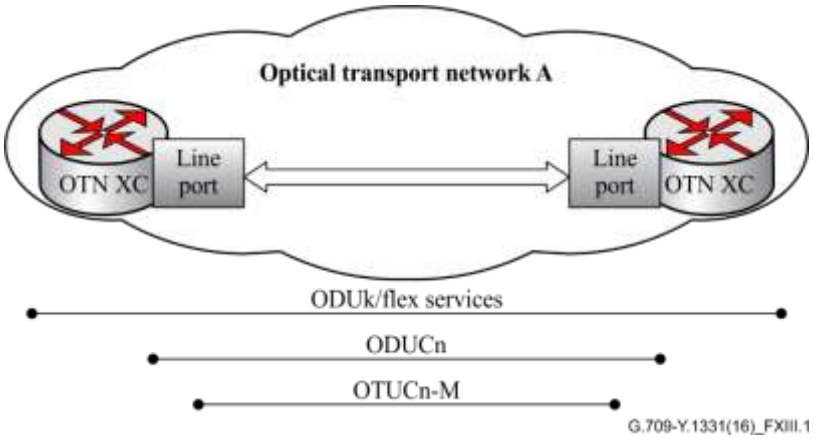


Figure XIII.1 – OTUCn sub rate application scenario A

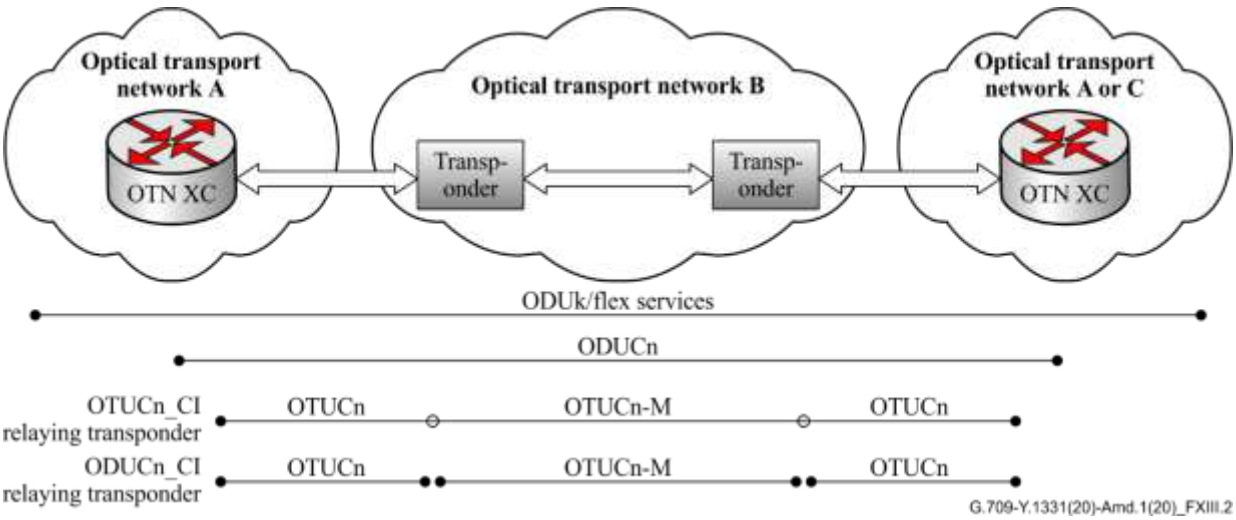


Figure XIII.2 – OTUCn sub rate application scenario B

XIII.2 OTUCn-M frame format and rates

As defined in Annex H, the OTUCn-M frame is a type of OTUCn frame which contains n instances of OTUC, ODUC and OPUC overhead and M 5 Gbit/s OPUCn tributary slots. If a particular value of M is not indicated, the frame contains 20×n tributary slots.

The use of this notation is illustrated below:

– OTUC2-25: a 125 Gbit/s OTU signal with 25 (5 Gbit/s) tributary slots and 2 instances of OxUC overhead

– OTUC2-30: a 150 Gbit/s OTU signal with 30 (5 Gbit/s) tributary slots and 2 instances of OxUC overhead

– OTUC2: a 200 Gbit/s OTU signal with 40 (5 Gbit/s) tributary slots and 2 instances of OxUC overhead

– OTUC3-50: a 250 Gbit/s OTU signal with 50 (5 Gbit/s) tributary slots and 3 instances of OxUC overhead

The bit rate of an OTUCn-M signal is (10n + 119 × M) / (10n + 119 × 20 × n) × OTUCn bit rate. The choice of which TS are assigned to each of the OTUC instances, and hence the signal rate of that OTUC instance, is vendor or application specific.

XIII.3 OTUCn-M fault condition

For the case where subrating is used between transponders the following checks should be made.

In the transmit (source) direction, transponder X should verify that the OPU tributary slots with the Availability bit of their OPUCn MSI overhead set to "available" are carried transparently by the OTUCn-M. For the case where one or more availability bits for the the tributary slots that are not carried transparently are set to "available", an OTUCn-AIS or ODUCn-AIS should be output by transponder Y. The behaviour between the two transponders is vendor specific and outside the scope of this Recommendation.

In the receive (sink) direction, transponder Y should check for a mismatch between received and expected OTUCn-M. If such a mismatch is detected an OTUCn-AIS or ODUCn-AIS should be output by Transponder Y.

In the transmit (source) direction, transponder X should verify if the OTUCn signal contains OTUCn-AIS or ODUCn-AIS. For such case, the OPUCn contains an all-1's pattern and OPU tributary slots can no longer be identified. OTUCn-AIS or ODUCn-AIS should be output by transponder Y. The behaviour between the two transponders is vendor specific and outside the scope of this Recommendation.

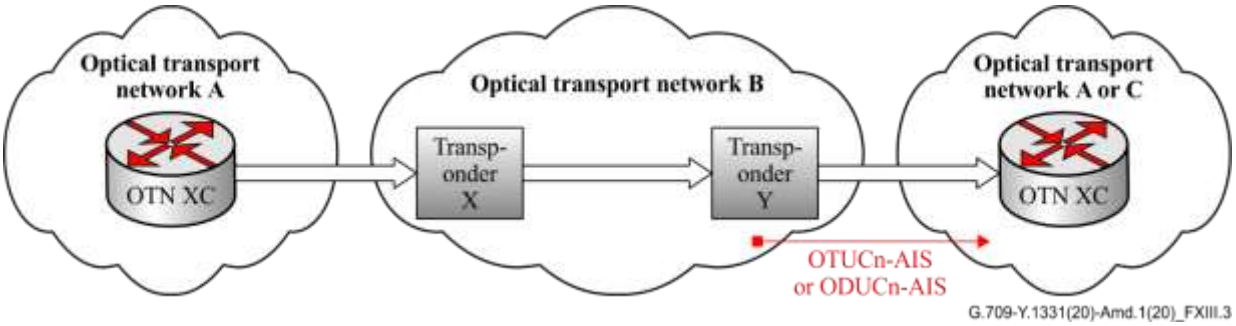


Figure XIII.3 – OTUCn sub rate mismatch fault condition

Appendix XIV

Examples of interconnection of Ethernet UNI and FlexE Group UNI in two administrative domains in the OTN for the case that these UNIs deploy different mapping methods

(This appendix does not form an integral part of this Recommendation.)

This appendix describes two cases to support the interconnection of an Ethernet UNI and a FlexE Group UNI in two administrative domains in the OTN, in which interconnection requires that the mapping method in one administrative domain is converted into the mapping method used in the other administrative domain.

Case I is to perform the mapping method conversion at the FlexE group UNI-N port as shown in Figure XIV.1.

Case II is to perform the mapping method conversion at an intermediate node along the connection as shown in Figure XIV.2.

These conversion approaches provide for backward compatibility with Ethernet UNI-N ports designed prior to the definition of FlexE.

Figure XIV.1 illustrates an implementation of case I. For the case of an Ethernet PCS encoded client service, the FlexE Group UNI-N port converts the FlexE Client signal into the corresponding Ethernet PCS encoded signal and maps it into an ODUk using BMP, TTT+GMP or GMP as specified in clause 17. The ODUk is transported through the OTN and is terminated in the Ethernet UNI-N port.

For the case of an Ethernet MAC client service with a bit rate up to and including 100 Gbit/s, the FlexE Group UNI-N port terminates the FlexE Client signal, extracts the Ethernet MAC frames stream and maps this MAC frames stream into an ODUk using GFP-F as specified in clause 17. The ODUk is transported through the OTN and is terminated in the Ethernet UNI-N port.

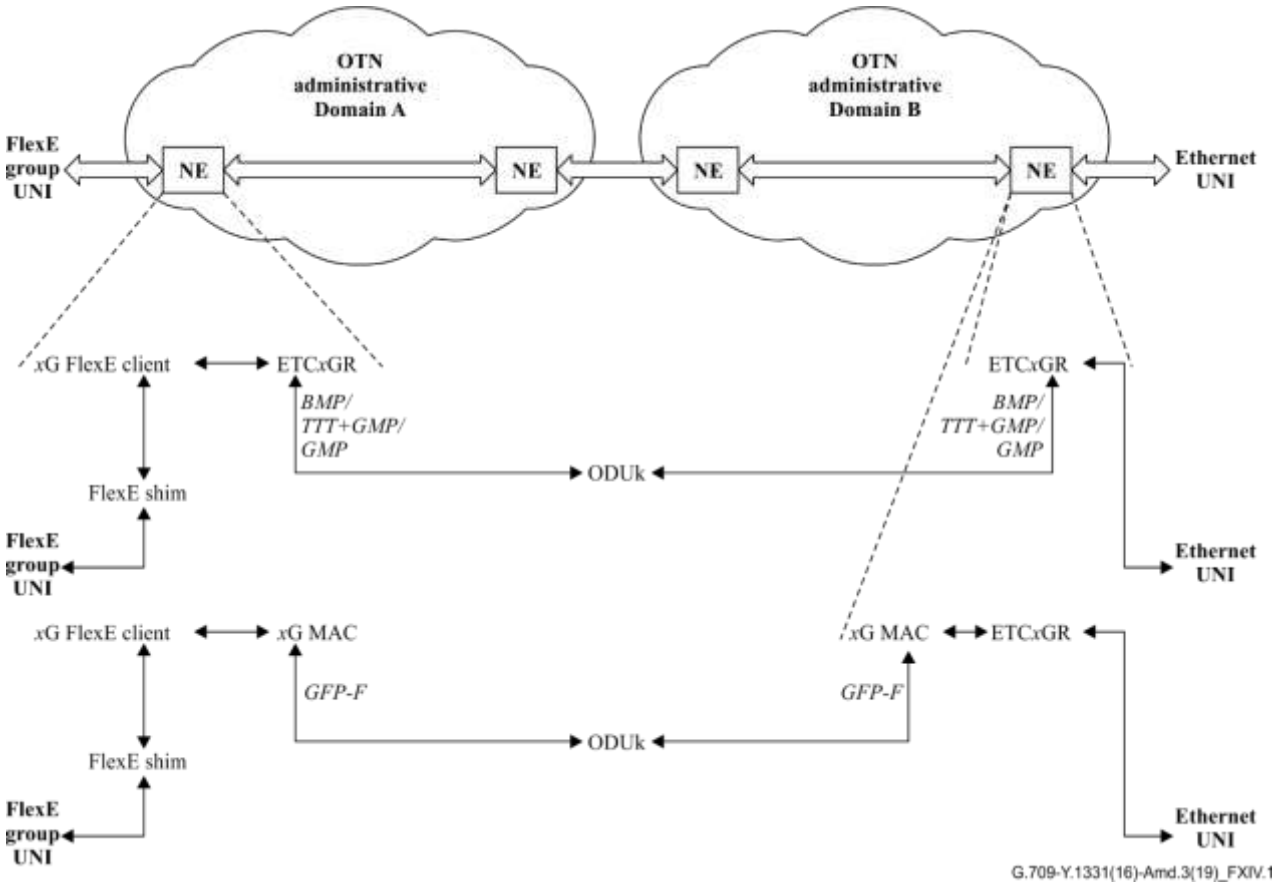


Figure XIV.1 – Ethernet PCS encoded or MAC signal interconnection between an Ethernet UNI and a FlexE UNI in case I

Figure XIV.2 illustrates an implementation of case II. For the case of an Ethernet PCS encoded client service, the mapping method conversion is performed at an intermediate node in the domain with the FlexE Group UNI. In this node, the FlexE Client signal is demapped from the ODUflex and then the FlexE Client signal is converted into the corresponding Ethernet PCS encoded signal and then mapped into an ODUk using BMP, TTT+GMP or GMP as specified in clause 17. At the same time, the overhead from ODUflex and OPUflex carrying the FlexE Client signal is converted into the overhead of ODUk and OPUk carrying the Ethernet PCS encoded signal. This conversion maintains the overhead information with the exception of the value of the Payload Type. The specific overhead conversion is vendor specific and outside the scope of this Recommendation.

For the case of an Ethernet MAC client service with a bit rate below and including 100 Gbit/s, the mapping method conversion is performed at an intermediate node in the domain with the FlexE Group UNI. In this node, the FlexE Client signal is demapped from the ODUflex and the Ethernet MAC frames stream is extracted. Then the Ethernet MAC frames stream is mapped into an ODUk using GFP-F as specified in clause 17. At the same time, the overhead from ODUflex and OPUflex carrying the FlexE Client signal is converted into the overhead of the ODUk and OPUk carrying the Ethernet MAC frames stream. This conversion maintains the overhead information with the exception of the value of the Payload Type. The specific overhead conversion is vendor specific and outside the scope of this Recommendation.

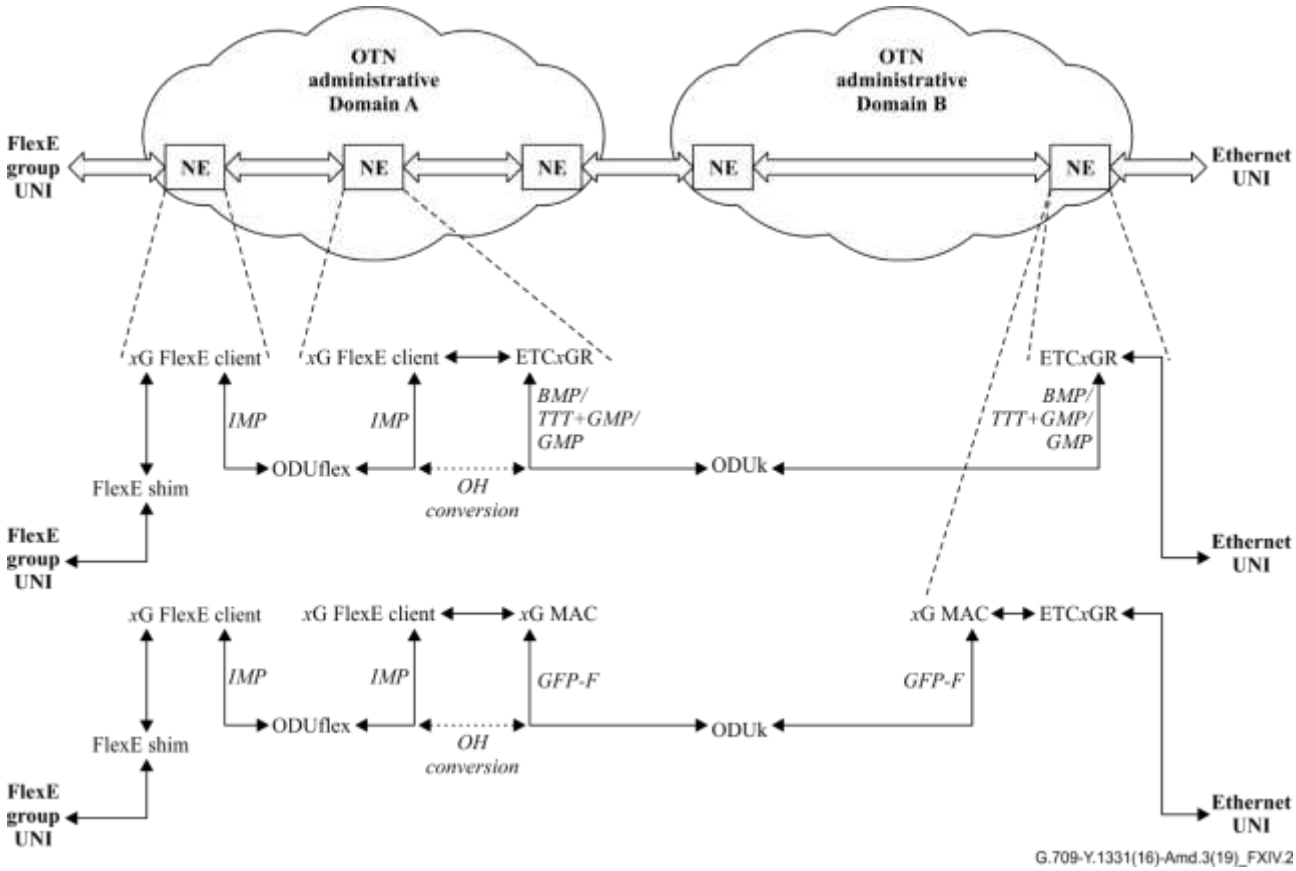


Figure XIV.2 – Ethernet PCS encoded or Ethernet MAC client services interconnection between an Ethernet UNI and a FlexE UNI in case II

Appendix XV

Examples of ODUflex(GFP) and ODUflex(IMP) clock generation methods

(This appendix does not form an integral part of this Recommendation.)

This appendix describes examples of clock generation methods that support ODUflex(GFP) and ODUflex(IMP) signals.

ODUflex(GFP) and ODUflex(IMP) signals are generated using a local clock or the timing of its client signal (if available). The local clock may be the local server ODUk (or OTUk) clock, the local ODUCn (or OTUCn) clock, or an equipment internal clock of the signal over which the ODUflex is carried through the equipment.

XV.1 Client timing based

In this method, the clock of the 64B/66B encoded packet client (e.g., FlexE Client) signal is used as reference clock for the generation of the ODUflex(IMP) signal and adjusted by means of a fixed X/Y value, with X = 239 and Y = 238. The ODUflex(IMP) bit rate is 239/238 × bit rate of 64B/66B encoded packet client.

XV.2 Local clock based

In this method, a local clock with a bit rate tolerance of ±20 ppm is used as reference clock for the generation of the ODUflex(GFP) and ODUflex(IMP) signals. This local clock is the same clock that is used to generate a server ODUk/OTUk or ODUCn/OTUCn or equipment specific server signal over which the ODUflex is carried. The bit rates of the ODUflex(GFP) and ODUflex(IMP) signals generated at the A and Z ends of the trail will be X_A ± 20 ppm and X_Z ± 20 ppm. These two rates do not have to be identical. In the worst case, these two rates could be [2 × (100 – 20)] = 160 ppm apart from each other to comply with the ODUflex(GFP) and ODUflex(IMP) bit rate of X Gbit/s ± 100 ppm, in which X represents the nominal bidirectional bit rate as illustrated in Figure XV.1.

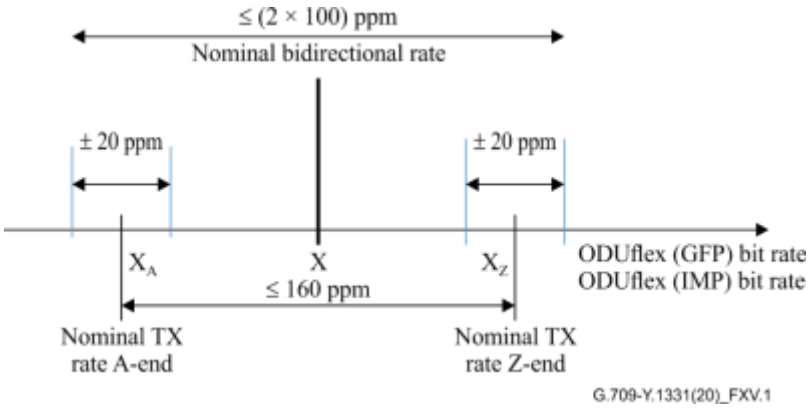


Figure XV.1 – General ODUflex(GFP) and ODUflex(IMP) bit rate and tolerance

In a first instance, the clock of the server ODU is used as reference clock to generate the ODUflex(GFP) and ODUflex(IMP) clock rate and adjusted by means of a fixed C_m value, similar to the case of ODUflex(GFP,n,k), or of two alternating C_m values X/X+1 (similar to the case of ODUflex(FlexE Aware)).

NOTE 1 – The main difference between the ODUflex(GFP) and ODUflex(IMP) clock rate and the ODUflex(GFP,n,k) clock rate is that this latter rate is the maximum rate that can be supported by M 1.25G OPUk tributary slots whereas the first clock rate is just the rate that is sufficient to carry the packet client signal in a GFP-F frame stream or in a 64B/66B encoded signal.

In a second instance, the bit rate of an ODUflex(GFP) and ODUflex(IMP) signal may be generated by means of a local clock circuit that multiplies a local free run clock rate with an Register1/Register2 multiplier of which the value of Register1 is configured via a first N-bit register and the value of Register2 is configured via a second N-bit register so that the ODUflex(GFP) and ODUflex(IMP) bit rate is equal to Register1/Register2 × Datapath_Width × System_Clock rate, with N ≥ 16. The Register1/Register2 value is related to the agreed bandwidth of the packet client signal (i.e., 238/239 × Register1/Register2). For the case of ODUflex(IMP), this clock is also provided to the packet client as reference clock for the generation of the 64B/66B encoded signal to restrict the difference in bit rate of the 64B/66B encoded packet client and the bandwidth of the OPUflex(IMP)_Payload to a value that is within the range that IMP can address.

NOTE 2 – ODUflex(GFP,n,k) signals can be supported by ODUflex(GFP) signals; for example for the case of a Datapath_Width × System_Clock rate being equal to 99.5328 Gbit/s ± 20 ppm the ODUflex(GFP,n,k=2) is supported by e.g., ODUflex(GFP, *REG1=n×28, REG2=2231*), the ODUflex(GFP,n,k=3) is supported by e.g., ODUflex(GFP, *REG1=n×35, REG2=2777*), and the ODUflex(GFP,n,k=4) is supported by e.g., ODUflex(GFP, *REG1=n×44, REG2=3365*).

NOTE 3 – ODUflex(IMP,s) signals can be supported by ODUflex(IMP) signals; for example, for the case of a Datapath_Width × System_Clock rate being equal to 99.5328 Gbit/s ± 20 ppm the ODUflex(IMP,s) is supported by e.g., ODUflex(IMP, *REG1=s×9, REG2=173*).

Appendix XVI

Implications of OTSiG (de)modulator processes and OCh|OTSiG overhead trail termination functions being located in adjacent network elements

(This appendix does not form an integral part of this Recommendation.)

This appendix describes the implications on fault management and its alarm suppression for the case that the OTSiG (de)modulator processes and OCh|OTSiG overhead trail termination functions are located in adjacent and thus different network elements.

XVI.1 Introduction

OTN equipment often consists of an electrical cross connect (EXC) subsystem and an optical cross connect or optical add/drop multiplexer (OXC/OADM) subsystem. These subsystems have traditionally been interconnected via fibres that connect the OTUk/OTUCn line cards with the OTSi(G) add/drop cards. More recently, these subsystems can be interconnected via an optical backplane that replaces the fibres.

The OCh/OTSiG overhead is typically terminated in the OXC/OADM subsystem (e.g., on the OTSiG add/drop cards), while the OTSiG is typically terminated in the EXC subsystem (on the OTUk/OTUCn line cards).

When both subsystems are from the same system vendor, fault management of both subsystems can be interconnected to support alarm suppression as specified for OTN in [ITU-T G.798]. From the functional model perspective, both subsystems behave as a single undivided system in which it does not matter how the system vendor has distributed the processes over the cards. Refer to Figure XVI.1.

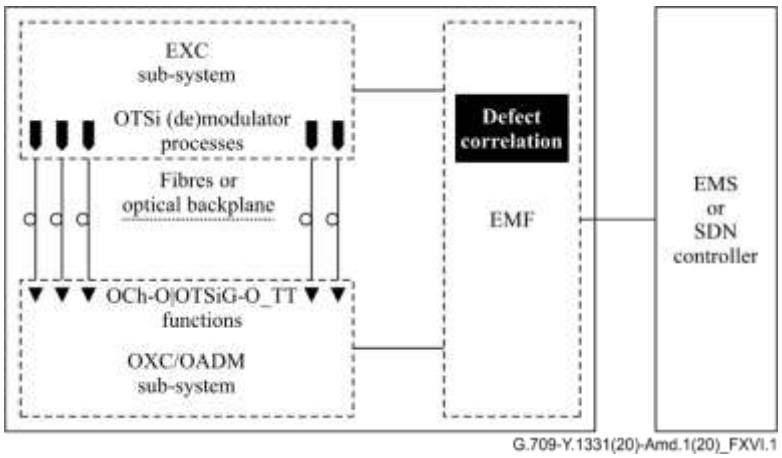


Figure XVI.1 – Single vendor OTN EXC and OXC/OADM scenario

When both subsystems are from different system vendors, fault management of both subsystems is typically not interconnected and alarm suppression as specified for OTN in G.798 is "broken". The OTN element management system or SDN controller that manages/controls both subsystems has to "glue together" the fault management and its alarm suppression. Refer to Figure XVI.2.

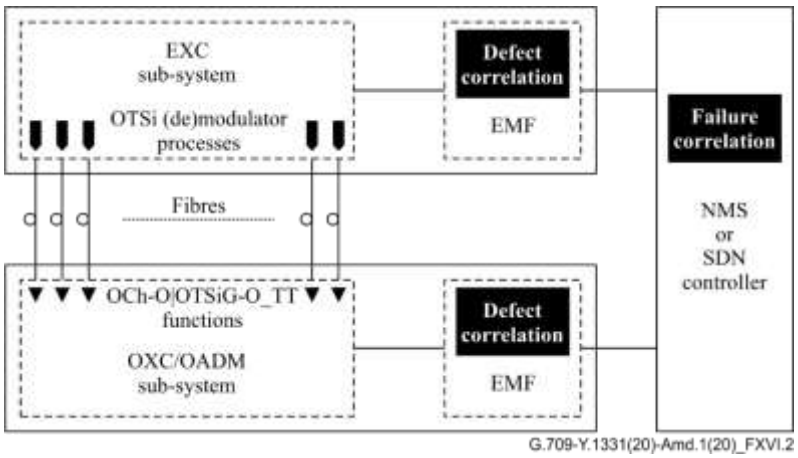


Figure XVI.2 – Multi vendor OTN EXC and OXC/OADM scenario

A similar situation is present when instead of an OTN EXC, a router or packet switch equipment's line cards (with OTSiG (de)modulator processes) are connected with the OTN OXC/OADM subsystem (with OCh|OTSiG overhead trail termination). For such case, different EMS or SDN controllers may be deployed, and therefore the network management system or SDN controller that manages/controls both circuit and packet layers has to "glue together" the fault management and alarm suppression. Refer to Figure XVI.3.

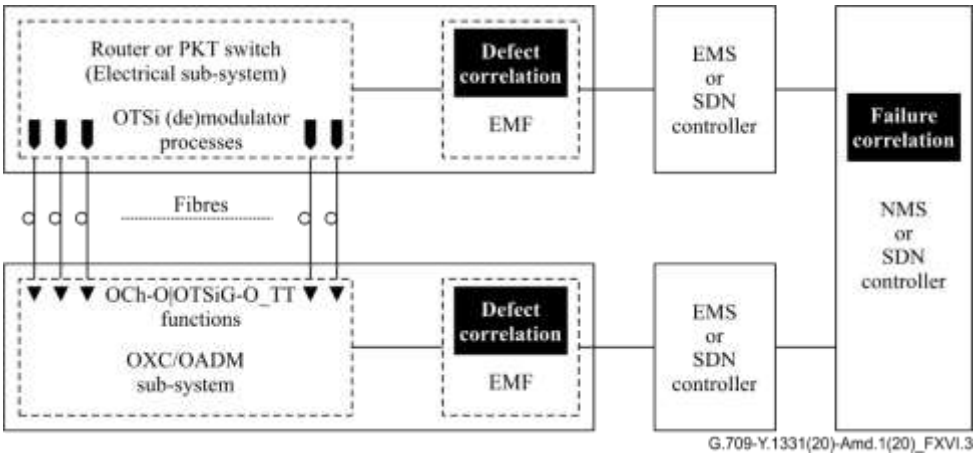


Figure XVI.3 – Multi vendor router/PKT switch and OTN OXC/OADM scenario

XVI.2 Black link deployment without co-location of OTSiG (de)modulator and OTSiG-O_TT/OCh-O_TT function

The two multi-vendor scenarios in Figures XVI.2 and XVI.3 are black link scenarios and ITU-T G.709 and [ITU-T G.7712] specify that the optical and electrical subsystems can be interconnected to transport the OCh|OTSiG overhead between the optical and electrical subsystems by means of an overhead communication channel (OCC) that is supported by the overhead communication network (OCN). The OCh-O_TT or OTSiG-O_TT functions and their OCh/OTSiG overhead processes would then be relocated to and supported in the electrical sub-system.

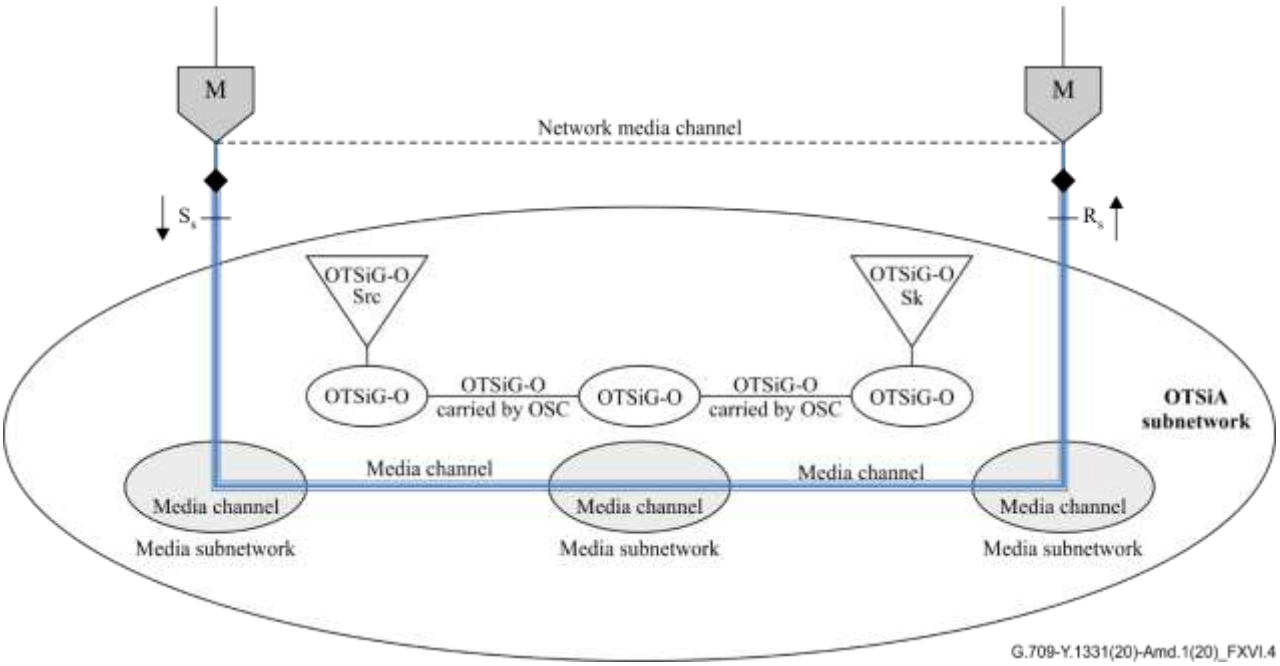


Figure XVI.4 – Black link deployment without colocation of OTSiG (de)modulator and OTSiG-O_TT/OCh-O_TT function

In black link scenarios without an OCC, as shown in Figure XVI.4, the OCh-O_TT and OTSiG-O_TT functions at the end of the black link are providing tandem connection monitoring for the OCh|OTSiA rather than path/network connection monitors.

Network connection monitoring for such case will rely on digital client (e.g., OTUk/OTUCn or ETH) related connection monitoring. Alarm suppression of OTSi loss of signal will have to be performed in the EMS, NMS or SDN controller by combining failure reports associated with the OCh-O_TT_Sk or OTSiG-O_TT_Sk functions at the edge of the OCh|OTSiA subnetwork with the loss-of-signal, -frame, -lock and -multi-frame failure reports associated with the OTSi(G)/<client>_A function in the client equipment.

XVI.3 Failure correlation for black link deployment without colocation of OTSiG (de)modulator and OTSiG-O_TT/OCh-O_TT function

Figure XVI.5 illustrates the defect correlation in the single vendor OTN EXC and OXC/OADM scenario, in which the defect correlation of the OTSi(G)/<client>_A_Sk function located in the EXC sub-system is performed in the common EMF, which mitigates for the missing AI_TSF-P signal. The AI_TSF-P term in the fault causes cXXX ← dXXX and ….. and (not AI_TSF-P) is replaced by its set of defect conditions detected in the OCh-O_TT_Sk or OTSiG-O_TT_Sk functions; i.e., by CI_SSF_P or dOCI or dFDI-P *or (dTIM and (not TIMActDIS))*.

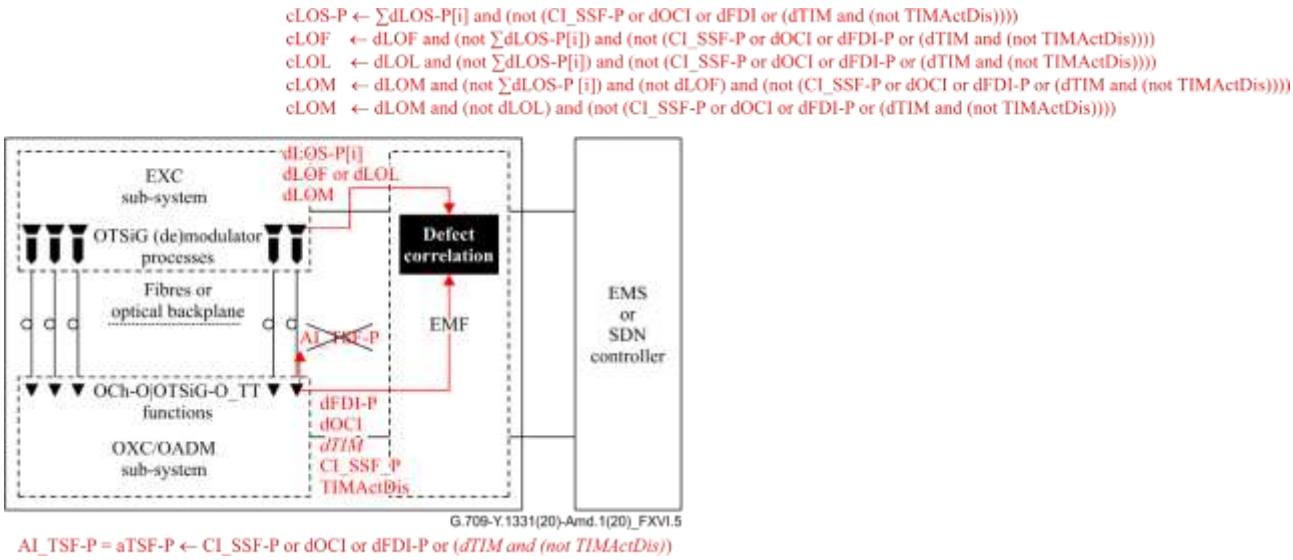


Figure XVI.5 – Defect correlation in single vendor OTN EXC and OXC/OADM scenario

Figure XVI.6 illustrates the failure correlation in the multi-vendor OTN EXC and OXC/OADM scenario, in which the defect correlation of the OTSi(G)/<client>_A_Sk function located in the EXC sub-system can only partly be performed in the EMF in absence of the AI_TSF-P input signal. The fault causes are therefore marked with a "*" (cXXX*), as well as the failures (fXXX*).

The correlation with the AI_TSF-P term is to be performed in a Failure Correlation process in the EMS or SDN controller. Instead of defects, this correlation has to use the failures from the OCh-O_TT_Sk or OTSiG-O_TT_Sk function: fOCI, fSSF, fSSF-P *and fTIM*.

The AI_TSF-P term in the failures fXXX ← fXXX* and (not AI_TSF-P) is replaced by its set of failures from the OCh-O_TT_Sk or OTSiG-O_TT_Sk functions; i.e., by fSSF or SSF-P or fOCI *or fTIM*.

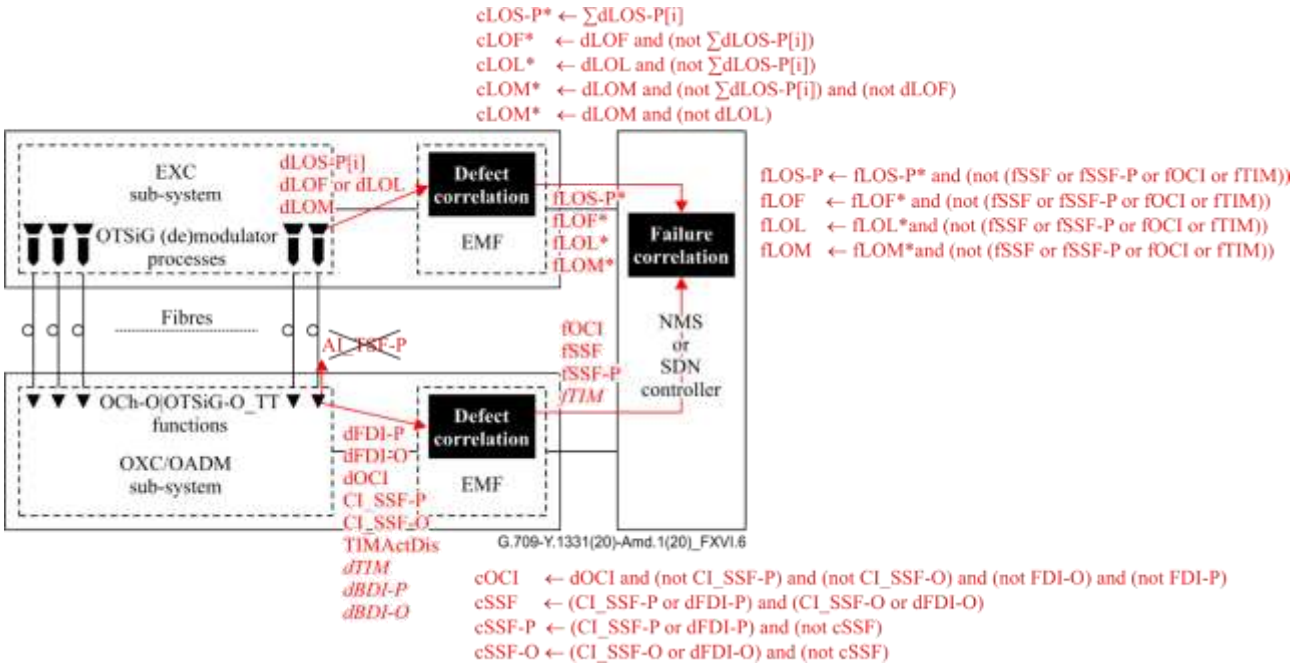


Figure XVI.6 – Failure correlation in multi-vendor OTN EXC and OXC/OADM scenario

Figure XVI.7 illustrates the failure correlation in the multi-vendor router/PKT switch and OXC/OADM scenario, in which the defect correlation of the OTSi(G)/<client>_A_Sk function located in the router/PKT switch sub-system can only partly be performed in the EMF in absence of the AI_TSF-P input signal. The fault causes are therefore marked with a "*" (cXXX*) as well as the failures (fXXX*).

The correlation with the AI_TSF-P term is to be performed in a Failure Correlation process in the EMS or SDN Controller. Instead of defects, this correlation has to use the failures from the OCh-O_TT_Sk or OTSiG-O_TT_Sk function: fOCI, fSSF, fSSF-P *and fTIM*.

The AI_TSF-P term in the failures fXXX ← fXXX* and (not AI_TSF-P) is replaced by its set of failures from the OCh-O_TT_Sk or OTSiG-O_TT_Sk functions; i.e. by fSSF or SSF-P or fOCI *or fTIM*.

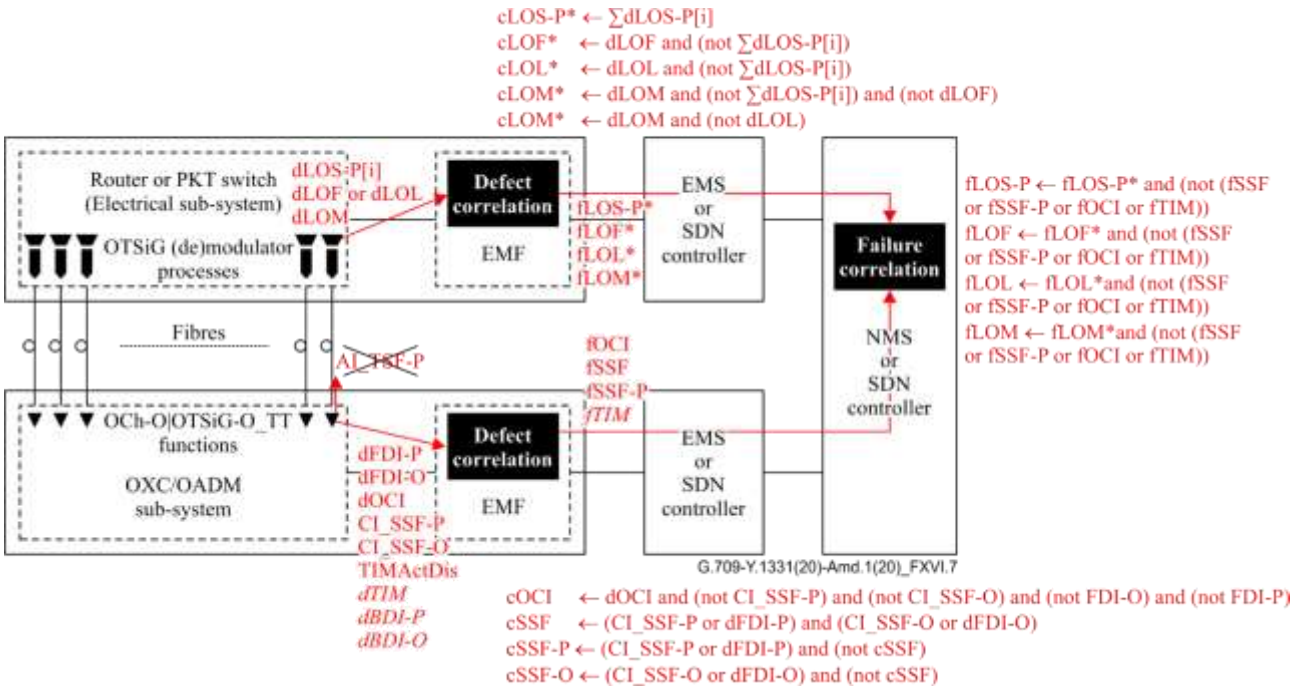


Figure XVI.7 – Failure correlation in multi-vendor router/PKT switch and OTN OXC/OADM scenario

Appendix XVII

fgOFCS test vectors for mapping of packet clients into fgOPUflex

(This appendix does not form an integral part of this Recommendation.)

This appendix shows fgOFCS test vectors with 3 consecutive 64B/66B encoded packets. All these three packets' DA is 0xFFFF_FFFF_FFFF, and SA is 0x0605_0403_0201. The first packet carries 64 bytes Ethernet frame, from DA to MAC FCS, excluding fgOFCS. The second packet carries 65 bytes Ethernet frame from DA to MAC FCS, excluding fgOFCS. The third packet carries 66 bytes from DA to MAC FCS. As there is no /I/ Block between the first packet and second packet, so the first packet's fgOFCS is inserted into the second packet's /S/ Block. The second packet's fgOFCS is inserted into the /I/ Block, following the second packet's /T/ Block. The third packet's fgOFCS is inserted into the first /I/ Block after the third packet. The second /I/ Block, after the third packet, carries no fgOFCS. The transmission order in Table XVII.1 is first from left to right and then from top to bottom. The values in Table XVII.1 are provided in hex code such that the MSB is on the left and the LSB right, excluding the sync-header and control code value "0b00011110" of /S/ block. The "0b00011110" in Table XVII.1 is a binary value of 0x78 control code value of /S/. The left bit "0" of "0b00011110" corresponding to the LSB that is transmitted first is at bit [2], and the right bit "0" corresponding to the MSB that is transmitted last is at bit [9] of the 66B /S/ block.

Table XVII.1 – Three consecutive encoded packets test vectors

66B blocks										
0	**1**	**2 … 9**	**10…17**	**18…25**	**26…33**	**34…41**	**42…49**	**50…57**	**58…65**	**Bit position / Comment**
1	0	78 (0b00011110)	55	55	55	55	55	55	D5	1st packet, /S/ Block
0	1	FF	FF	FF	FF	FF	FF	06	05	1st packet, /D/ Block, 64 bytes
0	1	04	03	02	01	00	2E	00	01	1st packet, /D/ Block
0	1	02	03	04	05	06	07	08	09	1st packet, /D/ Block
0	1	0A	0B	0C	0D	0E	0F	10	11	1st packet, /D/ Block
0	1	12	13	14	15	16	17	18	19	1st packet, /D/ Block
0	1	1A	1B	1C	1D	1E	1F	20	21	1st packet, /D/ Block
0	1	22	23	24	25	26	27	28	29	1st packet, /D/ Block
0	1	2A	2B	2C	2D	DE	E1	90	D0	1st packet, /D/ Block
1	0	87	00	00	00	00	00	00	00	1st packet, /T/ Block
1	0	78	29	FC	DA	4F	55	55	D5	2nd packet, /S/ Block with fgOFCS

Table XVII.1 – Three consecutive encoded packets test vectors

66B blocks										
0	**1**	**2 … 9**	**10…17**	**18…25**	**26…33**	**34…41**	**42…49**	**50…57**	**58…65**	**Bit position / Comment**
0	1	FF	FF	FF	FF	FF	FF	06	05	2nd packet, /D/ Block, 65 bytes
0	1	04	03	02	01	00	2F	00	01	2nd packet, /D/ Block
0	1	02	03	04	05	06	07	08	09	2nd packet, /D/ Block
0	1	0A	0B	0C	0D	0E	0F	10	11	2nd packet, /D/ Block
0	1	12	13	14	15	16	17	18	19	2nd packet, /D/ Block
0	1	1A	1B	1C	1D	1E	1F	20	21	2nd packet, /D/ Block
0	1	22	23	24	25	26	27	28	29	2nd packet, /D/ Block
0	1	2A	2B	2C	2D	2E	AC	61	DE	2nd packet, /D/ Block
1	0	99	52	00	00	00	00	00	00	2nd packet, /T/ Block
1	0	1E	B8	91	11	8A	00	00	00	/I/ Block with fgOFCS
1	0	78	55	55	55	55	55	55	D5	3rd packet, /S/ Block without fgOFCS
0	1	FF	FF	FF	FF	FF	FF	06	05	3rd packet, /D/ Block, 66 bytes
0	1	04	03	02	01	00	30	00	01	3rd packet, /D/ Block
0	1	02	03	04	05	06	07	08	09	3rd packet, /D/ Block
0	1	0A	0B	0C	0D	0E	0F	10	11	3rd packet, /D/ Block
0	1	12	13	14	15	16	17	18	19	3rd packet, /D/ Block
0	1	1A	1B	1C	1D	1E	1F	20	21	3rd packet, /D/ Block
0	1	22	23	24	25	26	27	28	29	3rd packet, /D/ Block
0	1	2A	2B	2C	2D	2E	2F	EE	93	3rd packet, /D/ Block
1	0	AA	D3	85	00	00	00	00	00	3rd packet, /T/ Block

Table XVII.1 – Three consecutive encoded packets test vectors

66B blocks										
0	**1**	**2 … 9**	**10…17**	**18…25**	**26…33**	**34…41**	**42…49**	**50…57**	**58…65**	**Bit position / Comment**
1	0	1E	89	FA	11	79	00	00	00	/I/ Block with fgOFCS
1	0	1E	00	00	00	00	00	00	00	/I/ Block without fgOFCS

Bibliography

[b-ITU-T G.8312] Recommendation ITU-T G.8312 (2020), *Interfaces for metro transport networks*.

[b-ITU-T G-Sup.43] ITU-T G-series Recommendations – Supplement 43 (2011), *Transport of IEEE 10GBASE-R in optical transport networks (OTN)*.

[b-ITU-T G-Sup.56] ITU-T G-series Recommendations – Supplement 56 (2016), *OTN Transport of CPRI signals*.

[b-ITU-T G-Sup.58] ITU-T G-series Recommendations – Supplement 58 (2020), Optical transport network (*OTN) module framer interfaces (MFIs)*.

[b-ANSI INCITS 296] ANSI INCITS 296-1997, *Information Technology – Single-Byte Command Code Sets CONnection (SBCON) Architecture)*.

[b-ANSI INCITS 364] ANSI INCITS 364-2003, *Information Technology – Fibre Channel* 10 Gigabit (10GFC).

[b-CPRI] CPRI Specification V5.0 (2011-09-21), *Common Public Radio Interface (CPRI), Interface Specification*.

[b-IB ARCH] InfiniBand Trade Association (2006), *InfiniBand Architecture Specification, Volume 2, Release 1.2.1*.

[b-IEEE 1588] IEEE 1588:2013, *Precision Time Protocol Time Synchronization Performance*.

[b-IETF RFC 6205] IETF RFC 6205 (2011), *Generalized Labels for Lambda-Switch-Capable (LSC) Label Switching Routers*.

[b-INCITS 488] INCITS 488:2016, *Information Technology – Fibre Channel Framing and Signaling – 4 (FC-FS-4)*.

[b-INCITS 470] INCITS 470:2011, *Information Technology – Fibre Channel – Framing and Signaling – 3 (FC-FS-3)*.

ITU-T Y-SERIES RECOMMENDATIONS

Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities

GLOBAL INFORMATION INFRASTRUCTURE	Y.100-Y.999
INTERNET PROTOCOL ASPECTS	Y.1000-Y.1999
General	Y.1000-Y.1099
Services and applications	Y.1100-Y.1199
Architecture, access, network capabilities and resource management	Y.1200-Y.1299
Transport	**Y.1300-Y.1399**
Interworking	Y.1400-Y.1499
Quality of service and network performance	Y.1500-Y.1599
Signalling	Y.1600-Y.1699
Operation, administration and maintenance	Y.1700-Y.1799
Charging	Y.1800-Y.1899
IPTV over NGN	Y.1900-Y.1999
NEXT GENERATION NETWORKS	Y.2000-Y.2999
FUTURE NETWORKS	Y.3000-Y.3499
CLOUD COMPUTING	Y.3500-Y.3599
BIG DATA	Y.3600-Y.3799
QUANTUM KEY DISTRIBUTION NETWORKS	Y.3800-Y.3999
INTERNET OF THINGS AND SMART CITIES AND COMMUNITIES	Y.4000-Y.4999

For further details, please refer to the list of ITU-T Recommendations.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	**Transmission systems and media, digital systems and networks**
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities**
Series Z	Languages and general software aspects for telecommunication systems

Published in Switzerland
Geneva, 2024