International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.8021/Y.1341

Amendment 1

(07/2011)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

Packet over Transport aspects – Ethernet over Transport aspects

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

Internet protocol aspects – Transport

Characteristics of Ethernet transport network equipment functional blocks

Amendment 1

Recommendation ITU-T G.8021/Y.1341 (2010) – Amendment 1

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
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
Ethernet over Transport aspects	**G.8000–G.8099**
MPLS over Transport aspects	G.8100–G.8199
Quality and availability targets	G.8200–G.8299
Service Management	G.8600–G.8699
ACCESS NETWORKS	G.9000–G.9999

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

Recommendation ITU-T G.8021/Y.1341

Characteristics of Ethernet transport network equipment functional blocks

Amendment 1

Summary

Amendment 1 to Recommendation ITU-T G.8021/Y.1341 (2010) presents enhancements concerning ETH performance monitoring functions, Client Signal Failure function, and ODU server to ETH adaptation functions.

History

Edition	Recommendation	Approval	Study Group
1.0	ITU-T G.8021/Y.1341	2004-08-22	15
1.1	ITU-T G.8021/Y.1341 (2004) Amd. 1	2006-06-06	15
2.0	ITU-T G.8021/Y.1341	2007-12-22	15
2.1	ITU-T G.8021/Y.1341 (2007) Amd. 1	2009-01-13	15
2.2	ITU-T G.8021/Y.1341 (2007) Amd. 2	2010-02-22	15
3.0	ITU-T G.8021/Y.1341	2010-10-22	15
3.1	ITU-T G.8021/Y.1341 (2010) Amd. 1	2011-07-22	15

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.

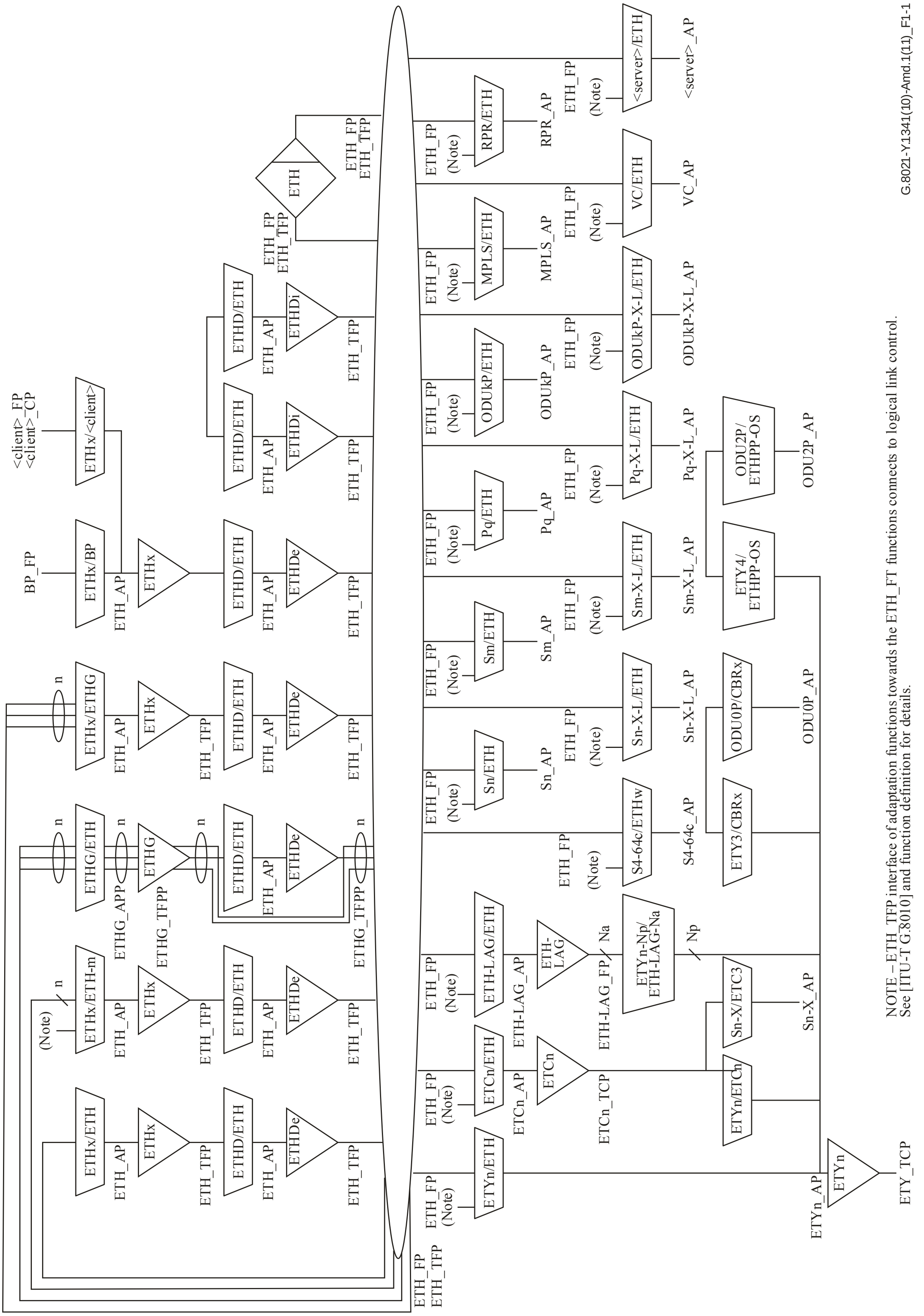
As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged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

Recommendation ITU-T G.8021/Y.1341

Characteristics of Ethernet transport network equipment functional blocks

Amendment 1

1) Figure 1-1



NOTE – ETH_TFP interface of adaptation functions towards the ETH_FT functions connects to logical link control. See [ITU-T G.8010] and function definition for details.

G.8021-Y.1341(10)-Amd.1(11)_F1-1

Figure 1-1 – Overview of ITU-T G.8021/Y.1341 atomic model functions

2) Add new definition

Add the following definition:

3.1.69 timing point: [ITU-T G.806].

Existing clauses 3.1.69 to 3.1.75 are to be renumbered.

3) Clause 4

Add the following abbreviations:

CSF	Client Signal Fail
DCI	Defect Clear Indication
FDI	Forward Defect Indication
PI	Replication Information
PP	Replication Point
SL	Synthetic Loss
SLM	Synthetic Loss Message
SLR	Synthetic Loss Reply
TCP	Trail Connection Point
TP	Timing Point

4) Table 6-1

Add the following rows to Table 6-1 for CSF and dFOP-TO

CSF-LOS	Reception of a CSF frame that indicates client loss of signal.
CSF-FDI	Reception of a CSF frame that indicates client forward defect indication.
CSF-RDI	Reception of a CSF frame that indicates client reverse defect indication.
expRAPS	Reception of a valid R-APS frame.

5) Table 6-2

Add the following rows to Table 6-2 for CSF and dFOP-TO and modify the first paragraph after the table as shown.

dCSF-LOS	CSF-LOS	#CSF-LOS == 0 (K*CSF_Period or CSF-DCI)
dCSF-FDI	CSF-FDI	#CSF-FDI == 0 (K*CSF_Period or CSF-DCI)
dCSF-RDI	CSF-RDI	#CSF-RDI == 0 (K*CSF_Period or CSF-DCI)
dFOP-TO	#expAPS==0 (K * long APS interval) or #expRAPS==0 (K * long R-APS frame interval)	expAPS or expRAPS

Note that for the case of CCM_Period, AIS_Period,~~ and~~ LCK_Period, and CSF_Period the values for the CCM, AIS,~~ and~~ LCK, and CSF periods are based on the periodicity as indicated in the CCM, AIS~~ or~~, LCK, or CSF frame that triggered the timer to be started.

6) **Figure 6-2**

Replace Figure 6-2 with the following figure:

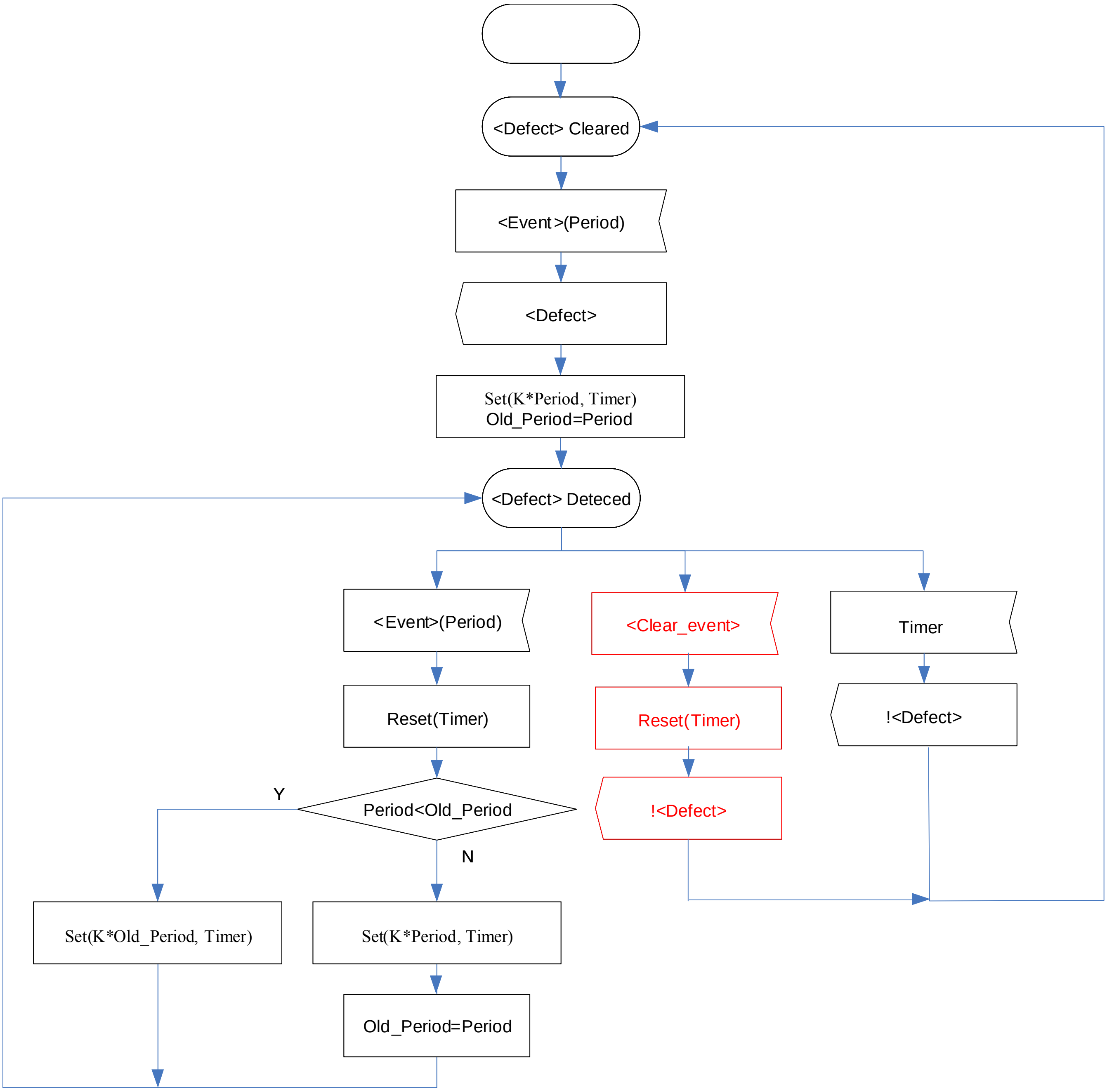


Figure 6-2 – Defect detection and clearance process for dUNL, dMMG, dUNM, dUNP, dUNPr, dAIS, dLCK, and dCSF

7) **New clause 6.1.4.3.4** *for dFOP-TO*

Add the following new clause with respect to dFOP-TO:

6.1.4.3.4 Linear or ring protection failure of protocol time out (dFOP-TO)

The Failure of Protocol Time Out defect is calculated at the ETH layer. It monitors time-out defect of:

- linear protection by detecting the prolonged absence of expected APS frames; or
- ring protection by detecting the prolonged absence of expected R-APS frames.

Its detection and clearance are defined in Table 6-2.

In the case of linear protection, dFOP-TO is detected on receipt of no expAPS event during K times the long APS interval defined in ITU-T G.8031/Y.1342 (where K ≥ 3.5) when neither dLOC nor CI_SSF are reported. dFOP-TO is cleared on receipt of an expAPS event. These events are generated by the subnetwork connection protection process (clause 9.1.2).

In the case of ring protection, dFOP-TO is detected on receipt of no expRAPS event during K times the long R-APS frame intervals defined in ITU-T G.8032/Y.1344 (where K ≥ 3.5) on a ring port reporting no link level failure and neither administratively disabled, nor blocked from R-APS message reception. dFOP-TO is cleared on receipt of an expRAPS event. These events are generated by the ring protection control process (clause 9.1.3).

8) New clause 6.1.5.4 *for CSF*

Add the following new clause with respect to CSF:

6.1.5.4 Client Signal Fail defect (dCSF)

The CSF (CSF-LOS, CSF-FDI, and CSF-RDI) defect is calculated at the ETH layer. It monitors the presence of a CSF maintenance signal.

Its detection and clearance conditions are defined in Figure 6-2. The <Defect> in Figure 6-2 is dCSF-LOS, dCSF-FDI, or dCSF-RDI. The <Event> in Figure 6-2 is the CSF event (as generated by the CSF reception process in clause 9.3.2.2) and the period is the period carried in the CSF frame that triggered the event, unless an earlier CSF frame carried a greater period.

The <Clear_event> in Figure 6-2 is the CSF event which indicates detect clearance indication (DCI).

9) Clause 8.1.7.2

Update clause 8.1.7.2 with respect to in-profile as follows:

8.1.7.2 CCM Generation process

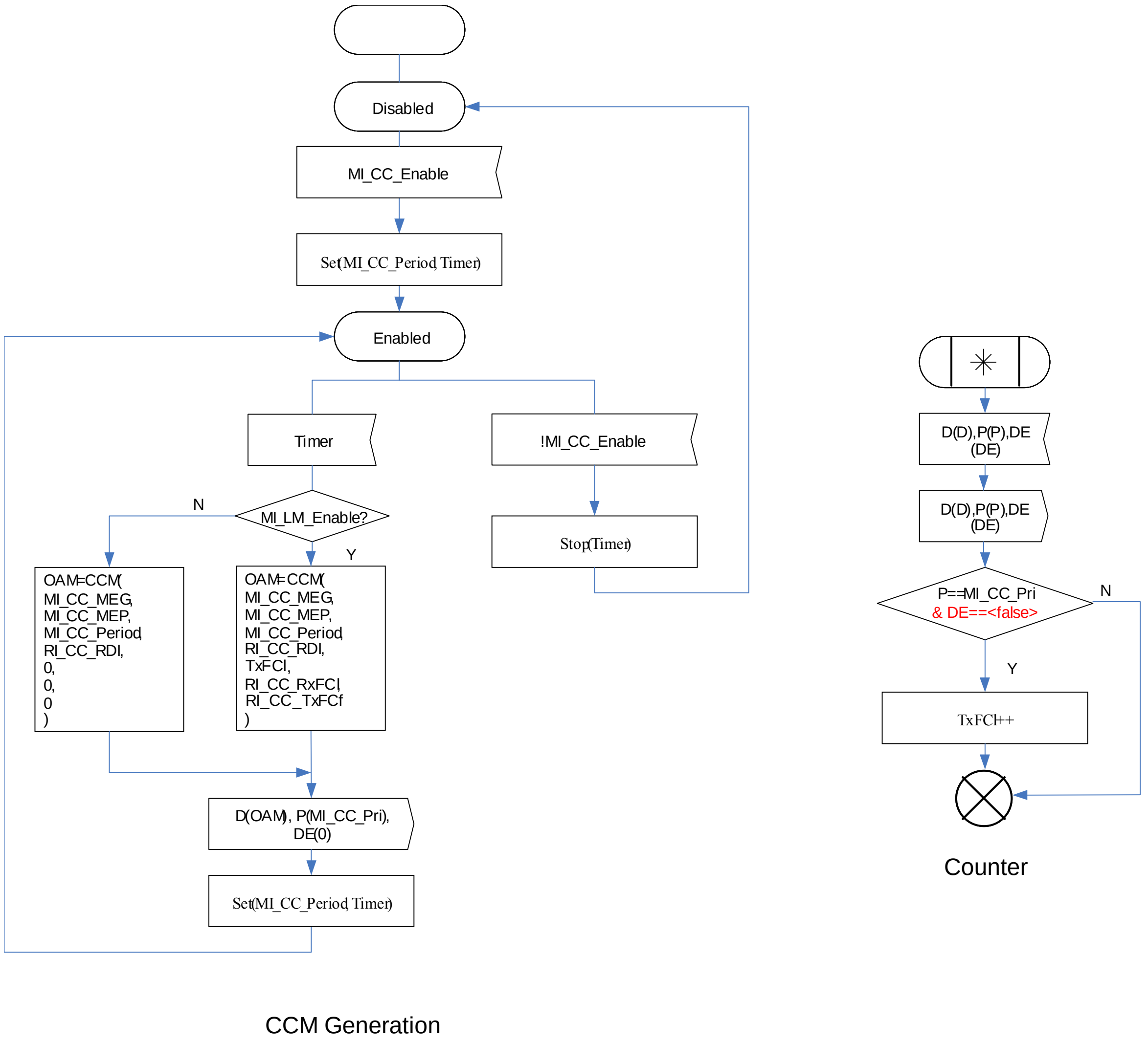


Figure 8-17 – CCM Generation behaviour

Figure 8-17 shows the state diagram for the CCM Generation process. The CCM Generation process can be enabled and disabled using the MI_CC_Enable signal, where the default value is FALSE.

In the Enabled state there are two main parts:

– counter part that is triggered by the receipt of a data frame;

– CCM generation part that is triggered by the expiration of the timer.

Counter part

The counter part of the CCM Generation process forwards data frames and counts all ETH_AI frames with Priority (P) (i.e., ETH_AI_P) equal to MI_CC_Pri and Drop Eligibility (DE) (i.e., ETH_AI_DE) equal to <false (0)>. The D, P and DE signals are forwarded unchanged as indicated by the dotted lines in Figure 8-16.

10) Clause 8.1.7.3

Update clause 8.1.7.3 with respect to in-profile as follows:

8.1.7.3 CCM Reception process

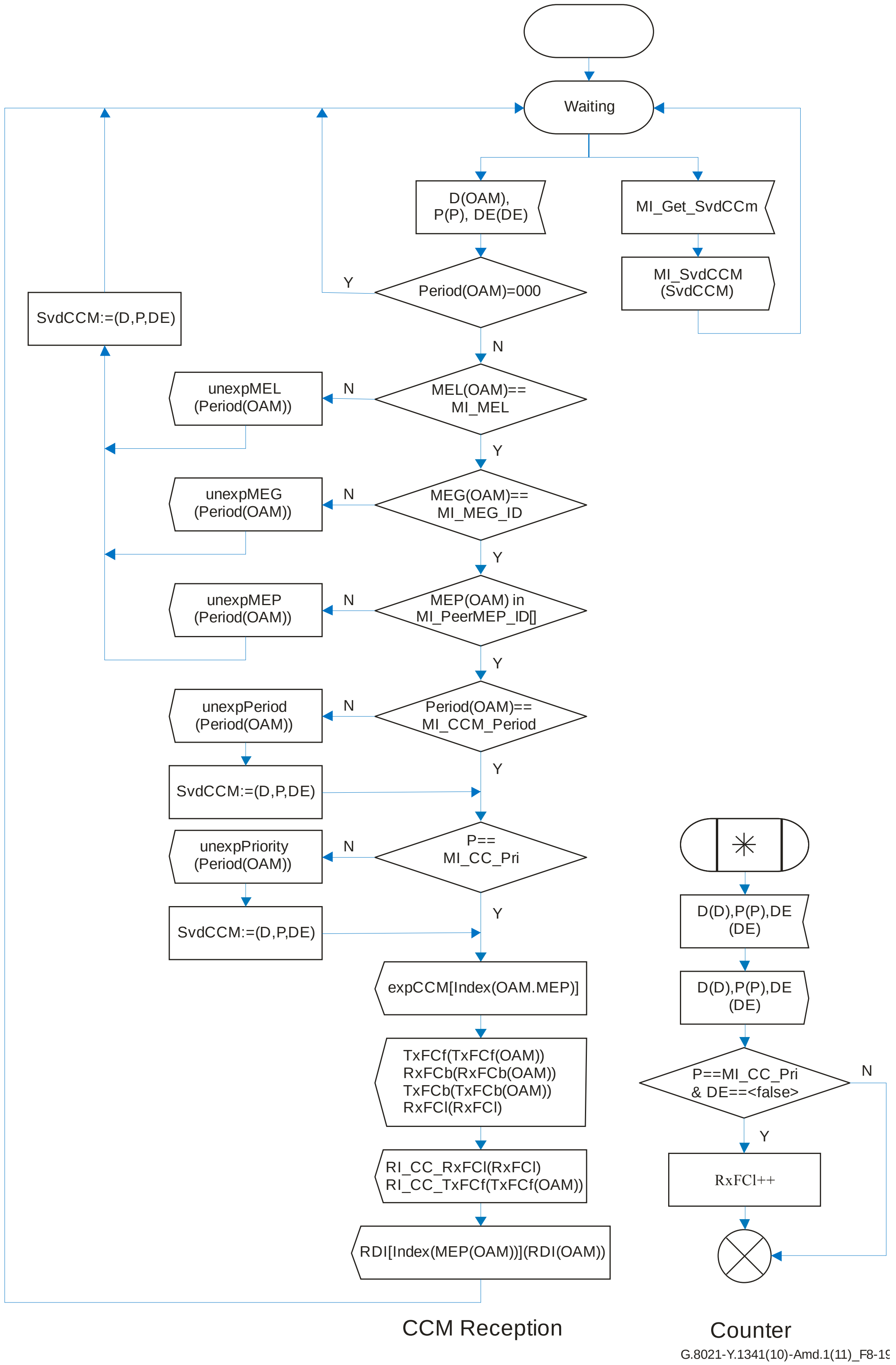


Figure 8-19 – CCM Reception behaviour

The CCM reception process consists of two parts: Counter and CCM Reception.

Counter part

The counter part of the CCM reception process receives ETH_CI, extracts pro-active ETH OAM frames and forwards remainder as ETH_AI traffic units. It ~~the data frames and~~ counts this number of ETH_AI traffic units ~~all data frames~~ that have priority (P) (i.e., ETH_AI_P) equal to MI_CC_Pri and Drop Eligibility (DE) (i.e., ETH_AI_DE) equal to <false (0)>.

CCM Reception part

The CCM reception part of the CCM reception process processes CCM OAM frames. It checks the various fields of the frames and generates the corresponding events (as defined in clause 6). If the Version, MEL, MEG and MEP are valid, the values of the frame counters are sent to the performance counter process.

Note that unexpPriority and unexpPeriod events do not prevent the CCM from being processed, since the MEL, MEG and MEP are as expected.

11) Clause 8.1.9.3

Update clause 8.1.9.3 with respect to in-profile as shown:

8.1.9.3 LMx Generation process

The LMx Generation process contains both the LMM Generation and LMR Generation functionalities. Figure 8-35 shows the LMx Generation process.

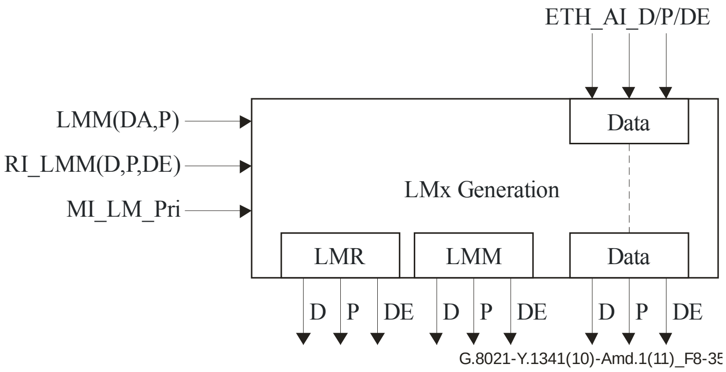


Figure 8-35 – LMx Generation process

Figure 8-36 defines the behaviour of the LMx process. The behaviour consists of three parts:

– LMM Generation part that is triggered by the receipt of the LMM(DA,P) signal;

– LMR Generation part that is triggered by the receipt of RI_LMM(D,P,DE) signals;

– Counter part that is triggered by the receipt of a normal data signal.

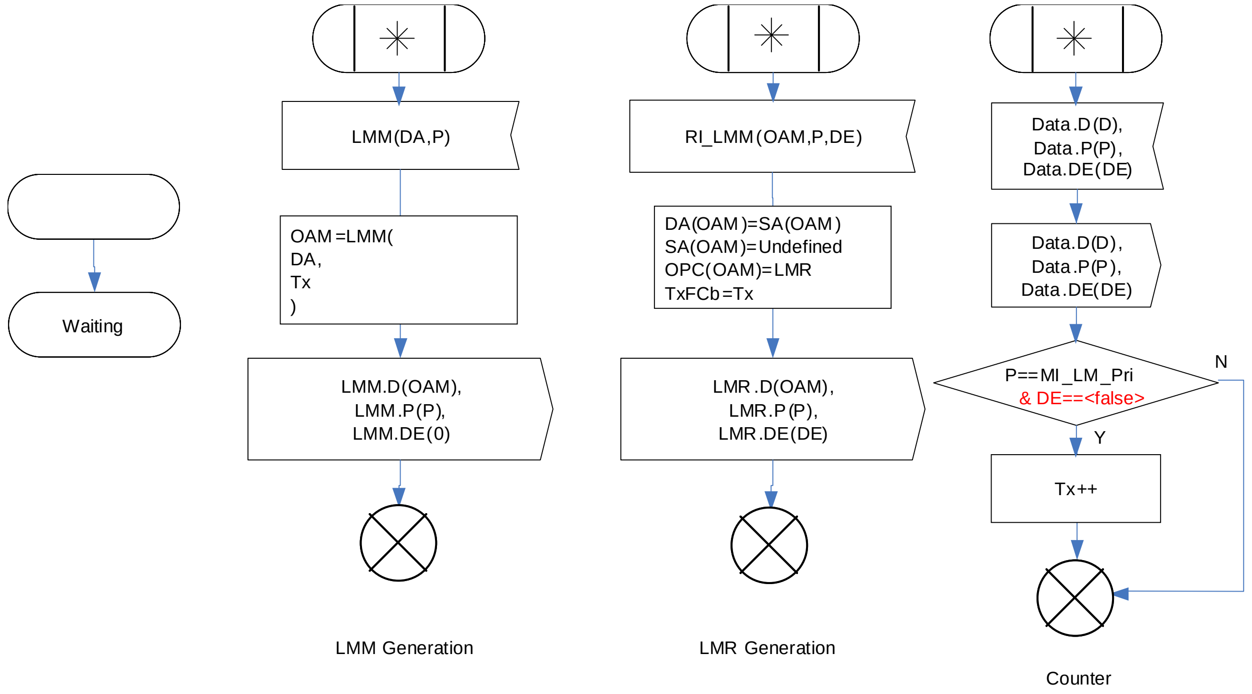


Figure 8-36 – LMx Generation behaviour

Counter part

This part receives ETH_~~CI~~AI and forwards it. It counts the number of ETH_~~CI~~AI traffic units received with ETH_~~CI~~AI_P signal equal to MI_LM_Pri and ETH_AI_DE to <false (0)>.

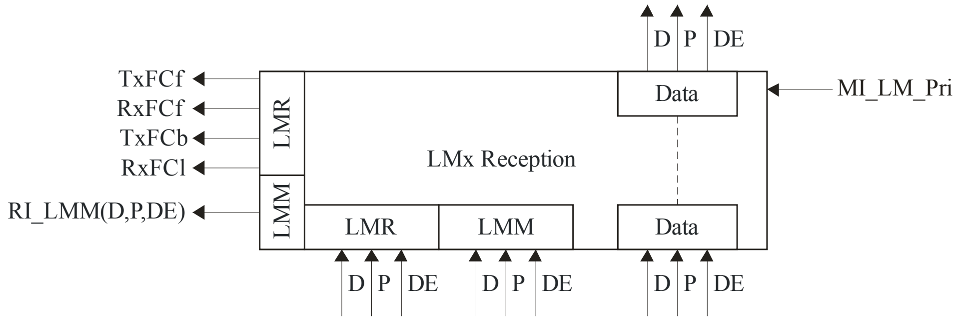
• • •

12) Clause 8.1.9.4

Update clause 8.1.9.4 with respect to in-profile as shown:

8.1.9.4 LMx Reception process

The LMx Reception process contains both the LMM Reception and LMR Reception functionalities. Figure 8-39 shows the LMx Reception process.



G.8021-Y.1341(10)-Amd.1(11)_F8-39

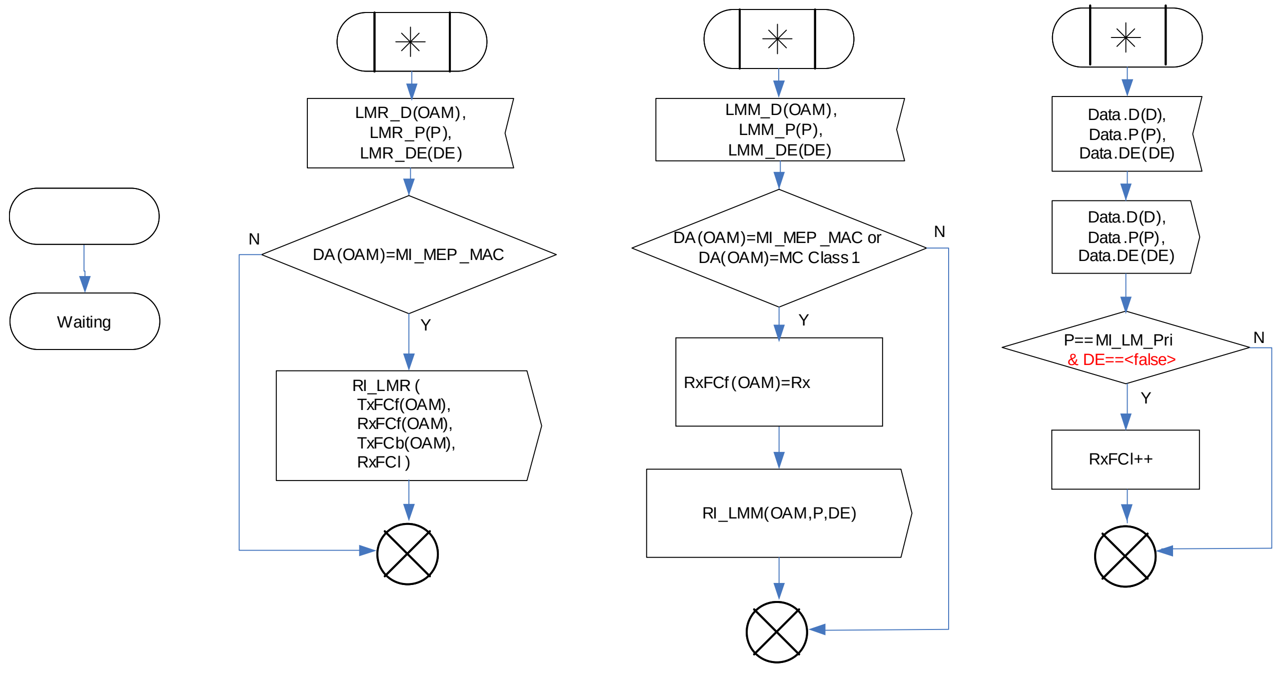
Figure 8-39 – LMx Reception process

Figure 8-40 defines the behaviour of the LMx Reception process. The behaviour consists of three parts:

– LMM Reception part that is triggered by the receipt of an LMM traffic unit;

– LMR Reception part that is triggered by the receipt of an LMR traffic unit;

– Counter part that is triggered by the receipt of a normal data signal.



LMR Reception LMM Reception Counter

Figure 8-40 – LMx Reception behaviour

Counter part

This part receives ETH_CI, extracts on-demand ETH OAM frames and forwards the remainder as ETH_AI traffic units~~it~~. It counts this the number of ETH_~~CI~~ AI instances ~~received~~ with ETH_~~CI~~AI_P signal equal to MI_LM_Pri and ETH_AI_DE equal to <false (0)>.

LMM Reception part

This part processes received LMM Traffic Units. It checks the destination address, the DA must be either the Local MAC address or it should be a Multicast Class 1 Destination Address. If this is the case the LMM Reception process writes the Rx Counter value to the received traffic unit in the RxFCf field, and forwards the received traffic unit and complementing P and DE signals as Remote Information to the LMR Generation process.

LMR Reception part

This part process received LMR traffic units. If the DA equals the local MAC address, it extracts the counter values TxFCf, RxFCf, TxFCb from the received traffic unit as well as the SA field. These values together with the value of the Rx counter(RxFCl) are forwarded as RI signals.

13) Clause 8.1.10

Update clause 8.1.10 with respect to DM as follows:

8.1.10 Delay Measurement (DM) processes

8.1.10.1 Overview

Figure 8-41 shows the different processes inside MEPs and MIPs that are involved in the on-demand delay measurement protocol.

The MEP OnDemand-OAM Source insertion process is defined in clause 9.4.1.1, the MEP OnDemand-OAM Sink extraction process in clause 9.4.1.2, the MIP OnDemand-OAM Sink Extraction process in clause 9.4.2.2, and the MIP OnDemand-OAM Source insertion process in clause 9.4.2.1. In summary, they insert and extract ETH_CI OAM signals into and from the stream of ETH_C_D traffic units and the complementing P and D signals going through an MEP and MIP; the extraction is based on MEL and Opcode. Furthermore, the insertion process inserts the correct MEL and SA values into the OAM traffic units.

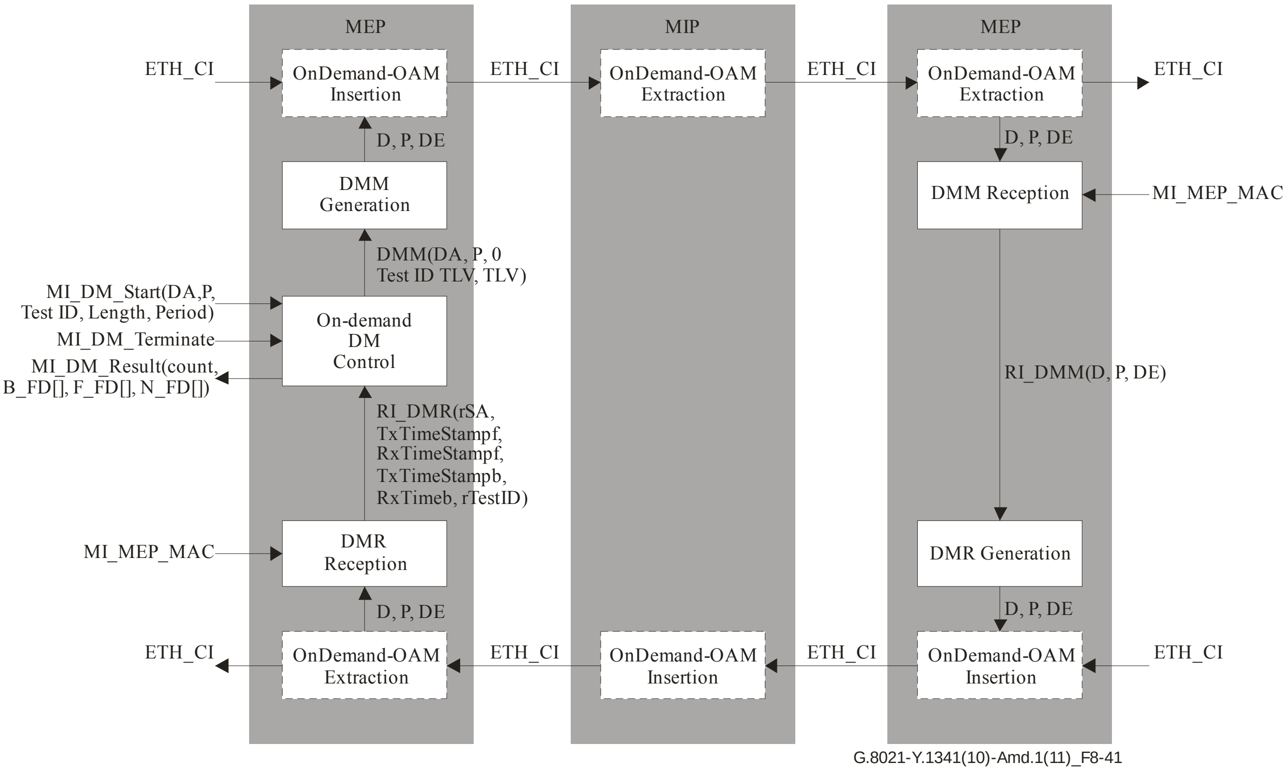


Figure 8-41 – Overview of processes involved with on-demand delay measurement

The MEP on-demand-OAM Source insertion process is defined in clause 9.4.1.1, the MEP on-demand-OAM Sink extraction process in clause 9.4.1.2.

The on-demand DM control process controls the on-demand DM protocol. The protocol is activated upon receipt of the MI_DM_Start(DA,P,Test ID,Length,Period) signal and remains activated until the MI_DM_Terminate signal is received. The result is communicated via the MI_DM_Result(count, B_FD[], F_FD[], N_FD[]) signal. If the on-demand DM control process activates the multiple monitoring on different CoS levels simultaneously, each result is independently managed per CoS level. Optional Test ID TLV can be utilized to distinguish each measurement if multiple measurements are simultaneously activated in an ME. If the protocol is used in multipoint-to-multipoint environments, the multicast class 1 address is used for DA and the test result is independently managed per peer node.

The DMM generation process generates DMM traffic units that pass through MIPs transparently, but are received and processed by DMM Reception processes in MEPs. The DMR Generation process may generate a DMR traffic unit in response. This DMR traffic unit also passes transparently through MIPs, but is received and processed by DMR Reception processes in MEPs.

At the Source MEP side, the DMM generation process stamps the value of the local time to the TxTimeStampf field in the DMM message when the first bit of the frame is transmitted. Note well that at the sink MEP side, the DMM reception process stamps the value of the local time to the RxTimeStampf field in the DMM message when the last bit of the frame is received.

The DMR generation and reception process stamps with the same way as the DMM generation and reception process.

Figure 8-41bis shows the different processes inside MEPs and MIPs that are involved in the proactive delay measurement protocol.

The MEP proactive OAM insertion process is defined in clause 9.2.1.1, the MEP OAM proactive extraction process in clause 9.2.1.2, the MIP OAM extraction process in clause 9.4.2.2, and the MIP OAM insertion process in clause 9.4.2.1. In summary, they insert and extract ETH_CI OAM signals into and from the stream of ETH_C_D traffic units and the complementing P and D signals going through a MEP and MIP; the extraction is based on MEL and Opcode. Furthermore, the insertion process inserts the correct MEL and SA values into the OAM traffic units.

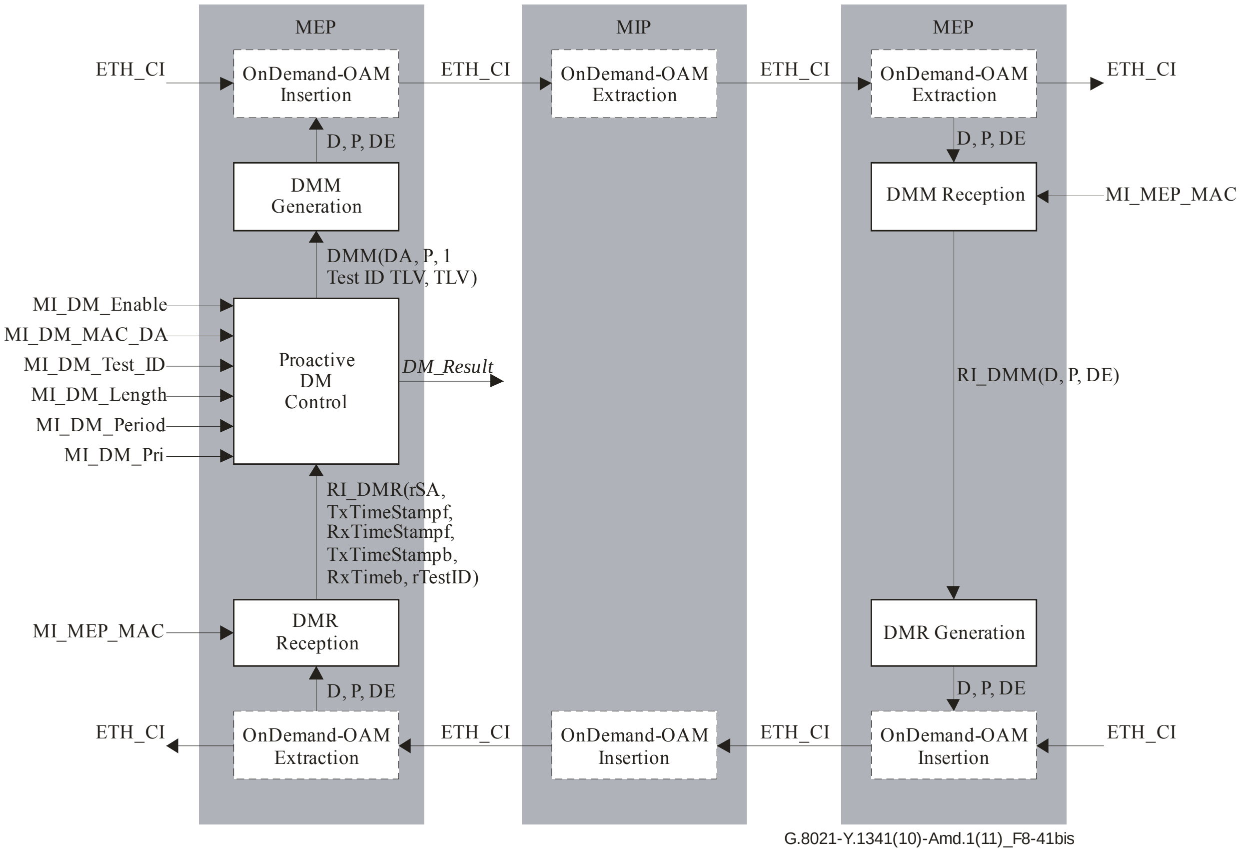


Figure 8-41bis – Overview of processes involved with proactive delay measurement

The MEP Proactive OAM Source insertion process is defined in clause 9.2.1.1, the MEP Proactive-OAM Sink extraction process in clause 9.2.1.2.

The proactive DM control process controls the proactive DM protocol. If MI_DM_Enable is set the DMM frames are sent periodically. The DMM frames are generated with a periodicity determined by MI_DM_Period and with a priority determined by MI_DM_Pri. The result (B_FD, F_FD, N_FD) is reported per a DMR reception. If the proactive DM control process activates the multiple monitoring on different CoS levels simultaneously, each result is independently managed per CoS level. Optional Test ID TLV can be utilized to distinguish each measurement if multiple measurements are simultaneously activated in an ME. If the protocol is used in multipoint-to-multipoint environments, the multicast class 1 address is used for DA and the test result is independently managed per peer node.

8.1.10.2 DM Control process

The behaviour of the on-demand DM Control process is defined in Figure 8-42.

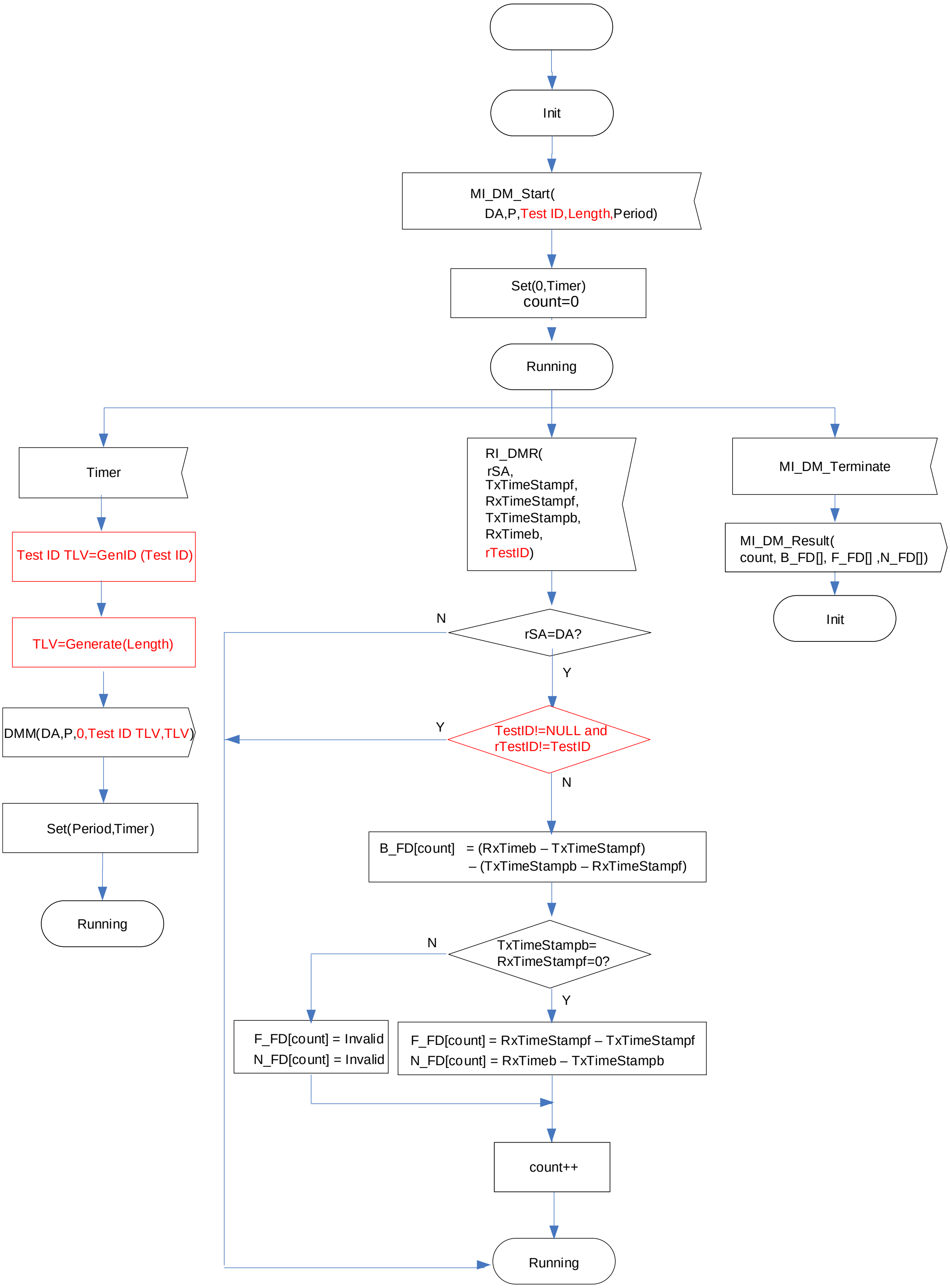


Figure 8-42 – On-demand DM Control behaviour

Upon receipt of the MI_DM_Start(DA,P,Test ID,Length,Period), the DM protocol is started. Every Period the generation of a DMM frame is triggered (using the DMM(DA,P,0,Test ID TLV,TLV) signal), until the MI_DM_Terminate signal is received. The TLV field of the DMM frames can have two types of TLVs. The first one is the Test ID TLV, which is optionally used for a discriminator of each test and the value 'Test ID' is included in the TLV. The second one is the Data

TLV, which is determined by the Generate(Length) function. Generate(Length) generates a Data TLV with length 'Length' of arbitrary bit pattern to be included in the DMM frame.

Upon receipt of a DMR traffic unit the delay value recorded by this particular DMR traffic unit is calculated. This result is reported using the MI_DM_Result(count, B_FD[], F_FD[] ,N_FD[]) signal after the receipt of the MI_DM_Terminate signal. Note that the measurements of F_FD and N_FD are not supported by peer MEP if both TxTimeStampb and TxTimeStampf are zero.

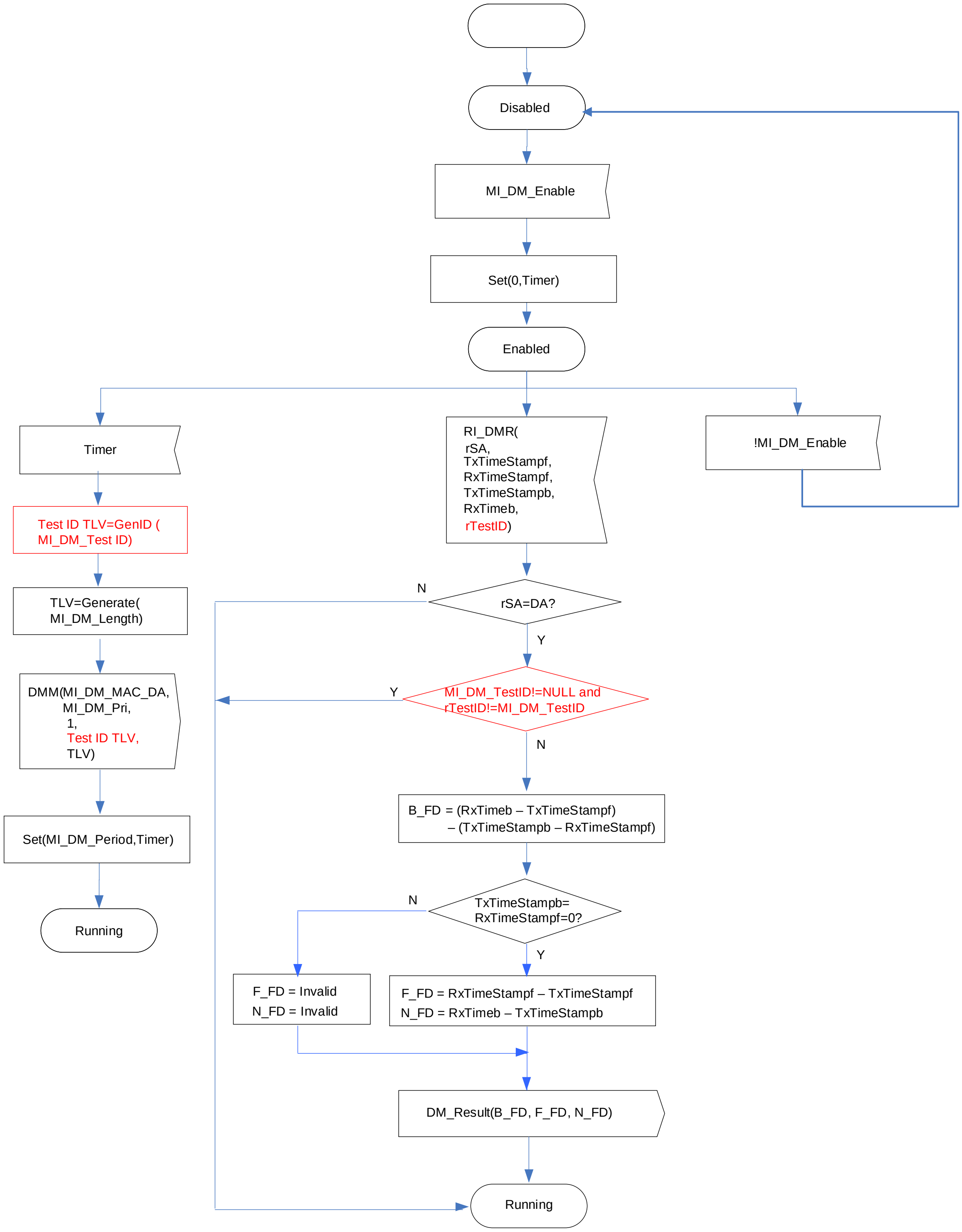


Figure 8-42bis – Proactive DM Control behaviour

The behaviour of the proactive DM Control process is defined in Figure 8-42bis. If the MI_DM_Enable is asserted, the process starts to generate DMM frames (using the DMM(MI_DM_MAC_DA,MI_DM_Pri,1,Test ID TLV,TLV) signal). The result (B_FD, F_FD, N_FD) is reported per a DMR reception.

8.1.10.3 DMM Generation process

The behaviour of the DMM Generation process is defined in Figure 8-43.

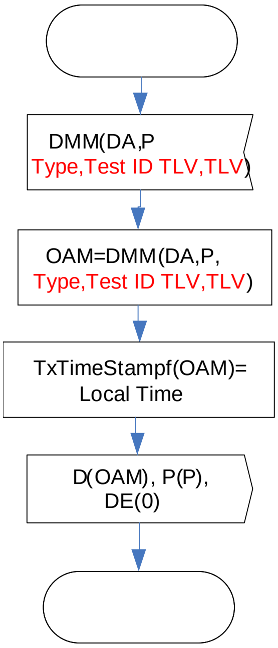


Figure 8-43 – DMM Generation behaviour

Upon receiving the DMM(DA,P,Type,Test ID TLV,TLV), a single DMM traffic unit is generated together with the complementing P and DE signals. The DA of the generated traffic unit is determined by the DMM(DA) signal. The TxTimeStampf field is assigned the value of the local time.

The P signal value is defined by DMM(P). The DE signal is set to 0. The Type signal is set to 1 if it is the proactive OAM, or set to 0 if it is the on-demand OAM operation. The Test ID signal is determined by the DMM(Test ID TLV) signal. The TLV signal is determined by the DMM(TLV) signal. If both Test ID TLV and Data TLV are included in the DMM PDU, it is recommended that Test ID TLV be located at the beginning of the optional TLV field. It makes easier the classification of the Test ID in the received PDUs.

	1	2	3	4
	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	DA=DMM(DA)			
5			SA=Undefined	
9				
13	Ethertype=89-02		MEL=Undef Version=~~0~~1	Opcode=47 (DMM)
17	0 0 0 0 0 0 0 Type	TLV Offset =32	TxTimeStampf=Local Time	
21				
25			0 (Reserved for DMM receiving equipment)	
29				
33			0 (Reserved for DMR)	
37				
41			0 (Reserved for DMR receiving equipment)	
45				
49			Test ID TLV=DMM(Test ID TLV) if exists	
53	Test ID TLV Continued		Data TLV= DMM (TLV) if exists	
57				
61				
:				
Last				END TLV (0)
~~49~~			~~END TLV=0~~	

Figure 8-44 – DMM traffic unit

8.1.10.4 DMM Reception process

The DMM Reception process processes the received DMM traffic units and the complementing P and DE signals. The behaviour is defined in Figure 8-45.

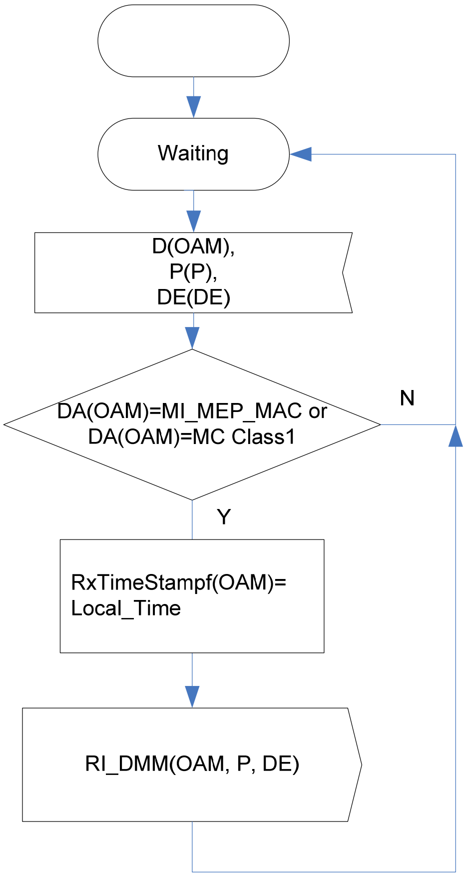


Figure 8-45 – DMM Reception behaviour

First the DA is checked, it should be the Local MAC address or a Multicast Class 1 address, otherwise the frame is ignored.

If the DA is the Local MAC or a Multicast Class 1 address the RxTimeStampf field is assigned the value of the local time and traffic unit and the complementing P and DE signals are forwarded as remote information to the DMR Generation process.

8.1.10.5 DMR Generation process

The DMR Generation process generates a DMR traffic unit and its complementing P and DE signals. The behaviour is defined in Figure 8-46.

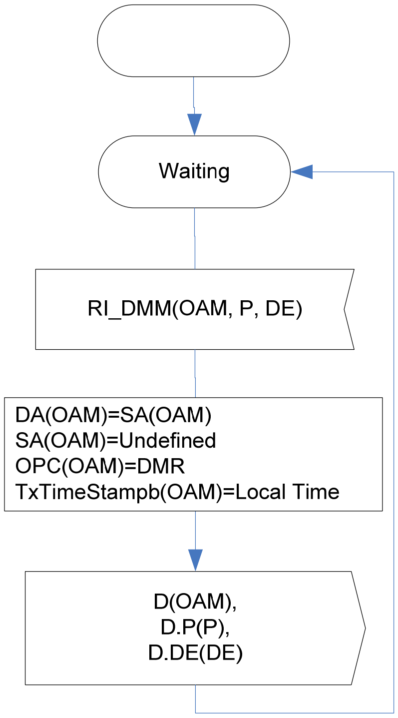


Figure 8-46 – DMR Generation behaviour

Upon the receipt of remote information containing a DMM traffic unit, the DMR Generation process generates a DMR traffic unit and forwards it to the OAM Insertion process.

As part of the DMR generation the:

- DA of the DMR traffic unit is the SA of the original DMM traffic unit.
- The Opcode is changed into DMR Opcode.
- The TxTimeStampb field is assigned the value of the local time.
- All the other fields (including TLVs and padding after the End TLV) are copied from the remote information containing the original DMM traffic unit.

The resulting DMR traffic unit is shown in Figure 8-47.

NOTE – In the generated DMR, in the OAM (MEP) Insertion process, the SA will be overwritten with the local MAC address, and the MEL will be over written with MI_MEL.

The TLVs are copied from the remote information containing the original DMM traffic unit. If multiple TLVs exist, the order of the TLVs is unchanged.

	1	2	3	4
	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	DA=SA(RI_DMM(D))			
5			SA=Undefined	
9				
13	Ethertype=89-02		MEL= Undef Version=~~0~~ Version (RI_DMM(D))	Opcode=46 (DMR)
17	Flags= Flags(RI_DMM(D))	TLV Offset= TLV Offset(RI_DMM(D))	TxTimeStampf=TxTimeStampf(RI_DMM(D))	
21				
25			RxTimeStampf=RxTimeStampf(RI_DMM(D))	
29				
33			TxTimeStampb=Local Time	
37				
41			0 (Reserved for DMR reception process)	
45				
49			Test ID TLV=Test ID(RI_DMM(D)) if exists~~TLV=TLV(RI_DMM(D))~~	
53	Test ID TLV Continued		Data TLV= TLV (RI_DMM(D)) if exists	
57				
61				
:				
last				END TLV= END TLV(RI_DMM(D))
~~49~~			~~END TLV= END TLV(RI_DMM(D))~~	

Figure 8-47 – DMR traffic unit

8.1.10.6 DMR Reception process

The DMR Reception process processes the received DMR traffic units and the complementing P and DE signals. The behaviour is defined in Figure 8-48.

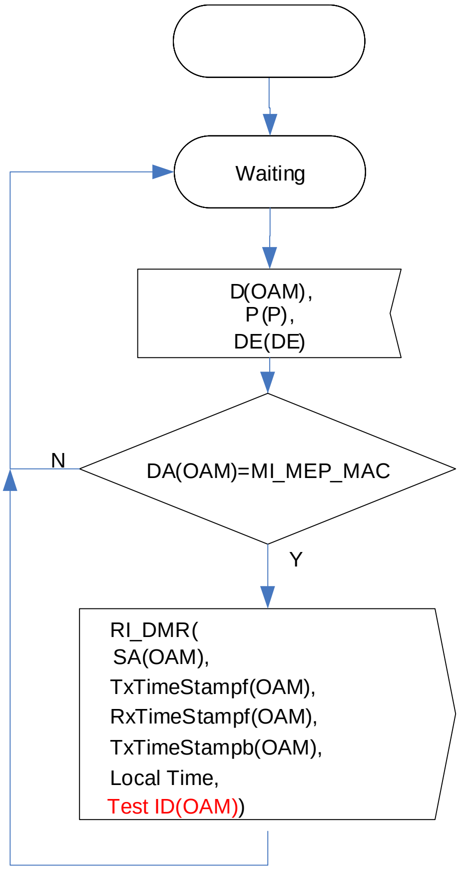


Figure 8-48 – DMR Reception behaviour

Upon receipt of a DMR traffic unit the DA field of the traffic unit is checked. If the DA field equals the local MAC address, the DMR traffic unit is processed further, otherwise it is ignored.

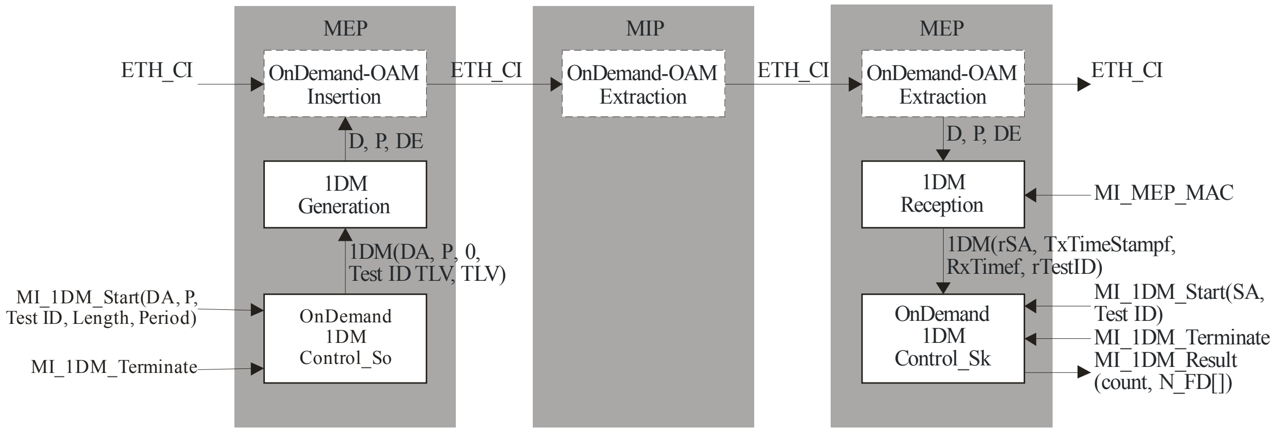
If the DMR traffic unit is processed, the TxTimeStampf, RxTimeStampf, ~~and~~ TxTimeStampb and Test ID are extracted from the traffic unit and signalled together with the local time.

8.1.11 One-way delay measurement (1DM) processes

8.1.11.1 Overview

Figure 8-49 shows the different processes inside MEPs and MIPs that are involved in the on-demand one-way delay measurement protocol.

The MEP OnDemand-OAM Source insertion process is defined in clause 9.4.1.1, the MEP OnDemand-OAM Sink extraction process in clause 9.4.1.2, the MIP OnDemand-OAM Sink Extraction process in clause 9.4.2.2, and the MIP OnDemand-OAM Source insertion process in clause 9.4.2.1. In summary, they insert and extract ETH_CI OAM signals into and from the stream of ETH_CI_D traffic units and the complementing P and DE signals going through an MEP and MIP; the extraction is based on MEL and Opcode. Furthermore, the insertion process inserts the correct MEL and SA values into the OAM traffic units.



G.8021-Y.1341(10)-Amd.1(11)_F8-49

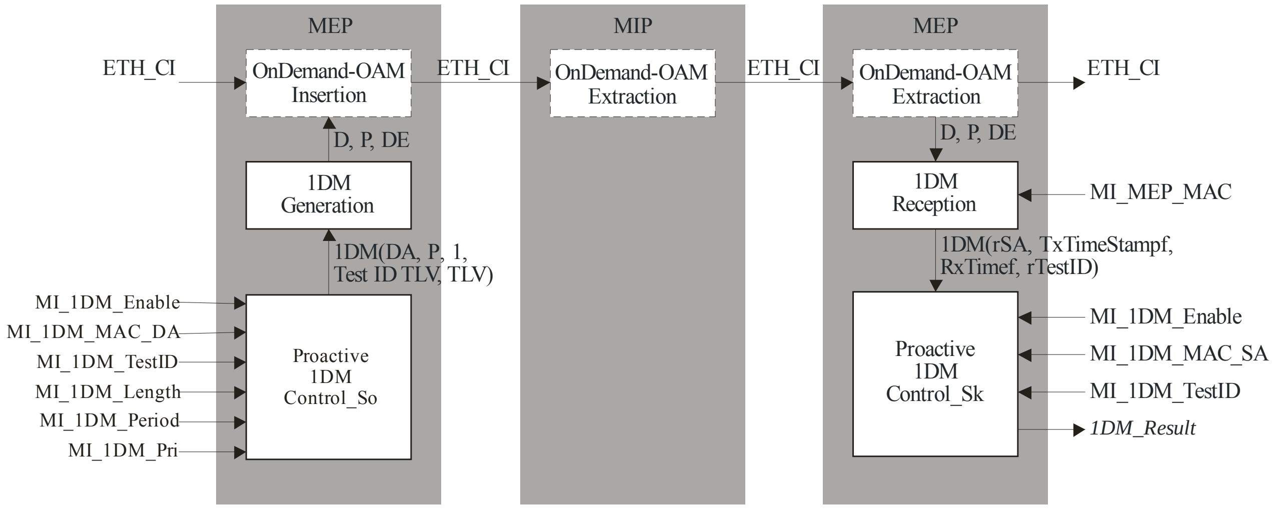
Figure 8-49 – Overview of processes involved with on-demand one-way delay measurement

The on-demand 1DM protocol is controlled by the on-demand 1DM Control_So and 1DM Control_Sk processes. The on-demand 1DM Control_So process triggers the generation of 1DM Traffic Units upon the receipt of an MI_1DM_Start(DA,P,Test ID,Length,Period) signal. The on-demand 1DM Control_Sk process processes the information from received 1DM Traffic Units after receiving the MI_1DM_Start(SA,Test ID) signal.

The 1DM generation process generates 1DM messages that pass transparently through MIPs and are received and processed by the 1DM Reception Process in MEPs.

At the Source MEP side, ~~The~~ the 1DM generation process stamps the value of the Local Time to the TxTimeStampf field in the 1DM message when the first bit of the frame is transmitted. Note well that at the sink MEP side, the 1DM reception process records the value of the Local Time when the last bit of the frame is received.

Figure 8-49bis shows the different processes inside MEPs and MIPs that are involved in the proactive delay measurement protocol.



G.8021-Y.1341(10)-Amd.1(11)_F8-49bis

Figure 8-49bis – Overview of processes involved with proactive one-way delay measurement

The MEP Proactive-OAM Source insertion process is defined in clause 9.2.1.1, and the MEP Proactive-OAM Sink extraction process in clause 9.2.1.2.

The proactive 1DM Control_So process triggers the generation of 1DM traffic units if MI_1DM_Enable signal is set. The 1DM frames are generated with a periodicity determined by MI_1DM_Period and with a priority determined by MI_1DM_Pri. The result (N_FD) is reported per a 1DM reception by the 1DM Control_Sk process.

8.1.11.2 1DM Control_So process

Figure 8-50 shows the behaviour of the on-demand 1DM Control_So process. Upon receipt of the MI_1DM_Start(DA,P,Test ID,Length,Period) signal the 1DM protocol is started. The protocol will run until the receipt of the MI_1DM_Terminate signal.

If the DM protocol is running every period (as specified in the MI_1DM_Start signal) the generation of a 1DM message is triggered by generating the 1DM(DA,P,0,Test ID TLV,TLV) signal towards the 1DM Generation process. The TLV field of the 1DM frames can have two types of TLVs. The first one is the Test ID TLV, which is optionally used for a discriminator of each test and the value 'Test ID' is included in the TLV. The second one is the Data TLV, which is determined by the Generate(Length) function. Generate(Length) generates a Data TLV with length 'Length' of arbitrary bit pattern to be included in the 1DM frame.

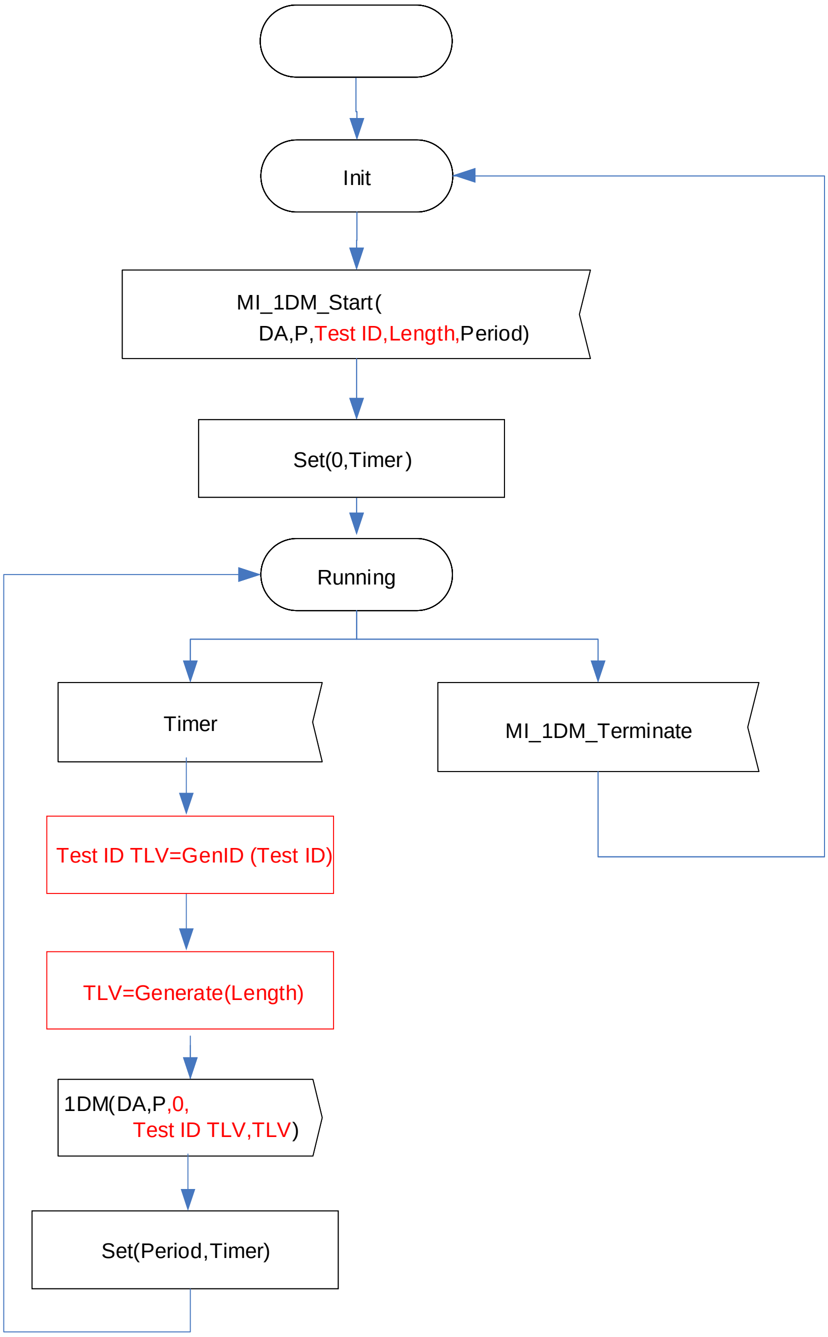


Figure 8-50 – On-demand 1DM Control_So behaviour

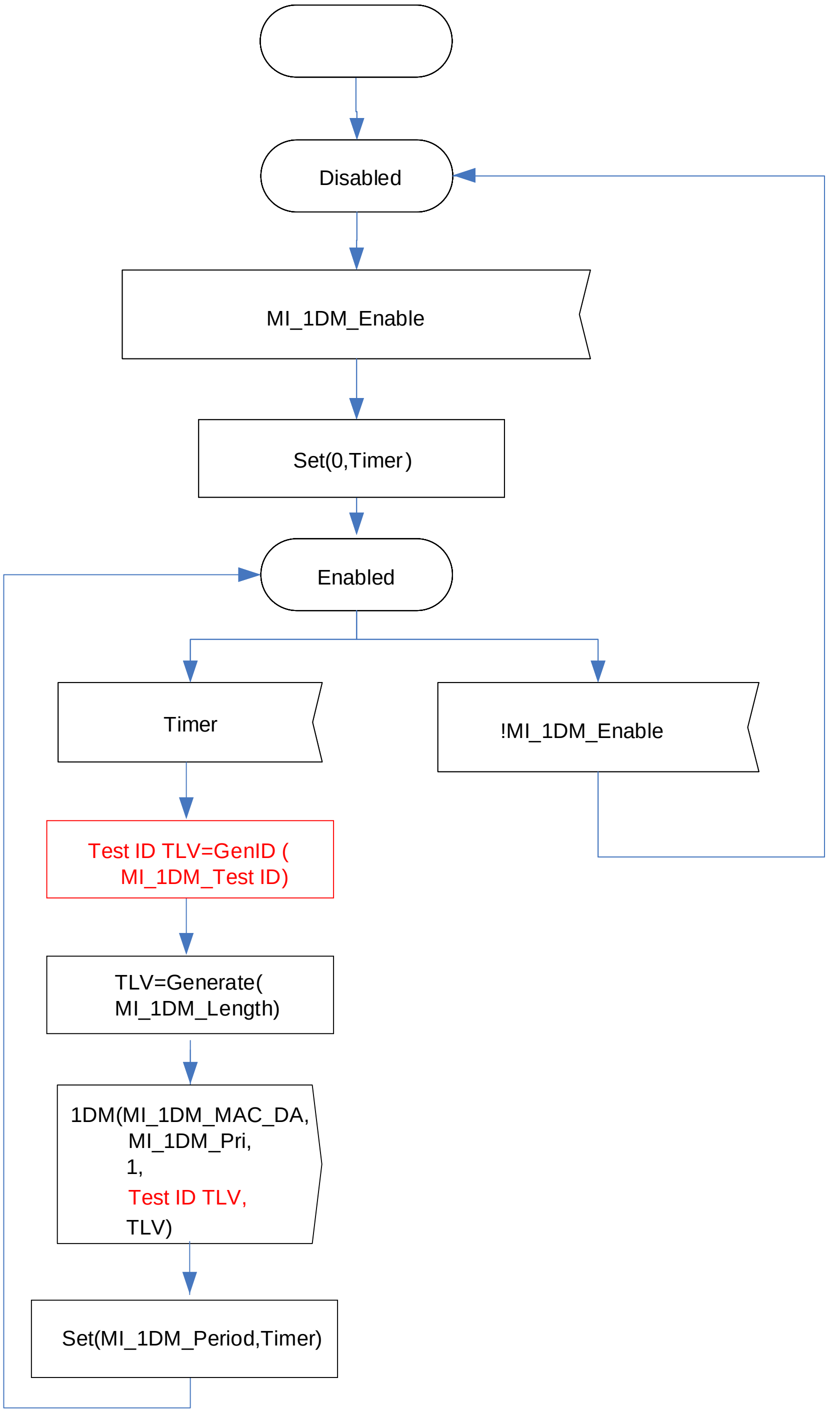


Figure 8-50bis – Proactive 1DM Control_So Behaviour

The behaviour of the proactive 1DM Control process is defined in Figure 8-50bis.

If the MI_1DM_Enable is asserted, the process starts to generate 1DM frames (using the 1DM(MI_1DM_MAC_DA,MI_1DM_Pri,1,Test ID TLV,TLV) signal.

8.1.11.3 1DM Generation process

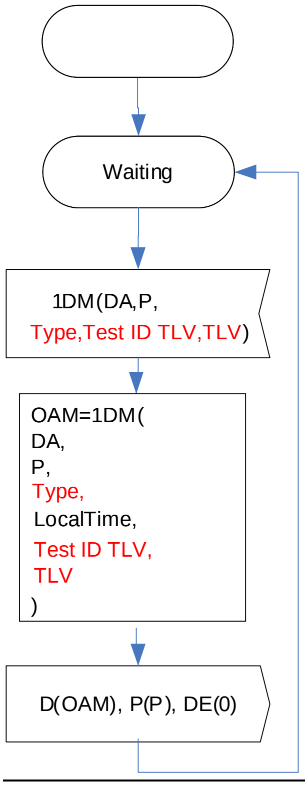


Figure 8-51 – 1DM Generation behaviour

Figure 8-51 shows the 1DM Generation process. Upon receiving the 1DM(DA,P,Type,Test ID TLV,TLV) signal a single 1DM traffic unit is generated by the OAM=1DM (DA,P,Type, LocalTime, Test ID TLV, TLV) call.

Together with this 1DM traffic unit the complementing P and DE signals are generated. The DA of the generated 1DM traffic unit is determined by the 1DM(DA) signal. -The TxTimeStampf field is assigned the value of the local time. The value of the P signal is determined by the 1DM(P) signal. The DE signal is set to 0. The Type signal is set to 1 if it is the proactive OAM, or set to 0 if it is the on-demand OAM operation. The Test ID signal is determined by the 1DM(Test ID TLV) signal. The TLV signal is determined by the 1DM(TLV) signal.

The resulting traffic unit is shown in Figure 8-52.

NOTE – In the generated 1DM traffic unit, in the OAM (MEP) Insertion process, the SA will be assigned the local MAC address, and the MEL will be assigned by MI_MEL.

If both Test ID TLV and Data TLV are included in the 1DM PDU, it is recommended that Test ID TLV be located at the beginning of the optional TLV field. It makes for the easier classification of the Test ID in the received PDUs.

	1	2	3	4
	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	DA=1DM(DA)			
5			SA=Undefined	
9				
13	Ethertype=89-02		MEL= Undef Version=~~0~~1	Opcode=45 (1DM)
17	0 0 0 0 0 0 Type	TLV Offset =16	TxTimeStampf=Local Time	
21				
25			0 (Reserved for 1DM receiving equipment)	
29				
33			Test ID TLV=1DM(Test ID TLV)~~TLV=1DM(TLV)~~ if exists	
37	Test ID TLV Continued		Data TLV=1DM(TLV) if exists	
41				
45				
:				
last				END TLV (0)
~~33~~			~~END TLV=0~~	

Figure 8-52 – 1DM traffic unit

8.1.11.4 1DM Reception process

The 1DM Reception process processes the received 1DM traffic units and the complementing P and DE signals. The behaviour is defined in Figure 8-53.

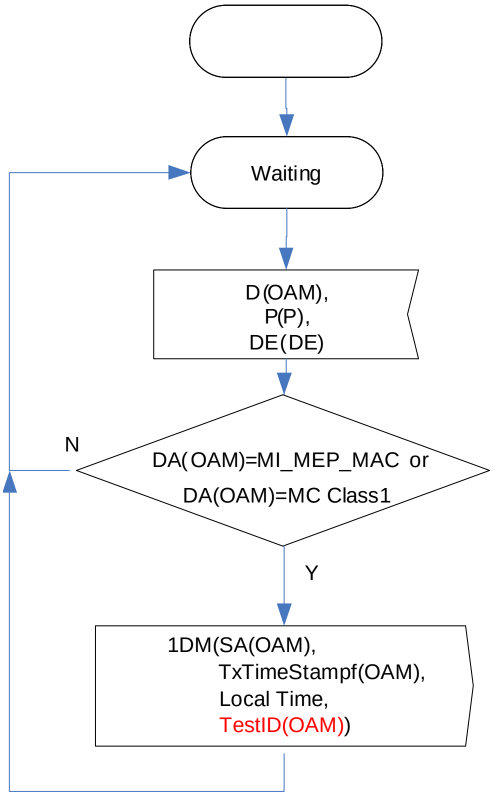


Figure 8-53 – 1DM Reception behaviour

Upon receipt of a~~n~~ 1DM traffic unit the DA field is checked. The 1DM traffic unit is processed if the DA is equal to the local MAC address or Multicast Class 1 MAC address. Otherwise, the traffic unit is ignored.

If the 1DM traffic unit is processed the SA and TxTimeStampf fields are extracted and forwarded to the 1DM Control_Sk process together with the local time using the 1DM(rSA,TxTimeStampf,RxTimef,rTestID) signal.

8.1.11.5 1DM Control_Sk process

Figure 8-54 shows the behaviour of the on-demand 1DM Control_Sk process. The MI_1DM_Start(SA) signal starts the processing of 1DM messages coming from a MEP with SA as MAC address. The protocol runs until the receipt of the MI_1DM_Terminate signal.

While running the process processes the received 1DM(rSA,TxTimeStampf,RxTimef,rTestID) information. First the rSA is compared with the SA from the MI_1DM_Start (SA) signal. If the rSA is not equal to this SA, the information is ignored. Next the rTestID is compared with the TestID from the MI_1DM_Start (Test ID) signal. If the MI_1DM_Start (Test ID) signal is configured and rTestID is available but both values are different, the information is ignored. Otherwise the delay from the single received 1DM traffic unit is calculated. This result is reported using the MI_1DM_Result(count, N_FD[]) signal after the receipt of the MI_1DM_Terminate signal.

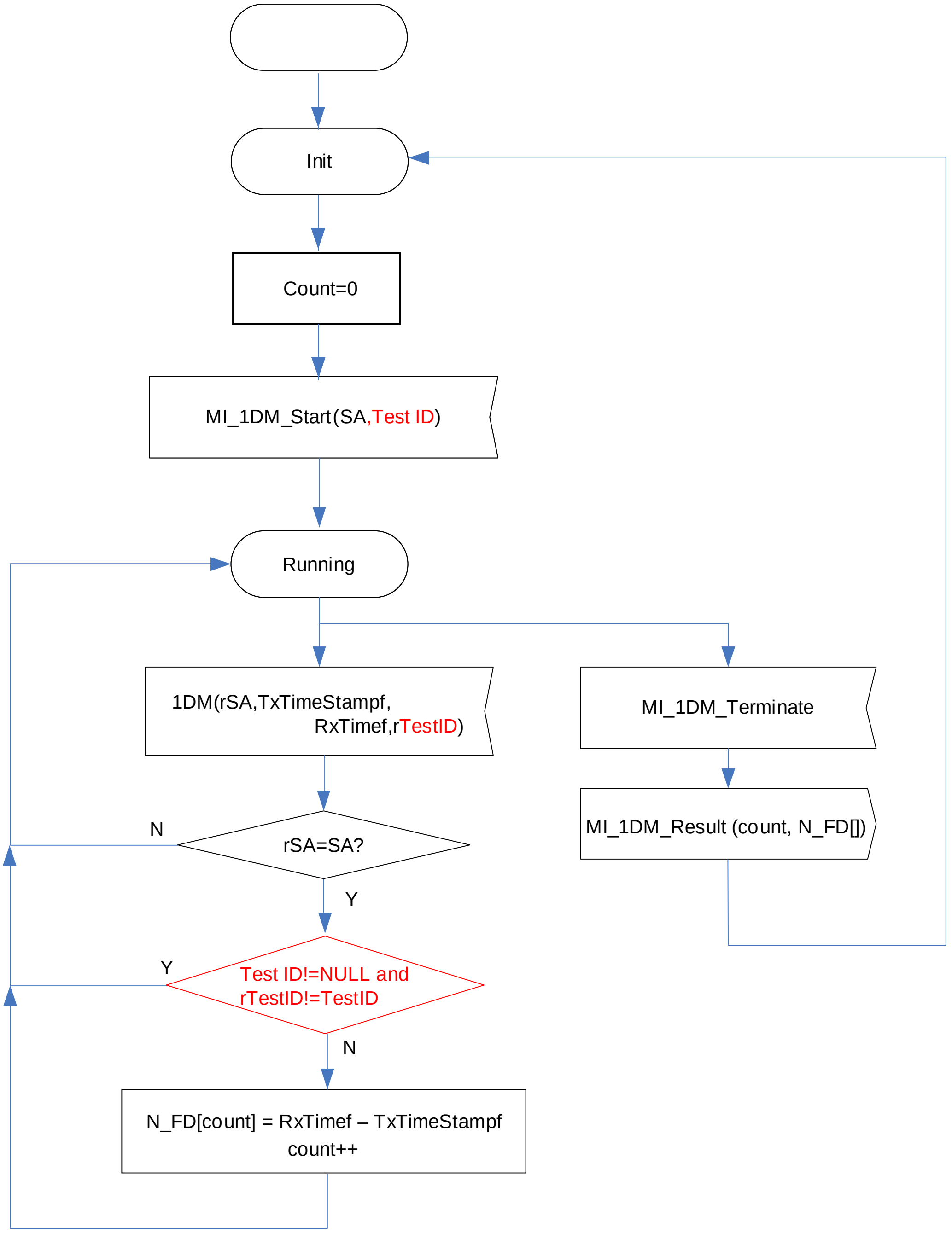


Figure 8-54 – On-demand 1DM Control_Sk process

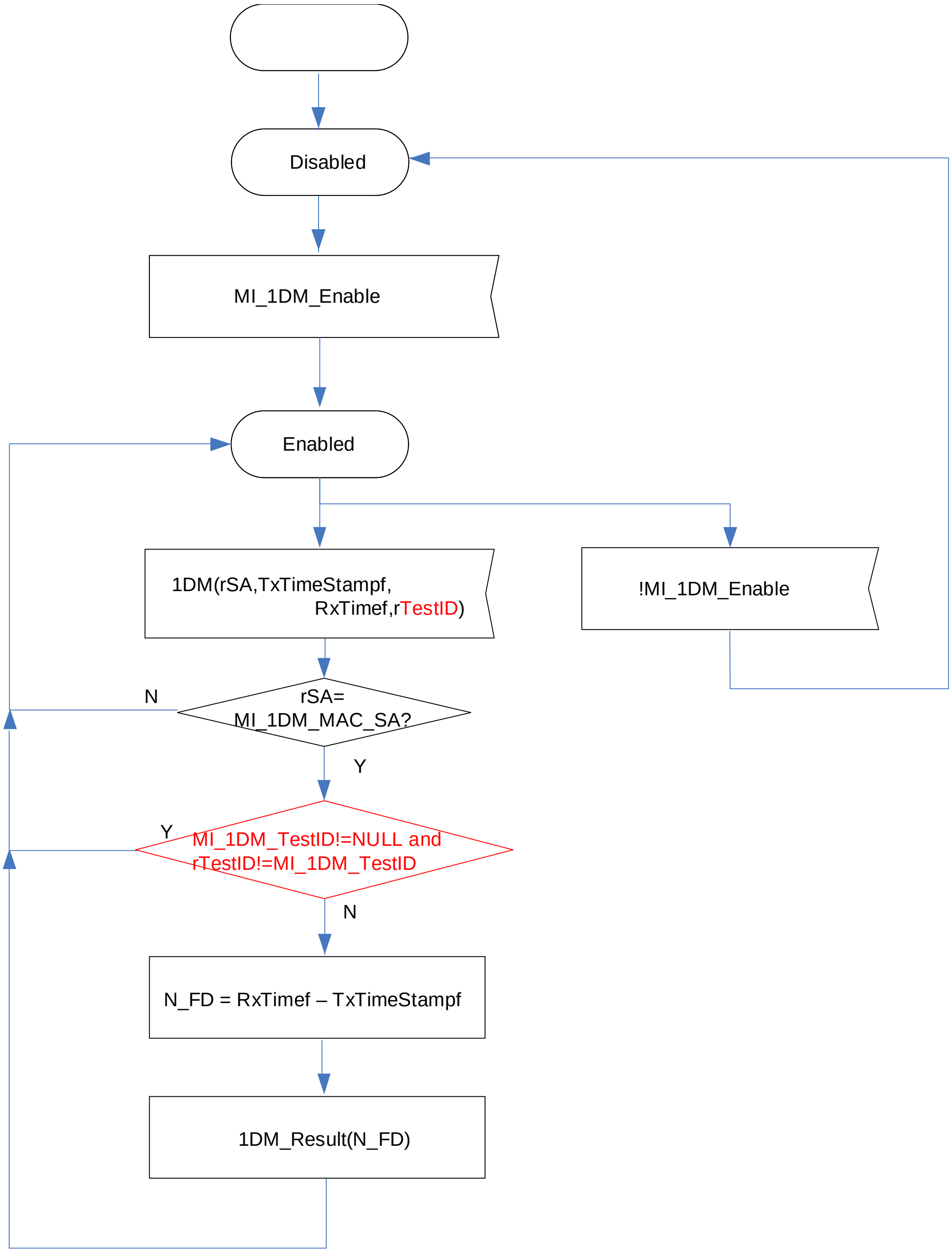


Figure 8-54bis – Proactive 1DM Control_Sk process

The behaviour of the proactive 1DM Control_Sk Process is defined in Figure 8-54bis. If the MI_1DM_Enable is asserted, the result (N_FD) is reported per a 1DM reception.

14) New clause 8.1.14

Add the following clause with respect to SLM:

8.1.14 Synthetic loss measurement (SL) processes

8.1.14.1 Overview

Figure 8.1.14-1 shows the different processes inside MEPs and MIPs that are involved in the on_demand synthetic loss measurement protocol.

The MEP On-demand OAM Insertion process is defined in clause 9.4.1.1, the MEP OAM on-demand Extraction process in clause 9.4.1.2, the MIP OAM Extraction process in clause 9.4.2.2, and the MIP OAM Insertion process in clause 9.4.2.1. In summary, they insert and extract ETH_CI OAM signals into and from the stream of ETH_C_D traffic units and the complementing P and D signals going through a MEP and MIP; the extraction is based on MEL and Opcode. Furthermore, the Insertion process inserts the correct MEL and SA values into the OAM traffic units.

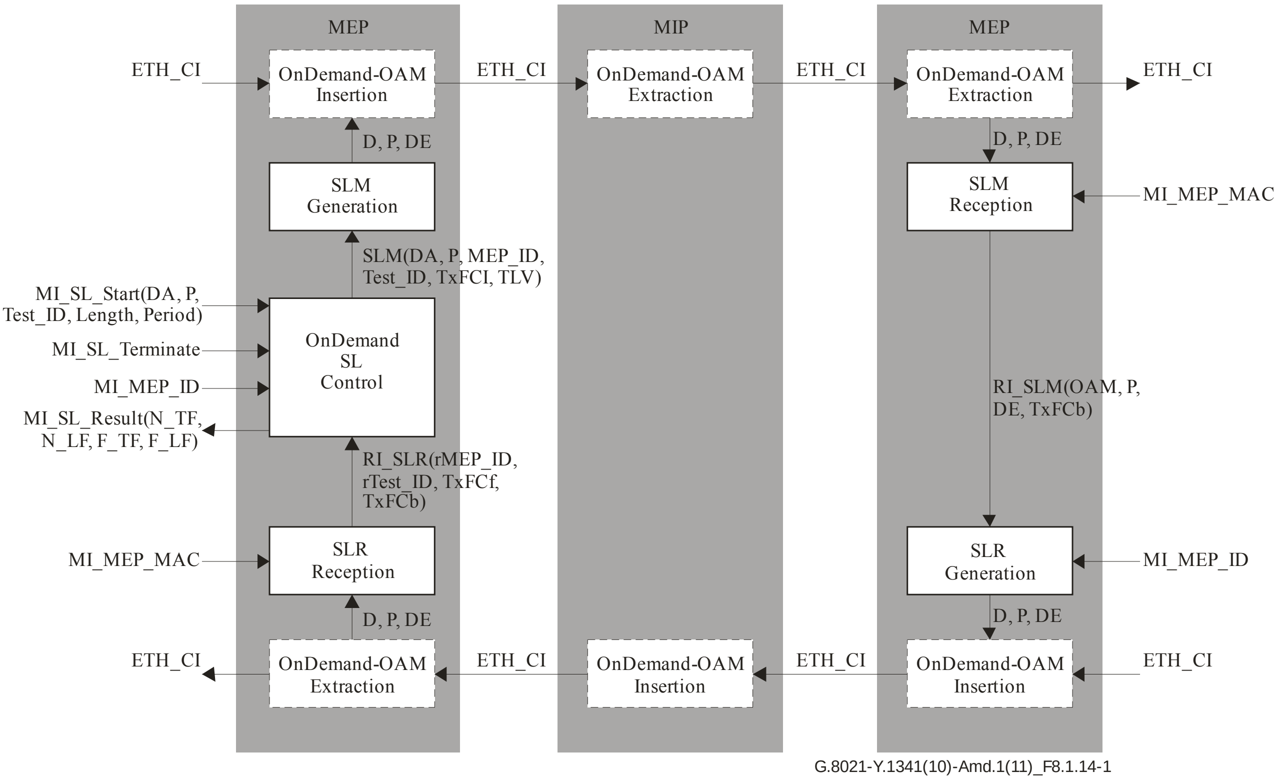


Figure 8.1.14-1 – Overview of processes involved with on-demand synthetic loss measurement protocol

The SL protocol is controlled by the SL Control process.

The On-demand SL Control process is activated upon receipt of the MI_SL_Start(DA,P,Test_ID,Length,Period) signal and remains activated until the MI_SL_Terminate signal is received. The measured synthetic loss values are output after the MI_SL_Terminate signal via the MI_SL_Result(N_TF,N_LF,F_TF,F_LF) signal.

The SLM generation process generates SLM traffic units that pass through MIPs transparently, but are received and processed by SLM reception processes in MEPs. The SLR generation process may generate an SLR traffic unit in response. This SLR traffic unit also passes transparently through MIPs, but is received and processed by SLR reception processes in MEPs.

Figure 8.1.14-2 shows the different processes inside MEPs and MIPs that are involved in the proactive synthetic loss measurement protocol.

The MEP proactive OAM Insertion process is defined in clause 9.2.1.1, the MEP OAM proactive Extraction process in clause 9.2.1.2, the MIP OAM Extraction process in clause 9.4.2.2, and the MIP OAM Insertion process in clause 9.4.2.1. In summary, they insert and extract ETH_CI OAM signals into and from the stream of ETH_C_D traffic units and the complementing P and D signals going through a MEP and MIP; the extraction is based on MEL and Opcode. Furthermore, the Insertion process inserts the correct MEL and SA values into the OAM traffic units.

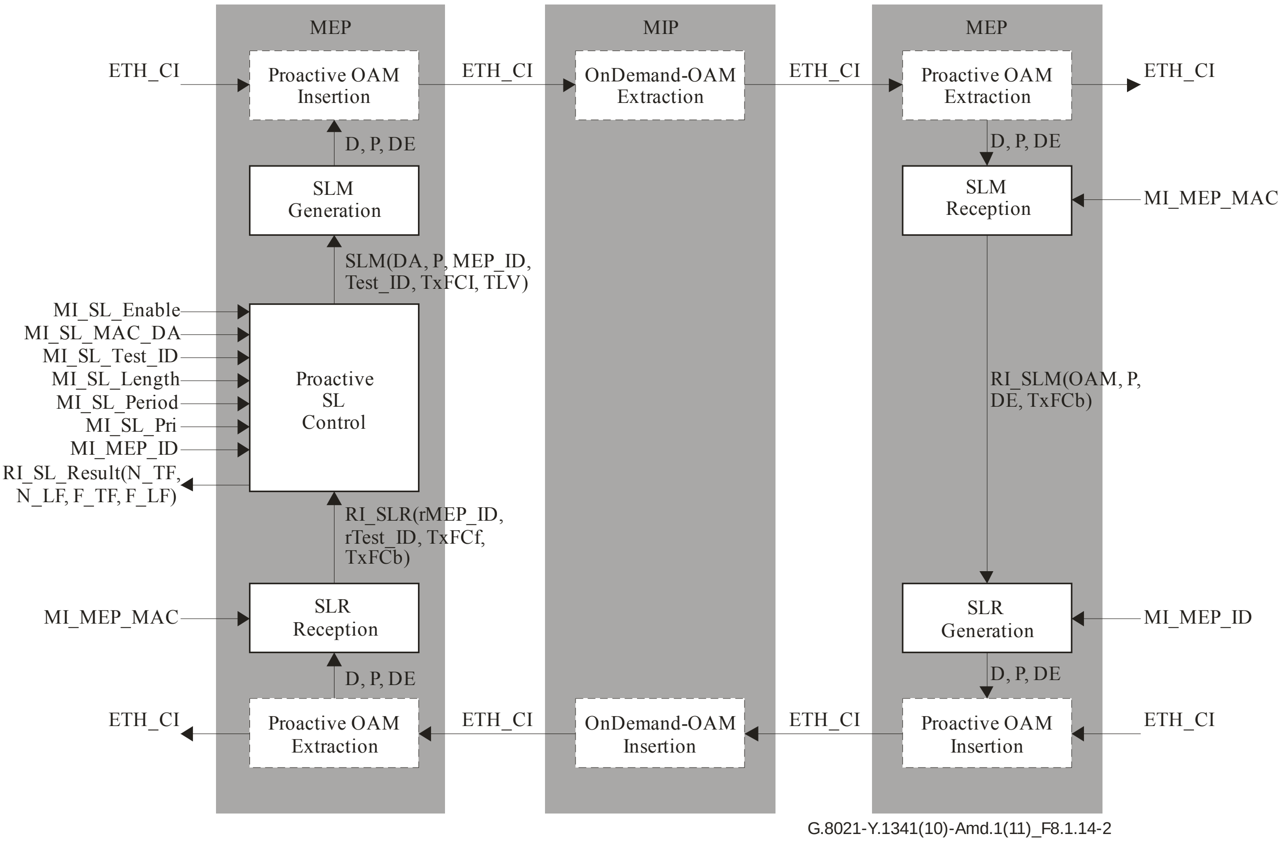


Figure 8.1.14-2 – Overview of processes involved with proactive synthetic loss measurement protocol

The SL protocol is controlled by the proactive SL control processes.

The Proactive SL Control process is activated upon receipt of the MI_SL_Enable signal and remains activated until the signal is deactivated. The measured results are output every 1 s using the RI_SL_Result (N_TF, N_LF, F_TF, F_LF) signal.

8.1.14.2 SL Control process

The behaviour of the on-demand SL Control process is defined in Figure 8.1.14-3. There are multiple instances of the on-demand SL Control process, each handling an independent stream of SLM frames.

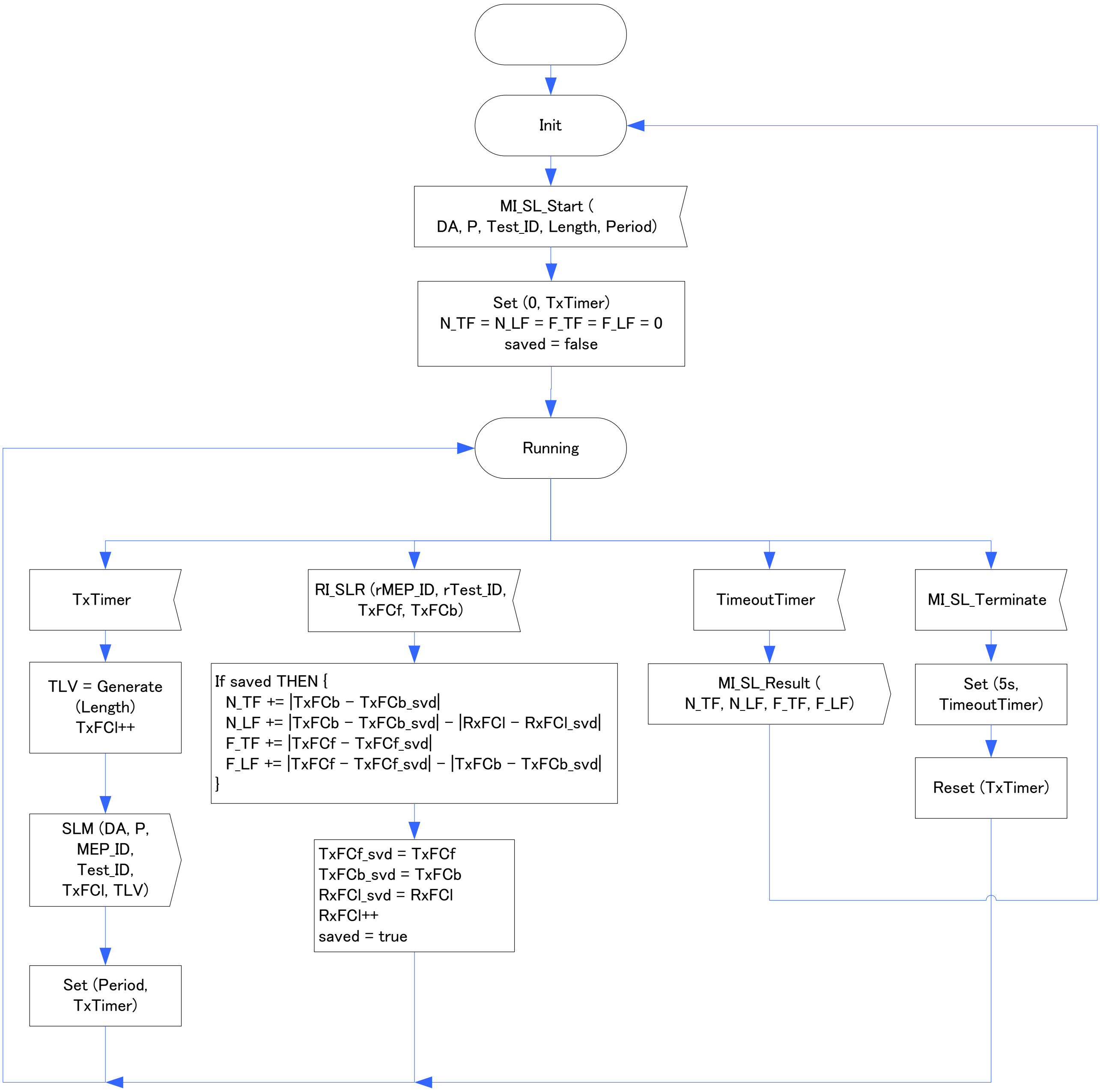


Figure 8.1.14-3 – On-demand SL Control behaviour

Upon receipt of the MI_SL_Start(DA,P,Test ID,Length,Period), the SL protocol is started. Every designated 'period' the generation of an SLM frame is triggered (using the SLM(DA,P,MEP_ID,Test_ID,TxFCl,TLV) signal), until the MI_SL_Terminate signal is received. The MEP_ID is the MI_MEP_ID of the MEP itself. The TLV field of the SLM frames is determined by the Generate(Length) function. Generate(Length) generates a Data TLV with length 'Length' of arbitrary bit pattern, as described in clause 8.1.8.2. If the Length is 0, the TLV is set to Null.

Upon receipt of an SLR traffic unit, the received counter values are used to count the near-end and far-end transmitted and lost synthetic frames. This result is reported using the MI_SL_Result(N_TF,N_LF,F_TF,F_LF) signal after the receipt of the MI_SL_Terminate signal.

The behaviour of the Proactive SL Control process is defined in Figure 8.1.14-4. There are multiple instances of the Proactive SL Control process, each handling an independent stream of SLM frames.

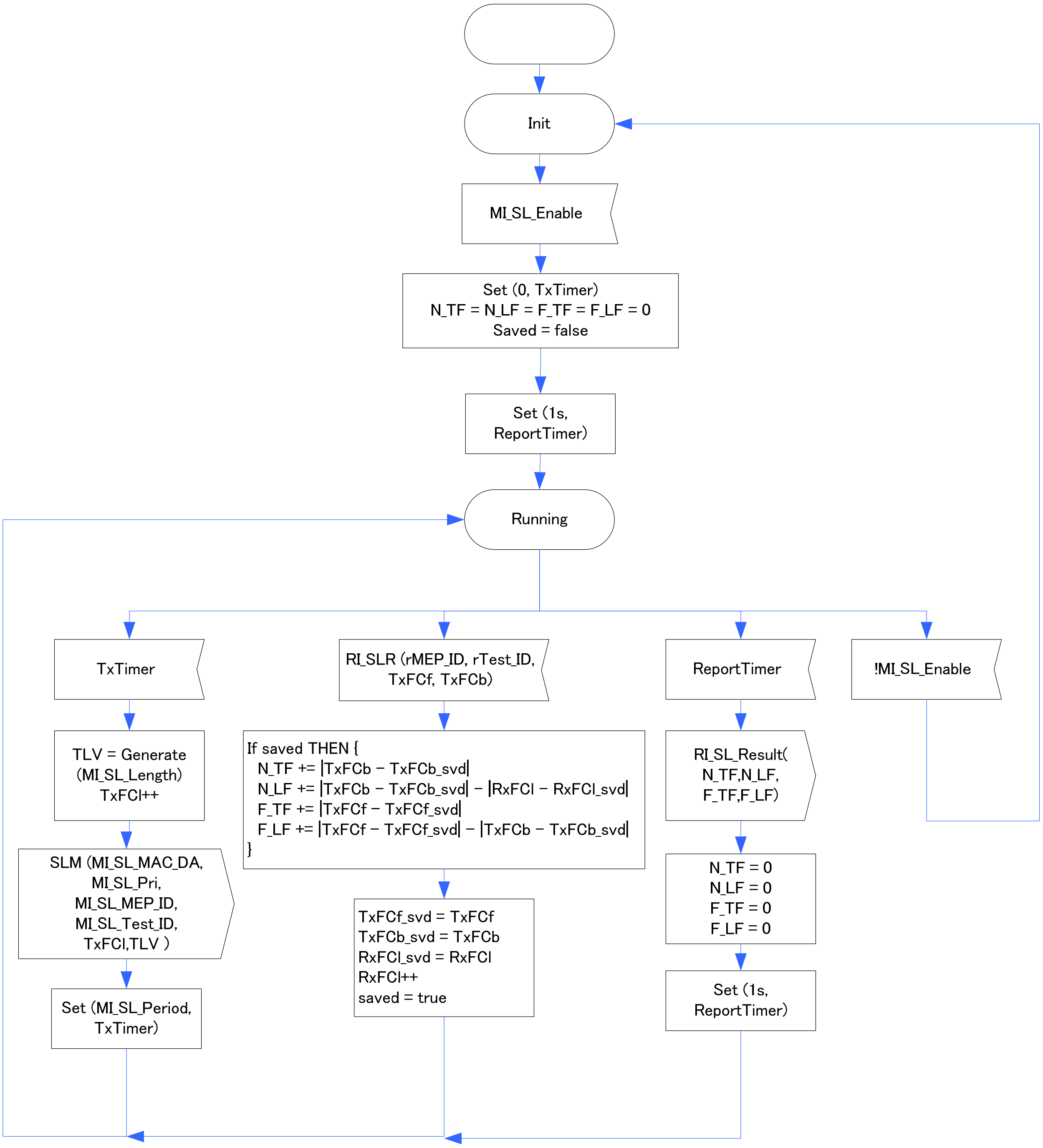


Figure 8.1.14-4 – Proactive SL Control behaviour

Upon receipt of the MI_SL_Enable, the SL protocol is started. Every designated MI_SL_Period the generation of an SLM frame is triggered (using the SLM(MI_SL_MAC_DA,MI_SL_Pri,MI_MEP_ID,MI_SL_Test_ID,TxFCl,TLV) signal). The TLV field of the SLM frames is determined by the Generate(MI_SL_Length) function. Generate(MI_SL_Length) generates a Data TLV with MI_SL_ Length of arbitrary bit pattern, as described in clause 8.1.8.2. If the MI_SL_Length is 0, the TLV is set to Null.

Upon receipt of an SLR traffic unit, the received counter values are used to count the near-end and far-end transmitted and lost synthetic frames. The calculation is performed every 1 s and the RI_SL_Result(N_TF, N_LF, F_TF, N_LF) signal is generated.

8.1.14.3 SLM Generation process

The behaviour of the SLM Generation process is defined in Figure 8.1.14-5.

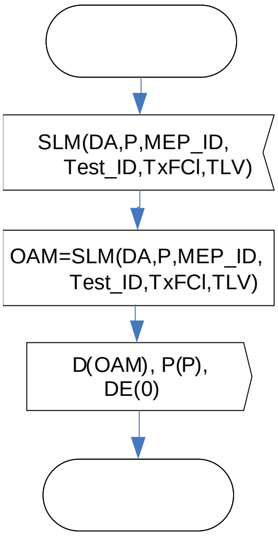


Figure 8.1.14-5 – SLM Generation behaviour

Upon receiving the SLM(DA,P,MEP_ID,Test_ID,TxFCl,TLV), a single SLM traffic unit is generated together with the complementing P and DE signals. The DA, Source MEP_ID, Test_ID and TxFCf of the generated traffic unit are determined by the DA, MEP_ID, Test_ID and TxFCl respectively in the SLM(DA,P,MEP_ID,Test_ID,TxFCl,TLV) signal. If not Null, the specified TLV is appended to the traffic unit as shown in Figure 8.1.14-6.

The P signal value is defined by SLM(P). The DE signal is set to 0.

	1	2	3	4
	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	DA=SLM(DA)			
5			SA=Undefined	
9				
13	Ethertype=89-02		MEL= Undef Version=0	Opcode=55 (SLM)
17	Flags=0	TLV Offset = 16	Source_MEP_ID = SLM(MI_MEP_ID)	
21	0 (reserved for Responder_MEP_ID)		Test_ID = SLM(Test_ID)	
25	Test_ID Continued		TxFCf = SLM(TxFCl)	
29	TxFCf Continued		Reserved for TxFCb	
33	Reserved Continued		TLV = SLM(TLV) if exists	
37				
41				
45				
:				
last				END TLV (0)

Figure 8.1.14-6 – SLM traffic unit

8.1.14.4 SLM Reception process

The SLM Reception process processes the received SLM traffic units and the complementing P and DE signals. The behaviour is defined in Figure 8.1.14-7.

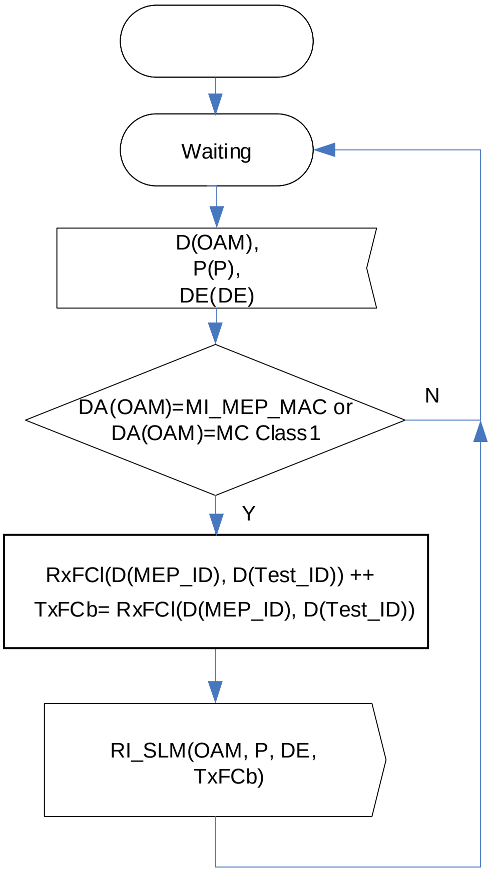


Figure 8.1.14-7 – SLM Reception behaviour

First the DA is checked, it should be the local MAC address or a Multicast Class 1 address, otherwise the frame is ignored.

If the DA is the local MAC or a Multicast Class 1 address, the MEP_ID and the Test_ID fields are extracted from the traffic unit. The local received counter RxFCl, maintained per MEP_ID and Test_ID values, is incremented. The received OAM information, P and DE signals, as well as local TxFCb value are forwarded as remote information to the SLR generation process using the RI_SLM(OAM,P,DE, TxFCb) signal.

NOTE – The SLM reception process allocates and maintains local resources for the counter RxFCl per MEP_ID and Test_ID. To facilitate the automatic release of local resources, a timer for monitoring no receipt of SLM can be utilized. The SLM reception process must ensure there is no discontinuity in RxFCl for the given MEP ID and Test ID for some interval (e.g., 5 minutes) after the last received SLM for that MEP ID and Test ID. The detailed mechanism for the release is out of scope of this Recommendation.

8.1.14.5 SLR Generation process

The SLR Generation process generates an SLR traffic unit and its complementing P and DE signals. The behaviour is defined in Figure 8.1.14-8.

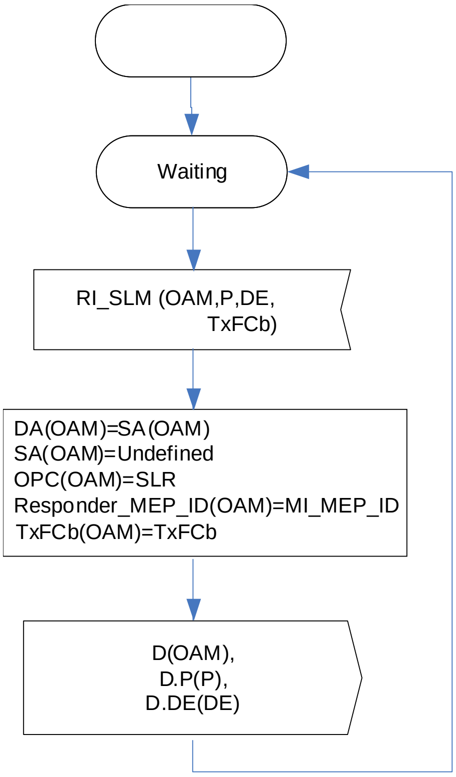


Figure 8.1.14-8 – SLR Generation behaviour

Upon the receipt of the RI_SLM (P,DE,OAM, TxFCb) signal containing an SLM traffic unit, the SLR generation process generates an SLR traffic unit and forwards it to the MEP OAM Insertion process.

As part of the SLR generation:

– The DA of the SLR traffic unit is the SA of the original SLM traffic unit.

– The Opcode is changed into SLR Opcode.

– The responder MEP_ID is set to MI_MEP_ID.

– TxFCb field is assigned the TxFCb value passed in the SLR(TxFCb).

– The other fields and optional TLVs are copied from the SLM.

The resulting SLR traffic unit is shown in Figure 8.1.14-9.

NOTE – In the generated SLR, in the OAM (MEP) Insertion process, the SA will be overwritten with the local MAC address, and the MEL will be overwritten with MI_MEL.

	1	2	3	4
	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
1	DA=SA(RI_SLM (OAM))			
5			SA=Undefined	
9				
13	Ethertype=89-02		MEL= Undef Version=0	Opcode=54(SLR)
17	Flags=Flags (RI_SLM(OAM))	TLV Offset = TLV Offset((RI_SLM(OAM))	Source_MEP_ID = Source_MEP_ID((RI_SLM(OAM))	
21	Responder_MEP_ID = MI_MEP_ID		Test_ID = Test_ID((RI_SLM(OAM))	
25	Test_ID Continued		TxFCf = TxFCf((RI_SLM(OAM))	
29	TxFCf Continued		TxFCb = RI_SLM(TxFCb)	
33	TxFCb Continued		TLV = TLV((RI_SLM(OAM)) if exists	
37				
41				
45				
:				
last				END TLV = END TLV(RI_SLM (OAM))

Figure 8.1.14-9 – SLR traffic unit

8.1.14.6 SLR Reception process

The SLR reception process processes the received SLR traffic units and the complementing P and DE signals. The behaviour is defined in Figure 8.1.14-10.

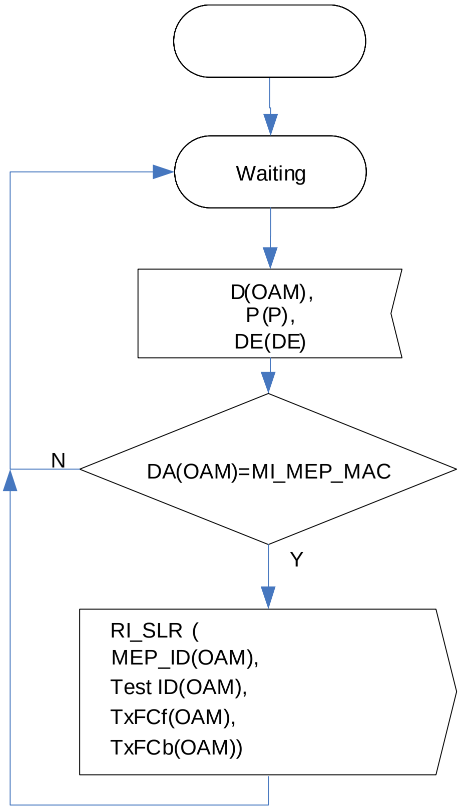


Figure 8.1.14-10 – SLR Reception behavior

Upon receipt of an SLR traffic unit, the DA field of the traffic unit is checked. If the DA field equals the local MAC address, the SLR traffic unit is processed further, otherwise it is ignored.

If the SLR traffic unit is processed, Test_ID, TxFCf, TxFCb, Responder MEP_ID, are extracted from the traffic unit and signalled, using the RI_SLR(MEP_ID, Test_ID,TxFCf,TxFCb) signal.

15) New clause 8.1.15

Add the following new clause with respect to CSF:

8.1.15 CSF Insert process

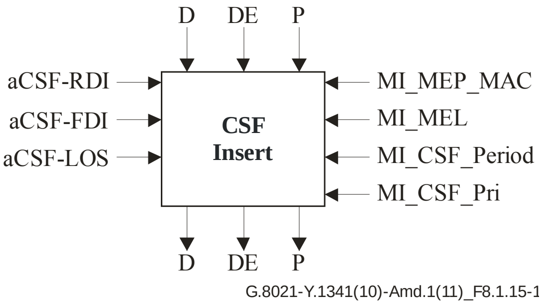


Figure 8.1.15-1 – CSF Insert process

Figure 8.1.15-1 shows the CSF Insert process symbol and Figure 8.1.15-2 defines the behaviour. If the aCSF signal is true, the CSF Insert process continuously generates ETH_CI traffic units where the ETH_CI_D signal contains the CSF signal, until the aCSF signal is false. The generated CSF traffic units are inserted in the incoming stream, i.e., the output stream contains the incoming traffic units and the generated CSF traffic units.

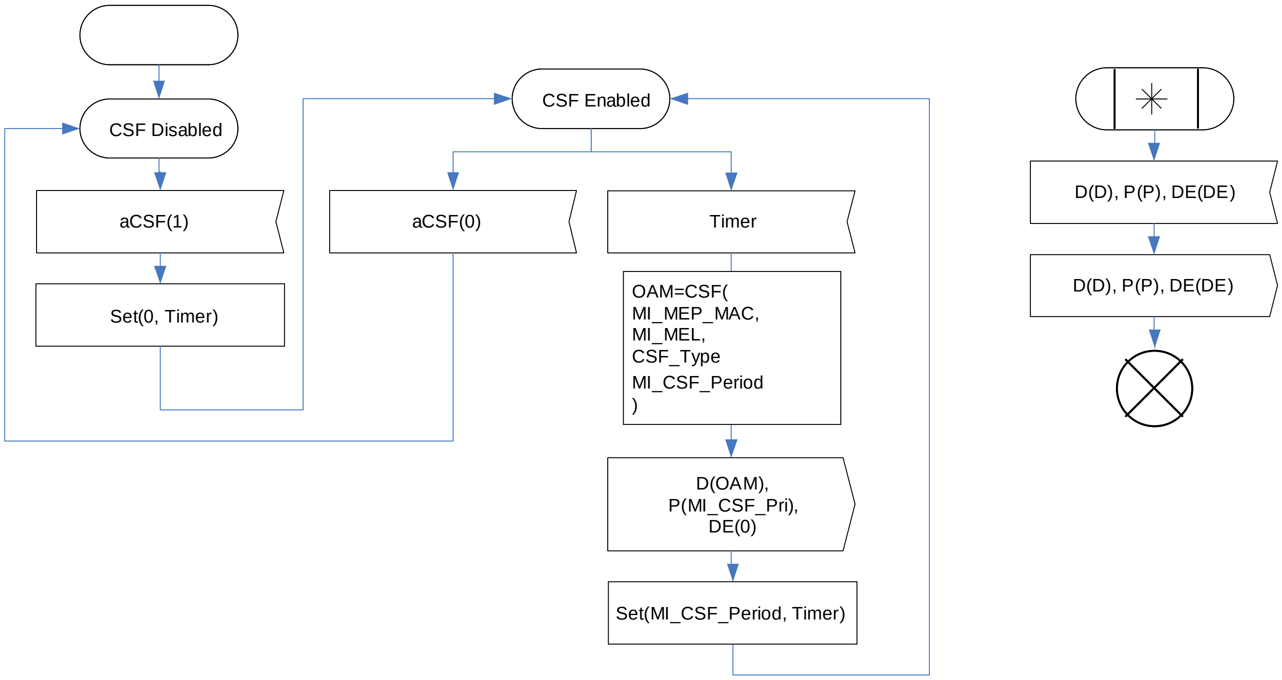


Figure 8.1.15-2 – CSF Insert behaviour

The period between consecutive CSF traffic units is determined by the MI_CSF_Period parameter. Allowed values are once per second and once per minute; the encoding of these values is defined in Table 8.1.15-1. Note that these encoding are the same as for the LCK/AIS generation process.

Table 8.1.15-1 – CSF period values

3-bits	Period value	Comments
000	Invalid value	Invalid value for CSF PDUs
001	FFS	FFS
010	FFS	FFS
011	FFS	FFS
100	1 s	1 frame per second
101	FFS	FFS
110	1 min	1 frame per minute
111	FFS	FFS

The ETH_CI_D signal contains a source and a destination address field and an M_SDU field. The format of the M_SDU field for CSF traffic units is defined in clauses 9.1 and 9.21 of [ITU-T Y.1731]. The MEL in the M_SDU field is determined by the MI_ MEL input parameter.

The values of the Source and Destination address fields in the ETH_CI_D signal are determined by the Local MAC address (SA) and the Multicast class 1 DA as described in [ITU-T Y.1731] (DA). The value of the Multicast class 1 DA is 01-80-C2-00-00-3x, where x is equal to MI_MEL as defined in [IEEE 802.1ag]. The value of MI_MEP_MAC should be a valid unicast MAC address.

The CSF_Type is encoded in the three bits of the Flags field in the CSF PDU using the values from Table 8.1.15-2.

Table 8.1.15-2 – CSF type values

Value	Type	Comments
000	LOS	Client loss of signal
001	FDI/AIS	Client forward defect indication
010	RDI	Client reverse defect indication
011	DCI	Client defect clear indication

The periodicity (as defined by MI_CSF_Period) is encoded in the three least significant bits of the Flags field in the CSF PDU using the values from Table 8.1.15-1.

The CSF (SA, MEL, Type, Period) function generates a CSF traffic unit with the SA, MEL, Type and Period fields defined by the values of the parameters. Figure 8.1.15-3 below shows the ETH_CI_D signal format resulting from the function call from Figure 8.1.15-2:

```
OAM=CSF(
MI_MEP_MAC,
MI_MEL,
CSF_Type,
MI_CSF_Period
)
```

<table>
<tr><td></td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td><td colspan="8">4</td></tr>
<tr><td></td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td></tr>
<tr><td>1</td><td colspan="32">DA=01-80-C2-00-00-3x, where x=MI_MEL</td></tr>
<tr><td>5</td><td colspan="16"></td><td colspan="16">SA=MI_MEP_MAC</td></tr>
<tr><td>9</td><td colspan="32"></td></tr>
<tr><td>13</td><td colspan="16">Ethertype=89-02</td><td colspan="3">MEL=
MI_MEL</td><td colspan="5">Version=0</td><td colspan="8">Opcode=52 (CSF)</td></tr>
<tr><td>17</td><td>0</td><td>0</td><td colspan="3">CSF Type</td><td colspan="3">Period=
MI_CSF_
Period</td><td colspan="8">TLV Offset = 0</td><td colspan="8">END TLV=0</td><td colspan="8"></td></tr>
</table>

Figure 8.1.15-3 – CSF traffic unit

8.1.16 CSF Extract process

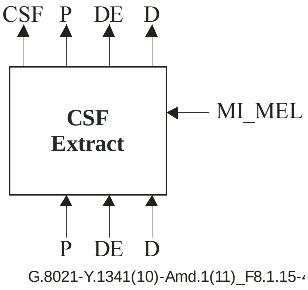


Figure 8.1.15-4 – CSF Extract process

The CSF Extract process extracts ETH_CI_CSF signals from the incoming stream of ETH_CI traffic units. ETH_CI_CSF signals are only extracted if they belong to the MEL as defined by the MI_MEL input parameter.

If an incoming traffic unit is a CSF traffic unit belonging to the MEL defined by MI_MEL, the ETH_CI_CSF signal will be extracted from this traffic unit and the traffic unit will be filtered. The ETH_CI_CSF is the CSF specific information contained in the received traffic unit. All other traffic units will be transparently forwarded. The encoding of the ETH_CI_D signal for CSF frames is defined in clause 9.21 of [ITU-T Y.1731].

The criteria for filtering are based on the values of the fields within the M_SDU field of the ETH_CI_D signal:

- length/type field equals the OAM Ethertype (89-02), and
- MEL field equals MI_MEL, and
- OAM type equals CSF (52), as defined in clause 9.21 of [ITU-T Y.1731].

This is defined in Figure 8.1.15-4. The function CSF(D) extracts the CSF specific information from the received traffic unit.

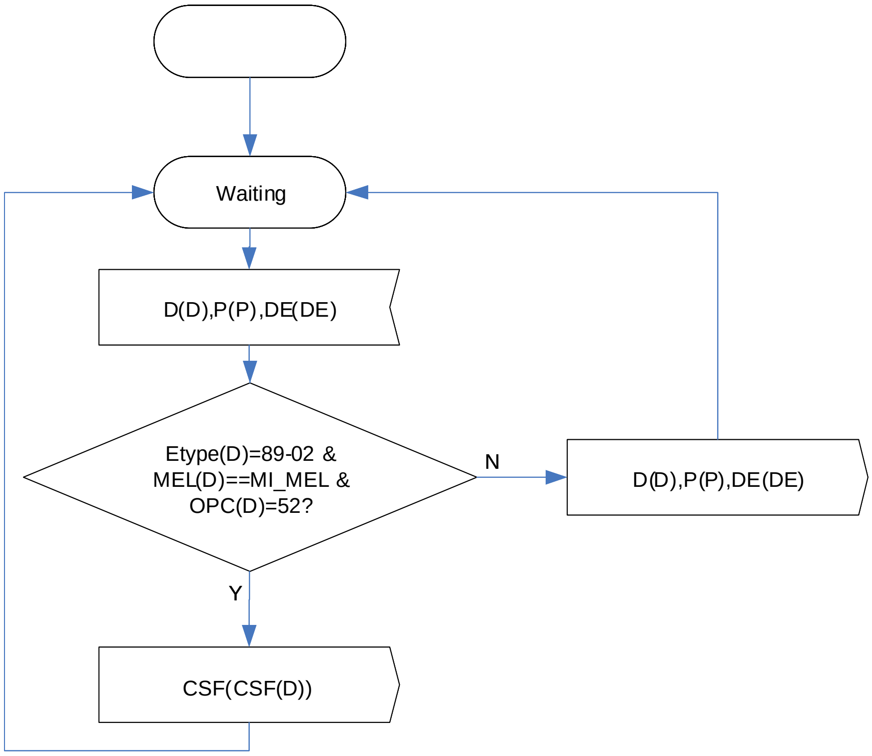


Figure 8.1.15-5 – CSF Extract behaviour

16) Clause 8.6

Revise the first paragraph of clause 8.6 as follows:

This process checks whether the length of the MAC frame is allowed. When the processed signal is ETYn_AI frames shorter than 64 bytes are discarded. Frames longer than MI_MAC_Length are ~~passed~~discarded.

17) Clause 9.1.2

Revise clause 9.1.2 with respect to dFOP-TO as follows:

9.1.2 Subnetwork Connection Protection process

SNC Protection with Sublayer monitoring based on TCM is supported.

Figure 9-9 shows the involved atomic functions in SNC/S. The ETH_FT_Sk provides the TSF/TSD protection switching criterion via the ETH/ETH_A_Sk function (SSF/SSD) to the ETH_C function.

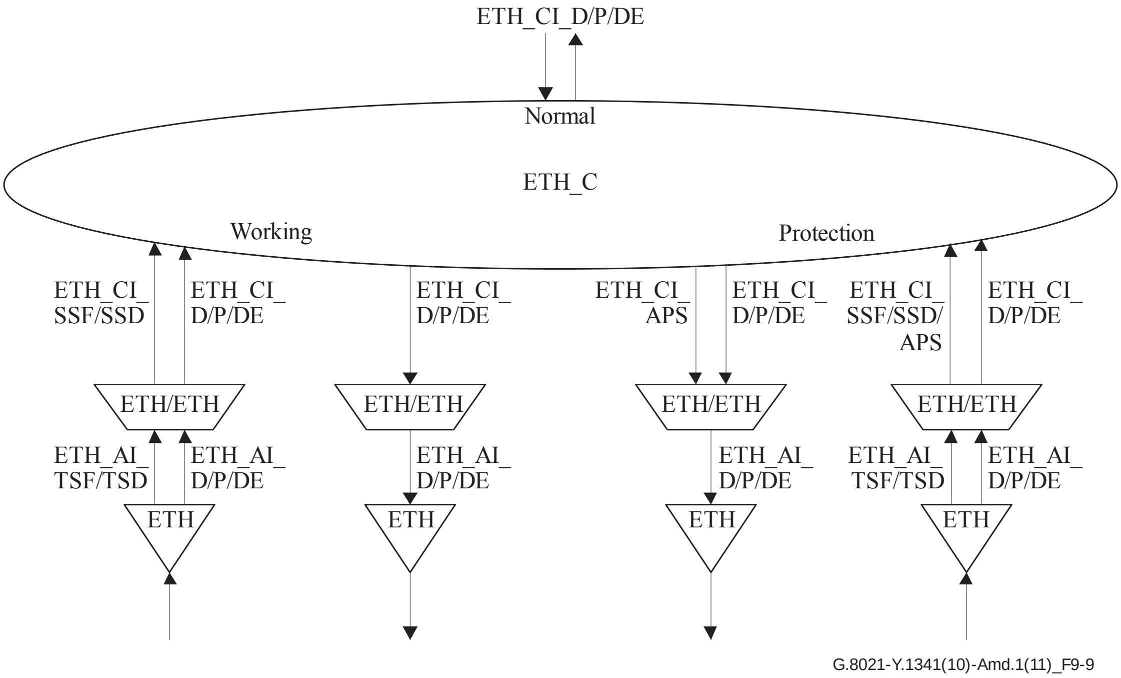


Figure 9-9 – SNC/S atomic functions

The protection functions at both ends operate the same way, by monitoring the working and protection subnetwork connections for defects, evaluating the system status taking into consideration the priorities of defect conditions and of external switch requests, and switching the appropriate subnetwork flow point (i.e., working or protection) to the protected (sub)network flow point.

The signal flows associated with the ETH_C SNC protection process are described with reference to Figure 9-10. The protection process receives control parameters and external switch requests at the MP reference point. The report of status information at the MP reference point is for further study.

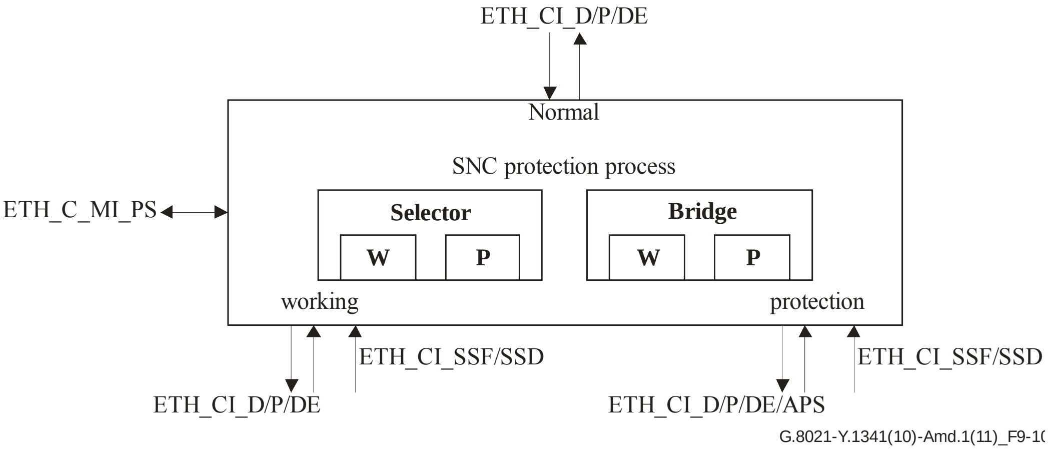


Figure 9-10 – SNC/S Protection process

Source direction:

For a 1+1 architecture, the CI coming from the normal (protected) ETH_FP is bridged permanently to both the working and protection ETH_FP.

For a 1:1 architecture, the CI coming from the normal (protected) ETH_FP is switched to either the working or the protection ETH_FP. A switch-over from working to protection ETH_FP or vice versa is initiated by the switch initiation criteria defined below.

Sink direction:

For a 1+1 or 1:1 architecture, the CI coming from either the working or protection ETH_FP is switched to the normal (protected) ETH_FP. A switch-over from working to protection ETH_FP or vice versa is initiated by the switch initiation criteria defined below.

Switch initiation criteria:

Automatic protection switching is based on the defect conditions of the working and protection (sub)network connections, for SNC/S protection server signal fail (SSF) and server signal degrade (SSD).

In order to allow interworking between nested protection schemes, a hold-off timer is provided. The hold-off timer delays switch initiation, in case of signal fail, in order to allow a nested protection to react and clear the fault condition. The hold-off timer is started by the activation of signal fail and runs for the hold-off time. Protection switching is only initiated if signal fail is still present at the end of the hold-off time. The hold-off time shall be provisionable between 0 and 10 s in steps of 100 ms; this is defined in clause 11.12 of [ITU-T G.8031].

Protection switching can also be initiated by external switch commands received via the MP or a request from the far end via the received ETH_CI_APS. Depending on the mode of operation, internal states (e.g., wait-to-restore) may also affect a switch-over.

See the switching algorithm described in [ITU-T G.8031].

Switching time:

Refer to [ITU-T G.8031].

Switch restoration:

In the revertive mode of operation, the protected signal shall be switched back from the protection (sub)network connection to the working (sub)network connection when the working (sub)network connection has recovered from the fault.

To prevent frequent operation of the protection switch due to an intermittent fault, a failed working (sub)network connection must become fault-free for a certain period of time before it is used again. This period, called the wait-to-restore (WTR) period, should be of the order of 5-12 minutes and should be capable of being set. The WTR is defined in clause 11.13 of [ITU-T G.8031].

In the non-revertive mode of operation no switch back to the working (sub)network connection is performed when it has recovered from the fault.

Configuration:

The following configuration parameters are defined in [ITU-T G.8031]:

ETH_C_MI_PS_WorkingPortId configures the working port.

ETH_C_MI_PS_ProtectionPortId configures the protection port.

ETH_C_MI_PS_ProtType configures the protection type.

ETH_C_MI_PS_OperType configures to be in revertive mode.

ETH_C_MI_PS_HoTime configures the hold off timer.

ETH_C_MI_PS_WTR configures the wait-to-restore timer.

ETH_C_MI_PS_ExtCMD configures the protection group command.

Defects:

The function detects dFOP-PM, dFOP-CM, ~~and~~ dFOP-NR and dFOP-TO defects in case the APS protocol is used.

Consequent actions None.

Defect correlations cFOP-TO ← dFOP-TO and (not dFOP-CM)

18) Clause 9.1.3

Revise clause 9.1.3 with respect to dFOP-TO as follows:

9.1.3 Ring protection control process

Ring protection with inherent, sub-layer, or test trail monitoring is supported.

Figure 9-11 shows a subset of the atomic functions involved, and the signal flows associated with the ring protection control process. This is only an overview of the Ethernet ring protection control process as specified in [ITU-T G.8032]. The ETH_FT_Sk provides the TSF protection switching criterion via the ETH/ETH_A_Sk function (SSF). [ITU-T G.8032] specifies the requirements, options and the ring protection protocol supported by the ring protection control process.

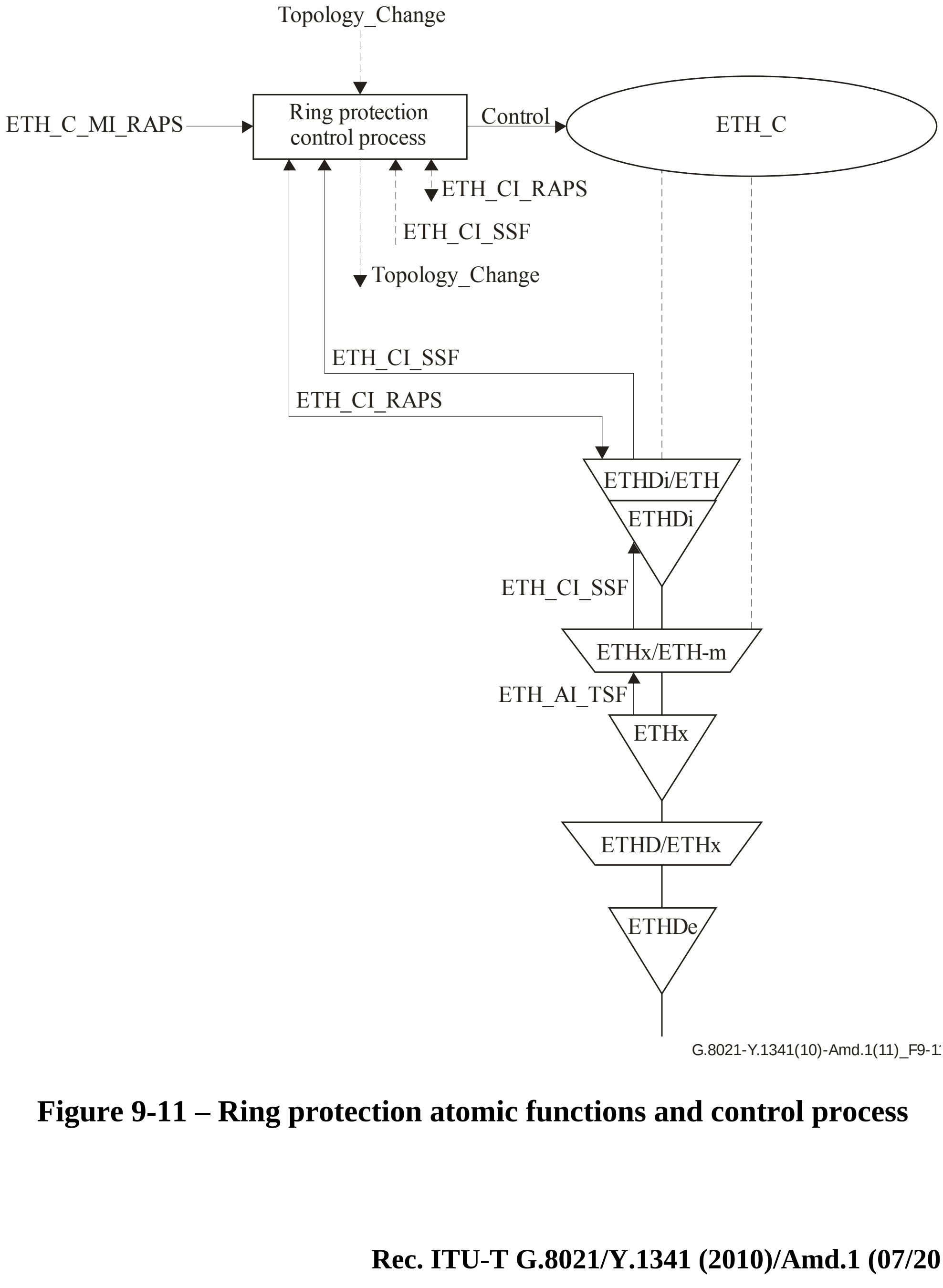


Figure 9-11 – Ring protection atomic functions and control process

Configuration:

The following configuration parameters are defined in [ITU-T G.8032]:

ETH_C_MI_RAPS_RPL_Owner_Node configures the node type.

ETH_C_MI_RAPS_RPL_Neighbour_Node configures the adjacency of a node to the RPL Owner.

ETH_C_MI_RAPS_Propagate_TC[1…M] configures the flush logic of an interconnection node.

ETH_C_MI_RAPS_Compatible_Version configures the Backward compatibility logic.

ETH_C_MI_RAPS_Revertive configures the revertive mode.

ETH_C_MI_RAPS_Sub_Ring_Without_Virtual_Channel configures the sub-ring type.

ETH_C_MI_RAPS_HoTime configures the hold off timer.

ETH_C_MI_RAPS_WTR configures the wait-to-restore timer.

ETH_C_MI_RAPS_GuardTime configures the guard timer.

ETH_C_MI_RAPS_ExtCMD configures the protection command.

Defects:

The function detects dFOP-PM and dFOP-TO in case the R-APS protocol is used.

Consequent actions None.

Defect correlations

cFOP-PM ← dFOP-PM

cFOP-TO ← dFOP-TO

19) Clause 9.2.1.1

Revise clause 9.2.1.1 with respect to DM as shown:

9.2.1.1 ETHx Flow Termination source function (ETHx_FT_So)

Symbol

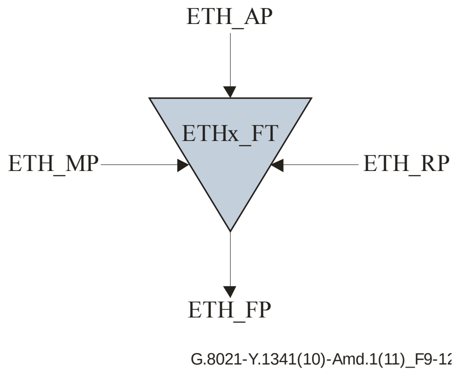


Figure 9-12 – ETHx_FT_So symbol

Interfaces

Table 9-2 – ETHx_FT_So interfaces

Inputs	Outputs
ETH_AP: ETH_AI_D ETH_AI_P ETH_AI_DE ETH_RP: ETH_RI_CC_RxFCl ETH_RI_CC_TxFCf ETH_RI_CC_RDI ETH_RI_CC_Blk ETH~~x_FT_So~~_RI_DMM(~~D~~OAM,P,DE) ETH~~x_FT_So~~_RI_DMR(rSA,Tx, TimeStamp~~D,P,DE)~~, RxTimeStampf,TxTimeStampb,RxTimeb) ETH_RI_SLM(OAM,P,DE,TxFCb) ETH_RI_SLR(MEP_ID,Test ID,TxFCf, TxFCb) ETHx_FT_So_MP: ETHx_FT_So_MI_MEL ETHx_FT_So_MI_MEP_MAC ETHx_FT_So_MI_CC_Enable ETHx_FT_So_MI_LM_Enable ETHx_FT_So_MI_MEG_ID ETHx_FT_So_MI_MEP_ID ETHx_FT_So_MI_CC_Period ETHx_FT_So_MI_CC_Pri ETHx_FT_So_MI_DM_Enable ETHx_FT_So_MI_DM_MAC_DA ETHx_FT_So_MI_DM_Test_ID ETHx_FT_So_MI_DM_Length ETHx_FT_So_MI_DM_Period ETHx_FT_So_MI_DM_Pri ETHx_FT_So_MI_1DM_Enable ETHx_FT_So_MI_1DM_MAC_DA ETHx_FT_So_MI_1DM_Test_ID ETHx_FT_So_MI_1DM_Length ETHx_FT_So_MI_1DM_Period ETHx_FT_So_MI_1DM_Pri ETHx_FT_So_MI_SL_Enable ETHx_FT_So_MI_SL_MAC_DA ETHx_FT_So_MI_SL_Test_ID ETHx_FT_So_MI_SL_Length ETHx_FT_So_MI_SL_Period ETHx_FT_So_MI_SL_Pri	ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_RP: ETH_RI_DM_Result(B_FD,F_FD,N_FD) ETH_RI_SL_Result(N_TF,N_LF,F_TF,F_LF)

Processes

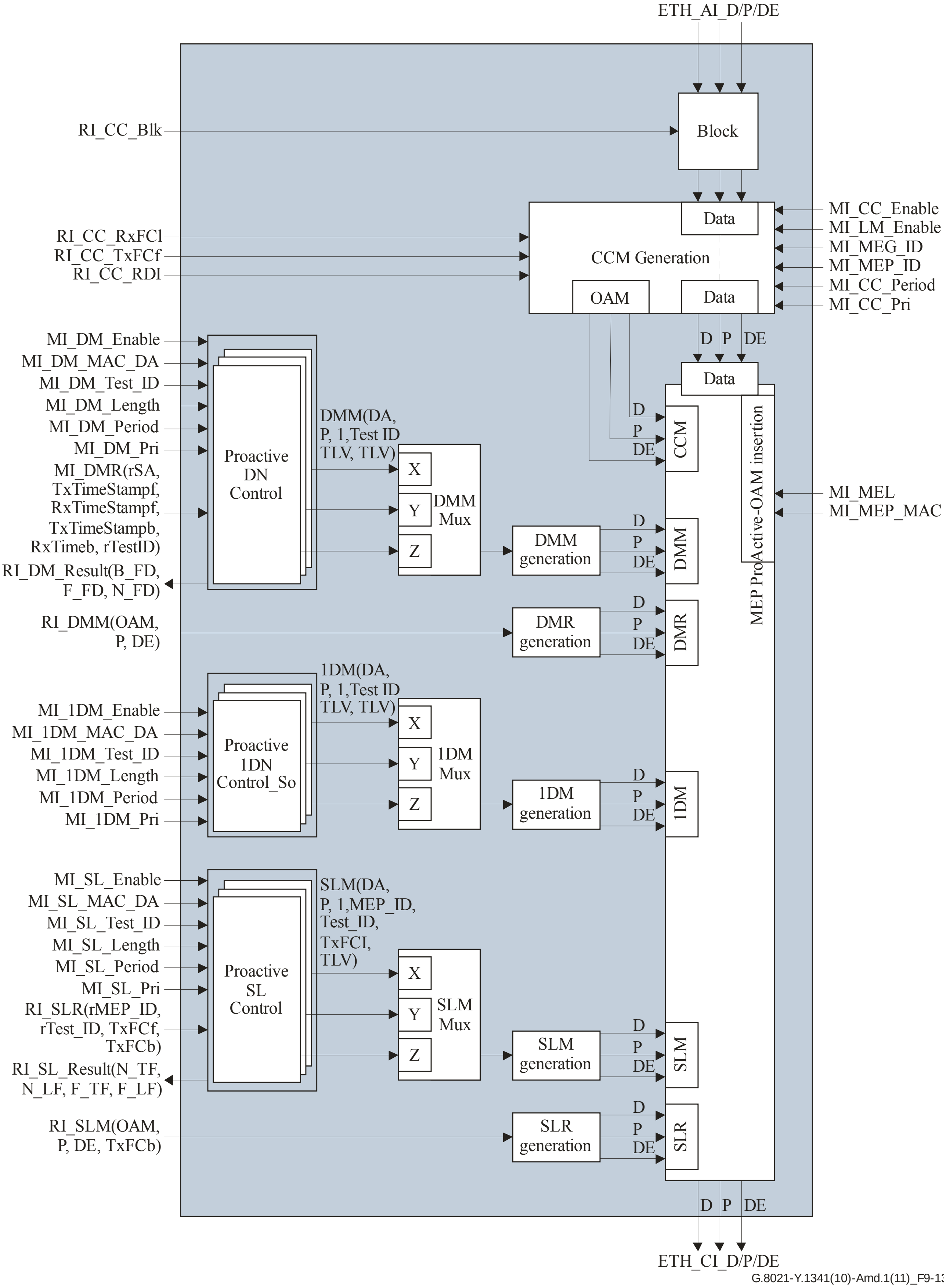


Figure 9-13 – ETHx_FT_So process

MEP ProActive-OAM Insertion process:

This process inserts the OAM traffic units in the stream of ETH_CI, sets the MEL field to MI_MEL and sets the SA field to MI_MEP_MAC.

If the DA of the OAM Traffic Unit is a Class 1 Multicast DA, the OAM insertion process updates the DA to reflect the correct MEL.

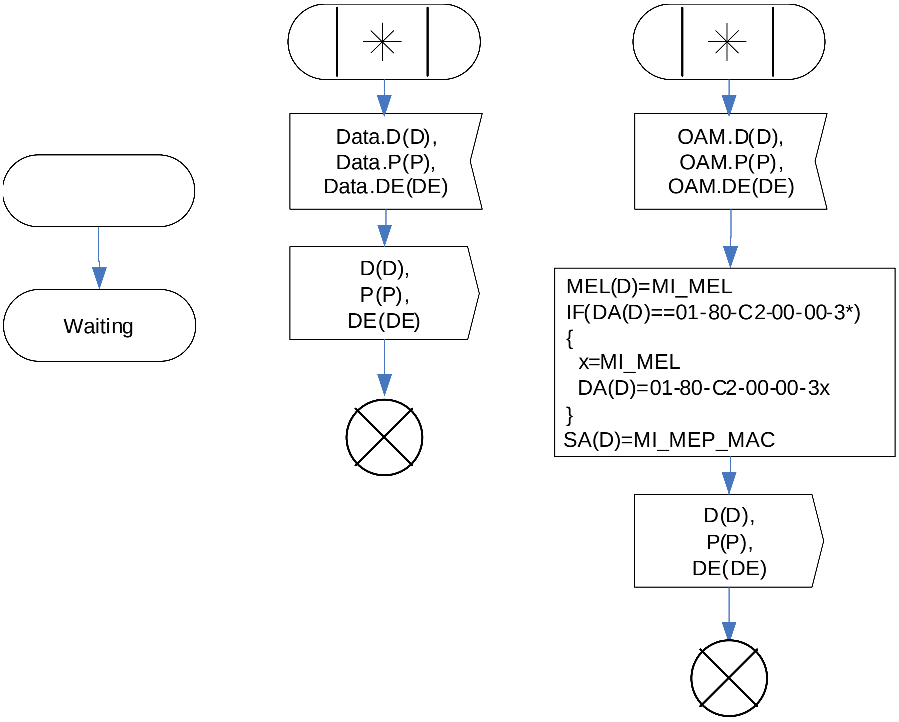


Figure 9-14 – OAM MEP Insertion behaviour

CCM Generation process:

This process is defined in clause 8.1.7, where the CC protocol is defined. Clause 8.1.7.2 defines the CCM Generation process.

Block process:

When RI_CC_Blk is raised, the Block process will discard all ETH_CI information it receives. If RI_CC_Blk is cleared, the received ETH_CI information will be passed to the output port.

Proactive DM Control:

This process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.2 defines the DM Control process.

DMM Generation:

This process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.3 defines the DMM Generation process.

DMR Generation:

This process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.5 defines the DMR Generation process.

DMM Mux:

The DMM Mux process interleaves the signal sets DMM(DA,P,1,Test ID TLV, TLV) from the input ports (X, Y, Z).

Proactive1DM Control_So:

This process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.2 defines the 1DM Control_So Process.

1DM Generation:

This process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.3 defines the 1DM Generation process.

1DM Mux:

The 1DM Mux process interleaves the signal sets 1DM(DA,P,1,Test ID TLV, TLV) from the input ports (X, Y, Z).

Proactive SL Control:

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.2 defines the SL Control process.

SLM Generation:

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.3 defines the SLM generation process.

SLR Generation:

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.5 defines the SLR Generation process.

SLM Mux:

The SLM Mux process interleaves the signal sets SLM(DA,P,MEP_ID,Test_ID,TxFCl,TLV) from the input ports (X, Y, Z).

Defects	None.
Consequent actions	None.
Defect correlations	None.
Performance monitoring	None.

20) Clause 9.2.1.2

Revise clause 9.2.1.2 for DM as shown:

9.2.1.2 ETHx Flow Termination sink function (ETHx_FT_Sk)

The ETHx_FT_Sk Process diagram is shown in Figure 9-15.

Symbol

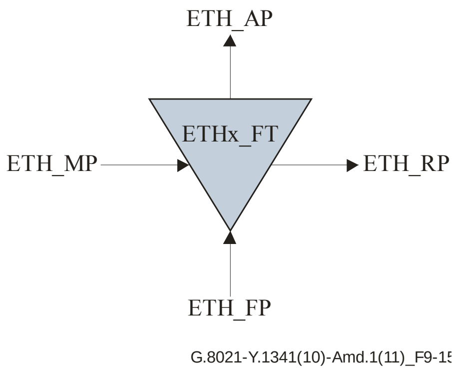


Figure 9-15 – ETHx_FT_Sk symbol

Interfaces

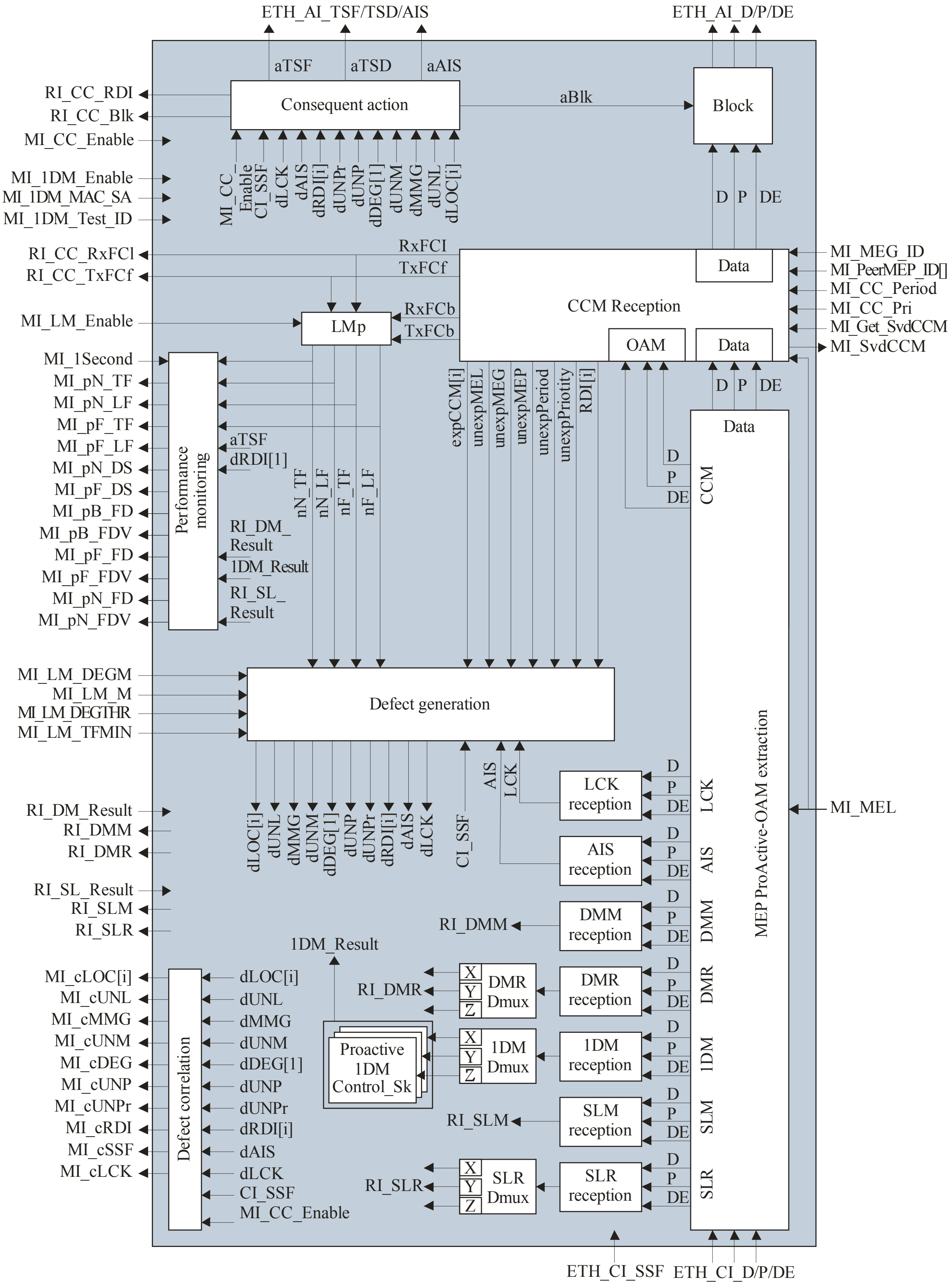
Table 9-3 – ETHx_FT_Sk interfaces

Inputs	Outputs
ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_RP: ETH_RI_DM_Result(B_FD,F_FD,N_FD) ETH_RI_SL_Result(N_TF,N_LF,F_TF,F_LF) ETHx_FT_Sk_MP: ETHx_FT_Sk_MI_CC_Enable ETHx_FT_Sk_MI_LM_Enable ETHx_FT_Sk_MI_1Second ETHx_FT_Sk_MI_LM_DEGM ETHx_FT_Sk_MI_LM_M ETHx_FT_Sk_MI_LM_DEGTHR ETHx_FT_Sk_MI_LM_TFMIN ETHx_FT_Sk_MI_MEL ETHx_FT_Sk_MI_MEG_ID ETHx_FT_Sk_MI_PeerMEP_ID[i] ETHx_FT_Sk_MI_CC_Period ETHx_FT_Sk_MI_CC_Pri ETHx_FT_Sk_MI_GetSvdCCM ETHx_FT_Sk_MI_1DM_Enable ETHx_FT_Sk_MI_1DM_MAC_SA ETHx_FT_Sk_MI_1DM_Test_ID	ETH_AP: ETH_AI_D ETH_AI_P ETH_AI_DE ETH_AI_TSF ETH_AI_TSD ETH_AI_AIS ETH_RP: ETH_RI_CC_RxFCl ETH_RI_CC_TxFCf ETH_RI_CC_RDI ETH_RI_CC_Blk ETH_RI_DMM(OAM,P,DE) ETH_RI_DMR(SA,TxTimeStampf,RxTimeStampf,TxTimeStampb,LocalTime) ETH_RI_SLM(OAM,P,DE,TxFCb) ETH_RI_SLR(MEP_ID,Test ID,TxFCf,TxFCb) ETHx_FT_Sk_MP: ETHx_FT_Sk_MI_cLOC[i] ETHx_FT_Sk_MI_cUNL ETHx_FT_Sk_MI_cMMG ETHx_FT_Sk_MI_cUNM ETHx_FT_Sk_MI_cDEG ETHx_FT_Sk_MI_cUNP ETHx_FT_Sk_MI_cUNPr ETHx_FT_Sk_MI_cRDI ETHx_FT_Sk_MI_cSSF ETHx_FT_Sk_MI_cLCK ETHx_FT_Sk_MI_pN_TF ETHx_FT_Sk_MI_pN_LF ETHx_FT_Sk_MI_pF_TF

Table 9-3 – ETHx_FT_Sk interfaces

Inputs	Outputs
	ETHx_FT_Sk_MI_pF_LF ETHx_FT_Sk_MI_pF_DS ETHx_FT_Sk_MI_pN_DS ETHx_FT_Sk_MI_pB_FD ETHx_FT_Sk_MI_pB_FDV ETHx_FT_Sk_MI_pF_FD ETHx_FT_Sk_MI_pF_FDV ETHx_FT_Sk_MI_pN_FD ETHx_FT_Sk_MI_pN_FDV ETHx_FT_Sk_MI_SvdCCM

Processes



G.8021-Y.1341(10)-Amd.1(11)_F9-16

Figure 9-16 – ETHx_FT_Sk process

MEP Proactive-OAM Extraction process:

The MEP Proactive-OAM Extraction process extracts OAM traffic units that are processed in the ETHx_FT_Sk process from the stream of traffic units according to the following pseudo code:

```
if (TYPE=<ETHOAM>) and (MEL=MI_MEL) then
  switch(OPC) {
  case <CCM>: extract ETH-CCM OAM traffic unit and forward to CCM Port
  case <AIS>: extract ETH-AIS OAM traffic unit and forward to AIS Port
  case <LCK>: extract ETH-LCK OAM traffic unit and forward to LCK Port
  case <DMM>: extract ETH-DMM OAM traffic unit and forward to DMM Port
  case <DMR>: extract ETH-DMR OAM traffic unit and forward to DMR Port
  case <1DM>: extract ETH-1DM OAM traffic unit and forward to 1DM Port
  case <SLM>: extract ETH-SLM OAM traffic unit and forward to SLM port
  case <SLR>: extract ETH-SLR OAM traffic unit and forward to SLR port

else if (TYPE=<ETH0AM>) and (MEL<MI_MEL) and (OPC=CCM) then
  extract ETH-CCM OAM traffic unit and forward to CCM Port
else
  forward ETH CI traffic unit to Data Port
end if
```

ETH_AIS Reception process:

This process generates the AIS event upon the receipt of the AIS Traffic Unit from the OAM MEP Extraction process.

ETH_LCK Reception process:

This process generates the LCK event upon the receipt of the LCK Traffic Unit from the OAM MEP Extraction Process.

DMM Reception:

This Process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.4 defines the DMM Reception process.

DMR Reception:

This Process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.6 defines the DMR Reception process.

DMR Demux:

The DMR Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

1DM Reception:

This Process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.4 defines the 1DM Reception process.

1DM Demux:

The 1DM Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

Proactive 1DM Control_Sk:

This Process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.5 defines the 1DM Control_Sk process.

SLM Reception:

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.4 defines the SLM Reception process.

SLR Reception:

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.6 defines the SLR Reception process.

SLR Demux:

The SLR Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

Block process:

When aBlk is raised, the Block process will discard all ETH_CI information it receives. If aBLK is cleared, the received ETH_CI information will be passed to the output port.

LMp process:

This process is defined in clause 8.1.7.4.

Defect Generation process:

This process detects and clears the defects (dLOC[i], dUNL, dMMG, dUNM, dDEG, dUNP, dUNPr, dRDI[i], dAIS, dLCK) as defined in clause 6, where [i] = maintenance entity.

CCM Reception process:

This process is defined in clause 8.1.7.3.

Defects

This function detects dLOC[i], dUNL, dMMG, dUNM, dDEG, dUNP, dUNPr, dRDI[i], dAIS, dLCK.

Consequent actions

aBLK ← (dUNL or dMMG or dUNM)

Note that dUNP and dUNPr does not contribute to aBLK, because a mismatch of periodicity is not considered to be a security issue.

aTSF ← (dLOC[1..n] and MI_CC_Enable) or (dAIS and not(MI_CC_Enable)) or (dLCK and not(MI_CC_Enable)) or dUNL or dMMG or dUNM or CI_SSF

aTSD ← dDEG[1] and (not aTSF)

aAIS ← aTSF

aRDI ← aTSF

Defect correlations

cLOC[i] ← dLOC[i] and (not dAIS) and (not dLCK) and (not CI_SSF) and (MI_CC_Enable)

cUNL ← dUNL

cMMG ← dMMG

cUNM ← dUNM

cDEG[1] ← dDEG[1] and (not dAIS) and (not dLCK) and (not CI_SSF) and (not (dLOC[1..n] or dUNL or dMMG or dUNM)) and (MI_CC_Enable)) cUNP ← dUNP

cUNPr ← dUNPr

cRDI ← (dRDI[1..n]) and (MI_CC_Enable)

cSSF ← CI_SSF or dAIS

cLCK ← dLCK and (not dAIS)

Performance monitoring

pN_TF ← N_TF

pN_LF ← N_LF

pF_TF ← F_TF

pF_LF ← F_LF

pN_DS ← aTSF

pF_DS ← aRDI[1]

nB_FD ← B_FD

nB_FDV ← B_FDV

nF_FD ← F_FD

nF_FDV ← F_FDV

nN_FD ← N_FD

nN_FDV ← N_FDV

NOTE – A detail calculation formula for FDV is for further study.

21) Clause 9.2.2.1

Update Figure 9-18 in clause 9.2.2.1 for technical clarification, and the paragraph that follows it.

Processes

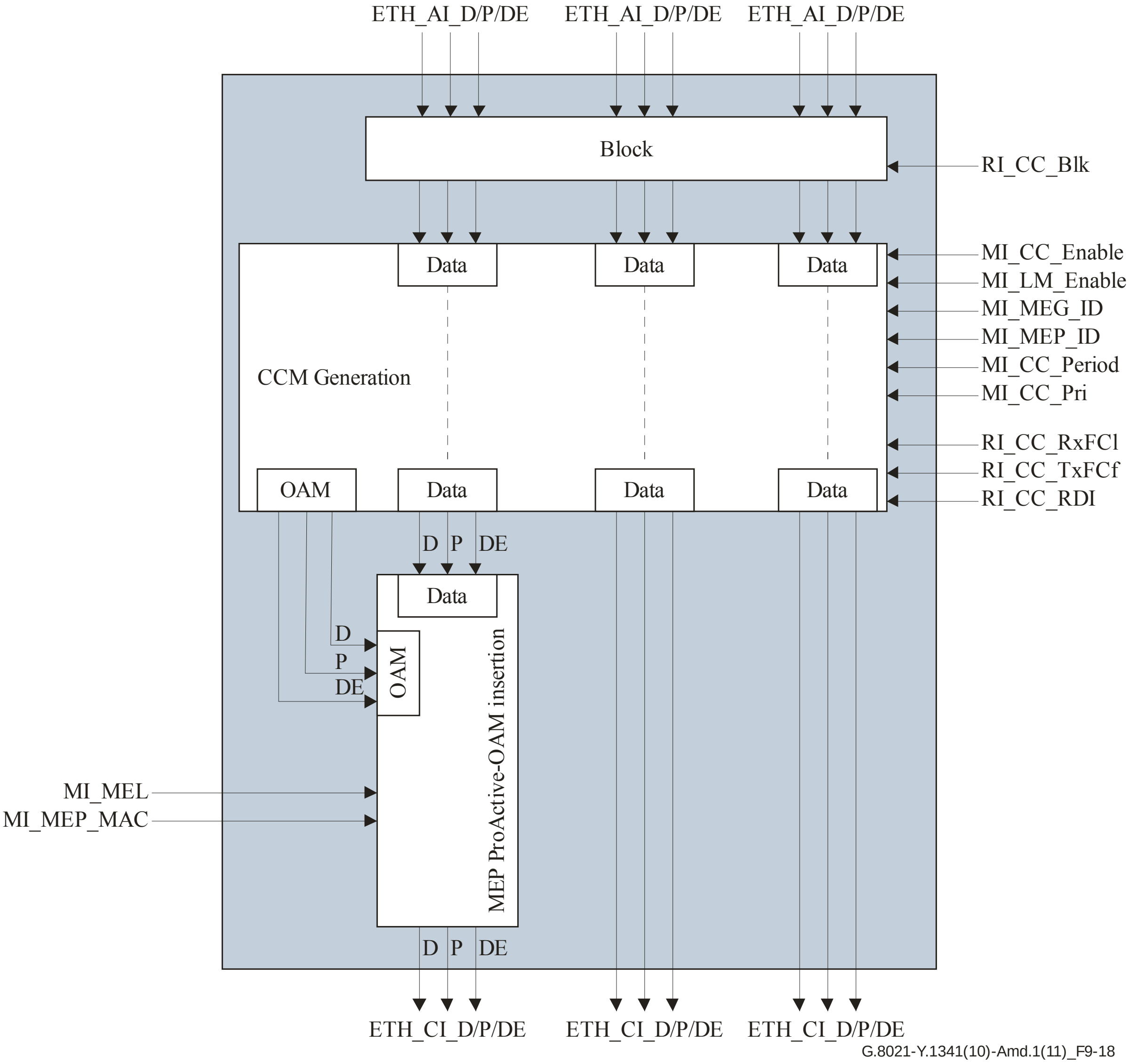


Figure 9-18 – ETHG_FT_So process

MEP ProActive-OAM Insertion process:

This process inserts the OAM traffic units in the stream of ETH_CI, sets the MEL field to MI_MEL and sets the SA field to MI_MEP_MAC. This process resides only in the lowest number in the contiguous range of ETH_FPs or a selected ETH_FP within the group of arbitrary ETH_FPs~~(CCM Generation Process as well)~~. The detail of the OAM Insertion behaviour is described in clause 9.2.1.1.

22) Clause 9.2.2.2

Update Figure 9-20 in clause 9.2.2.2 for technical clarification, and the paragraph that follows it.

Interfaces

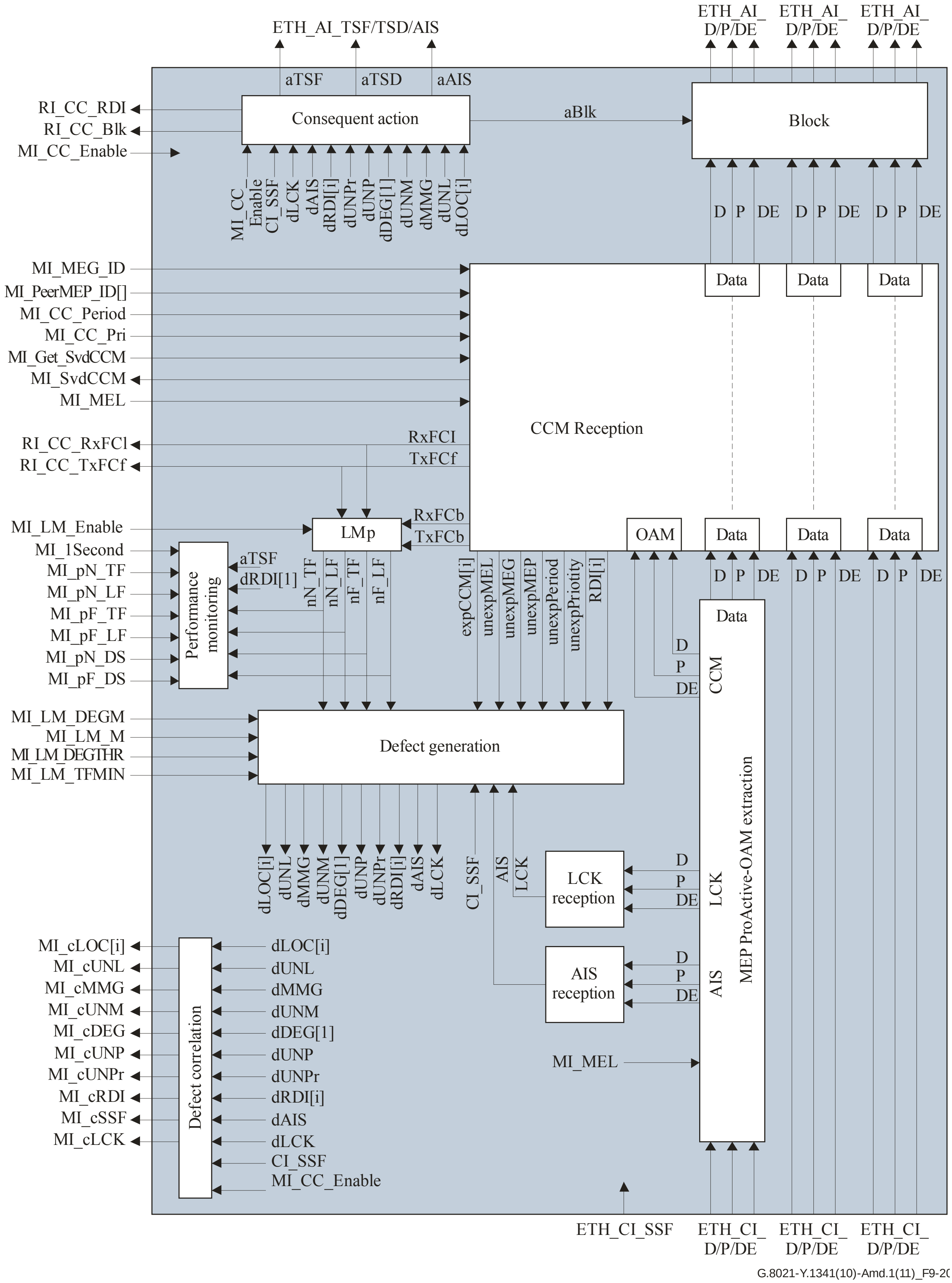


Figure 9-20 – ETHG_FT_Sk process

MEP Proactive-OAM Extraction process:

The MEP Proactive-OAM Extraction process extracts OAM traffic units that are processed in the ETHx_FT_Sk process from the stream of traffic units. This process resides only in the lowest number in the contiguous range of ETH_FPs or a selected ETH_FP within the group of arbitrary ETH_FPs (AIS Reception, LCK Reception, LMp, and Defect Generation ~~and CCM Reception~~ processes as well). The detail of this process is described in clause 9.2.1.2.

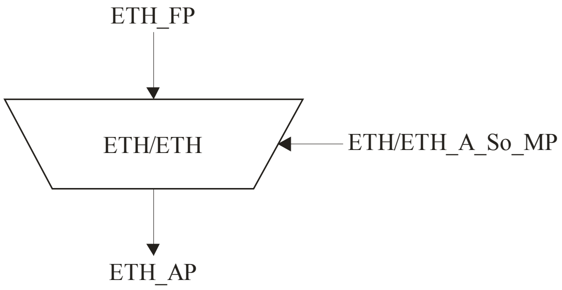
23) Clause 9.3.2.1

Revise clause 9.3.2.1 with respect to CSF as follows:

9.3.2.1 ETH to ETH adaptation source function (ETHx/ETH_A_So)

This function maps client ETH_CI traffic units into server ETH_AI traffic units.

Symbol



G.8021-Y1341(10)-Amd.1(11)_F9-21

Figure 9-21 – ETHx/ETH_A_So symbol

Interfaces

Table 9-6 – ETHx/ETH_A_So interfaces

Inputs	Outputs
ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_APS ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ETHx/ETH_A_So_MP: ETHx/ETH_A_So_MI_Active ETHx/ETH_A_So_MI_MEP_MAC ETHx/ETH_A_So_MI_Client_MEL ETHx/ETH_A_So_MI_LCK_Period ETHx/ETH_A_So_MI_LCK_Pri ETHx/ETH_A_So_MI_Admin_State ETHx/ETH_A_So_MI_MEL ETHx/ETH_A_So_MI_APS_Pri ETHx/ETH_A_So_MI_CSF_Enable ETHx/ETH_A_So_MI_CSFrdifdiEnable	ETH_AP: ETH_AI_D ETH_AI_P ETH_AI_DE

Processes

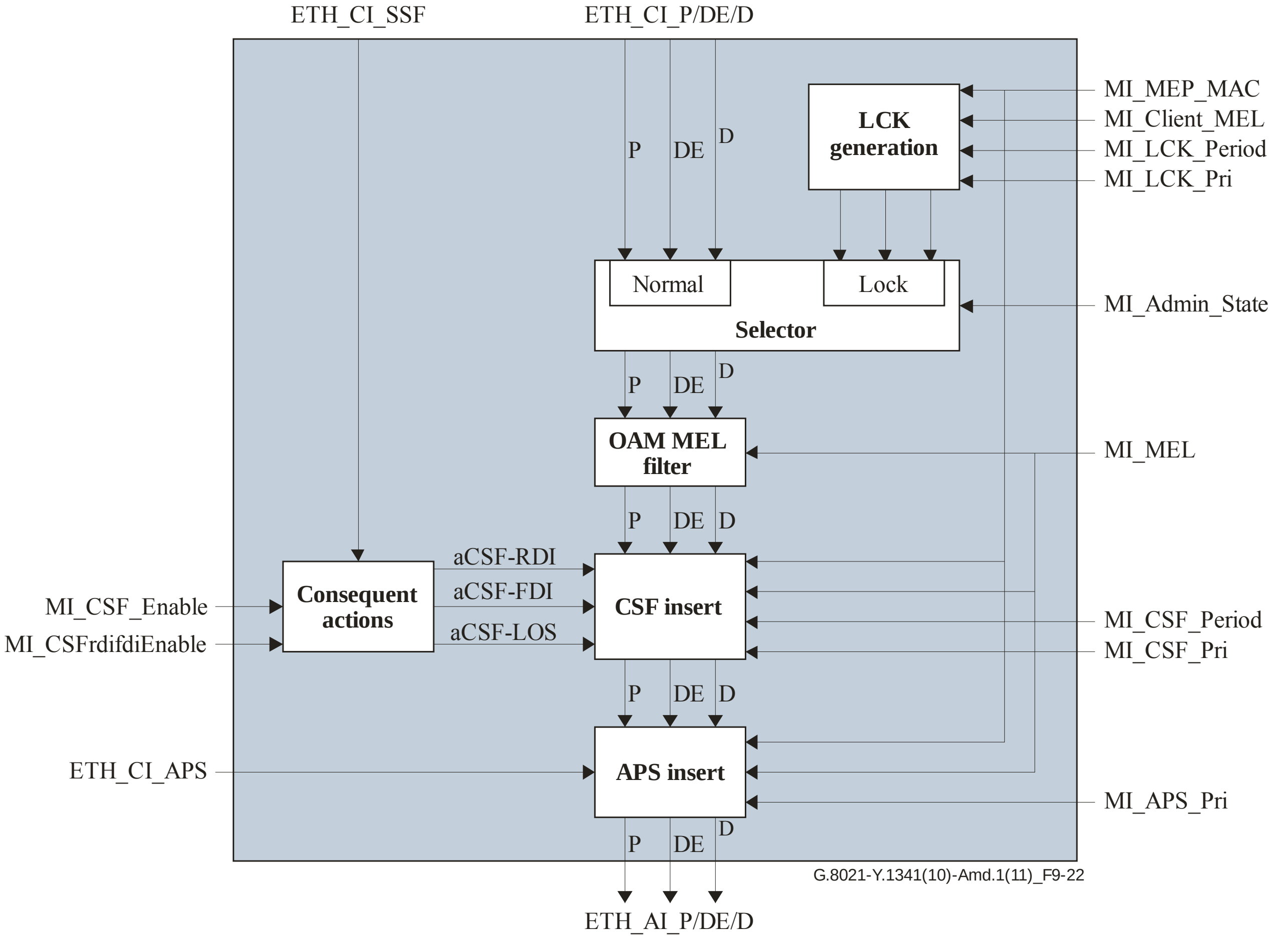


Figure 9-22 – ETHx/ETH_A_So process

LCK Generation process:

As defined in clause 8.1.2.

Selector process:

As defined in clause 8.1.3.

OAM MEL Filter process:

As defined in clause 8.1.1.

APS Insert process:

As defined in clause 8.1.5.

When this process is activated, LCK admin state shall be unlocked. See clause 7.5.2.2 of [ITU-T G.8010].

Defects None.

Consequent actions ~~None.~~

aCSF-LOS ← CI_SSF and MI_CSFEnable

aCSF-RDI ← CI_SSFrdi and MI_CSFrdifdiEnable and MI_CSFEnable

aCSF-FDI ← CI_SSFfdi and MI_CSFrdifdiEnable and MI_CSFEnable

Defect correlations None.

24) Clause 9.3.2.2

Revise clause 9.3.2.2 with respect to CSF as follows:

9.3.2.2 ETH to ETH adaptation sink function (ETHx/ETH_A_Sk)

This function retrieves client ETH_CI traffic units from server ETH_AI traffic units.

Symbol

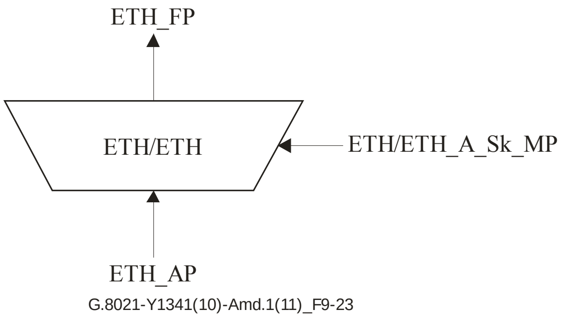


Figure 9-23 – ETHx/ETH_A_Sk symbol

Interfaces

Table 9-7 – ETHx/ETH_A_Sk interfaces

Inputs	Outputs
ETH_AP: ETH_AI_D ETH_AI_P ETH_AI_DE ETH_AI_TSF ETH_AI_TSD ETH_AI_AIS ETHx/ETH_A_Sk_MP: ETHx/ETH_A_Sk_MI_Active ETHx/ETH_A_Sk_MI_MEP_MAC ETHx/ETH_A_Sk_MI_Client_MEL ETHx/ETH_A_Sk_MI_LCK_Period ETHx/ETH_A_Sk_MI_LCK_Pri ETHx/ETH_A_Sk_MI_Admin_State ETHx/ETH_A_Sk_MI_AIS_Period ETHx/ETH_A_Sk_MI_AIS_Pri ETHx/ETH_A_Sk_MI_MEL ETHx/ETH_A_Sk_MI_CSF_Reported ETHx/ETH_A_Sk_MI_CSFrdifdiEnable	ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_APS ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ETH_CI_SSD ETHx/ETH_A_Sk_MP: ETHx/ETH_A_Sk_MI_cCSF

Processes

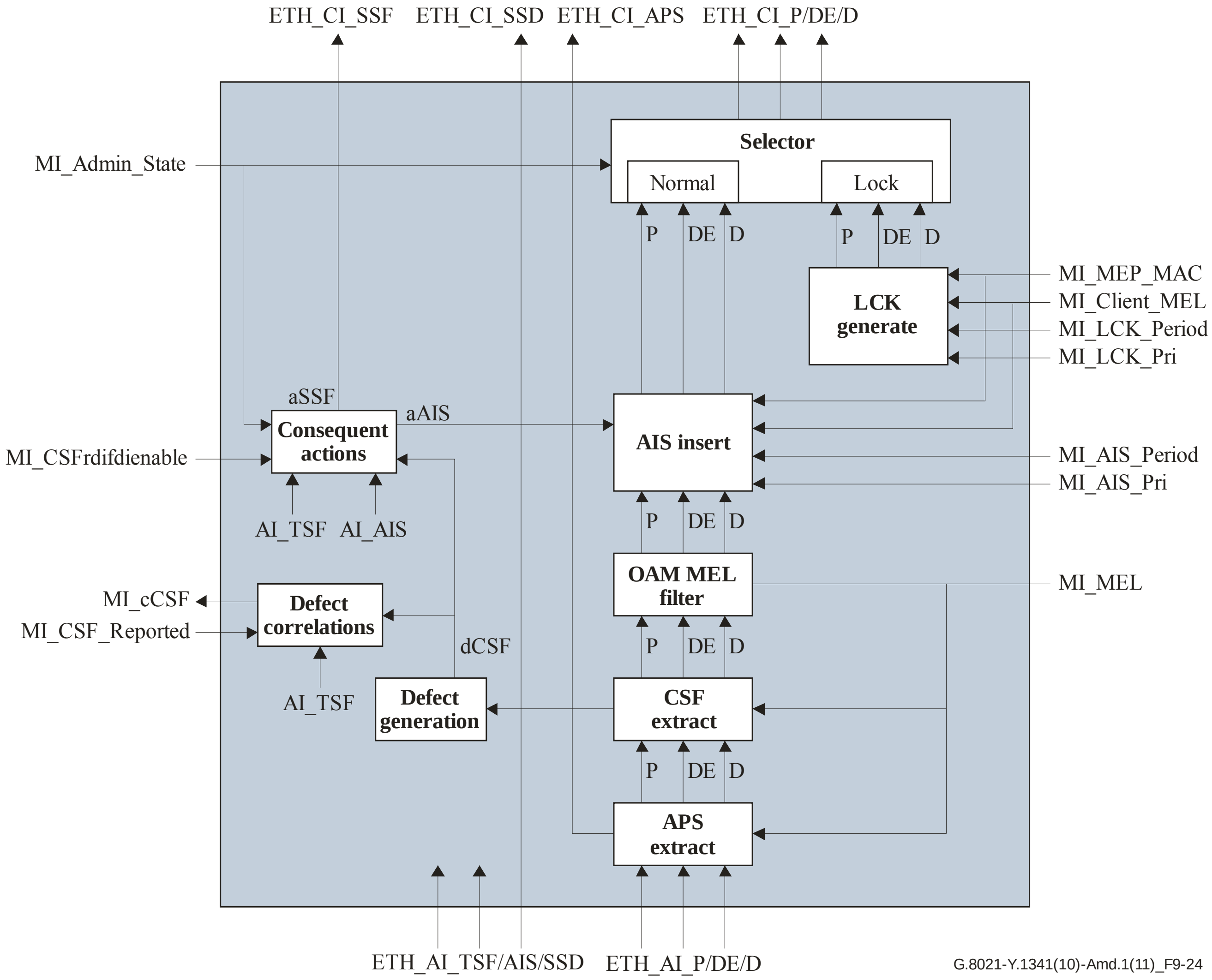


Figure 9-24 – ETHx/ETH_A_Sk process

APS Extract process:

As defined in clause 8.1.6.

OAM MEL Filter process:

As defined in clause 8.1.1.

AIS Insert process:

As defined in clause 8.1.4.

LCK Generation process:

As defined in clause 8.1.2.

Selector process:

As defined in clause 8.1.3.

Defects~~ None.~~

dCSF-LOS – See clause 6.1.5.4.

dCSF-RDI – See clause 6.1.5.4.

dCSF-FDI – See clause 6.1.5.4.

Consequent actions

aSSF ← (AI_TSF or dCSF-LOS) and (not MI_Admin_State == Locked)

aSSFrdi ← dCSF-RDI and MI_CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and MI_CSFrdifdiEnable

aAIS ← AI_AIS

Defect correlations ~~None.~~

cCSF ← (dCSF-LOS or dCSF-RDI or dCSF-FDI) and (not AI_TSF) and MI_CSF_Reported

Performance monitoring None.

25) Clause 9.3.3.2

In clause 9.3.3.2, revise the description for the VID Demux process as follows:

9.3.3.2 ETH to ETH multiplexing adaptation sink function (ETHx/ETH-m_A_Sk)

• • •

VID Demux process :

The VID Demux Process deinterleaves the incoming signal set (DE, P, D) to the different ports (X, Y, Z in Figure 9-27). The VID signal determines the port to be selected, based on the MI_Vlan_Config input parameter.

The MI_Vlan_Config parameter specifies the possible VID values for the ports to be used. If there is no port assigned to a specific VID value, and this VID value is used, the VID Demux process will filter the incoming signal set.

Disabling the Ingress VID Filtering is modelled by setting MI_Vlan_Config [1…4094]. Refer to Appendix VIII.

• • •

26) Clause 9.4.1

Revise clause 9.4.1 with respect to DM and other processes, as follows:

9.4.1 ETH Diagnostic Flow Termination functions for MEPs (ETHDe_FT)

The bidirectional ETHDe Flow Termination (ETHDe_FT) function is performed by a co-located pair of ETHDe flow termination source (ETHDe_FT_So) and sink (ETHDe_FT_Sk) functions.

9.4.1.1 ETH Diagnostic Flow Termination Source function for MEPs (ETHDe_FT_So)

The ETHDe_FT_So process diagram is shown in Figure 9-41.

Symbol

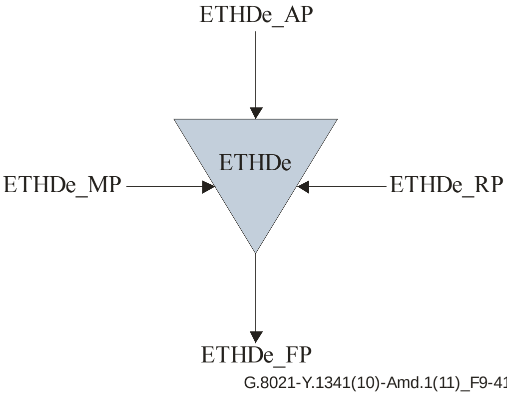


Figure 9-41 – ETHDe_FT_So symbol

Interfaces

Table 9-14 – ETHDe_FT_So interfaces

Inputs	Outputs
ETH_AP: ETH_AI_D ETH_AI_P ETH_AI_DE	ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE
ETH_RP: ETH_RI_LMM(D,P,DE) ETH_RI_LMR(TxFCf,RxFCf,TxFCb,RxFCl) ETH_RI_LBM(D,P,DE) ETH_RI_LBR(SA,rTLV,TID) ETH_RI_DMM(D,P,DE) ETH_RI_DMR(rSA,TxTimeStampf,RxTimeStampf, TxTimeStampb,RxTimeb, rTestID) ETH_RI_LTM(D,P,DE) ETH_RI_LTR(SA,TTL,TID,TLV) ETH_RI_SLM(OAM,P,DE,TxFCb) ETH_RI_SLR(rMEP_ID,rTest_ID,TxFCf,TxFCb)	
ETHDe_FT_So_MP: ETHDe_FT_So_MI_LM_Start(DA,P,Period) ETHDe_FT_So_MI_LM_Terminate ETHDe_FT_So_MI_LB_Discover(P) ETHDe_FT_So_MI_LB_Series(DA,DE,P,N, Length, Period) ETHDe_FT_So_MI_LB_Test (DA,DE,P,Pattern, Length, Period) ETHDe_FT_So_MI_LB_Test_Terminate ETHDe_FT_So_MI_DM_Start(DA,P,TestID,Length, Period) ETHDe_FT_So_MI_DM_Terminate ETHDe_FT_So_MI_1DM_Start(DA,P, TestID,Length,Period) ETHDe_FT_So_MI_1DM_Terminate ETHDe_FT_So_MI_TST(DA,DE,P,Pattern, Length, Period) ETHDe_FT_So_MI_TST_Terminate ETHDe_FT_So_MI_LT(TA,TTL,P) ETHDe_FT_So_MI_MEP_MAC ETHDe_FT_So_MI_MEL ETHDe_FT_So_MI_MEP_ID ETHDe_FT_So_MI_LM_Pri ETHDe_FT_So_MI_SL_Start(DA,P, TestID,Length,Period) ETHDe_FT_So_MI_SL_Terminate	ETHDe_FT_So_MP: ETHDe_FT_So_MI_LM_Result(N_TF, N_LF, F_TF, F_LF) ETHDe_FT_So_MI_LB_Discover_Result(MACs) ETHDe_FT_So_MI_DM_Result(count,B_FD[],F_FD[], N_FD[]) ETHDe_FT_So_MI_LB_Series_Result(REC,ERR,OO) ETHDe_FT_So_MI_LB_Test_Result (Sent, REC, CRC, BER, OO) ETHDe_FT_So_MI_TST_Result(Sent) ETHDe_FT_So_MI_LT_Results(Results) ETHDe_FT_So_MI_SL_Result(N_TF,N_LF,F_TF,F_LF)

Processes

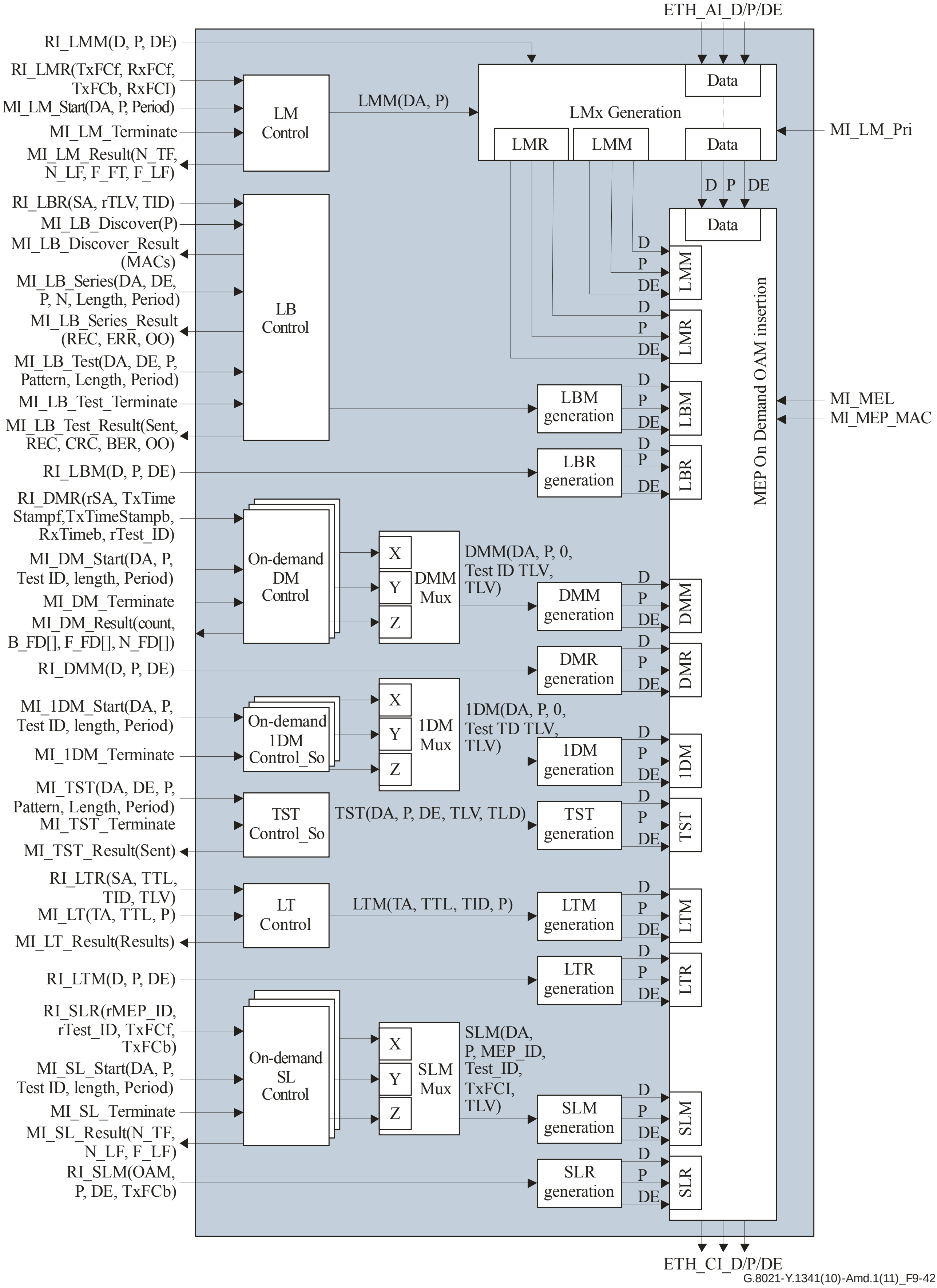


Figure 9-42 – ETHDe_FT_So process

MEP On Demand-OAM Insertion process:

The MEP On Demand OAM Insertion process inserts OAM traffic units that are generated in the ETHDe_FT_So process into the stream of traffic units.

For all ETH_CI_D received on any but the data input port, the SA field is overwritten with the MI_MEP_MAC value. In the M_SDU field, the MEL field is overwritten with the MI_MEL value.

If the DA of the OAM traffic unit is a Class1 or Class 2 Multicast DA the OAM insertion process updates the DA to reflect the right MEL.

This ensures that every generated OAM field has the correct SA, DA and MEL.

LB Control:

This Process is defined in clause 8.1.8 where the LB protocol is defined. Clause 8.1.8.2 defines the LB Control Process.

LBM Generation:

This Process is defined in clause 8.1.8 where the LB protocol is defined. Clause 8.1.8.3 defines the LBM Generation Process.

LBR Generation:

This Process is defined in clause 8.1.8 where the LB protocol is defined. Clause 8.1.8.6 defines the LBR Generation Process.

LM Control:

This Process is defined in clause 8.1.9 where the LM protocol is defined. Clause 8.1.9.2 defines the LM Control Process.

LMx Generation:

This Process is defined in clause 8.1.9 where the LM protocol is defined. Clause 8.1.9.3 defines the LMx Generation Process.

On-demand DM Control:

This Process is defined in clause 8.1.10 where the DM protocol is defined. Clause 8.1.10.2 defines the DM Control Process.

DMM Generation:

This Process is defined in clause 8.1.10 where the DM protocol is defined. Clause 8.1.10.3 defines the DMM Generation Process.

DMM Mux:

The DMM Mux process interleaves the signal sets DMM(DA,P,0,Test ID TLV, TLV) from the input ports (X, Y, Z).

DMR Generation:

This Process is defined in clause 8.1.10 where the DM protocol is defined. Clause 8.1.10.5 defines the DMR Generation Process.

On-demand 1DM Control_So:

This Process is defined in clause 8.1.11 where the 1DM protocol is defined. Clause 8.1.11.2 defines the 1DM Control_So Process.

1DM Generation:

This Process is defined in clause 8.1.11 where the 1DM protocol is defined. Clause 8.1.11.3 defines the 1DM Generation Process.

1DM Mux:

The 1DM Mux process interleaves the signal sets 1DM(DA,P,0,Test ID TLV, TLV) from the input ports (X, Y, Z).

TST Control_So:

This Process is defined in clause 8.1.12 where the TST protocol is defined. Clause 8.1.12.2 defines the TST Control Process.

TST Generation:

This Process is defined in clause 8.1.12 where the TST protocol is defined. Clause 8.1.12.3 defines the TST Generation Process.

LT Control:

This Process is defined in clause 8.1.13 where the LT protocol is defined. Clause 8.1.13.2 defines the LT Control Process.

LTM Generation:

This Process is defined in clause 8.1.13 where the LT protocol is defined. Clause 8.1.13.3 defines the LTM Generation Process.

LTR Generation:

This Process is defined in clause 8.1.13 where the LT protocol is defined. Clause 8.1.13.6 defines the LTR Generation Process.

On-demand SL Control:

This process is defined in clause 8.1.14 where the SL protocol is defined. Clause 8.1.14.2 defines the SL Control process.

SLM Generation:

This process is defined in clause 8.1.14 where the SL protocol is defined. Clause 8.1.14.3 defines the SLM Generation process.

SLR Generation:

This process is defined in clause 8.1.14 where the SL protocol is defined. Clause 8.1.14.5 defines the SLR Generation process.

SLM Mux:

The SLM Mux process interleaves the signal sets SLM(DA,P,MEP_ID,Test_ID,TxFCl,TLV) from the input ports (X, Y, Z).

Defects None.

Consequent actions None.

Defect correlations None.

Performance monitoring None.

9.4.1.2 ETH Diagnostic Flow Termination Sink Function for MEPs (ETHDe_FT_Sk)

The ETHDe_FT_Sk process diagram is shown in Figure 9-43.

Symbol

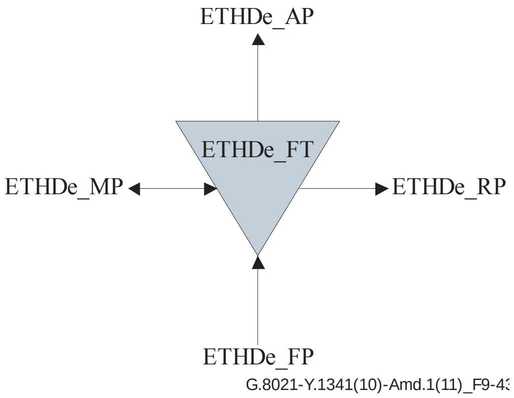


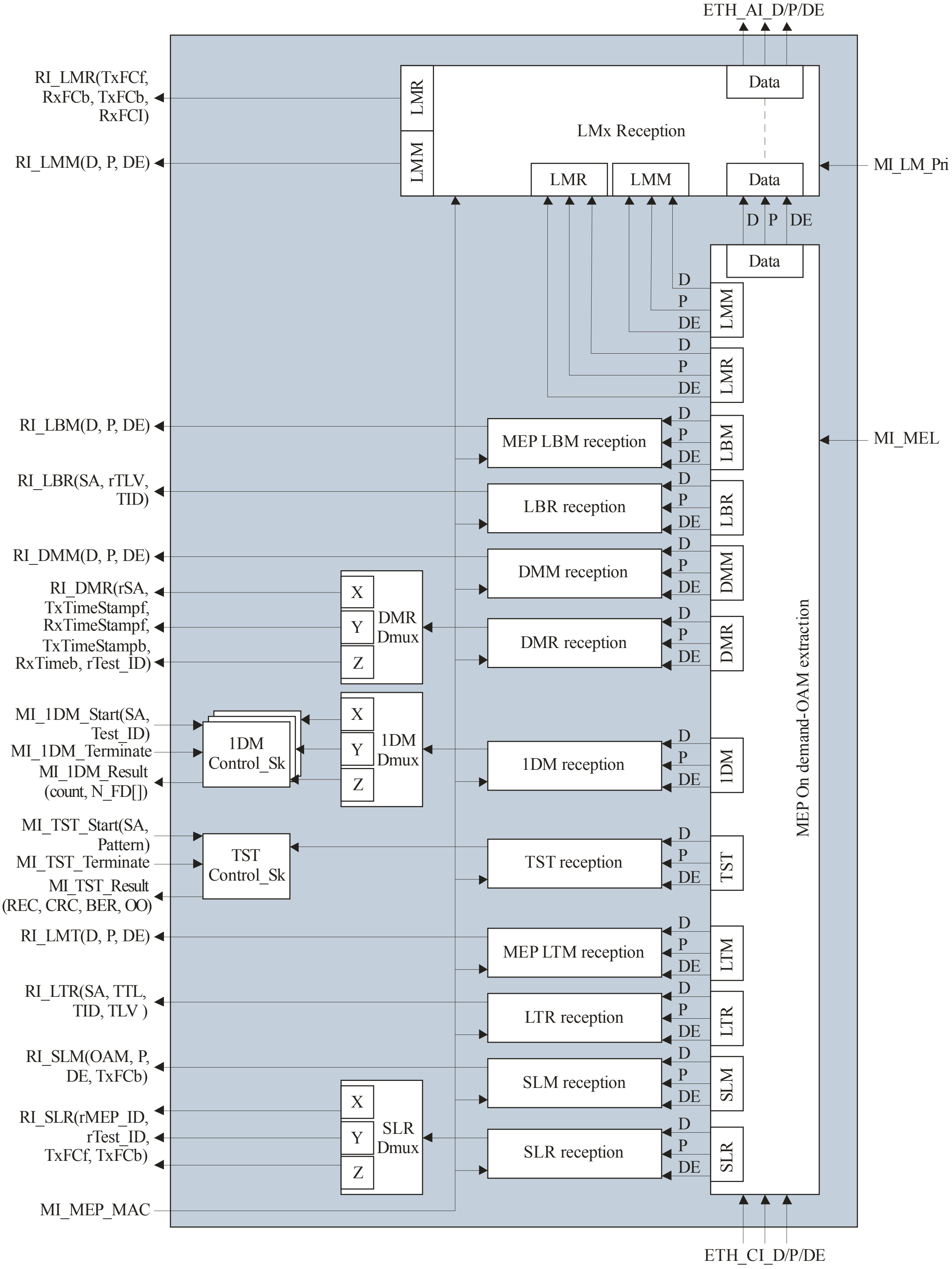
Figure 9-43 – ETHDe_FT_Sk symbol

Interfaces

Table 9-15 – ETHDe_FT_Sk interfaces

Inputs	Outputs
ETH_FP:	ETH_AP:
ETH_CI_D	ETH_AI_D
ETH_CI_P	ETH_AI_P
ETH_CI_DE	ETH_AI_DE
	ETH ~~De_FT_Sk~~_RP:
	ETH~~De_FT_Sk~~_RI_LMM(D,P,DE)
	ETH~~De_FT_Sk~~_RI_LMR(TxFCf,RxFCb,TxFCb,RxFCl)
	ETH~~De_FT_Sk~~_RI_LBM(D,P,DE)
	ETH~~De_FT_Sk~~_RI_LBR(SA,rTLV,TID)
	ETH~~De_FT_Sk~~_RI_DMM(D,P,DE)
ETHDe_FT_Sk_MP:	ETH~~De_FT_Sk~~_RI_DMR(SA,TxTimestampf,RxTimeStampf, TxTimeStampb,RxTimeb,TestID)
ETHDe_FT_Sk_MI_Active	
ETHDe_FT_Sk_MI_LM_Pri	ETH~~De_FT_Sk~~_RI_LTM(D,P,DE)
ETHDe_FT_Sk_MI_MEL	ETH~~De_FT_Sk~~_RI_LTR(SA,TTL,TID,TLV)
ETHDe_FT_Sk_MI_MEP_MAC	ETH_RI_SLM(OAM,P,DE,TxFCb)
ETHDe_FT_Sk_MI_1DM_Start(SA,Test_ID)	ETH_RI_SLR(rMEP_ID,rTest_ID,TxFCf,TxFCb)
ETHDe_FT_Sk_MI_1DM_Terminate	ETHDe_FT_Sk_MP:
ETHDe_FT_Sk_MI_TST_Start(SA,Pattern)	ETHDe_FT_Sk_MI_1DM_Result(count,N_FD[])
ETHDe_FT_Sk_MI_TST_Terminate	ETHDe_FT_Sk_MI_TST_Result(REC,CRC,BER,OO)

Processes



G.8021-Y.1341(10)-Amd.1(11)_F9-44

Figure 9-44 – ETHDe_FT_Sk processes

MEP On Demand-OAM extraction process:

The MEP On Demand-OAM Extraction process extracts OAM traffic units that are processed in the ETHDe_FT_Sk process from the stream of traffic units as defined in the following pseudo code:

```
if (TYPE=<ETHOAM>) and (MEL=MI_MEL) then
  switch(OPC) {
  case <LMM>: extract ETH-LMM OAM traffic unit and forward to LMM Port
  case <LMR>: extract ETH-LMR OAM traffic unit and forward to LMR Port
  case <DMM>: if (Flag.Type=0) then
                extract ETH-DMM OAM traffic unit and forward to DMM Port
                endif
  case <DMR>: if (Flag.Type=0) then
                extract ETH-DMR OAM traffic unit and forward to DMR Port
              endif
  case <1DM>: extract ETH-1DM OAM traffic unit and forward to 1DM Port
  case <LTM>: extract ETH-LTM OAM traffic unit and forward to LTM Port
  case <LTR>: extract ETH-LTR OAM traffic unit and forward to LTR Port
  case <LBM>: extract ETH-LBM OAM traffic unit and forward to LBM Port
  case <LBR>: extract ETH-LBR OAM traffic unit and forward to LBR Port
  case <TST>: extract ETH-TST OAM traffic unit and forward to TST Port
  case <SLM>: extract ETH-SLM OAM traffic unit and forward to SLM port
  case <SLR>: extract ETH-SLR OAM traffic unit and forward to SLR port
else
    forward ETH_CI_traffic unit to Data Port
endif
```

NOTE – If both ETHDe_FT and ETHx_FT are involved in synthetic loss measurements, the MEP On Demand-OAM Extraction process needs to determine to which flow termination the received ETH-SLM PDU belongs. The detailed mechanism is for further study.

MEP LBM Reception:

This Process is defined in clause 8.1.8, where the LB protocol is defined. Clause 8.1.8.5 defines the LBM MEPReception process.

LBR Reception:

This Process is defined in clause 8.1.8, where the LB protocol is defined. Clause 8.1.8.7 defines the LBR Reception process.

LMx Reception:

This Process is defined in clause 8.1.9, where the LM protocol is defined. Clause 8.1.9.4 defines the LMx Reception process.

DMM Reception:

This Process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.4 defines the DMM Reception process.

DMR Reception:

This Process is defined in clause 8.1.10, where the DM protocol is defined. Clause 8.1.10.6 defines the DMR Reception process.

DMR Demux:

The DMR Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

1DM Reception:

This Process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.4 defines the 1DM Reception process.

1DM Demux:

The 1DM Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

1DM Control_Sk:

This Process is defined in clause 8.1.11, where the 1DM protocol is defined. Clause 8.1.11.5 defines the 1DM Control_Sk process.

TST Reception:

This Process is defined in clause 8.1.12, where the TST protocol is defined. Clause 8.1.12.4 defines the TST Reception process.

TST Control_Sk:

This Process is defined in clause 8.1.12, where the TST protocol is defined. Clause 8.1.12.5 defines the TST Control_Sk process.

MEP LTM Reception:

This Process is defined in clause 8.1.13, where the LT protocol is defined. Clause 8.1.13.5 defines the MEP LTM Reception process.

LTR Reception:

This Process is defined in clause 8.1.13, where the LT protocol is defined. Clause 8.1.13.7 defines the LTR Reception process.

SLM Reception

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.4 defines the SLM Reception process.

SLR Reception

This process is defined in clause 8.1.14, where the SL protocol is defined. Clause 8.1.14.6 defines the SLR Reception process.

SLR Demux:

The SLR Demux process deinterleaves the incoming signal set (D,P,DE) to the different output ports (X, Y, Z). P and/or Test_ID signal can be used for the selection of the port.

Defects None.

Consequent actions None.

Defect correlations None.

Performance monitoring None.

27) Clause 9.6.2.2

Revise clause 9.6.2.2 with respect to TCS as follows:

9.6.2.2 ETH Group Traffic Conditioning function (ETH_GTCS_Sk)

~~For Further Study~~

For ETH Group Traffic, the traffic conditioning process is performed per flow point, but there is no correlation between the various processes. Therefore, an ETH_GTCS_Sk function can be modelled by multiple ETH_TCS_Sk functions. No specific function is defined in this Recommendation.

28) Clause 10.3

Revise the first paragraph of clause 10.3 with respect to ETYn/ETH-m adaptation as shown:

Figures 10-3 and 10-4 illustrate the Ethernet trail termination to ETH adaptation function (ETYn/ETH_A and ETYn/ETH-m_A). Information crossing the ETH flow point (ETH_FP) and ETH termination flow point (ETH_TFP) is referred to as ETH characteristic information (ETH_CI). Information crossing the ETYn access point (ETY_AP) is referred to as ETYn adapted information (ETYn_AI). Note that ETYn/ETH-m_A is a compound function of ETYn/ETH_A and ETHx/ETH-m_A (see clause 9.3.3).

29) Clause 10.4

Revise the first paragraph of clause 10.4 with respect to ETY3/ETC3 as shown:

This adaptation function adapts 1000BASE-SX, -LX, or -CX physical layer signals from/to GMII data octets. The combination of ETY3_TT and ETY3/ETC3_A represents the functions up to and including the PCS sublayer in the 802.3 model. The GMII data octets~~8B/10B-encoded codewords. Codewords~~ may be extracted from or mapped into GFP-T frames, per clause 11.2, SDH to ETC Adaptation functions (Sn-X/ETC3_ A). It may also be extracted from and mapped into ODU0, per clause 14.3.7.1/G.798 (ODU0P/CBRx_A). In the latter case, the ETC3_CP from the ETY3/ETC3_A function is bound to the CBRx_CP of the ODU0P/CBRx_A function.

30) Clauses 11.5.4 and 10.7

Delete clause 11.5.4 and create clause 10.5 related to ETH PP-OS:

~~11.5.4~~

~~For Further Study.~~

10.5 ETY4 to Ethernet PP-OS adaptation functions (ETY4/ETHPP-OS_A)

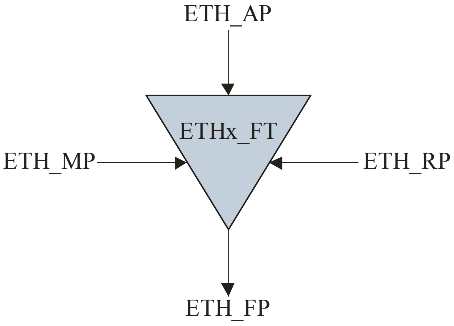
The ETY4 to Ethernet PP-OS adaptation function supports transporting preamble and ordered set information of the 10GBASE-R signals over enhanced OPU2 payload area.

It adapts 10GBASE-R signals from/to data frames which include the preamble and start-of-frame delimiter and ordered sets from the inter-frame gap into ETHPP-OS_CI for subsequent mapping into an OPU2 with extended payload area as described in clause 11.5.3.

Note that there is no Ethernet MAC termination function. Consequently, since no error checking is performed on the Ethernet MAC frames, errored MAC frames are forwarded in both ingress and egress directions.

10.5.1 ETY4 to Ethernet PP-OS adaptation source function (ETY4/ETHPP-OS_A_So)

Symbol



G.8021-Y.1341(10)-Amd.1(11)_F9-12

Figure 10-12 – ETY4/ETHPP-OS_A_So symbol

Interfaces

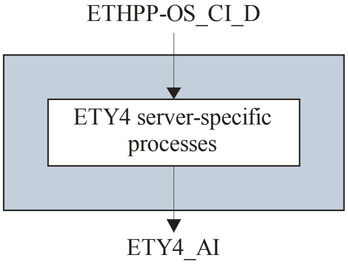
Table 10-8 – ETY4/ETHPP-OS_A_So interfaces

Inputs	Outputs
ETHPP-OS_FP: ETHPP-OS_CI_D ETHPP-OS_CI_SSF ETY4/ETHPP-OS_A_So_MP: ETY4/ETHPP-OS_A_So_MI_Active	ETY4_AP: ETY4_AI_Data ETY4_AI_ClocK ETY4_AI_SSF

NOTE – ETHPP-OS_CI_D is composed of Preamble, Payload and Ordered Set information as described in [ITU-T G.7041].

Processes

A process diagram of this function is shown in Figure 10-13.



G.8021-Y.1341(10)-Amd.1(11)_F10-13

Figure 10-13 – ETY4/ETHPP-OS_A_So process diagram

Activation: The ETY4/ETHPP-OS_A_So function shall access the ETY4 access point and perform the processes specified below when it is activated (MI_Active is true). Otherwise, it shall not access the ETY4 access point.

ETY4 Server-specific processes: None.

NOTE – All source processes related to the Ethernet physical layer are encapsulated in this Recommendation by the ETYn_TT_So function.

Defects	None.
Consequent actions	None.
Defect correlations	None.
Performance monitoring	For further study.

10.5.2 ETY4 to Ethernet PP-OS adaptation sink function (ETY4/ETHPP-OS_A_Sk)

Symbol

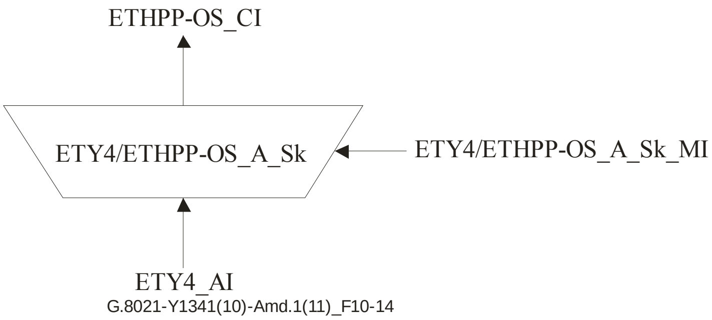


Figure 10-14 – ETY4/ETHPP-OS_A_Sk symbol

Interfaces

Table 10-9 – ETY4/ETHPP-OS_A_Sk interfaces

Inputs	Outputs
ETY4_API: ETY4_AI_Data ETYn_AI_ClocK ETYn_AI_TSF ETY4/ETHPP-OS_A_Sk_MP: ETY4/ETHPP-OS_A_Sk_MI_Active	ETHPP-OS_FP: ETHPP-OS_CI_D ETHPP-OS_CI_SSF

Processes

A process diagram of this function is shown in Figure 10-15.

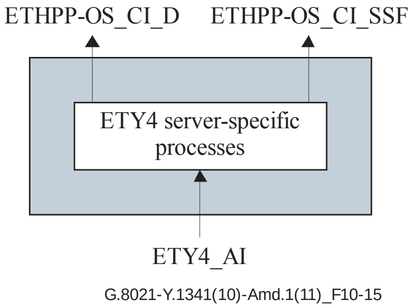


Figure 10-15 – ETY4/ETHPP-OS_A_Sk process diagram

Activation: The ETY4/ETHPP-OS_A_Sk function shall access the ETY4 access point and perform the processes specified below when it is activated (MI_Active is true). Otherwise, it shall activate the SSF signal and not report its status via the management point.

ETY4 Server-specific processes: None.

NOTE – All sink processes related to the Ethernet physical layer are encapsulated in this Recommendation by the ETYn_TT_Sk function.

Defects None.

Consequent actions

aSSF ← AI_TSF

Note that the replacement signal is generated in the subsequent adaptation source function ODU2P/ETHPP-OS_A_So.

Defect correlations None.

Performance monitoring For further study.

31) Clause 11.1.1.2

Revise clause 11.1.1.2 with respect to aSSFfdi as follows:

11.1.1.2 VC-n to ETH Adaptation sink function (Sn/ETH_A_Sk)

•••

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

•••

32) Clause 11.1.2.2

Revise clause 11.1.2.2 with respect to aSSFfdi as follows:

11.1.2.2 LCAS-capable VC-n-Xv to ETH Adaptation sink function (Sn-X-L/ETH_A_Sk)

•••

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

NOTE 1 – XAR = 0 results in AI_TSF being asserted, so there is no need to include it as additional contributor to aSSF.

•••

33) Clause 11.1.3.2

Revise clause 11.1.3.2 with respect to aSSFfdi as follows:

11.1.3.2 VC-m to ETH adaptation sink function (Sm/ETH_A_Sk)

• • •

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

• • •

34) Clause 11.1.4.2

Revise clause 11.1.4.2 with respect to aSSFfdi as follows:

11.1.4.2 LCAS-capable VC-m-Xv to ETH Adaptation sink function (Sm-X-L/ETH_A_Sk)

• • •

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

NOTE 1 – XAR = 0 results in AI_TSF being asserted, so there is no need to include it as additional contributor to aSSF.

• • •

35) Clause 11.2.1

Revise clause 11.2.1 with respect to VC-n Adaptation as follows:

11.2.1 VC-n-X to ETC3 Adaptation Source function (Sn-X/ETC3_A_So)

This function maps ETC_CI information from an ETC3 onto an Sn-X_AI signal (n=3, 4). This mapping is currently only defined for X=7 for VC-4 and X=22 for VC-3.

Data at the Sn-X_AP is a VC-n-Xv, having a payload as described in [ITU-T G.707], but with indeterminate POH bytes: J1, B3, G1.

Symbol

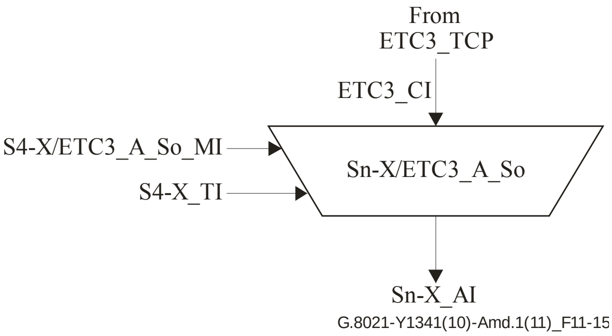


Figure 11-15 – Sn-X/ETC3_A_So symbol

Interfaces

Table 11-9 – Sn-X/ETC3_A_So interfaces

Inputs	Outputs
ETC3_TCP: ETC3_CI_Data_Control ETC3_CI_Clock ETC3_CI_Control_Ind ETC3_CI_SSF Sn-X_TP: Sn-X _TI_Clock Sn-X _TI_FrameStart Sn-X/ETC3_A_So_MP: Sn-X/ETC3_A_So_MI_Active Sn-X/ETC3_A_So_MI_CSFEnable	Sn-X _AP: Sn-X _AI_Data Sn-X _AI_Clock Sn-X _AI_FrameStart

Processes

A process diagram of this function is shown in Figure 11-16.

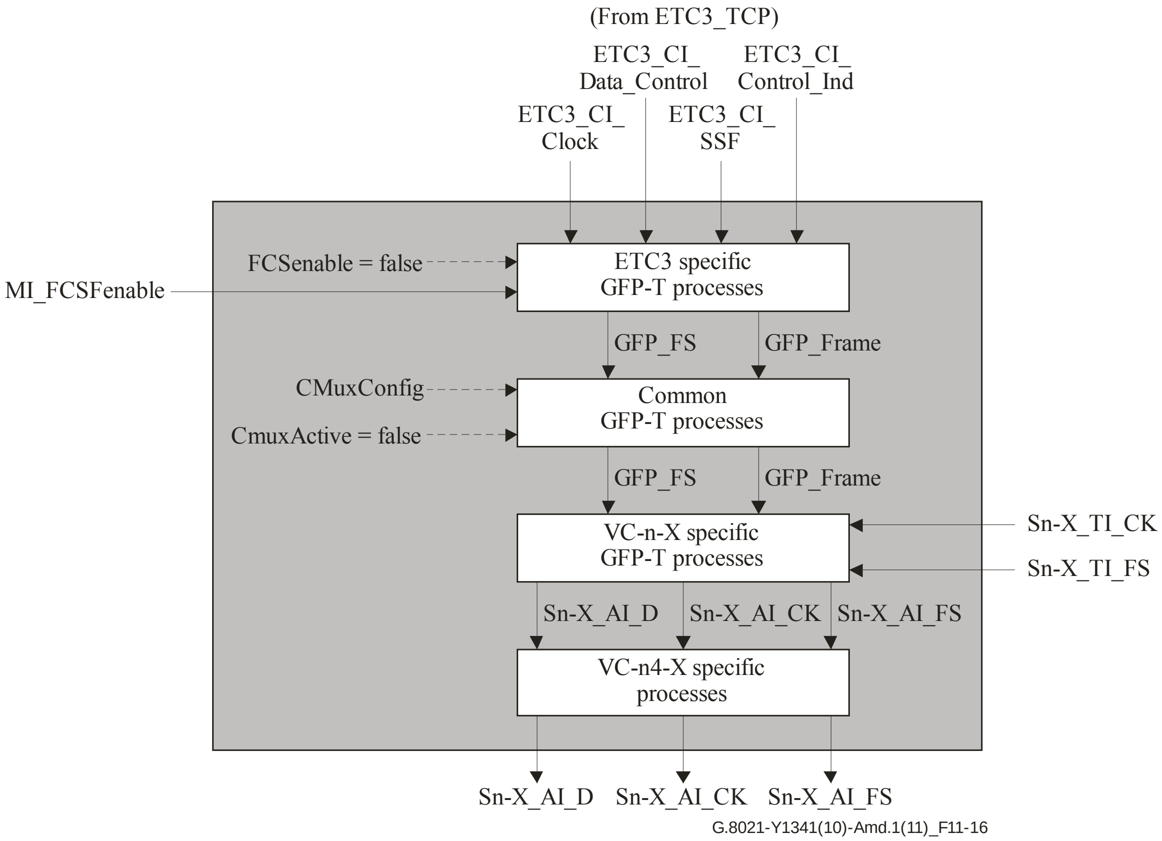


Figure 11-16 – Sn-X/ETC3_A_So process

Ethernet specific GFP-T source process:

See clause 8.5.4.2.1 of [ITU-T G.806]. GFP pFCS generation is disabled (FCSenable=false). The UPI value for Transparent Gb Ethernet shall be inserted (Table 6-3 of [ITU-T G.7041]). The Ethernet codeword information is inserted into the client payload information field of the GFP-T frames according to clause 8 of [ITU-T G.7041]. 65B rate adaptation is enabled (RAdisable=false).

NOTE – Equipment designed prior to this Amendment may not support configuration of RAdisable; in such equipment the use of 65B rate adaptation is implicitly enabled.

Response to ETC3_CI_SSF is according to the principles in clauses 8.3 and 8.3.4 of [ITU-T G.7041] and Appendix VIII of [ITU-T G.806]. Details are for further study.

Common GFP source process:

See clause 8.5.3.1 of [ITU-T G.806]. GFP channel multiplexing is not supported (CMuxActive=false).

VC-n-X specific GFP source process:

See clause 8.5.2.1 of [ITU-T G.806]. The GFP frames are mapped into the VC-n-X (n=3,4) payload area according to clause 10.6 of [ITU-T G.707].

VC-n-X specific source process:

C2: Signal label information is derived directly from the Adaptation function type. The value for "GFP mapping" in Table 9-11 of [ITU-T G.707] is placed in the C2 byte position.

NOTE – For Sn-X/ETC3_A_So, the H4, K3, F2, and F3 bytes are undefined at the Sn-X_AP output of this function (as per clause 12 of [ITU-T G.783]).

Defects None.

Consequent actions

aCSF-RDI ← CI_SSFrdi and CSFrdifdiEnable and CSFEnable

aCSF-FDI ← CI_SSFfdi and CSFrdifdiEnable and CSFEnable

aCSF-LOS ← CI_SSF and CSFEnable

Defect correlations None.

Performance monitoring For further study.

36) Clause 11.4.1.2

Revise clause 11.4.1.2 with respect to aSSFfdi as follows:

• • •

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

• • •

37) Clause 11.4.2.2

Revise clause 11.4.2.2 with respect to aSSFfdi as follows:

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

NOTE 3 – X_{AR} = 0 results in AI_TSF being asserted, so there is no need to include it as additional contributor to aSSF.

38) Clause 11.5.1

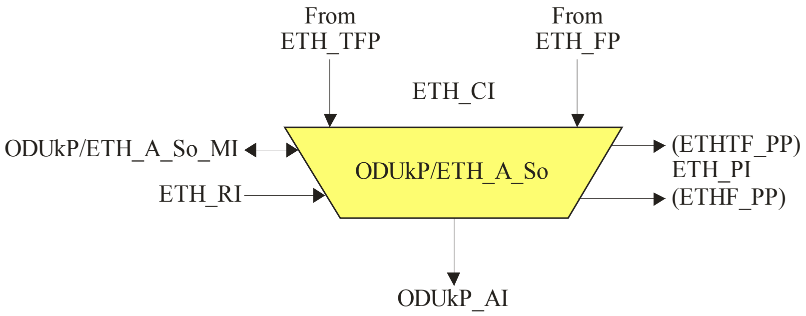
Revise clause 11.5.1 with respect to ODU Adaptation as follows:

11.5.1 ODUk to ETH adaptation functions (ODUkP/ETH_A~~; k = 1, 2, 3~~)

11.5.1.1 ODUk to ETH adaptation source function (ODUkP/ETH_A_So)

The ODUkP/ETH_A_So function creates the ODUk signal from a free running clock. It maps the ETH_CI information into the payload of the OPUk~~ (k = 1, 2, 3)~~, adds OPUk Overhead (RES, PT) and default ODUk Overhead.

Symbol



G.8021-Y1341(10)-Amd.1(11)_F11-27

Figure 11-27 – ODUkP/ETH_A_So symbol

Interfaces

Table 11-15 – ODUkP/ETH_A_So interfaces

Inputs	Outputs
ETH_TFP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ~~ETH_RP:~~ ~~ETH_RI_RSF~~ ODUkP/ETH_A_So_~~MI~~MP: ODUkP/ETH_A_So_MI_Active ODUkP/ETH_A_So_MI_CSFEnable ODUkP/ETH_A_So_MI_CSFrdifdiEnable	ODUkP_AP: ODUkP_AI_Data ODUkP_AI_Clock ODUkP_AI_FrameStart ODUkP_AI_MultiframeStart ETHTF_PP: ETH_PI_D ETH_PI_P ETH_PI_DE ETH~~TF~~_PP: ETH_PI_D ETH_PI_P ETH_PI_DE

Processes

A process diagram of this function is shown in Figure 11-28.

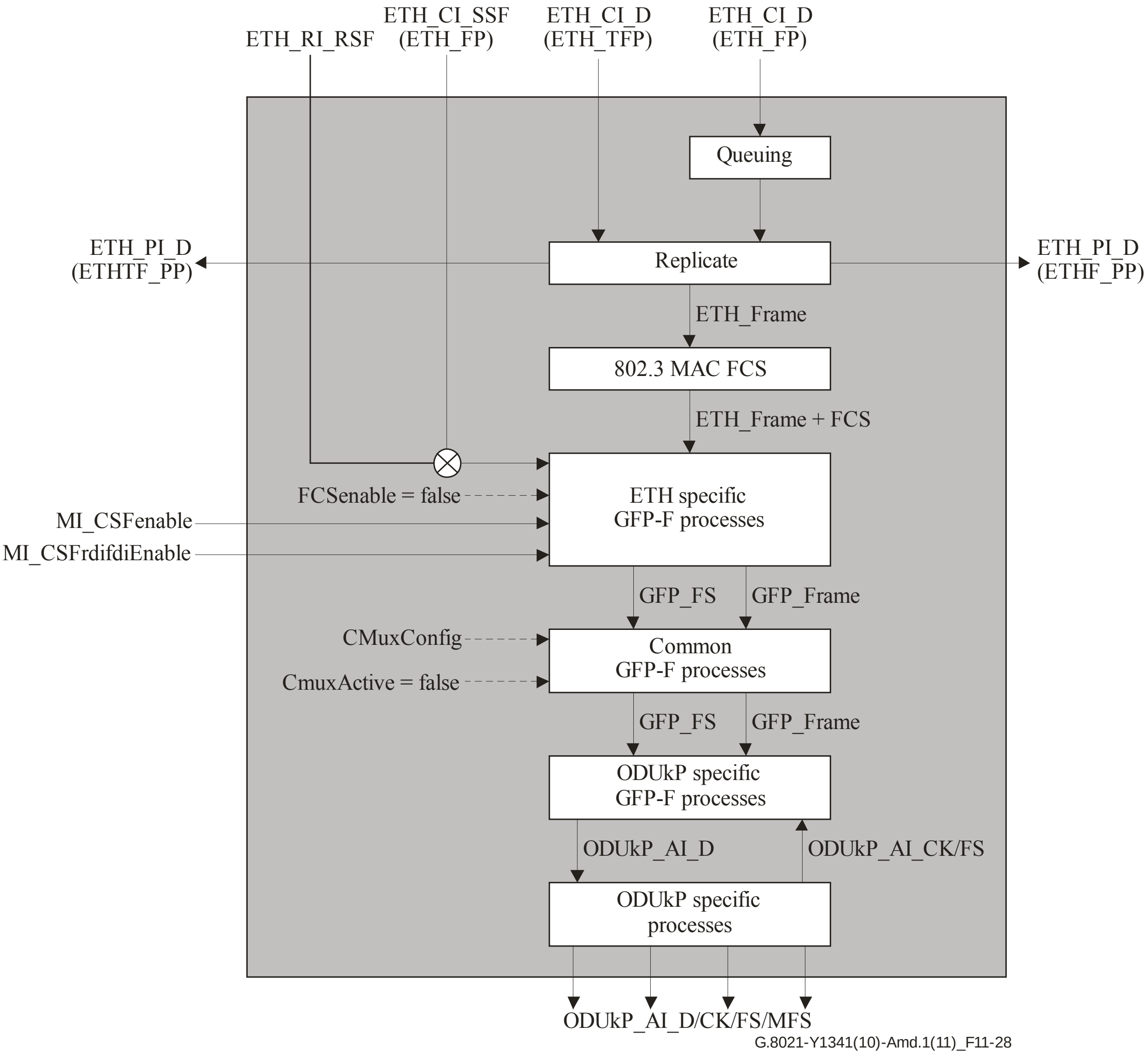


Figure 11-28 – ODUkP/ETH_A_So process

"Queuing" process:

See clause 8.2.

"Replicate" process:

See clause 8.4.

802.3 MAC FCS generation:

See clause 8.8.1.

Ethernet specific GFP-F source process:

See clause 8.5.4.1.1 of [ITU-T G.806]. GFP pFCS generation is disabled (FCSenable=false). The UPI value for frame-mapped Ethernet shall be inserted (Table 6-3 of [ITU-T G.7041]). The Ethernet frames are inserted into the client payload information field of the GFP-F frames according to clause 7.1 of [ITU-T G.7041].

~~Response to ETH_CI_SSF asserted is *for further study*.~~

Common GFP source process:

See clause 8.5.3.1 of [ITU-T G.806]. GFP channel multiplexing is not supported (CMuxActive=false).

ODUkP specific GFP source process:

See clause 8.5.2.1 of [ITU-T G.806]. The GFP frames are mapped into the ODUk payload area according to clause 17.4~~3~~ of [ITU-T G.709].

ODUkP specific source process:

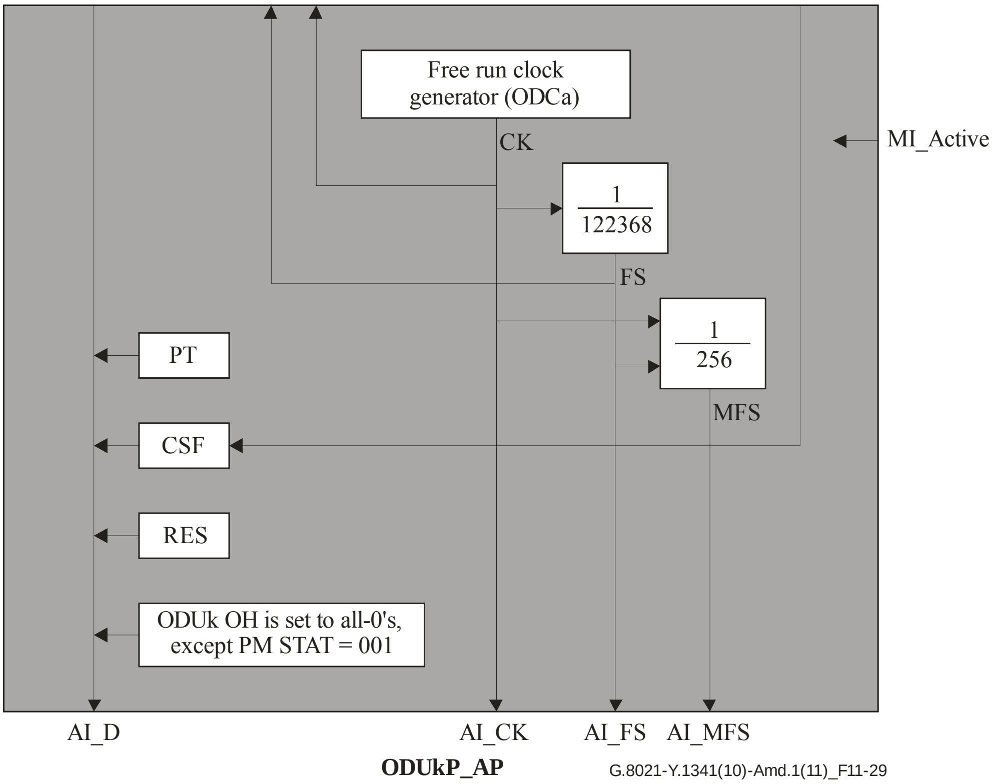


Figure 11-29 – ODUkP specific source process

Clock and (Multi)Frame Start signal generation:

The function shall generate a local ODUk clock (ODUkP_AI_CK) with a clock rate within the minimum to maximum clock rate of the specified ODU signal as given in Table 14-2 of [ITU-T G.798]. ~~of "239/(239 – k) * 4$^{(k-1)}$ * 2 488 320 kHz ± 20 ppm" from a free running oscillator.~~ The jitter and wander requirements as defined in Annex A of [ITU-T G.8251] (ODCa clock) apply.

The function shall generate the (multi)frame start reference signals AI_FS and AI_MFS for the ODUk signal. The AI_FS signal shall be active once per 122 368 clock cycles. AI_MFS shall be active once every 256 frames.

PT: The payload type information is derived directly from the Adaptation function type. The value for "GFP mapping" shall be inserted into the PT byte position of the PSI overhead as defined in clause 15.9.2.1.1 of [ITU-T G.709].

RES: The function shall insert all-0's into the RES bytes.

CSF: The function shall signal the failure of the client signal to the far end by using Bit 1 of the PSI[2] byte of the Payload Structure Identifier as defined in clause 17.1 of [ITU-T G.709].

All other bits of the ODUk overhead should be sourced as "0"s, except the ODUk-PM STAT field which should be set to the value "normal path signal" (001).

Counter processes:

For further study.

Defects None.

Consequent actions

aCSF-RDI ← CI_SSFrdi and CSFrdifdiEnable and CSFEnable

aCSF-FDI ← CI_SSFfdi and CSFrdifdiEnable and CSFEnable

aCSF-LOS ← CI_SSF and CSFEnable

aCSF-OPU ← CI_SSF and CSFEnable

Defect correlations None.

Performance monitoring For further study.

11.5.1.2 ODUk to ETH adaptation sink function (ODUkP/ETH_A_Sk)

The ODUkP/ETH_A_Sk extracts ETH_CI information from the ODUkP payload area, delivering ETH_CI to ETH_TFP and ETH_FP. It extracts the OPUk Overhead (PT and RES) and monitors the reception of the correct payload type.

Symbol

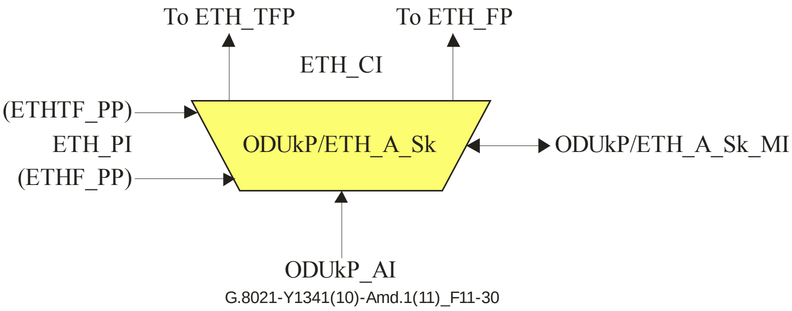


Figure 11-30 – ODUkP/ETH_A_Sk symbol

Interfaces

Table 11-16 – ODUkP/ETH_A_Sk interfaces

Inputs	Outputs
ODUkP_AP: ODUkP_AI_Data ODUkP_AI_ClocK ODUkP_AI_FrameStart ODUkP_AI_MultiframeStart ODUkP_AI_TSF ETHTF_PP: ETH_PI_D ETH_PI_P ETH_PI_DE ETH~~T~~F_PP: ETH_PI_D ETH_PI_P ETH_PI_DE ODUkP/ETH_A_Sk_MP~~I~~: ODUkP/ETH_A_Sk_MI_Active ODUkP/ETH_A_Sk_MI_FilterConfig ODUkP/ETH_A_Sk_MI_CSF_Reported ODUkP/ETH_A_Sk_MI_MAC_Length ODUkP/ETH_A_Sk_MI_CSFrdifdiEnable	ETH_TFP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ~~ETH_RP:~~ ~~ETH_RI_RSF~~ ODUkP/ETH_A_Sk_~~MI~~MP: ODUkP/ETH_A_Sk_MI_AcPT ODUkP/ETH_A_Sk_MI_AcEXI ODUkP/ETH_A_Sk_MI_AcUPI ODUkP/ETH_A_Sk_MI_cPLM ODUkP/ETH_A_Sk_MI_cLFD ODUkP/ETH_A_Sk_MI_cUPM ODUkP/ETH_A_Sk_MI_cEXM ODUkP/ETH_A_Sk_MI_cCSF ODUkP/ETH_A_Sk_MI_pFCSErrors

Processes

A process diagram of this function is shown in Figure 11-31.

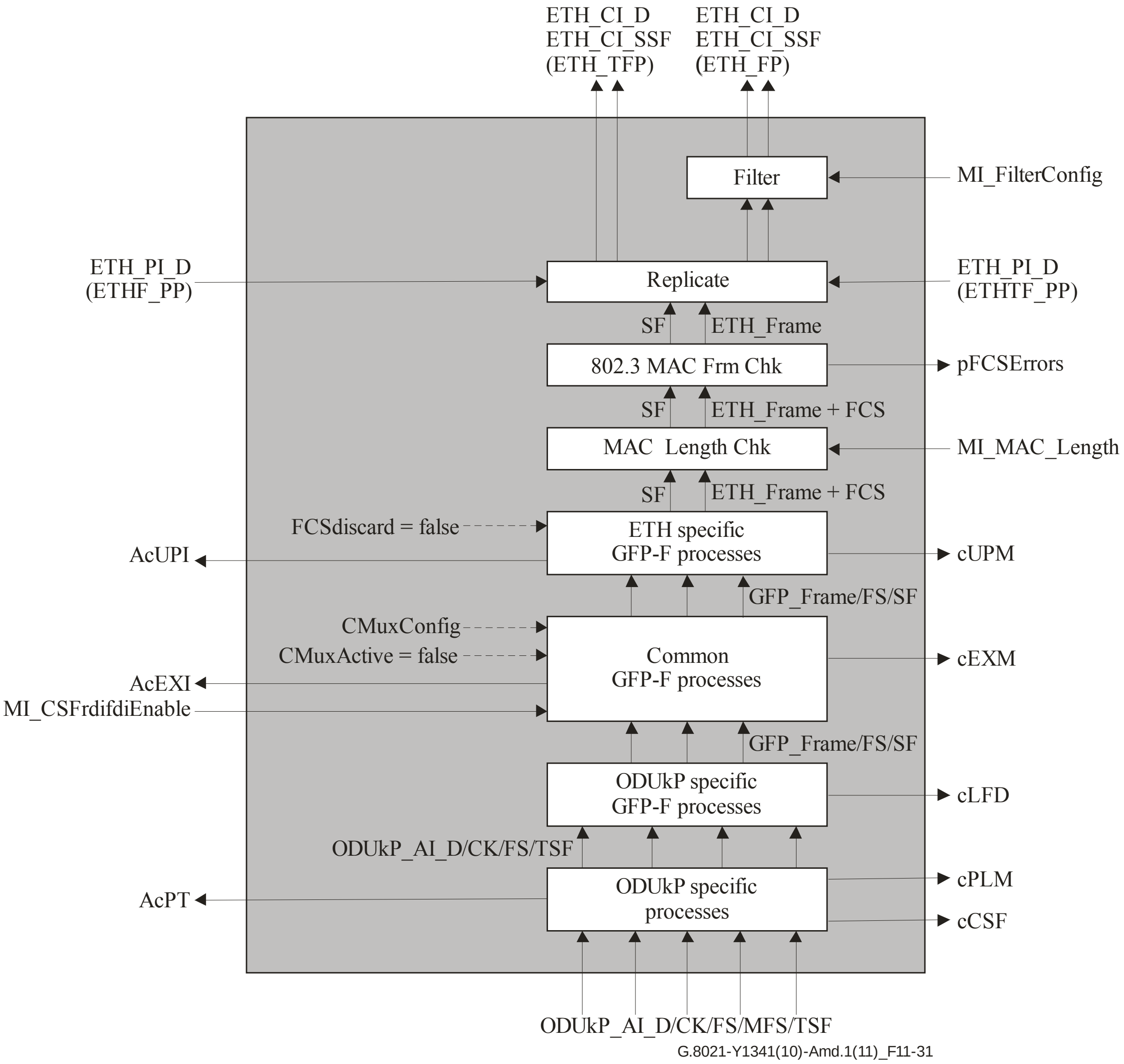


Figure 11-31 – ODUkP/ETH_A_Sk process

"Filter" process:

See clause 8.3.

"Replicate" process:

See clause 8.4.

"802.3 MAC FCS Check" process:

See clause 8.8.2.

Ethernet specific GFP-F sink process:

See clause 8.5.4.1.2 of [ITU-T G.806]. GFP pFCS checking, GFP p_FCSError, p_FDis are not supported (FCSdiscard=false). The UPI value for Frame-Mapped Ethernet shall be expected (Table 6-3 of [ITU-T G.7041]). The Ethernet frames are extracted from the client payload information field of the GFP-F frames according to clause 7.1 of [ITU-T G.7041].

Common GFP sink process:

See clause 8.5.3.2 of [ITU-T G.806]. GFP channel multiplexing is not supported (MI_CMuxActive=false).

ODUkP specific GFP sink process:

See clause 8.5.2.2 of [ITU-T G.806]. The GFP frames are demapped from the ODUk payload area according to clause 17.~~3~~4 of [ITU-T G.709].

ODUkP specific sink process:

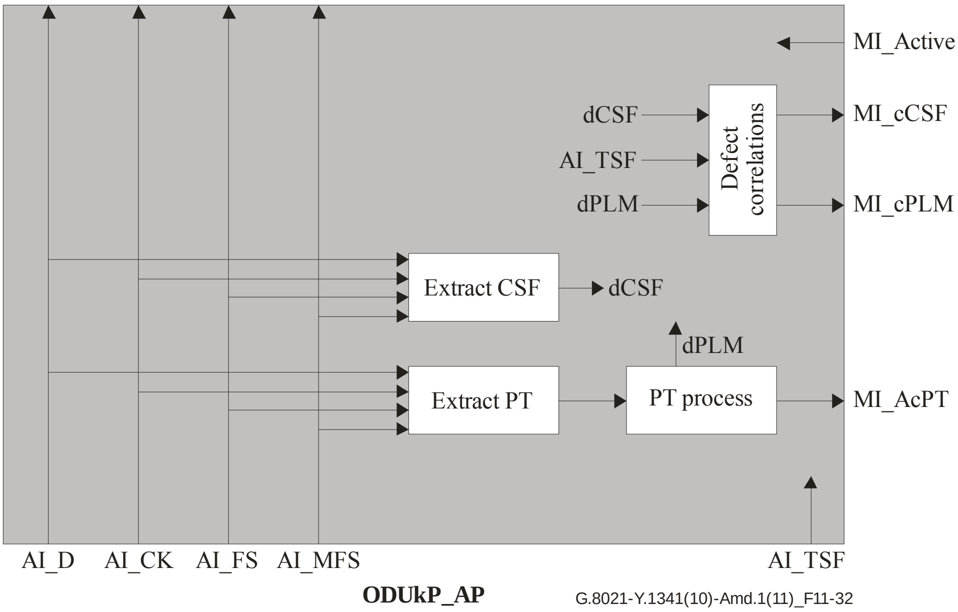


Figure 11-32 – ODUkP specific sink process

PT: The function shall extract the PT byte from the PSI overhead as defined in clause 8.7.1 of [ITU-T G.798]. The payload type value for "GFP mapping" in clause 15.9.2.1.1 of [ITU-T G.709] shall be expected. The accepted PT value is available at the MP (MI_AcPT) and is used for PLM defect detection.

RES: The value in the RES bytes shall be ignored.

CSF: The function shall extract the CSF signal indicating the failure of the client signal from Bit 1 of the PSI[2] byte of the Payload Structure Identifier as defined in clause 17.1 of [ITU-T G.709].

Defects

dPLM – See clause 6.2.4.1 of [ITU-T G.798].

dLFD – See clause 6.2.5.2 of [ITU-T G.806].

dUPM – See clause 6.2.4.3 of [ITU-T G.806].

dEXM – See clause 6.2.4.4 of [ITU-T G.806].

dCSF-LOS – See clause 8.8.6.2.

dCSF-RDI – See clause 8.8.6.2.

dCSF-FDI – See clause 8.8.6.2.

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

Defect correlations

The function shall perform the following defect correlations to determine the most probable fault cause (see clause 6.4 of [ITU-T G.806]). This fault cause shall be reported to the EMF.

cPLM ← dPLM and (not AI_TSF);

cLFD ← dLFD and (not dPLM) and (not AI_TSF);

cUPM ← dUPM and (not dEXM) and (not dPLM) and (not dLFD) and (not AI_TSF);

cEXM ← dEXM and (not dPLM) and (not dLFD) and (not AI_TSF)

cCSF ← (dCSF-LOS or dCSF-OPU ~~dCSF-RDI~~ or dCSF-FDI) and (not dEXM) and (not dUPM) and (not dPLM) and (not dLFD) and (not AI_TSF) and CSF_Reported

Performance monitoring

The function shall perform the following performance monitoring primitives processing. The performance monitoring primitives shall be reported to the EMF.

pFCSErrors: count of FrameCheckSequenceErrors per second.

NOTE – This primitive is calculated by the MAC FCS Check process.

39) Clause 11.5.2

Revise clause 11.5.2 with respect to ODU Adaptation as follows:

11.5.2 LCAS-capable ODUk-Xv to ETH adaptation functions (ODUkP-X-L/ETH_A; k = 1, 2, 3)

11.5.2.1 LCAS-capable ODUk-Xv to ETH adaptation source function (ODUkP-X-L/ETH_A_So)

The ODUkP-X-L/ETH_A_So function creates the ODUk-X-L signal from a free running clock. It maps the ETH_CI information into the payload of the OPUk-Xv (k = 1, 2, 3), adds OPUk-Xv Overhead (RES, vcPT).

Symbol

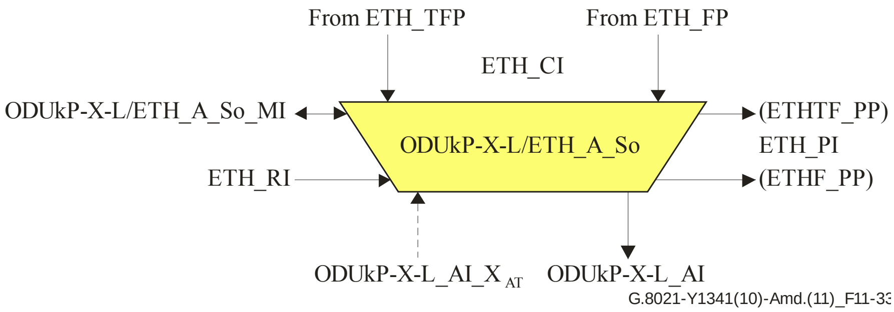


Figure 11-33 – ODUkP-X-L/ETH_A_So symbol

Interfaces

Table 11-17 – ODUkP-X-L/ETH_A_So interfaces

Inputs	Outputs
ETH_TFP: ETH_CI_D ETH_CI_DE ETH_CI_P ETH_FP: ETH_CI_D ETH_CI_DE ETH_CI_P ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ODUkP-X-L_AP: ODUkP-X-L_AI_X_{AT} ODUkP-X-L/ETH_A_So_~~MI~~MP: ODUkP-X-L/ETH_A_So_MI_Active ODUkP-X-L/ETH_A_So_MI_CSFEnable ODUkP-X-L/ETH_A_So_MI_CSFrdifdiEnable	ODUkP-X-L_AP: ODUkP-X-L_AI_Data ODUkP-X-L_AI_ClocK ODUkP-X-L_AI_FrameStart ODUkP-X-L_AI_MultiframeStart ETHTF_PP: ETH_PI_D ETH_PI_DE ETH_PI_P ETH~~T~~F_PP: ETH_PI_D ETH_PI_P ETH_PI_DE

Processes

A process diagram of this function is shown in Figure 11-34.

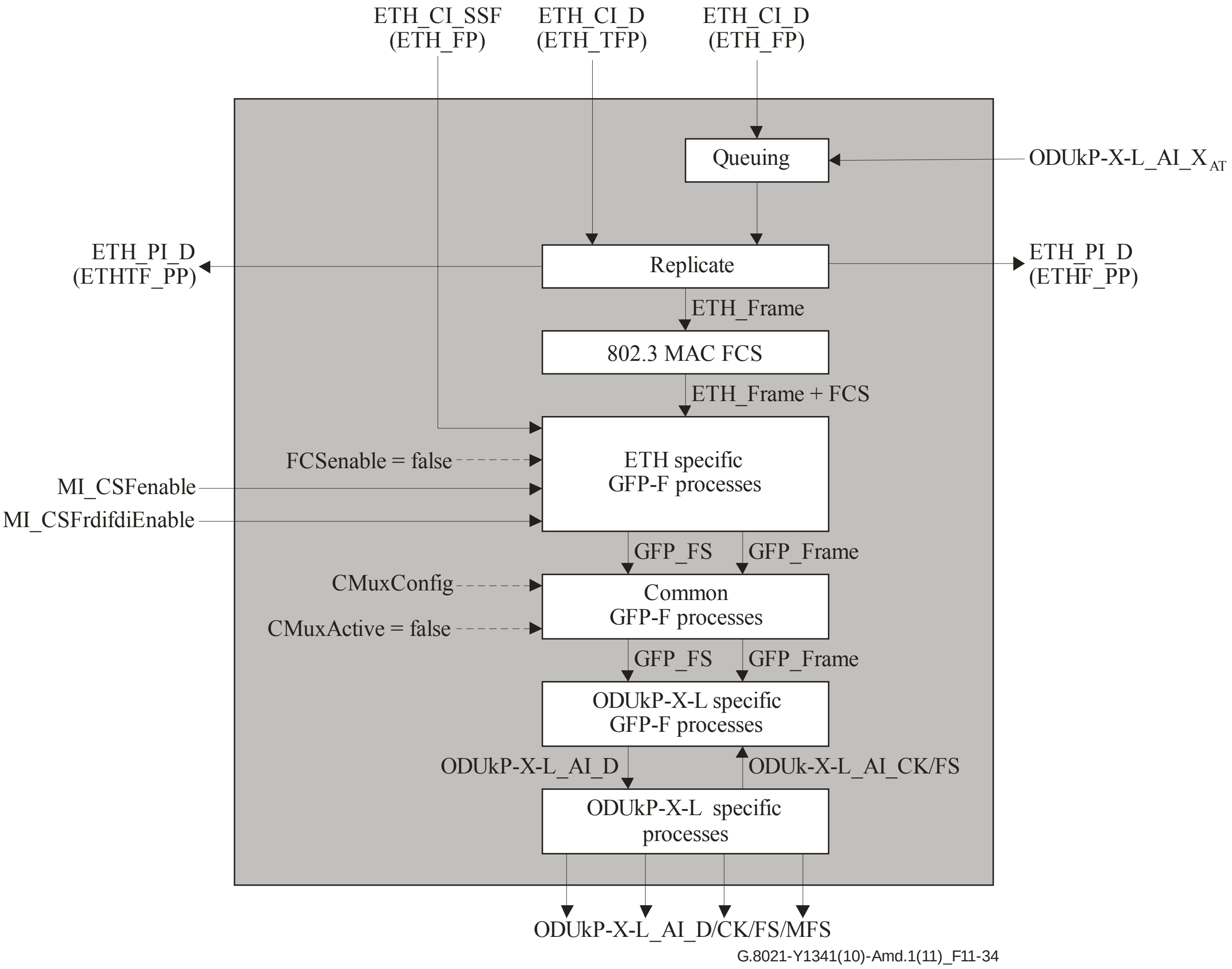


Figure 11-34 – ODUkP-X-L/ETH_A_So process

See clause 11.5.1.1 for a description of ODUkP-X-L/ETH_A processes.

ODUkP-X-L specific source process:

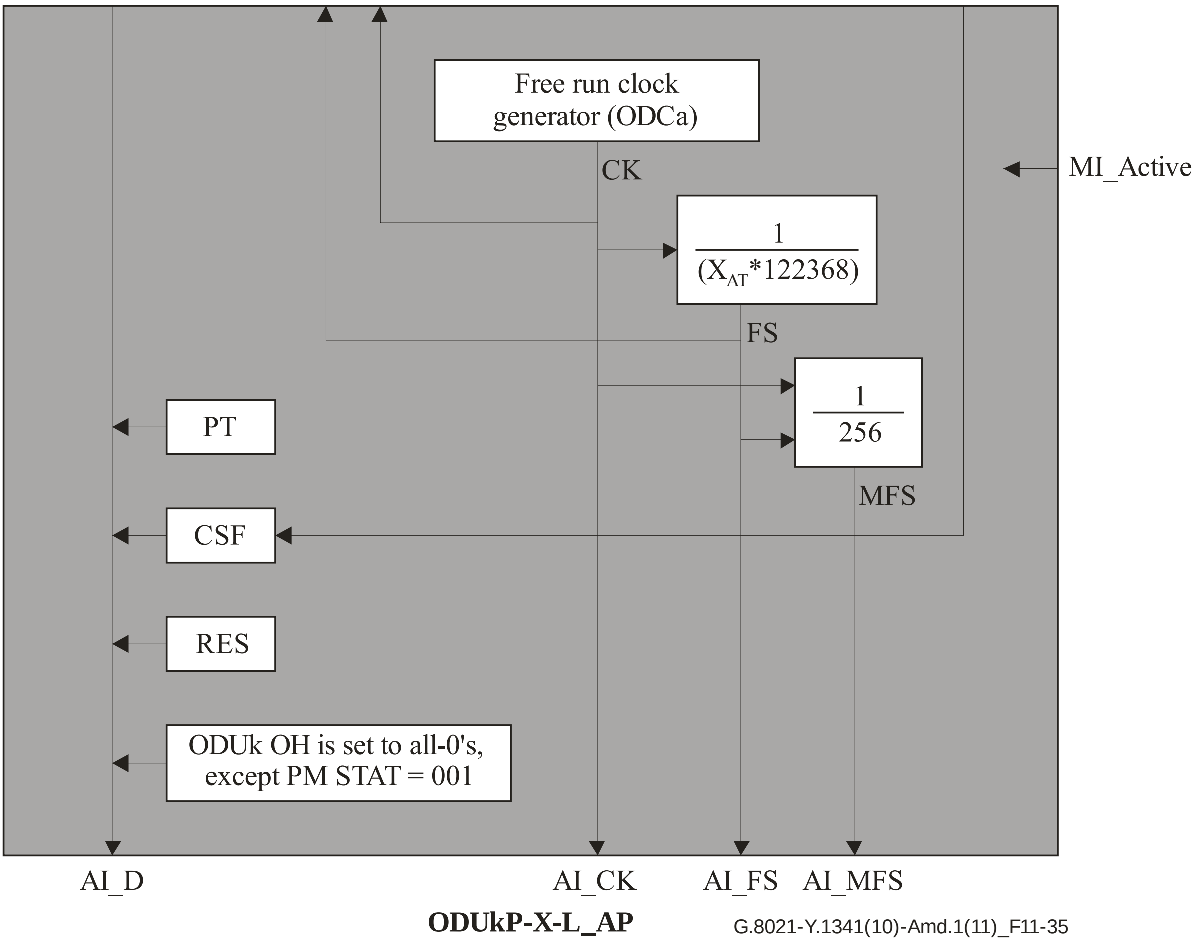


Figure 11-35 – ODUkP-X-L specific source process

Clock and (Multi)Frame Start signal generation:

The function shall generate a local ODUk clock (ODUkP_AI_CK) with a clock rate within the minimum to maximum clock rate of the specified ODU signal as given in Table 14-2 of [ITU-T G.798] ~~of " X_{AT} * 239/(239 – k) * $4^{(k-1)}$ * 2 488 320 kHz ± 20 ppm" from a free running oscillator~~. The jitter and wander requirements as defined in Annex A of [ITU-T G.8251] (ODCa clock) apply.

The function shall generate the (multi)frame start reference signals AI_FS and AI_MFS for the ODUk signal. The AI_FS signal shall be active once per X_{AT} *122 368 clock cycles. AI_MFS shall be active once every 256 frames.

vcPT: The payload type information is derived directly from the Adaptation function type. The value for "GFP mapping" shall be inserted into the vcPT byte position of the PSI overhead as defined in clause 18.1.2.2 of [ITU-T G.709].

RES: The function shall insert all-0's into the RES bytes.

CSF: The function shall signal the failure of the client signal to the far end by using Bit 1 of the PSI[2] byte of the Payload Structure Identifier as defined in clause 18.1.2.2.1.3 of [ITU-T G.709].

All other bits of the ODUk overhead should be sourced as "0"s, except the ODUk-PM STAT field which should be set to the value "normal path signal" (001).

Counter processes:

For further study.

Defects None.

Consequent actions

aCSF-RDI ← CI_SSFrdi and CSFrdifdiEnable and CSFEnable

aCSF-FDI ← CI_SSFfdi and CSFrdifdiEnable and CSFEnable

aCSF-LOS ← CI_SSF and CSFEnable

aCSF-OPU ← CI_SSF and CSFEnable

Defect correlations None.

Performance monitoring For further study.

11.5.2.2 LCAS-capable ODUk-Xv to ETH adaptation sink function (ODUkP-X-L/ETH_A_Sk)

The ODUkP-X-L/ETH_A_Sk extracts ETH_CI information from the ODUkP-Xv payload area, delivering ETH_CI to ETH_TFP and ETH_FP. It extracts the OPUk-Xv Overhead (vcPT and RES) and monitors the reception of the correct payload type.

Symbol

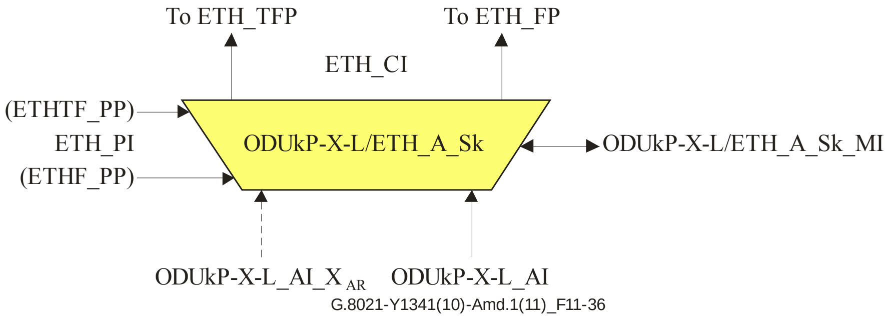


Figure 11-36 – ODUkP-X-L/ETH_A_Sk symbol

Interfaces

Table 11-18 – ODUkP-X-L/ETH_A_Sk interfaces

Inputs	Outputs
ODUkP-X-L_AP: ODUkP-X-L_AI_Data ODUkP-X-L_AI_ClocK ODUkP-X-L_AI_FrameStart ODUkP-X-L_AI_MultiframeStart ODUkP-X-L_AI_TSF ODUkP-X-L_AI_X_{AR} ETHF_PP: ETH_PI_D ETH_PI_P ETH_PI_DE ETHTF_PP: ETH_PI_D ETH_PI_P ETH_PI_DE ODUkP-X-L/ETH_A_Sk_MI: ODUkP-X-L/ETH_A_Sk_MI_Active ODUkP-X-L/ETH_A_Sk_MI_FilterConfig ODUkP-X-L/ETH_A_Sk_MI_CSF_Reported ODUkP-X-L/ETH_A_Sk_MI_MAC_Length ODUkP-X-L/ETH_A_Sk_MI_CSFrdifdiEnable	ETH_TFP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_FP: ETH_CI_D ETH_CI_P ETH_CI_DE ETH_CI_SSF ETH_CI_SSFrdi ETH_CI_SSFfdi ODUkP-X-L/ETH_A_Sk_MI: ODUkP-X-L/ETH_A_Sk_MI_AcVcPT ODUkP-X-L/ETH_A_Sk_MI_AcEXI ODUkP-X-L/ETH_A_Sk_MI_AcUPI ODUkP-X-L/ETH_A_Sk_MI_cVcPLM ODUkP-X-L/ETH_A_Sk_MI_cLFD ODUkP-X-L/ETH_A_Sk_MI_cUPM ODUkP-X-L/ETH_A_Sk_MI_cEXM ODUkP-X-L/ETH_A_Sk_MI_cCSF ODUkP-X-L/ETH_A_Sk_MI_pFCSError

Processes

See process diagram and process description in clause 11.5.1.2. The additional ODUkP-X-L_AI_X_{AR} interface is not connected to any of the internal processes.

ODUkP-X-L specific sink process:

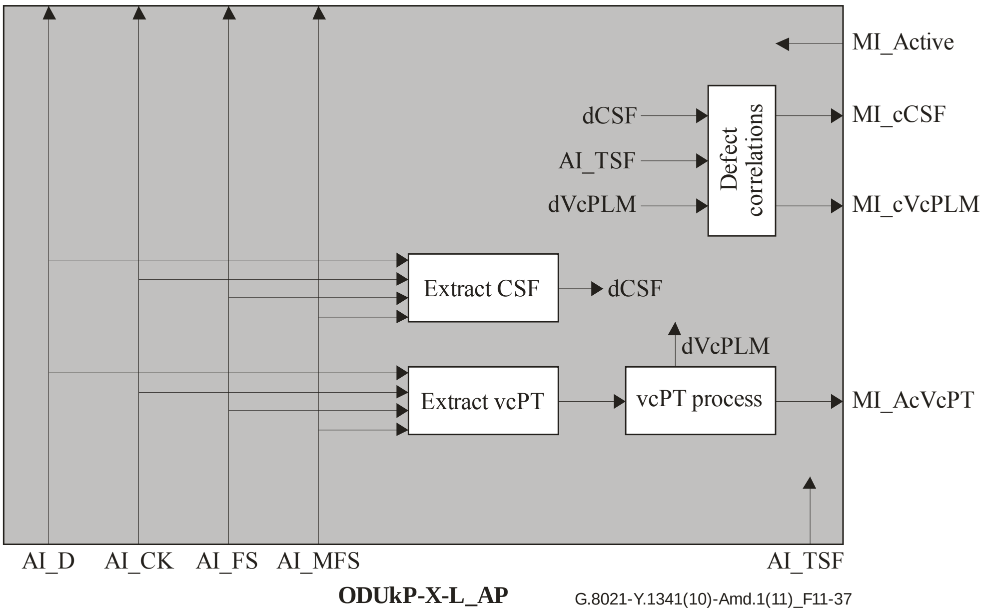


Figure 11-37 – ODUkP-X-L specific sink process

PT: The function shall extract the vcPT byte from the PSI overhead as defined in clause 8.7.3 of [ITU-T G.798]. The payload type value for "GFP mapping" in clause 18.1.2.2 of [ITU-T G.709] shall be expected. The accepted PT value is available at the MP (MI_AcPT) and is used for PLM defect detection.

RES: The value in the RES bytes shall be ignored.

CSF: The function shall extract the CSF signal indicating the failure of the client signal from Bit 1 of the PSI[2] byte of the Payload Structure Identifier as defined in clause 18.1.2.2.1.3 of [ITU-T G.709].

Defects

dVcPLM – See clause 6.2.4.2 of [ITU-T G.798].

dLFD – See clause 6.2.5.2 of [ITU-T G.806].

dUPM – See clause 6.2.4.3 of [ITU-T G.806].

dEXM – See clause 6.2.4.4 of [ITU-T G.806].

dCSF-LOS – See clause 8.8.6.2.

dCSF-RDI – See clause 8.8.6.2.

dCSF-FDI – See clause 8.8.6.2.

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dVcPLM or dLFD or dUPM or dEXM or dCSF-LOS

aSSFrdi ← dCSF-RDI and CSFrdifdiEnable

aSSFf~~r~~di ← dCSF-FDI and CSFrdifdiEnable

NOTE 1 – X_{AR} = 0 results in AI_TSF being asserted, so there is no need to include it as additional contributor to aSSF.

Defect correlations

The function shall perform the following defect correlations to determine the most probable fault cause (see clause 6.4 of [ITU-T G.806]). This fault cause shall be reported to the EMF.

cVcPLM ← dVcPLM and (not AI_TSF);

cLFD ← dLFD and (not dVcPLM) and (not AI_TSF);

cCSF ← (dCSF-LOS or dCSF-OPU~~dCSF-RDI~~ or dCSF-FDI) and (not dEXM) and (not dUPM) and (not dPLM) and (not dLFD) and (not AI_TSF) and CSF_Reported

Performance monitoring

The function shall perform the following performance monitoring primitives processing. The performance monitoring primitives shall be reported to the EMF.

pFCSError: count of FrameCheckSequenceErrors per second.

NOTE 2 – This primitive is calculated by the MAC FCS Check process.

40) Clause 11.5.3.1

Revise clause 11.5.3.1 with respect to ODU Adaptation as follows:

11.5.3.1 ODU2P to Ethernet PP-OS adaptation source function (ODU2P/ETHPP-OS_A_So)

The ODU2P/ETHPP-OS_A_So function creates the ODU2P signal from a free running clock. It maps the ETHPP-OS_CI information into the payload of the OPU2P, adds OPU2P Overhead (RES, PT) and default ODU2P Overhead.

Symbol

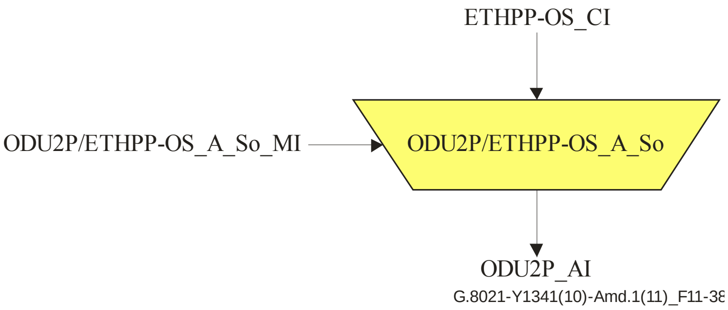


Figure 11-38 – ODU2P/ETHPP-OS_A_So symbol

Interfaces

Table 11-19 – ODU2P/ETHPP-OS_A_So interfaces

Inputs	Outputs
ETHPP-OS_CP: ETHPP-OS_CI_D ETHPP-OS_CI_SSF ODU2P/ETHPP-OS_A_So_MI: ODU2P/ETHPP-OS_A_So_MI_Active ODU2P/ETHPP-OS_A_So_MI_CSFEnable	ODU2P_AP: ODU2P_AI_Data ODU2P_AI_Clock ODU2P_AI_FrameStart ODU2P_AI_MultiframeStart

NOTE – ETHPP-OS_CI_D is composed of preamble, payload and order set information in [ITU-T G.7041].

Processes

A process diagram of this function is shown in Figure 11-39.

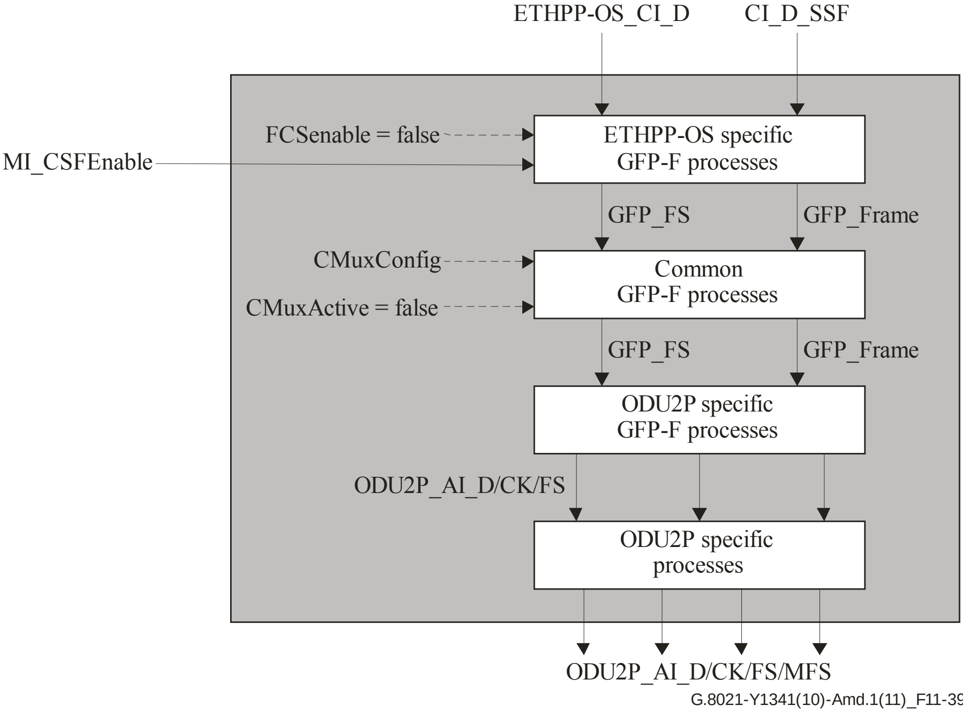


Figure 11-39 – ODU2P/ETHPP-OS_A_So process

Ethernet specific GFP-F source process:

The Ethernet frames are inserted into the client payload information field of the GFP-F frames according to clause 7.9.2 of [ITU-T G.7041].

The UPI values for frame-mapped Ethernet shall be inserted for data or Ordered Sets respectively. (Table 6-3 of [ITU-T G.7041]). The rest of the fields but UPI field in type header are static as:

- PTI = 000 (Client Data)
- PFI = 0 (No FCS)
- EXI = 0000 (Null Extension Header)

GFP client management frames (PTI = 100) are inserted if CI_SSF is input and GFP pFCS generation is disabled (FCSenable=false).

Common GFP source process:

See clause 8.5.3.1 of [ITU-T G.806]. GFP channel multiplexing is not supported (CMuxActive=false).

ODU2P specific GFP source process:

See clause 8.5.2.1 of [ITU-T G.806]. The GFP frames are mapped into the ODU2 payload area according to clause 17.3 of [ITU-T G.709]. OPU CSF may be generated if CI_SSF is input.

ODU2P specific source process:

See clause 11.5.1.1 (k=2).

Defects None.

Consequent actions aCSF-LOS ← CI_SSF and CSFEnable

aCSF-OPU ← CI_SSF and CSFEnable

Defect correlations	None.
Performance monitoring	For further study.

41) Clause 11.5.3.2

Revise clause 11.5.3.2 with respect to ODU Adaptation as follows:

11.5.3.2 ODU2P to Ethernet PP-OS adaptation sink function (ODU2P/ETHPP-OS_A_Sk)

The ODU2P/ETHPP-OS_A_Sk extracts ETHPP-OS_CI information from the ODU2P payload area, delivering ETHPP-OS_CI to ETHPP-OS_TCP. It extracts the OPU2P Overhead (PT and RES) and monitors the reception of the correct payload type.

Symbol

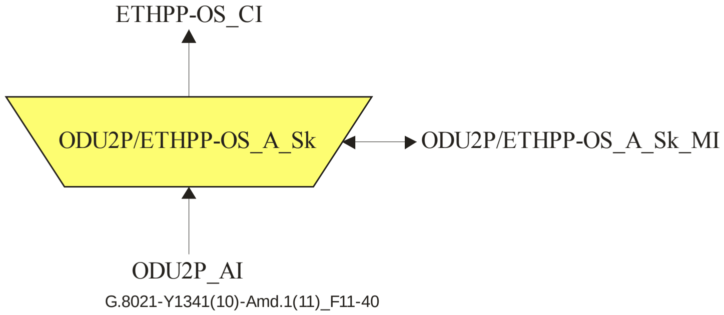


Figure 11-40 – ODU2P/ETHPP-OS_A_Sk symbol

Interfaces

Table 11-20 – ODU2P/ETHPP-OS_A_Sk interfaces

Inputs	Outputs
ODU2P_AP: ODU2P_AI_Data ODU2P_AI_ClocK ODU2P_AI_FrameStart ODU2P_AI_MultiframeStart ODU2P_AI_TSF ODU2P/ETHPP-OS_A_Sk_MP: ODU2P/ETHPP-OS_A_Sk_MI_Active ODU2P/ETHPP-OS_A_Sk_MI_CSF_Reported ~~ODU2P/ETHPP-OS_A_Sk_MI_MAC_Length~~	ETHPP-OS_CP: ETHPP-OS_CI_D ODU2P/ETHPP-OS_A_Sk_MP: ODU2P/ETHPP-OS_A_Sk_MI_AcPT ODU2P/ETHPP-OS_A_Sk_MI_AcEXI ODU2P/ETHPP-OS_A_Sk_MI_AcUPI ODU2P/ETHPP-OS_A_Sk_MI_cPLM ODU2P/ETHPP-OS_A_Sk_MI_cLFD ODU2P/ETHPP-OS_A_Sk_MI_cUPM ODU2P/ETHPP-OS_A_Sk_MI_cEXM ODU2P/ETHPP-OS_A_Sk_MI_cCSF ~~ODU2P/ETHPP-OS_A_Sk_MI_pFCSErrors~~ ~~ODU2P/ETHPP-OS_A_Sk_MI_AcSL~~ ~~ODU2P/ETHPP-OS_A_Sk_MI_AcPFI~~ ~~ODU2P/ETHPP-OS_A_Sk_MI_pCRC16Errors~~

Processes

A process diagram of this function is shown in Figure 11-41.

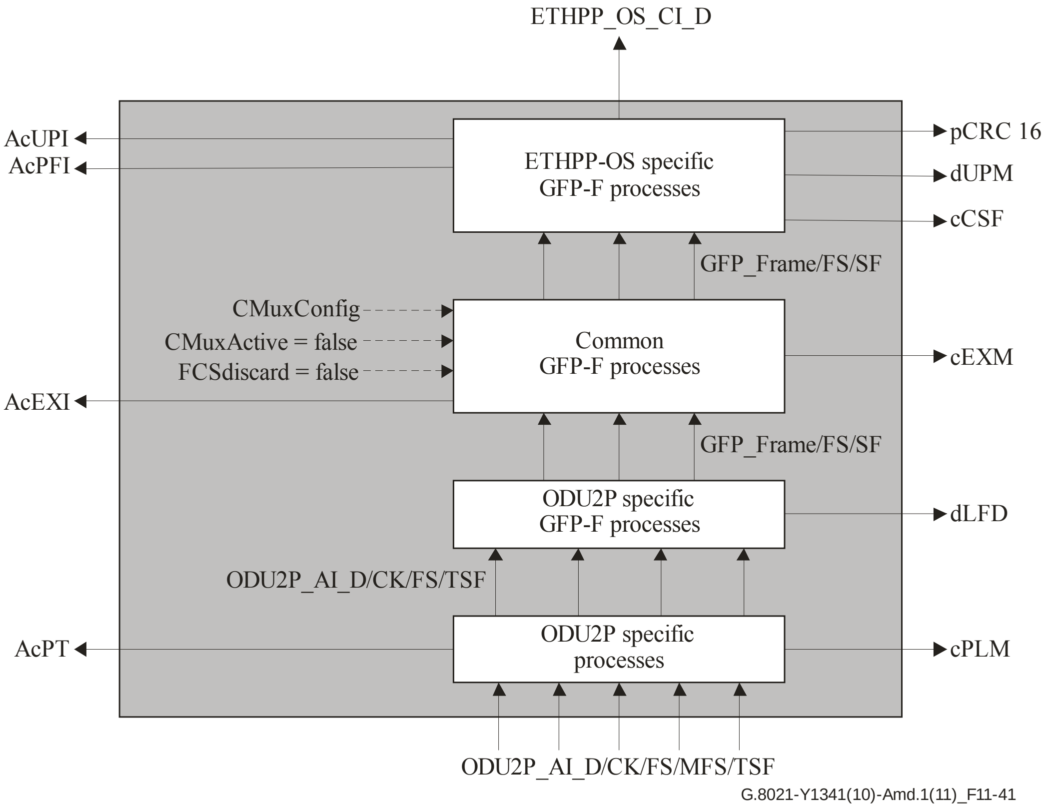


Figure 11-41 – ODU2P/ETHPP-OS_A_Sk process

Ethernet specific GFP-F sink process:

The Ethernet frames are extracted from the client payload information field of the GFP-F frames according to clause 7.9 of [ITU-T G.7041].

See clause 8.5.4.1.2 of [ITU-T G.806]. GFP pFCS checking, GFP p_FCSError, p_FDis are not supported (FCSdiscard=false). The UPI value for Frame-Mapped Ethernet shall be expected for data or Ordered Sets respectively (Table 6-3 of [ITU-T G.7041]).

Client signal fail from GFP-F or OPU may generate LF as included ETHPP-OS_CI_D.

Common GFP sink process:

See clause 8.5.3.2 of [ITU-T G.806]. GFP channel multiplexing is not supported (MI_CMuxActive=false).

ODU2 specific GFP sink process:

See clause 8.5.2.2 of [ITU-T G.806]. The GFP frames are demapped from the ODU2 payload area according to clause 17.4.1 of [ITU-T G.709].

ODU2P specific sink process:

See clause 11.5.1.2 (k=2).

Defects

dPLM – See clause 6.2.4.1 of [ITU-T G.798].

dLFD – See clause 6.2.5.2 of [ITU-T G.806].

dUPM – See clause 6.2.4.3 of [ITU-T G.806].

dEXM – See clause 6.2.4.4 of [ITU-T G.806].

dCSF-LOS – See clause 8.8.6.2.

dCSF-OPU – For further study.

Consequent actions

The function shall perform the following consequent actions:

aSSF ← AI_TSF or dPLM or dLFD or dUPM or dEXM or dCSF-LOS~~ or dCSF-OPU~~

Defect correlations

The function shall perform the following defect correlations to determine the most probable fault cause (see clause 6.4 of [ITU-T G.806]). This fault cause shall be reported to the EMF.

cPLM ← dPLM and (not AI_TSF);

cLFD ← dLFD and (not dPLM) and (not AI_TSF);

cUPM ← dUPM and (not dEXM) and (not dPLM) and (not dLFD) and (not AI_TSF);

cEXM ← dEXM and (not dPLM) and (not dLFD) and (not AI_TSF)

cCSF ← (dCSF-LOS or dCSF-OPU) and (not dEXM) and (not dUPM) and (not dPLM) and (not dLFD) and (not AI_TSF) and CSF_Reported

Performance monitoring

For further study.

~~The function shall perform the following performance monitoring primitives processing. The performance monitoring primitives shall be reported to the EMF.~~

~~pFCSErrors: count of FrameCheckSequenceErrors per second.~~

~~NOTE – This primitive is calculated by the MAC FCS Check process.~~

42) Renumbering and revision of clause 11.5.5

Renumber clause 11.5.5 to 11.5.4 and update clause 11.5.4 with respect to ODU Adaptation as shown:

11.5.4 ODU0P to 1 GbE client adaptation functions (ODU0P/CBRx_A)

The adaptation function that supports the transport of 1GbE signals in the OTN is the ODU0P to Client adaptation function (ODU0P/CBRx_A) (0≤x≤1.25G) described in [ITU-T G.798]. When the client is 1 GbE, the CBRx and ETC3 signals are equivalent; as such the ETY3/ETC3_A functions are bound to the ODU0P/CBRx_A functions.

43) Deletion of clause 11.5.6

Remove clause 11.5.6, ETY3 to 1-GbE Client Adaptation Functions (ETY3/CBRx_A)

~~11.5.6 ETY3 to 1 GbE client adaptation functions (ETY3/CBRx _A)~~

~~For Further Study.~~

44) New Appendix VIII

Add the following new Appendix VIII related to VID filtering:

Appendix VIII

Configurations for ingress VID filtering

(This appendix does not form an integral part of this Recommendation.)

This appendix describes an example of the configuration for ingress VID filtering described in [IEEE 802.1Q].

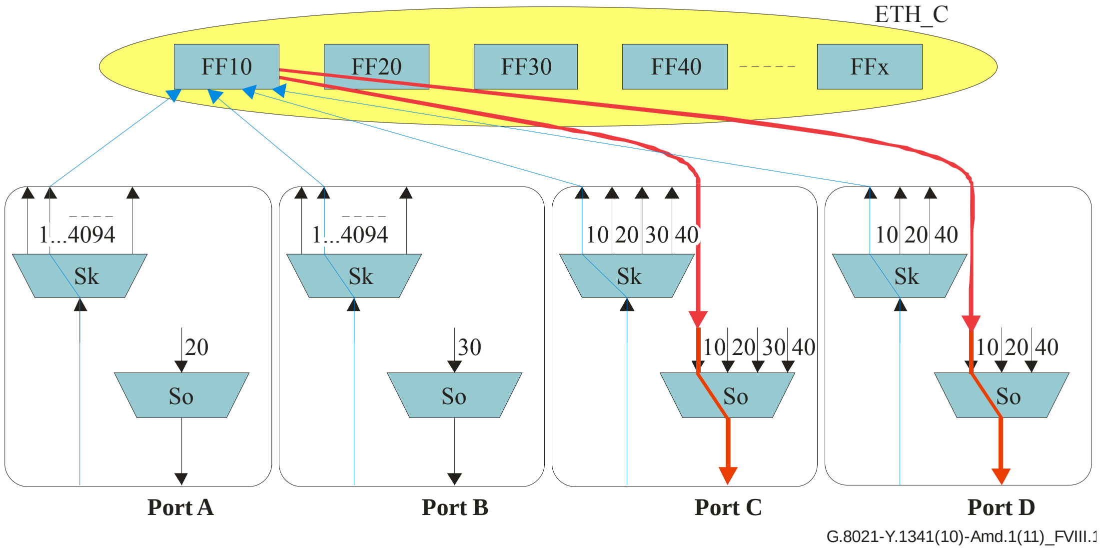


Figure VIII.1 – Example of configuration for ingress VID filtering

Table VIII-1 – VID Configuration

VID	Port A		Port B		Port C		Port D	
	Ingress	Egress	Ingress	Egress	Ingress	Egress	Ingress	Egress
10	✓		✓		✓	✓	✓	✓
20	✓	✓	✓		✓	✓	✓	✓
30	✓		✓	✓	✓	✓		
40	✓		✓		✓	✓	✓	✓
Others	✓		✓					

Figure VIII.1 and Table VIII-1 show an example of the configuration. For the ingress configuration, MI_Vlan_Config[] signal is set to ETHx/ETH-m_A_Sk function and ETH_CI signals corresponding VIDs are connected to ETH_FF processes in ETH_C function. For the egress configuration, MI_Vlan_Config[] signal is set to ETHx/ETH-m_A_So function and ETH_CI signals corresponding VIDs are connected to ETH_FF processes in ETH_C function.

On ports A and B in this example, MI_Vlan_Config[1…4094] are set to ETHx/ETH-m_A_Sk in order to disable Ingress VID filtering. In this case, all incoming VIDs traffic is forwarded to ETH_C. Since ETH_FF is connected to configured ingress and egress ports only, the traffic is forwarded to the appropriate ports.

45) Addition of MI_Active signals to various tables

Add the following MI_Active input signals to the respective table to disable/enable the whole features of its adaptation function.

- ETHx/ETH_A_So_MI_Active (Table 9-6)
- ETHx/ETH_A_Sk_MI_Active (Table 9-7)
- ETHx/ETH-m_A_So_MI_Active (Table 9-8)
- ETHx/ETH-m_A_Sk_MI_Active (Table 9-9)
- ETHG/ETH_A_So_MI_Active (Table 9-10)
- ETHG/ETH_A_Sk_MI_Active (Table 9-11)
- ETHx/ETHG_A_So_MI_Active (Table 9-12)
- ETHx/ETHG_A_Sk_MI_Active (Table 9-13)
- ETHD/ETH_A_So_MI_Active (Table 9-18)
- ETHD/ETH_A_Sk_MI_Active (Table 9-19)
- ETHDi/ETH_A_So_MI_Active (Table 9-20)
- ETHDi/ETH_A_Sk_MI_Active (Table 9-21)
- ETY-Np/ETH-LAG-Na_A_So_MI_Active (Table 9-25)
- ETY-Np/ETH-LAG-Na_A_Sk_MI_Active (Table 9-26)
- ETH-LAG/ETH_A_So_MI_Active (Table 9-29)
- ETH-LAG/ETH_A_Sk_MI_Active (Table 9-30)
- ETYn/ETH_A_So_MI_Active (Table 10-4)
- ETYn/ETH_A_Sk_MI_Active (Table 10-5)
- ETY3/ETC3_A_So_MI_Active (Table 10-6)
- ETY3/ETC3_A_Sk_MI_Active (Table 10-7)
- Sn/ETH_A_So_MI_Active (Table 11-1)
- Sn/ETH_A_Sk_MI_Active (Table 11-2)
- Sn-X-L/ETH_A_So_MI_Active (Table 11-3)
- Sn-X-L/ETH_A_Sk_MI_Active (Table 11-4)
- Sm/ETH_A_So_MI_Active (Table 11-5)
- Sm/ETH_A_Sk_MI_Active (Table 11-6)
- Sm-X-L/ETH_A_So_MI_Active (Table 11-7)
- Sm-X-L/ETH_A_Sk_MI_Active (Table 11-8)
- Sn-X/ETC3_A_So_MI_Active (Table 11-9)
- Sn-X/ETC3_A_Sk_MI_Active (Table 11-10)
- Pq/ETH_A_So_MI_Active (Table 11-11)
- Pq/ETH_A_Sk_MI_Active (Table 11-12)
- Pq-X-L/ETH_A_So_MI_Active (Table 11-13)
- Pq-X-L/ETH_A_Sk_MI_Active (Table 11-14)

46) Addition of the MI_MAC_Length signal to various tables

Add the following MI_MAC_Length input signals described in clause 8.6 of this Recommendation to the respective table, as indicated.

- Sn-X-L/ETH_A_Sk_MI_MAC_Length (Table 11-4)
- Sm/ETH_A_Sk_MI_MAC_Length (Table 11-6)
- Sm-X-L/ETH_A_Sk_MI_MAC_Length (Table 11-8)
- Pq/ETH_A_Sk_MI_MAC_Length (Table 11-12)
- Pq-X-L/ETH_A_Sk_MI_MAC_Length (Table 11-14)
- ODUkP-X-L/ETH_A_Sk_MI_MAC_Length (Table 11-18)

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
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	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Numbering, naming and addressing	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Smart ubiquitous networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
Future networks	Y.3000–Y.3099

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	General tariff principles
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	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	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
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 and next-generation networks**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2012