TSG-RAN Meeting #11 *RP-010026*
Palm Springs, CA, USA, 13 - 16 March 2001

Title: **Agreed CRs (Release '99) to TS 25.322**

Source: **TSG-RAN WG2**

Agenda item: **5.2.3**

Doc-1st-	Status-	Spec	CR	Rev	Phase	Subject	Cat	Version	Versio
R2-010647	agreed	25.322	097	1	R99	Clarification on LIST SUFI and RLIST SUFI	F	3.5.0	3.6.0
R2-010611	agreed	25.322	098	1	R99	Corrections and clarifications for SDU discard without explicit signalling	F	3.5.0	3.6.0
R2-010613	agreed	25.322	099	1	R99	Tr mode operation	F	3.5.0	3.6.0
R2-010612	agreed	25.322	100	1	R99	Timer based discard with explicit signalling	F	3.5.0	3.6.0
R2-010342	agreed	25.322	101		R99	Annex updates	F	3.5.0	3.6.0
R2-010351	agreed	25.322	103		R99	Clarification on MRW SUFI and SDU discard procedure	F	3.5.0	3.6.0
R2-010618	agreed	25.322	104	1	R99	General clarification on SN arithmetic comparison	F	3.5.0	3.6.0
R2-010682	agreed	25.322	105	2	R99	General clarification on RLC header and PDU header	F	3.5.0	3.6.0
R2-010615	agreed	25.322	106	1	R99	Clarification on the primitives between RLC and higher layers	F	3.5.0	3.6.0
R2-010616	agreed	25.322	107	1	R99	Clarification on the model of AM entity	F	3.5.0	3.6.0
R2-010683	agreed	25.322	109	2	R99	Clarification on UMD transfer procedure	F	3.5.0	3.6.0
R2-010617	agreed	25.322	110	1	R99	RLC status transmission in CELL_PCH and URA_PCH	F	3.5.0	3.6.0
R2-010376	agreed	25.322	111		R99	Re-establishment description	F	3.5.0	3.6.0
R2-010619	agreed	25.322	112	1	R99	Clarifications on the RESET and RESET ACK PDU sizes	F	3.5.0	3.6.0
R2-010620	agreed	25.322	113	1	R99	Editorial corrections and clarifications	F	3.5.0	3.6.0
R2-010655	agreed	25.322	114	1	R99	Clarifications on the RLC-AM-DATA-Conf primitive	F	3.5.0	3.6.0
R2-010449	agreed	25.322	116		R99	Removal of the payload unit concept	F	3.5.0	3.6.0
R2-010721	agreed	25.322	118	2	R99	Padding Blocks and TFC selection pre-empting	F	3.5.0	3.6.0

3GPP TSG-RAN2 Meeting #19 ***Tdoc R2-010647***

Sophia Antipolis, France, 19-23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **097** ⌘ rev **r1** ⌘ Curent version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarification on LIST SUFI and RLIST SUFI
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 13 February 2001
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	Possible ambiguity in the acknowledgment status for the in-between PDUs in a LIST SUFI and RLIST SUFI
Summary of change:	⌘	LIST SUFI and RLIST SUFI are used to indicate missing PDUs only
Consequences if not approved:	⌘	Interpretation differences in the meaning of LIST SUFI and RLIST SUFI

Clauses affected:	⌘	11.5.2.2
Other specs affected:	⌘	☐ Other core specifications ⌘ ☐ Test specifications ☐ O&M Specifications
Other comments:	⌘	

11.5.2.2 STATUS PDU contents to set

The size of the STATUS PDU shall be equal to one of the allowed PDU sizes. The information that needs to be transmitted in a status report can be split into several STATUS PDUs if one STATUS PDU does not accommodate all the information. A SUFI cannot be split into several STATUS PDUs. Indication of the same PU shall not be given in more than one STATUS PDU of a STATUS report, but the ACK SUFI can be present in more than one STATUS PDU of a status report.

Which SUFI fields to use is implementation dependent, but the status report shall include information about PUs that have been received and information about all PUs detected as missing. Bitmap SUFI is used to indicate both received and/or missing PDUs. List SUFI and/or Relative List SUFI are used to indicate missing PDUs only. Acknowledgement SUFI is used to indicate the received PDUs. (For SUFI details see 9.2.2.11.) No information shall be given for PUs with SN≥VR(H), i.e. PUs that have not yet reached the receiver.

Padding shall be inserted if the SUFI fields do not fill an entire STATUS PDU. If the PDU contains padding the last SUFI field shall be either an ACK SUFI or a NO_MORE SUFI. If there is no padding in the STATUS PDU, NO_MORE SUFI or ACK SUFI does not need to be included in the STATUS PDU.

3GPP TSG-RAN2 Meeting #19 ***Tdoc R2-010611***

Sophia Antipolis, France, 19-23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **098** ⌘ rev **r1** ⌘ Curent version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Corrections and clarifications for SDU discard without explicit signalling
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 21 February 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change: ⌘

1- The starting instance for the Timer_Discard timer is not indicated in the UM and TrM
2- Inconsistency between 25.322 and 25.331 for the use of Special LI

Summary of change: ⌘

1- Timer_Discard timer is started when a SDU is received from the upper layer for UM and TrM.

2- The LI dedicated for “Use special LI” will not be used as a tool in the course of SDU discard without explicit signaling in UM mode

Consequences if not approved: ⌘

1- A component of a procedure not defined within the text describing the procedure.

2- The specifications under the responsibility of RAN2 will include contradictory information

Clauses affected: ⌘ 11.1.2, 11.2.2, 11.2.4.3

Other specs affected: ⌘
☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘

9.5 Timers

a) Timer_Poll.

This timer is only used when the poll timer trigger is used. It is started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The timer is stopped when receiving a STATUS PDU that contains an acknowledgement of all AMD PDUs with SN up to and including VT(S)-1 at the time the poll was submitted to lower layer,or when a negative acknowledgement of the same PU is received. The value of the timer is signalled by RRC.

If the timer expires and no STATUS PDU fulfilling the criteria above has been received, the receiver is polled once more (either by the transmission of a PDU which was not yet sent, or by a retransmission) and the timer is restarted at the time specified above, with a new value of VT(S)-1.

If a new poll is sent when the timer is running the timer is restarted at the time specified above, with a new value of VT(S)-1.

b) Timer_Poll_Prohibit.

This timer is only used when the poll prohibit function is used. It is used to prohibit transmission of polls within a certain period. The timer shall be started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time a PDU containing a poll is submitted to lower layer until the timer has expired. A poll shall be delayed until the prohibit time expires if a poll is triggered during the prohibit time. Only one poll shall be transmitted when the prohibit time expires even if several polls were triggered during the prohibit time. This timer will not be stopped by a received STATUS PDU. The value of the timer is signalled by RRC.

c) Timer_EPC.

This timer is only used when the EPC function is used and it accounts for the roundtrip delay, i.e. the time when the first retransmitted PU should be received after a status report has been sent. The timer is started when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN) and when it expires VR(EP) can start its counting-down process (see subclause 9.7.4). The value of the timer is signalled by RRC.

d) Timer_Discard.

This timer is used for the SDU discard function. In the transmitter, the timer is activated upon reception of a SDU from higher layer. One timer is used for each SDU that is received from higher layer. If the SDU has not been acknowledged and/or transmitted when the timer expires, the SDU is discarded. Following which, if the SDU discard function uses explicit signalling, a Move Receiving Window request is sent to the receiver. The value of the timer is signalled by RRC.

e) Timer_Poll_Periodic.

This timer is only used when the timer based polling is used. The timer is started when the RLC entity is created. Each time the timer expires, the timer is restarted and a poll is triggered (either by the transmission of a PDU which was not yet sent, or by a retransmission). If there is no PU to be transmitted and all PUs have already been acknowledged, a poll shall not be triggered and the timer shall only be restarted. The value of the timer is signalled by RRC.

f) Timer_Status_Prohibit.

This timer is only used when the STATUS prohibit function is used. It prohibits the receiving side from sending status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. The timer is started when the successful or unsuccessful transmission of the last STATUS PDU in a status report is indicated by lower layer (in UE) or the last STATUS PDU in a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired and no new status report containing the mentioned SUFIs can be transmitted during the prohibit time. The

timer does not prohibit transmission of the SUFIs MRW, MRW_ACK, WINDOW or NO_MORE. The value of the timer is signalled by RRC.

g) Timer_Status_Periodic.

This timer is only used when timer based status report sending is used. The timer is started when the RLC entity is created. Each time the timer expires the transmission of a status report is triggered and the timer is restarted. The value of the timer is signalled by RRC.

h) Timer_RST.

This timer is used to detect the loss of RESET ACK PDU from the peer RLC entity. This timer is started when the successful or unsuccessful transmission of a RESET PDU is indicated by lower layer (in UE) or a RESET PDU is submitted to lower layer (in UTRAN). It will only be stopped upon reception of RESET ACK PDU, i.e. this timer is not stopped when an RLC reset occurs which was initiated from the peer RLC entity. If it expires, RESET PDU will be retransmitted. The value of the timer is signalled by RRC.

i) Timer_MRW.

This timer is used as part of the Move Receiving Window protocol. It is used to trigger the retransmission of a status report containing an MRW SUFI field. The timer is started when the successful or unsuccessful transmission of a STATUS PDU containing the MRW SUFI is indicated by lower layer (in UE) or a STATUS PDU containing the MRW SUFI is submitted to lower layer (in UTRAN). Each time the timer expires the MRW SUFI is retransmitted and the timer is restarted (at the time specified above). It shall be stopped when one of the termination criteria for the SDU discard is fulfilled. The value of the timer is signalled by RRC.

.

.

11.1.2 Initiation

The sender initiates this procedure upon a request of transparent mode data transfer from higher layer. When the sender is in data transfer ready state it shall put the data received from the higher layer into TrD PDUs. If required RLC shall perform segmentation.

Channels that can be used are DTCH, CCCH (uplink only), SHCCH (uplink only), BCCH and PCCH. The type of logical channel depends on if the RLC entity is located in the user plane (DTCH) or in the control plane (CCCH/BCCH/SHCCH/PCCH). One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can not be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

If timer based SDU discard is used, the timer Timer_Discard shall be started when the RLC entity receives an SDU from higher layer. One timer is used for each SDU that is received from higher layer.

11.2.2 Initiation

The sender initiates this procedure upon a request of unacknowledged mode data transfer from higher layer.

When the sender is in data transfer ready state it shall segment the data received from the higher layer into PDUs.

Channels that can be used are DTCH, DCCH, CCCH (downlink only), CTCH, SHCCH (downlink only). The type of logical channel depends on if the RLC entity is located in the user plane (DTCH, CTCH) or in the control plane (DCCH/CCCH(downlink only)/SHCCH(downlink only)). One or several PDUs may be transmitted in each transmission time interval (TTI). For each TTI, MAC decides which PDU size shall be used and how many PDUs shall be transmitted.

The VT(US) state variable shall be updated for each UMD PDU that is transmitted.

If timer based SDU discard is used, the timer Timer_Discard shall be started when the RLC entity receives an SDU from higher layer. One timer is used for each SDU that is received from higher layer.

11.2.4.3 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard the associated SDU. The next UMD PDU shall carry the first segment of the oldest SDU not discarded. The state variable VT(US) shall be updated so that the receiver can detect at least one missing PDUs. To ~~avoid that~~ prevent the receiver ~~should~~ from discarding one extra SDU, a LI field shall be added in the first PDU transmitted after a Discard Operation. The value of the LI field shall be ~~either~~ the value indicating that the previous SDU filled exactly the previous RLC PDU ~~or the value indicating that the first data octet in this RLC PDU is the first octet of a RLC SDU~~.

.

3GPP TSG-RAN WG2 Meeting #19
Sophia Antipolis, France, 19 - 23 Feb. 2001

R2-010613

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **099** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Tr mode operation
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 22 Feb. 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change:	⌘	The current RLC Tr model does not include the fact that multiple PDU size might be used. Some modifications are needed.
Summary of change:	⌘	The locations of 'Segmentation' and 'Transmission buffer' in RLC Tr model are exchanged.
Consequences if not approved:	⌘	Multiple PDU size can not be used in RLC Tr mode

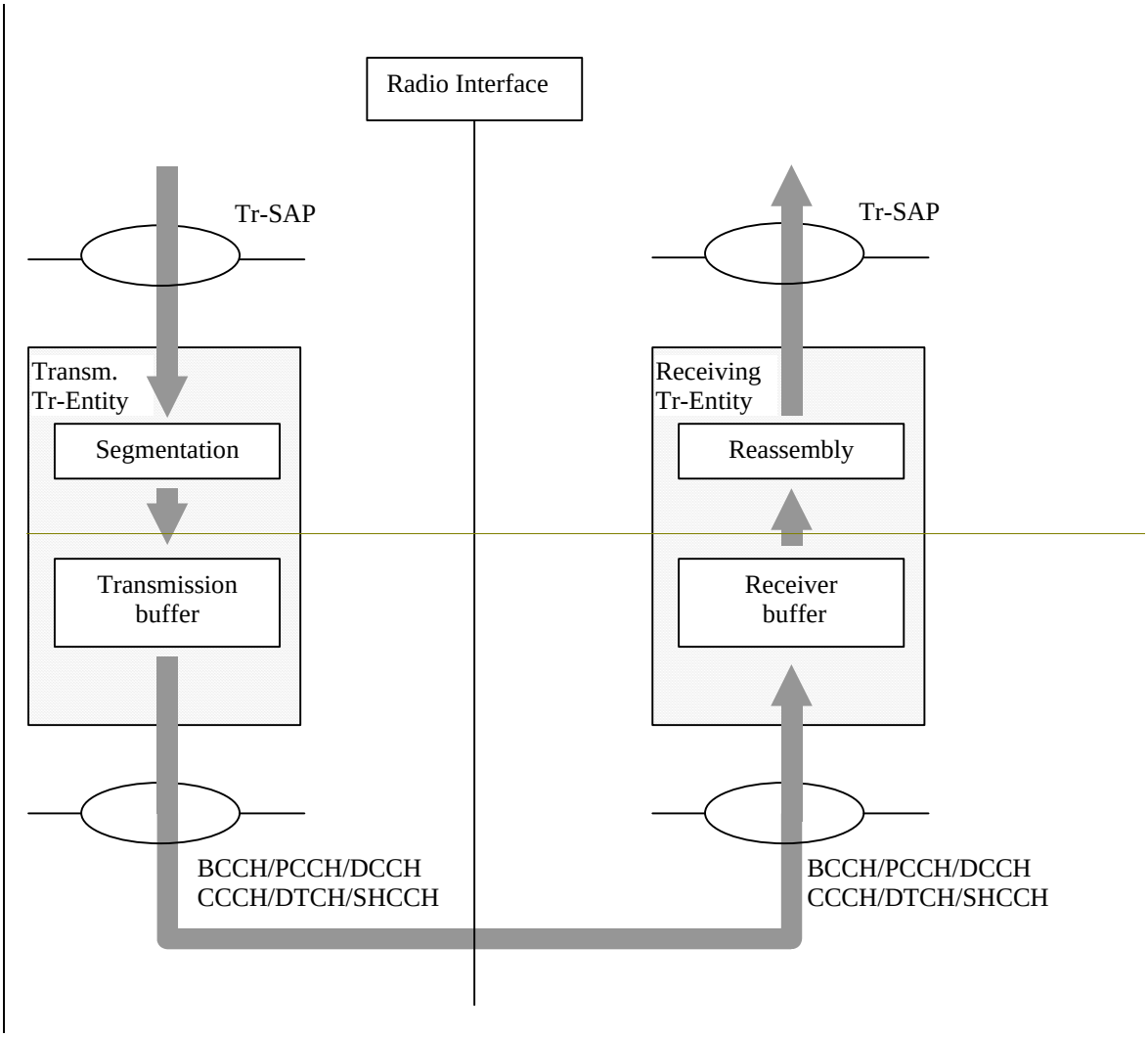
Clauses affected:	⌘	4.2.1.1, 11.1.2, 11.1.4.2		
Other specs Affected:	⌘	☐ Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.
2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.
3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

4.2.1.1 Transparent mode entities

Figure 4.2 below shows the model of two transparent mode peer entities.



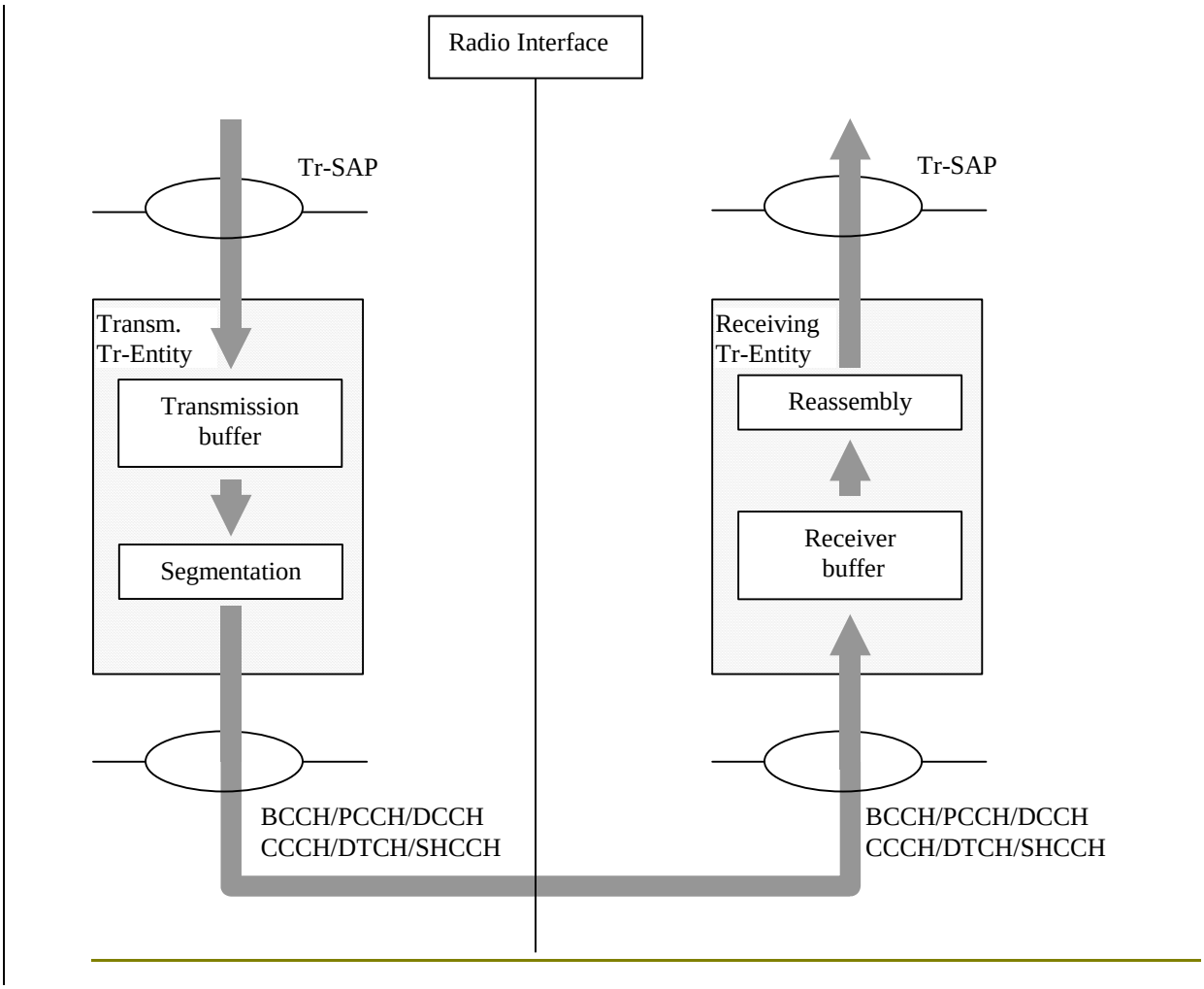


Figure 4.2: Model of two transparent mode peer entities

The transmitting Tr-entity receives SDUs from the higher layers through the Tr-SAP. RLC might segment the SDUs into appropriate RLC PDUs without adding any overhead. How to perform the segmentation is decided upon when the service is established. RLC delivers the RLC PDUs to MAC through either a BCCH, DCCH, PCCH, SHCCH or a DTCH. The CCCH and SHCCH also uses transparent mode, but only for the uplink. Which type of logical channel depends on if the higher layer is located in the control plane (BCCH, DCCH, PCCH, CCCH, SHCCH) or user plane (DTCH).

The Tr-entity receives PDUs through one of the logical channels from the MAC sublayer. RLC reassembles (if segmentation has been performed) the PDUs into RLC SDUs. How to perform the reassembling is decided upon when the service is established. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

11.1 Transparent mode data transfer procedure

11.1.1 Purpose

The transparent mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in transparent mode. Figure 11.1 below illustrates the elementary procedure for transparent mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.1: Transparent mode data transfer procedure

11.1.2 Initiation

The sender initiates this procedure upon a request of transparent mode data transfer from higher layer. When the sender is in data transfer ready state it shall put the data received from the higher layer into TrD PDUs. If required RLC shall perform segmentation.

Channels that can be used are DTCH, CCCH (uplink only), SHCCH (uplink only), BCCH and PCCH. The type of logical channel depends on if the RLC entity is located in the user plane (DTCH) or in the control plane (CCCH/BCCH/SHCCH/PCCH). One or several PDUs may be transmitted in each transmission time interval (TTI). ~~and~~ For each TTI, MAC decides which PDU size shall be used (applicable when segmentation is used) and how many PDUs shall be transmitted ~~in each TTI~~. ~~In the UE, t~~The ~~P~~SDUs that can not be transmitted in a TTI ~~(i.e. MAC has indicated that some of the available PDUs can not be transmitted)~~ shall be buffered according to the discard configuration set by RRC.

11.1.2.1 TrD PDU contents to set

The TrD PDU includes a complete SDU or a segment of an SDU. How to perform the segmentation is decided upon when the service is established. No overhead or header is added, instead segmentation is done based on which of the transport formats of the transport channel that will be used. A particular transport format informs the receiver how the segmentation was performed.

11.1.3 Reception of TrD PDU

Upon reception of a TrD PDU, the receiving entity reassembles (if segmentation was performed) the PDUs into RLC SDUs. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

11.1.4 Abnormal cases

11.1.4.1 Undefined SDU size at receiver

If the TrD PDUs are reassembled to a SDU which have a size that is not allowed the SDU shall be discarded.

11.1.4.2 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard ~~all PDUs that contain segments of~~ the associated SDU.

3GPP TSG-RAN WG2 Meeting #19
Sophia Antipolis, France, 19 - 23 Feb. 2001

R2-010612

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **100** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Timer based discard with explicit signalling
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 22 Feb. 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change: ⌘

Current timer based discard with explicit signalling procedure is inefficient for radio resource perspective. Suppose that Tx receives many SDUs (SDU1 ~ SDU100) from upper layer almost at the same time, and only a few SDUs are transmitted to the receiver (SDU1~SDU4). At this time, if Timer_discard expires, Tx should send many MRW_SUFIs to the receiver, even though it does not transmit the corresponding SDUs. It wastes radio resource much, and the procedure needs to be modified.

The main problem with this modification is that if AM RLC entity is connected to a PDCP layer which supports lossless SRNS relocation, the synchronization of sequence numbers between peer PDCP entities is not guaranteed.

So it is proposed that this modification should be applied only for an AM RLC entity which has no connection with a PDCP that supports lossless SRNS relocation. To do this, it is also proposed that a new parameter be added in CRLC-CONFIG-Req primitive.

Summary of change: ⌘

1. A new parameter Lossless_PDCP is added in CRLC_CONFIG_Req. This parameter indicates that the AM RLC entity is connected to a PDCP layer which supports lossless SRNS relocation.
2. The expired SDU whose segments have been transmitted is discarded in the transmitter, and a corresponding MRW SUFI is sent to the receiver.
3. The expired SDU whose segments have not been transmitted is discarded in the transmitter, and a corresponding MRW SUFI is sent to the receiver only if the AM RLC entity is connected to a PDCP layer which supports lossless SRNS relocation.

Consequences if not approved: ⌘

If Tx AM RLC entity receives many SDUs, and if they are timed out, many MRW_SUFI shall be sent to the receiver even if the SDUs are not actually transmitted to the receiver. It wastes radio resource much.

Clauses affected:	⌘	8.2, 9.5, 9.7.3.1, 11.6.1, 11.6.2		
Other specs Affected:	⌘	Other core specifications Test specifications O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

8 Elements for layer-to-layer communication

The interaction between the RLC layer and other layers are described in terms of primitives where the primitives represent the logical exchange of information and control between the RLC layer and other layers. The primitives shall not specify or constrain implementations.

8.1 Primitives between RLC and higher layers

The primitives between RLC and upper layers are shown in Table 8.1.

Table 8.1: Primitives between RLC and upper layers

Generic Name	Parameter			
	Req.	**Ind.**	**Resp.**	**Conf.**
RLC-AM-DATA	Data, CNF, MUI	Data, DiscardInfo	Not Defined	MUI
RLC-UM-DATA	Data, Use special LI	Data	Not Defined	Not Defined
RLC-TR-DATA	Data	Data	Not Defined	Not Defined
CRLC-CONFIG	E/R, Stop,Continue, Ciphering Elements (UM/AM only), AM_parameters (AM only)	Not Defined	Not Defined	Not Defined
CRLC-SUSPEND (UM/AM only)	N	Not Defined	Not Defined	VT(US) (UM only), VT(S) (AM only)
CRLC-RESUME (UM/AM only)	No Parameter	Not Defined	Not Defined	Not Defined
CRLC-STATUS	Not Defined	EVC	Not Defined	Not Defined

Each Primitive is defined as follows:

RLC-AM-DATA-Req/Ind/Conf

- RLC-AM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in acknowledged mode.
- RLC-AM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in acknowledged mode and to indicate higher layers of the discarded RLC SDU in the receiving RLC.
- RLC-AM-DATA-Conf is used by RLC to confirm to higher layers the transmission of a RLC SDU.

RLC-UM-DATA-Req/Ind

- RLC-UM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in unacknowledged mode.
- RLC-UM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in unacknowledged mode.

RLC-TR-DATA-Req/Ind

- RLC-TR-DATA-Req is used by higher layers to request transmission of a higher layer PDU in transparent mode.
- RLC-TR-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in transparent mode.

CRLC-CONFIG-Req

This primitive is used by RRC to establish, re-establish, release, stop, continue or reconfigure the RLC. Ciphering elements are included for UM and AM operation.

CRLC-SUSPEND-Req/Conf

This primitive is used by RRC to suspend the RLC. The N parameter indicates that RLC shall not send a PDU with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM, where N is an integer. RLC informs RRC of the VT(S) for AM and VT(US) for UM in the confirm primitive.

CRLC-RESUME-Req

This primitive is used by RRC to resume RLC when RLC has been suspended.

CRLC-STATUS-Ind

It is used by the RLC to send status information to RRC.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the correct transmission of the RLC SDU.

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC If it indicates (re-)establishment, the state variables in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state. If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the PU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (i.e., unrecoverable errors such as data link layer loss or recoverable status events such as reset, etc.).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Activation Time (SN to activate a new ciphering configuration) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains PU size, Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), SDU discard mode (see subclause 9.7.3), ~~and~~ Minimum WSN (see subclause 9.2.2.11.3), and Send MRW. The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set. The Send MRW indicates that the MRW SUFI shall be sent to the receiver even if no segments of the expired SDU were submitted to a lower layer.

8) The parameter DiscardInfo indicates the upper layer of each of the discarded RLC SDU. It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly or one octet short (only when 15 bit LI is used) in the end of the previous RLC PDU is present, the LI shall not be used.

9.5 Timers

a) Timer_Poll.

This timer is only used when the poll timer trigger is used. It is started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The timer is stopped when receiving a STATUS PDU that contains an acknowledgement of all AMD PDUs with SN up to and including VT(S)-1 at the time the poll was submitted to lower layer,or when a negative acknowledgement of the same PU is received. The value of the timer is signalled by RRC.

If the timer expires and no STATUS PDU fulfilling the criteria above has been received, the receiver is polled once more (either by the transmission of a PDU which was not yet sent, or by a retransmission) and the timer is restarted at the time specified above, with a new value of VT(S)-1.

If a new poll is sent when the timer is running the timer is restarted at the time specified above, with a new value of VT(S)-1.

b) Timer_Poll_Prohibit.

This timer is only used when the poll prohibit function is used. It is used to prohibit transmission of polls within a certain period. The timer shall be started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time a PDU containing a poll is submitted to lower layer until the timer has expired. A poll shall be delayed until the prohibit time expires if a poll is triggered during the prohibit time. Only one poll shall be transmitted when the prohibit time expires even if several polls were triggered during the prohibit time. This timer will not be stopped by a received STATUS PDU. The value of the timer is signalled by RRC.

c) Timer_EPC.

This timer is only used when the EPC function is used and it accounts for the roundtrip delay, i.e. the time when the first retransmitted PU should be received after a status report has been sent. The timer is started when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN) and when it expires VR(EP) can start its counting-down process (see subclause 9.7.4). The value of the timer is signalled by RRC.

d) Timer_Discard.

This timer is used for the SDU discard function. In the transmitter, the timer is activated upon reception of a SDU from higher layer. One timer is used for each SDU that is received from higher layer. ~~If the SDU has not been acknowledged and/or transmitted when the timer expires, the SDU is discarded.~~ For UM/Tr, if the timer expires before the SDU is submitted to a lower layer, “SDU discard without explicit signalling” specified in sections 11.2.4.3/11.1.4.2 shall be started. For AM, if the timer expires before the SDU is acknowledged, “SDU discard with explicit signalling” specified in section 11.6 shall be started. ~~Following which, if the SDU discard function uses explicit signalling, a Move Receiving Window request is sent to the receiver. The value of the timer is signalled by RRC.~~

e) Timer_Poll_Periodic.

This timer is only used when the timer based polling is used. The timer is started when the RLC entity is created. Each time the timer expires, the timer is restarted and a poll is triggered (either by the transmission of a PDU which was not yet sent, or by a retransmission). If there is no PU to be transmitted and all PUs have already been acknowledged, a poll shall not be triggered and the timer shall only be restarted. The value of the timer is signalled by RRC.

f) Timer_Status_Prohibit.

This timer is only used when the STATUS prohibit function is used. It prohibits the receiving side from sending status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. The timer is started when the successful or unsuccessful transmission of the last STATUS PDU in a status report is indicated by lower layer (in UE) or the last STATUS PDU in a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired and no new status report containing the mentioned SUFIs can be transmitted during the prohibit time. The timer does not prohibit transmission of the SUFIs MRW, MRW_ACK, WINDOW or NO_MORE. The value of the timer is signalled by RRC.

g) Timer_Status_Periodic.

This timer is only used when timer based status report sending is used. The timer is started when the RLC entity is created. Each time the timer expires the transmission of a status report is triggered and the timer is restarted. The value of the timer is signalled by RRC.

h) Timer_RST.

This timer is used to detect the loss of RESET ACK PDU from the peer RLC entity. This timer is started when the successful or unsuccessful transmission of a RESET PDU is indicated by lower layer (in UE) or a RESET PDU is submitted to lower layer (in UTRAN). It will only be stopped upon reception of RESET ACK PDU, i.e. this timer is not stopped when an RLC reset occurs which was initiated from the peer RLC entity. If it expires, RESET PDU will be retransmitted. The value of the timer is signalled by RRC.

i) Timer_MRW.

This timer is used as part of the Move Receiving Window protocol. It is used to trigger the retransmission of a status report containing an MRW SUFI field. The timer is started when the successful or unsuccessful transmission of a STATUS PDU containing the MRW SUFI is indicated by lower layer (in UE) or a STATUS PDU containing the MRW SUFI is submitted to lower layer (in UTRAN). Each time the timer expires the MRW SUFI is retransmitted and the timer is restarted (at the time specified above). It shall be stopped when one of the termination criteria for the SDU discard is fulfilled. The value of the timer is signalled by RRC.

9.7.3 SDU discard function for acknowledged, unacknowledged, and transparent mode

The SDU discard function allows to discharge RLC PDU from the buffer on the transmitter side, when the transmission of the RLC PDU does not success for a long time. The SDU discard function allows to avoid buffer overflow. There will be several alternative operation modes of the RLC SDU discard function, and which discard function to use will be given by the QoS requirements of the Radio Access Bearer.

The following is a list of operation modes for the RLC SDU discard function.

Table 9.2: List of criteria's that control when to perform SDU discard

Operation mode	Presence
Timer based discard, with explicit signalling	Network controlled
Timer based discard, without explicit signalling	Network controlled
SDU discard after MaxDAT number of retransmissions	Network controlled

9.7.3.1 Timer based discard, with explicit signalling

This alternative uses a timer based triggering of SDU discard (Timer_Discard). This makes the SDU discard function insensitive to variations in the channel rate and provides means for exact definition of maximum delay. However, the SDU loss rate of the connection is increased as SDUs are discarded.

For every SDU received from a higher layer, timer monitoring of the transmission time of the SDU is started. If the transmission time exceeds a predefined value for a SDU in acknowledged mode RLC, this SDU is discarded in the transmitter. Following which, if one or more segments of the SDU have been submitted to a lower layer, ~~and~~ a Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. If Send MRW is configured, an expired SDU whose segments were not submitted to a lower layer is also informed to the receiver by a MRW command. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded.

The MRW command is defined as a super-field in the RLC STATUS PDU (see subclause 9.2), and piggybacked to status information of transmissions in the opposite direction. If the MRW command has not been acknowledged by receiver, it will be retransmitted. Therefore, SDU discard variants requiring peer-to-peer signalling are only possible for full duplex connections.

9.7.3.2 Timer based discard, without explicit signalling

This alternative uses the same timer based trigger for SDU discard (Timer_Discard) as the one described in the subclause 9.7.3.1. The difference is that this discard method does not use any peer-to-peer signalling. This function is

applied only for unacknowledged and transparent mode RLC and peer-to-peer signalling is never needed. The SDUs are simply discarded in the transmitter, once the transmission time is exceeded.

9.7.3.3 SDU discard after MaxDAT number of retransmissions

This alternative uses the number of retransmissions as a trigger for SDU discard, and is therefore only applicable for acknowledged mode RLC. This makes the SDU discard function dependent of the channel rate. Also, this variant of the SDU discard function strives to keep the SDU loss rate constant for the connection, on the cost of a variable delay. SDU discard is triggered at the transmitter, and a MRW command is necessary to convey the discard information to the receiver, like in the timer based discard with explicit signalling.

11.6 SDU discard with explicit signalling procedure

11.6.1 Purpose

An SDU can be discarded with explicit signalling when MaxDAT number of retransmissions is reached or the transmission time exceeds a predefined value (Timer_Discard) for a SDU in acknowledged mode RLC. Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded. If Send MRW is not configured and no segments of a SDU were submitted to a lower layer, the SDU is simply discarded in the transmitter without notification to the receiver. If Send MRW is configured, a Move Receiving Window request shall be sent to the receiver even if no segments of the SDU were submitted to a lower layer. The Send MRW is used when the AM RLC entity is connected to a PDCP layer which supports lossless SRNS relocation.

The MRW command is defined as a super-field in the RLC STATUS PDU, and can be piggybacked to status information of transmissions in the opposite direction.

Figure 11.6 below illustrates the elementary procedure for SDU discard with explicit signalling. The sender is the sender of AMD PDUs and it is either the UE or the network and the receiver is the receiver of AMD PDUs and it is either the network or the UE.

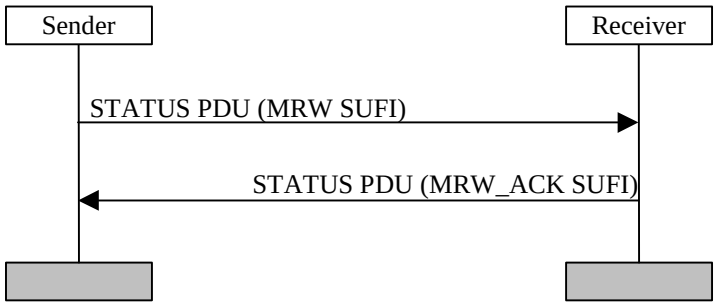


Figure 11.6: SDU discard with explicit signalling

11.6.2 Initiation

This procedure is initiated by the sender when the following conditions are fulfilled:

1) Timer based SDU discard with explicit signalling is used, ~~and~~ Timer_Discard expires for an SDU, and one or more segments of the SDU have been submitted to a lower layer.

2) Timer based SDU discard with explicit signalling is used, Timer_Discard expires for an SDU, and Send MRW is configured.

3~~2~~) SDU discard after MaxDAT number of retransmissions is used, and MaxDAT number of retransmissions is reached for an SDU.

The sender shall discard all PUs that contain segments of the associated SDUs. If the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded. VT(A) shall be updated when the procedure is terminated, and VT(S) shall be updated when a new MRW SUFI which includes $SN\_MRW_{LENGTH} \geq VT(S)$ is transmitted.

The sender shall transmit a status report on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

This status report is sent even if the 'STATUS prohibit' is used and the timer 'Timer_Status_Prohibit' or 'Timer_EPC' is active.

The STATUS PDUs have higher priority than data PDUs.

The sender shall start timer Timer_MRW. If a new SDU discard procedure is triggered when Timer_MRW is running, no new MRW SUFIs shall be sent before the current SDU discard procedure is terminated by one of the termination criteria.

3GPP TSG-RAN WG2 Meeting #19 ***R2-010342***

Sophia Antipolis, France, 19 - 23 Feb. 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **101** ⌘ rev **-** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Annex updates
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 19 Feb. 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	Annex A and Annex B are no longer usefull.
Summary of change:	⌘	Annex A and Annex B are removed
Consequences if not approved:	⌘	

Clauses affected:	⌘	Annex A, Annex B, Annex C		
Other specs Affected:	⌘	☐ Other core specifications	⌘	
		☐		
		☐ O&M Specifications		
Other comments:	⌘			

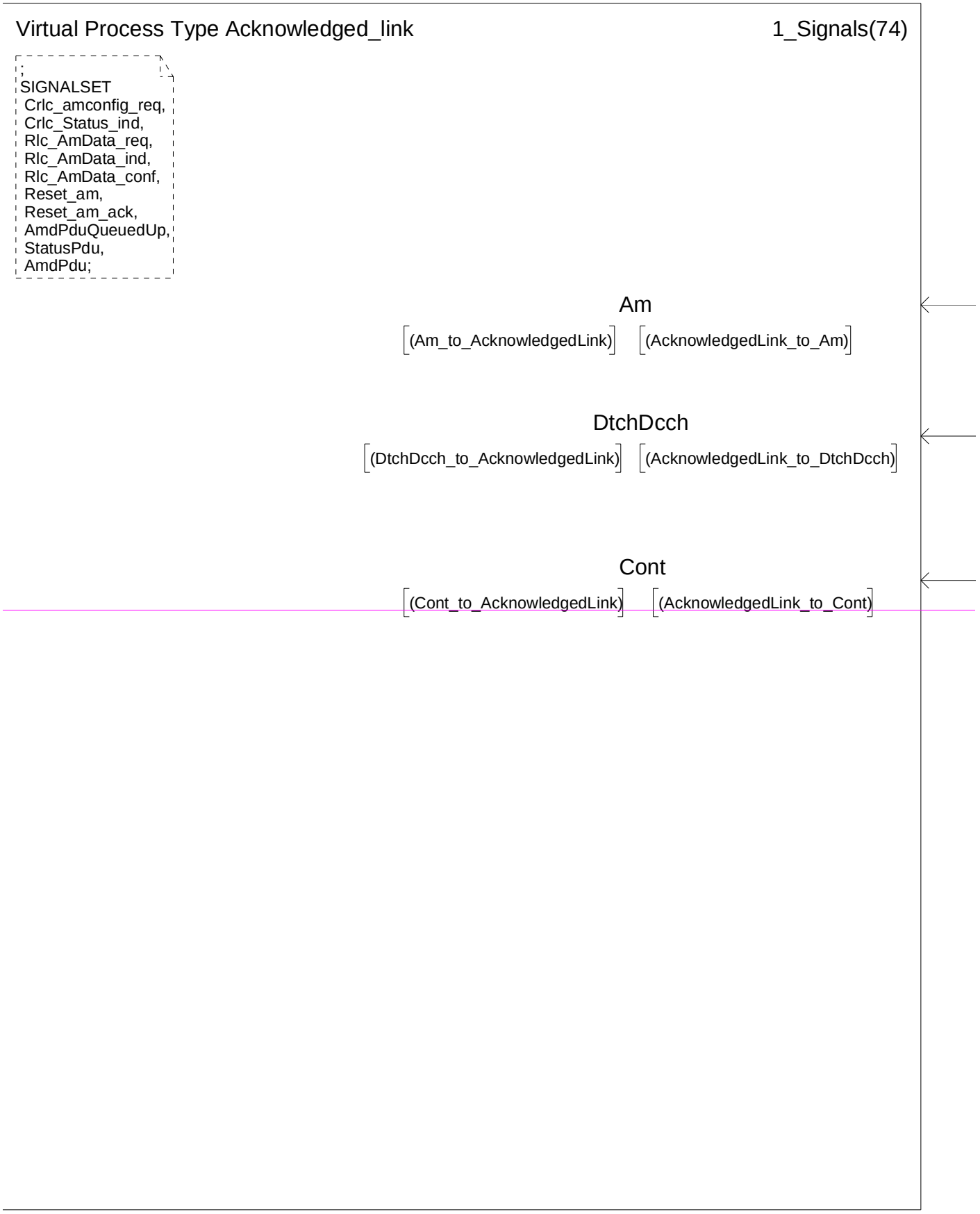
How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.
2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.
3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

Annex A (informative): SDL diagrams

This annex contains the SDL diagrams. For Release'99, it is meant for informative purposes only.

NOTE: All the SDL diagrams presented are [FFS].


Virtual Process Type Acknowledged_link
1_Signals(74)
SIGNALSET
Crlc_amconfig_req,
Crlc_Status_ind,
Rlc_AmData_req,
Rlc_AmData_ind,
Rlc_AmData_conf,
Reset_am,
Reset_am_ack,
AmdPduQueuedUp,
StatusPdu,
AmdPdu;
Am
(Am_to_AcknowledgedLink)
(AcknowledgedLink_to_Am)
DtchDcch
(DtchDcch_to_AcknowledgedLink)
(AcknowledgedLink_to_DtchDcch)
Cont
(Cont_to_AcknowledgedLink)
(AcknowledgedLink_to_Cont)

Virtual Process Type Acknowledged_link 1_Declarations(74

SIGNALSET

```
DCL


/*SDU, PDU, and PU declarations:_____*/

sdu                                          OctetType,
/*The sdu data from the upper layer protocol.*/

amd_pdu, pdu                                 AmPdu,
/*A representation of data contained within an AmPdu.*/

amd_pu                                       AmPuStructType,
/*A representation of a local am_pu*/

status_pdu, tx_status_pdu                    StatPdu,
/*A representation of data contained within a StatPdu.*/

rst_pdu                                      RstPdu,
/*A representation of data contained within a RstPdu.*/

/*SDU, PDU,  and PU  array declarations:_____*/

sdus                                         OctetArrayType,
/*An array containing SDUs.*/

pdus                                         AmPduArrayType,
/*An array containing AMD PDUs created by segmenting a SDU.*/

pus                                          AmPuArrayType,
/*An array containing PUs.*/

rem_pus                                      AmPuArrayType,
/*An array containing PDUs to be removed from queues.*/

status_pdus                                  StatusPduArrayType,
/*An array containing several STATUS PDUs.*/


/*Queue declarations:_____*/

receiver_queue                               Queue,
/*A queue used for storing PDUs as they arrive.*/

retransmission_queue                         Queue,
/*A queue used for PDUs that are to be retransmitted.*/

assembly_queue                               Queue,
/*A queue used for reassembly of received PDUs into an SDU.*/

transmitted_queue                            Queue,
/*A queue used for PDUs that have been transmitted.*/

amd_queue                                    Queue,
/*A queue used for PDUs to be transmitted.*/

mui_queue                                    Queue;
/*A queue used to store mui numbers for which confirmation
  has been requested.*/
```

Virtual Process Type Acknowledged_link 2_Declarations(74

SIGNALSET

```
DCL

/*Indicator declarations:_____________________________________*/

 epc_active                                    IndicatorType,
/*An indicator used to store whether the Timer_EPC is active or not.*/

 poll_periodic_active                          IndicatorType,
/*An indicator used to store whether the Timer_Poll_Periodic is active or not.*/

 poll_prohibit_active                          IndicatorType,
/*An indicator used to store whether the Timer_Poll_Prohibit is active or not.*/

 rst_active                                    IndicatorType,
/*An indicator used to store whether the Timer_RST is active or not.*/

 status_periodic_active                        IndicatorType,
/*An indicator used to store whether the Timer_Status_Periodic is active or not.*/

 status_prohibit_active                        IndicatorType,
/*An indicator used to store whether the Timer_Status_Prohibit is active or not.*/

 empty                                         IndicatorType,
/*An Indicator used to determine whether a queue is empty or not.*/

 exists                                        IndicatorType,
  /*An indicator used to determine whether a particular pdu exists
    within a queue or not.*/

 complete                                      IndicatorType,
  /*An indicator used to determine whether an SDU has been
     completely reassembled.*/

 cnf                                           IndicatorType,
  /*An indicator used to determine  whether an SDU requires
    confirmation.*/

 possible                                      IndicatorType,
/*An indicator used to indicate whether status piggyback is possible or not.
    An indicator used to indicate whether the PUs requested by the status report
   exsists in the que or not.*/

 create_status                                 IndicatorType,
  /*An indicator used to store whether a status report should be created or not.*/

 poll_triggered                                IndicatorType,
/*This variable is used to record if a poll is to be transmitted or not.*/

 status_triggered                              IndicatorType,
/*This variable is used to indicate whether a status report should be transmitted
    or not.*/

 suspend                                       IndicatorType,
/*This variable is used to indicate whether a local_suspend is in progress or not.*/

 piggyback                                     IndicatorType;
/*This variable indicates whether a piggybacked status report is included
    in the PDU or not.*/
```

Virtual Process Type Acknowledged_link 3_Declarations(74

SIGNALSET

```
DCL

/*Indicator declarations:_____*/

 MRW_active                                    IndicatorType,
/*An indicator used to store whether the Timer_MRW is active or not.*/

 poll_active                                   IndicatorType,
/*An indicator used to keep track of whether the Poll_Timer is active or not.*/

 contains, mrw_ans                             IndicatorType,
/*These indicators are used when checking the contents of a received
   status Pdu.*/

 discard_n_fli                                 IndicatorType,
/*This indicator is used to keep track of whether the first N length indicators of a given
   PU should be discarded or not when the receiving window is moved.*/

 retrans                                       IndicatorType,
/*This indicator keeps track of whether retransmissions should occur or not.*/

 missing_pu_detected                           IndicatorType;
/*This indicator is used to store whether he receive side has detected missing
   PUs.*/
```

Virtual Process Type Acknowledged_link 4_Declarations(74

SIGNALSET

```
DCL

/*Parameter declarations:_____________________________________*/

 e_r                                          ERParameterType,
/*The parameter indicating the desired end state.*/

 poll_triggers                                PollTriggArrType,
/*a configuration parameter dealing with when to issue poll requests.*/

 protocol_parameters                          ProtocolParametersStructType,
/*A struct variable containing the protocol parameters set.*/

 status_triggers                              StatusTriggArrType,
/*A configuraion parameter dealing with when to issue Status reports.*/

 timer_durations                              TimerDurationsStructType,
/*A struct containing the various timer durations.*/

 discard                                      DiscardArrayType,
 /*A configuration parameter identifying discard conditions.*/

 ciphering_mode                               CipheringModeType,
/*The ciphering mode.*/

 ciphering_key                                CipheringKeyType,
/*The ciphering key.*/

 hfn                                          CipheringSequenceNumberType,
/*The hyper frame number.*/

 leng                                         LengthType,
/*The number of SN_MRW fields in the MRW SUFI.*/

 pdu_size                                     OctetType,
/*The size in octets of an AMD PDU. It is indicated by MAC layer*/

 pu_size                                      OctetType,
/*The size in octets of a PU.*/

/*Sequence number variables:_____________________________________*/

 n,  sn_ack, sq                               SequenceNumberType,
 /*A local sequence number.*/

 poll_window                                  SequenceNumberType,
 /*The size of the poll_window.*/

 receive_window                               SequenceNumberType,
 /*The receive window size.*/

 transmit_window                              SequenceNumberType,
 /*The transmit window size.*/

 polled_sn                                    SequenceNumberType,
 /*This variable stores a sequence number associated with the PDU that contained
   a poll request.*/

n_susp, sn_suspend                            SequenceNumberType,
/*These variables contains sequence numbers used after  a local suspend has
  been initiated.*/

 sn_mrw                                       SequenceNumberType;
 /*This variable stores the sequence number associated with a MRW request.*/
```

Virtual Process Type Acknowledged_link 5_Declarations(74

SIGNALSET

```
DCL

/*Local variables declarations:_____*/

logical_channel                                    LogicalChannelType,
/*The logical channel associated with transmissions.*/

i, j                                                INTEGER,
/*A local counter.*/

mui                                                MuiType,
 /*The message unit identifier associated with a message to be transmitted.*/

muis                                               MuiArrayType,
 /*An array used to store message unit identifiers.*/

tx_rsn, rx_rsn                                     PduIndexType,
 /*A local variable for maintaining knowledge of the latest reset sequence number of
   the transmitted/received RESET PDU.*/

tot_mui, k,  tot_rem, n_sq                         PduIndexType,
/*Counters used to manage the amount of PUs and SDUs received.*/

tot_list                                           PduIndexType,
/*A local variable for maintaining knowledge of the total number of
   (SNi, Li)-pairs in a list super field.*/

tot_bitmap, tot_rlist                              PduIndexType,
/*A local variable for maintaining knowledge of the total length of a bitmap or codewords.*/

n_sdu                                              PduIndexType,
/*A local variable for maintaining knowledge of the number of SDUs reassembled PUs.*/

n_pdu                                              PduIndexType,
/*A local variable for maintaining knowledge of the number of AMD PDUs created from a SDU.*/

n_pu                                               PduIndexType,
/*A local variable for maintaining knowledge of the number of PUs included in a AMD PDU.*/

n_status                                           PduIndexType,
/*A local variable for maintaining knowledge of the number of STATUS PDUs
  which have been created.*/

n_pu_per_tti                                       PduIndexType,
/*A local variable for maintaining knowledge of the number of PUs received within a TTI.*/

end_state                                          EndStateType,
/*A variable used to ensure correct timer reset.*/

poll_win                                           REAL,
/*A local variable used to store the current transmit window usage.*/

 bitmap                                            IndicatorArrayType,
 /*This array of boolean values indicates losses experienced by the
   receiver.*/

codewords                                          IndicatorArrayType,
 /*This array is used to store the codewords in the rlsit super field.*/

mrw                                                SufiArrayType;
 /*This array is used to store the MRW super field or the MRW_N_IFL
     super field.*/
```

Virtual Process Type Acknowledged_link 6_Declarations(74

SIGNALSET

```
DCL
 /*State variable declarations:_____________________________________________*/

vt_s                                SequenceNumberType,
 /*Send state variable: The sequence number of the next pu to be transmitted for the first time (i.e
    excluding retransmissions). It is updated after transmission of a PDU which includes not earlier
     transmitted PUs. The initial value of this variable is 0.*/

 vt_a                               SequenceNumberType,
/*Acknowledge state variable: The sequence number of  the next in-sequence PU expected to
    be acknowledged, thus forming the lower edge of the window of acceptable acknowledgements.
    The variable vt_a is updated based on receipt of a STATUS PDU including an ACK super-field.
     The initial value of this variable is 0.*/

 vt_ms                              SequenceNumberType,
/*Maximum send state variable:  The sequence number of the first PU not allowed by the peer
    receiver  (i.e. the receiver will allow up t o vt_ms-1) vt_ms=vt_a+ window size. This value
     represents the upper edge of the transmit window.  The transmitter shall not transmit a
     new PU if vt_s >= vt_ms. The variable vt_ms is updated based on receipt of a STATUS PDU
     incluiding an ACK and/or WINDOW super-field.*/

 vt_pu                              SequenceNumberType,
 /*This state variable is used when the poll every Poll_PU PU function is used. It is incremented with
    1 for each PU that is transmitted. It should be incremented for both new and retransmitted PUs.
    When it reaches Poll_PU a new poll is transmitted and the state variable is set to zero. The initial
     value of this variable is 0.*/

 vt_sdu                             SequenceNumberType,
 /*This state variable is used when the poll every Poll_SDU SDU function is used. It is incremented
    with 1 for each SDU that is transmitted. When it reaches Poll_SDU a new poll is transmitted and
    the state variable is set to zero. The poll bit should be set in the PU that contains the last segment
     of the SDU. The initial value of this variable is 0.*/

 vt_rst                             SequenceNumberType,
 /*Reset state variable: This variable is used to count the number of times a RESET PDU is transmit-
    ted. It is incremented with 1 each time a RESET PDU is transmitted. It is reset upon reception of
    a RESET ACK PDU.  The initial value of this variable is 0.*/

 vr_r                               SequenceNumberType,
/*Receive state variable: The sequence number of the next in sequence PU expected to be received.
  It is updated upon receipt of the next in-sequence pdu. The initial value of this variable is 0.*/

vr_h                                SequenceNumberType,
/*Highest expected state variable: The sequence number of the next highest expected pdu. The vari-
  able is updated whenever a new pdu is received with SN>=vr_h. The initial value of this variable is 0.*/

vr_mr                               SequenceNumberType,
/*Maximum acceptable receive state variable: The sequence number of the first pdu not allowed
  by the receiver (i.e. the receiver will allow up to vr_mr-1), vr_mr=vr_r+window size. The receiver
  shall discard PUs with SN>=vr_mr, (in one case, such a PU may cause the transmission of an
  unsolicited STATUS PDU).*/

vr_ep                               SequenceNumberType;
/*Estimated PDU counter state variable: The number of PUs that should be received yet as
    a consequence of the transmission of the latest STATUS PDU. In acknowledged mode,
    this state variable is updated at the end of each transmission time interval. It is decremented
   by the number of PUs that should have been received during the transmission time interval. If
   VR(EP) is equal to zero, then check if all PUs requested for retransmission in the latest STATUS
   PDU have been received. */
```

Virtual Process Type Acknowledged_link 7_Declarations(74

SIGNALSET
Crlc_amconfig_req

```
DCL
/*State variable declarations:_____________________________________________*/

vt_dat                                          SequenceNumberType,
/*This state variable counts the number of times a PU has been transmitted. There is one
   VT(DAT) for each PU and it is incremented each time the PU is transmitted. The initial
   value of this variable is 0.*/

vt_mrw                                          SequenceNumberType;
/*It is used to count the number of times a MRW command is transmitted. VT(MRW) is
  incremented with 1 each time a  MRW command is transmitted. VT(MRW) is reset upon
  the reception of a STATUS PDU which suggests the acknowledgement of a MRW
  command in the receiver or the occurrence of discarding new SDU. The initial value
  of this variable is 0.*/
```

Virtual Process Type Acknowledged_link 8_Declarations(74

SIGNALSET

```
TIMER

 Timer_Poll,
  /*This timer is only used when the poll timer trigger is used. It is started when the transmitting side sends a
    poll to the peer entity. The timer is stopped when receiving a STATUS PDU that contains an acknowledge-
    ment or negative acknowledgement of the AMD PDU that triggered the timer. The value of the timer is sig-
    nalled by RRC. If the timer expires and no STATUS PDU containing an acknowledgement or negative
    acknowledgement of the AMD PDU that triggered the timer has been received, the receiver is polled once
    more (either by the transmission of a PDU which was not yet sent, or by a retransmission) and the timer is
   restarted. If there is no PU to be transmitted and all PUs have already been acknowledged, the receiver shall
   not be polled. If a new poll is sent when the timer is running it is restarted.*/

 Timer_Poll_Prohibit,
 /*This timer is only used when the poll prohibit function is used. It is used to prohibit transmission of polls within
   a certain period. A poll shall be delayed until the timer expires if a poll is triggered when the timer is active.
   Only one poll shall be transmitted when the timer expires even if several polls were triggered when the timer
   was active. If there is no PU to be transmitted and all PUs have already been acknowledged, a poll shall not
   be transmitted. This timer will not be stopped by a STATUS PDU. The value of the timer is signalled by RRC. */

 Timer_EPC,
 /*This timer is only used when the EPC function is used and it accounts for the roundtrip delay, i.e. the
    time when the first retransmitted PU should be received after a STATUS has been sent. The timer is
    started when a STATUS report is transmitted and when it expires EPC can start decrease. The value
    of the timer is signalled by RRC.*/

 Timer_EPC_check,
  /*This timer is used to count down  the state variable vr_ep at acertain interval.*/

  Timer_Discard(MuiType),
  /*This timer is used for the SDU discard function. In the transmitter, the timer is activated upon reception of a SDU
     from higher layer. If the SDU has not been acknowledged when the timer expires, the SDU is discarded. Following
     which, if the SDU discard function uses explicit signalling, a Move Receiving Window request is sent to the receiver.
    The value of the timer is signalled by RRC.*/

  Timer_Poll_Periodic;
  /*This timer is only used when the timer based polling is used. The timer is started when the RLC entity is created.
    Each time the timer expires a poll is transmitted and the timer is restarted. If there is no PU to be transmitted and
    all PUs have already been acknowledged, a poll shall not be transmitted and the timer shall only be restarted.
   The value of the timer is signalled by RRC.*/
```

Virtual Process Type Acknowledged_link 9_Declarations(74

SIGNALSET

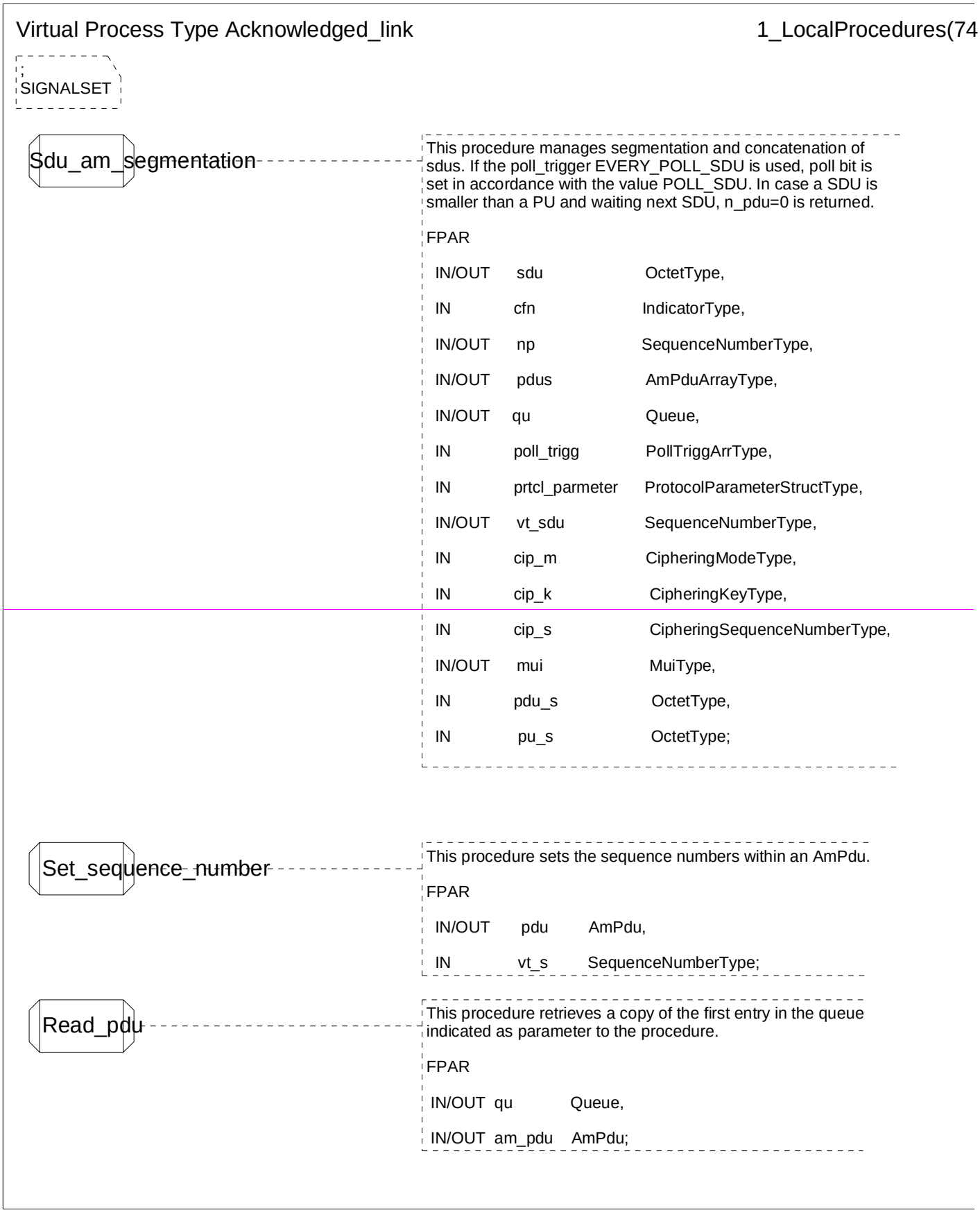
TIMER

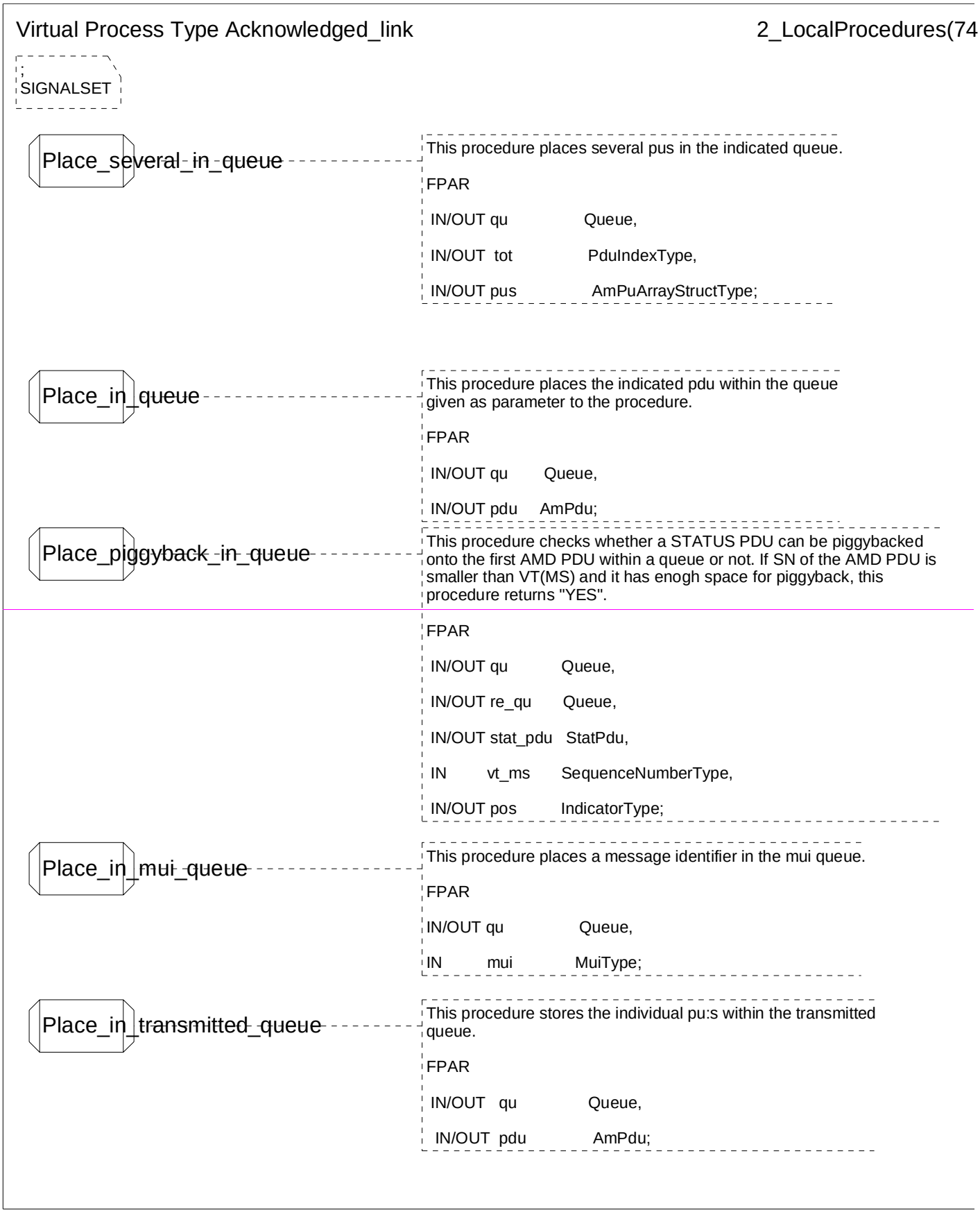
Timer_Status_Prohibit,
/*This timer is only used when the STATUS PDU prohibit function is used. It prohibits the receiving side from sending STATUS PDUs. The timer is started when a STATUS PDU is transmitted and no new STATUS PDU can be transmitted before the timer has expired. The value of the timer is signalled by RRC.*/

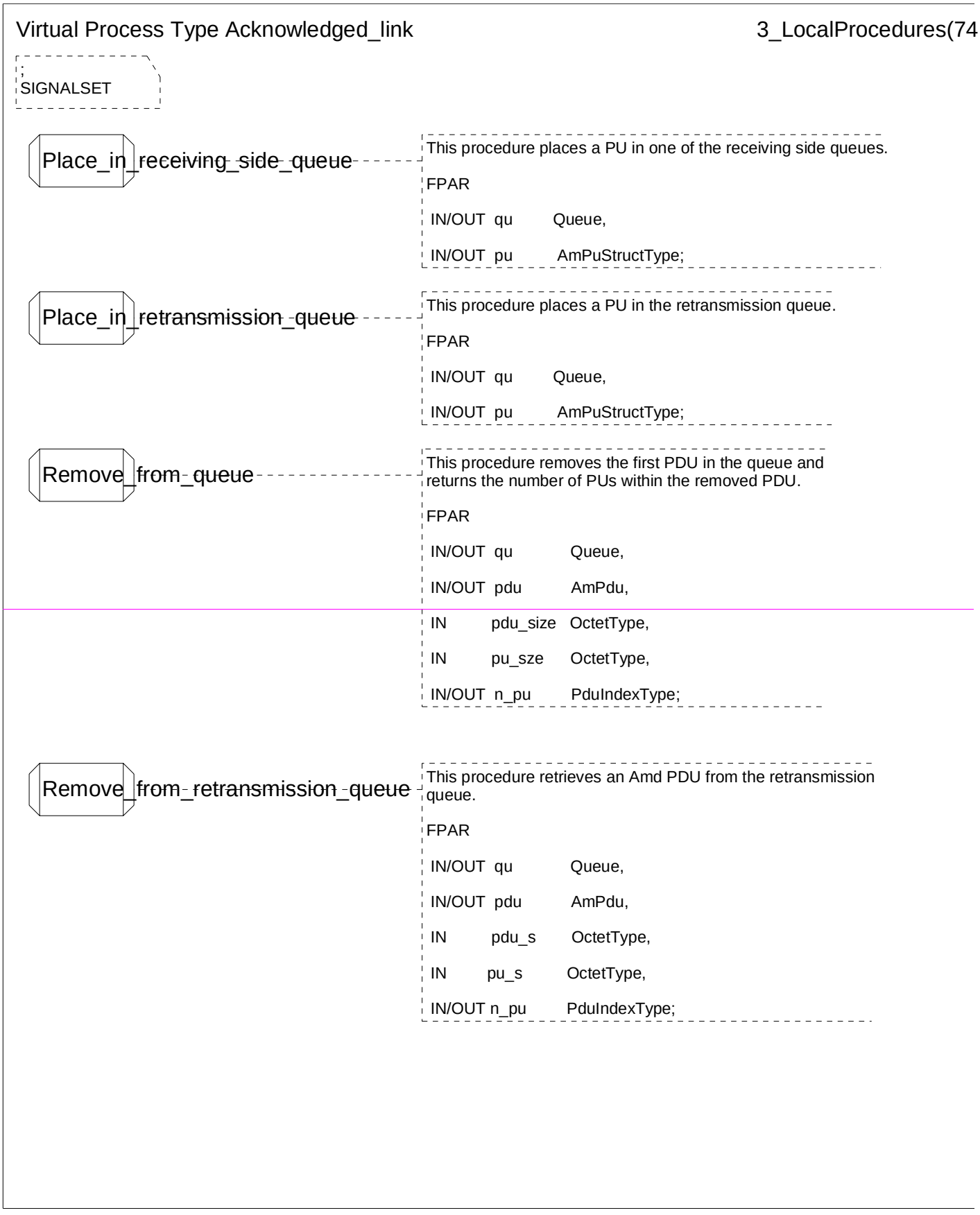
Timer_Status_Periodic,
/*This timer is only used when timer based STATUS PDU sending is used. The timer is started when the RLC entity is created. Each time the timer expires a STATUS PDU is transmitted and the timer is restarted. The value of the timer is signalled by RRC.*/

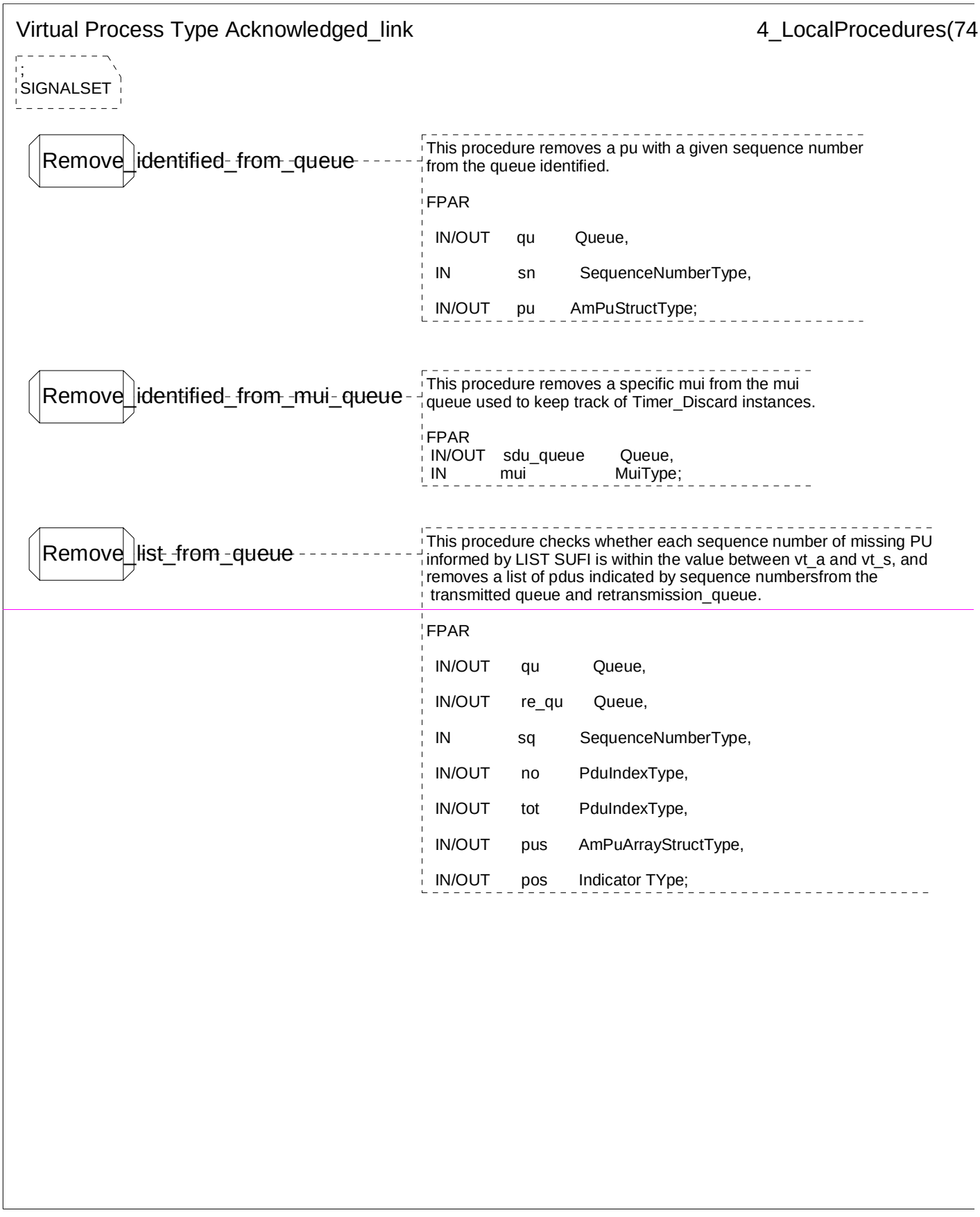
Timer_MRW,
/*This timer is used as part of the Move Receiving Window protocol. It is used to trigger the retransmission of a STATUS PDU containing an MRW SUFI field. The timer is started when the STATUS PDU is first transmitted. Each time the timer expires the STATUS PDU is retransmitted and the timer is restarted. It shall be stopped when a STATUS PDU is received that indicates that VR(R) ³ SN_MRW. It shall also be stopped if a new MRW procedure is triggered whilst it is running.*/

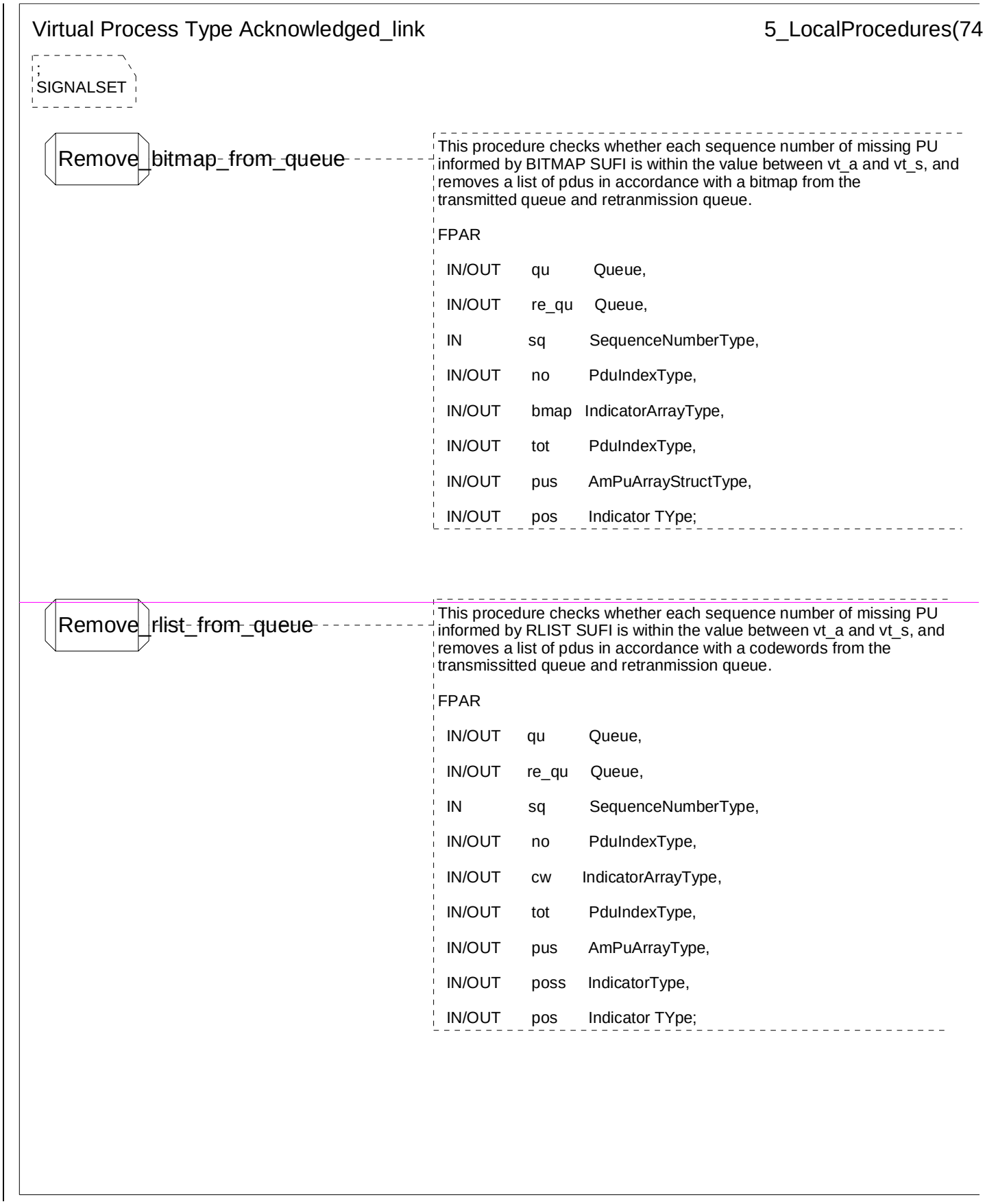
Timer_RST;
/*It is used to detect the loss of RESET ACK PDU from the peer RLC entity. This timer is set when the RESET PDU is transmitted. And it will be stopped upon reception of RESET ACK PDU. If it expires, RESET PDU will be retransmitted.*/

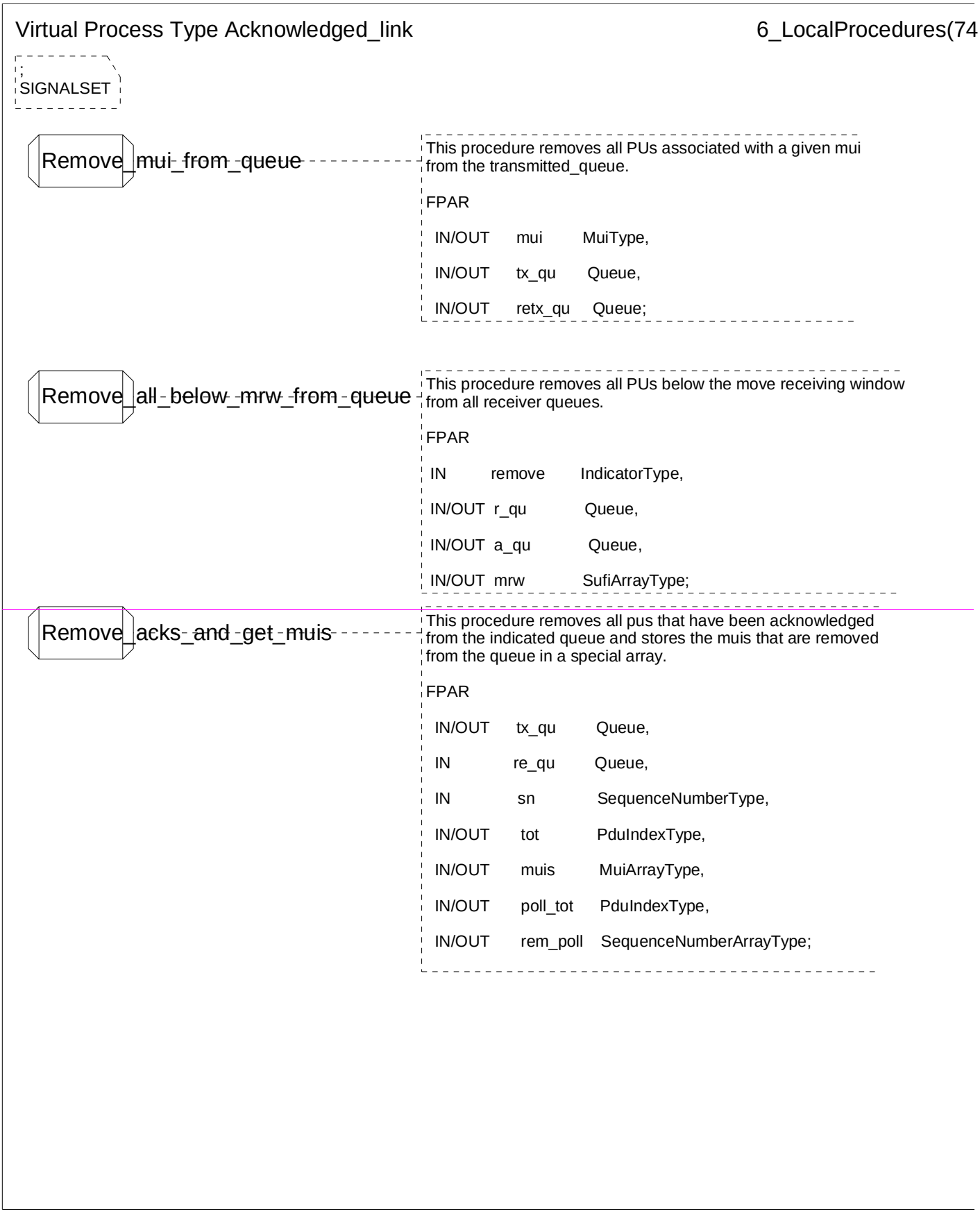

Virtual Process Type Acknowledged_link
1_LocalProcedures(74
SIGNALSET
Sdu_am_segmentation
This procedure manages segmentation and concatenation of sdus. If the poll_trigger EVERY_POLL_SDU is used, poll bit is set in accordance with the value POLL_SDU. In case a SDU is smaller than a PU and waiting next SDU, n_pdu=0 is returned.
FPAR
IN/OUT sdu OctetType,
IN cfn IndicatorType,
IN/OUT np SequenceNumberType,
IN/OUT pdus AmPduArrayType,
IN/OUT qu Queue,
IN poll_trigg PollTriggArrType,
IN prtcl_parmeter ProtocolParameterStructType,
IN/OUT vt_sdu SequenceNumberType,
IN cip_m CipheringModeType,
IN cip_k CipheringKeyType,
IN cip_s CipheringSequenceNumberType,
IN/OUT mui MuiType,
IN pdu_s OctetType,
IN pu_s OctetType;
Set_sequence_number
This procedure sets the sequence numbers within an AmPdu.
FPAR
IN/OUT pdu AmPdu,
IN vt_s SequenceNumberType;
Read_pdu
This procedure retrieves a copy of the first entry in the queue indicated as parameter to the procedure.
FPAR
IN/OUT qu Queue,
IN/OUT am_pdu AmPdu;

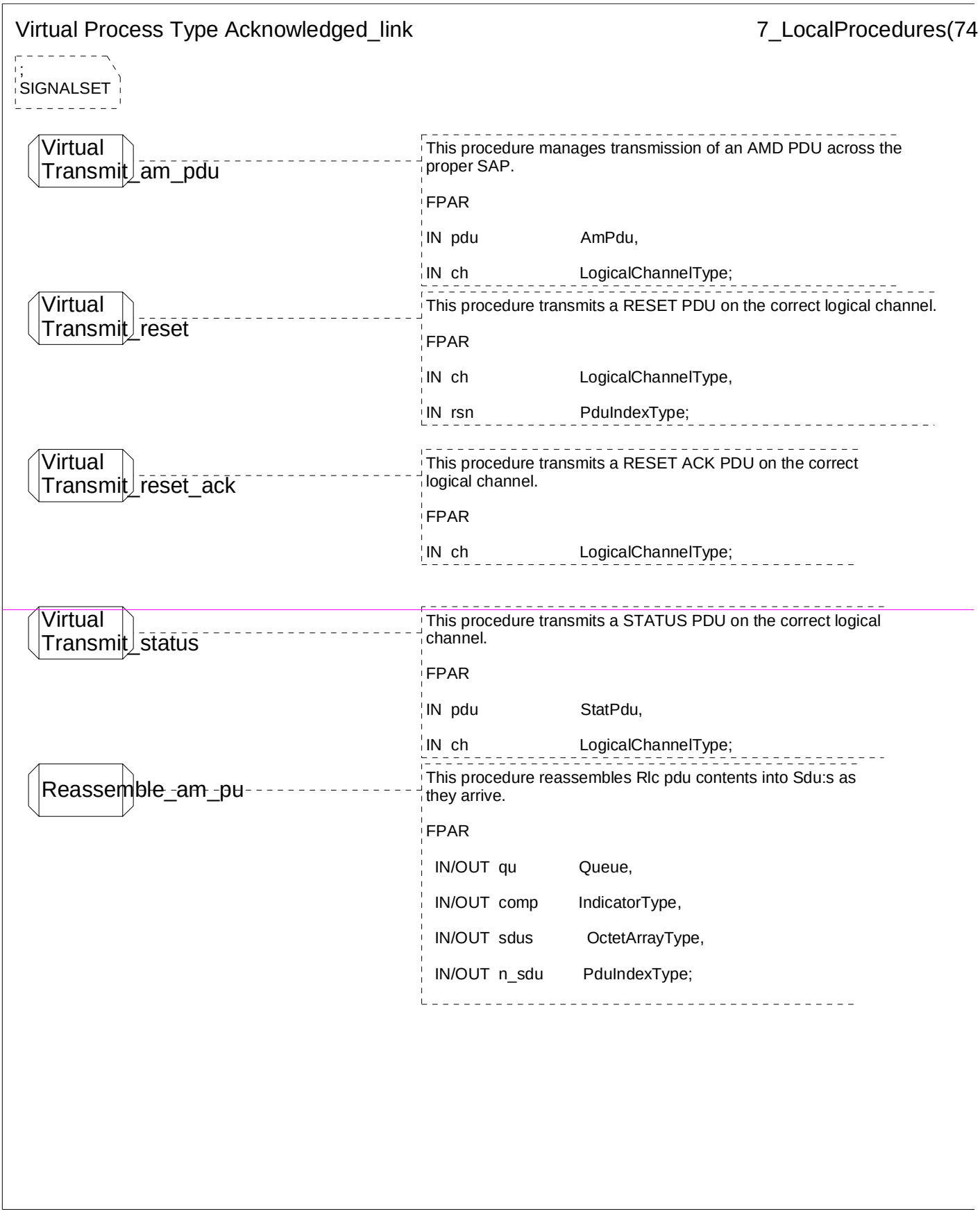

Virtual Process Type Acknowledged_link
2_LocalProcedures(74
SIGNALSET
Place_several_in_queue
This procedure places several pus in the indicated queue.
FPAR
IN/OUT qu Queue,
IN/OUT tot PduIndexType,
IN/OUT pus AmPuArrayStructType;
Place_in_queue
This procedure places the indicated pdu within the queue given as parameter to the procedure.
FPAR
IN/OUT qu Queue,
IN/OUT pdu AmPdu;
Place_piggyback_in_queue
This procedure checks whether a STATUS PDU can be piggybacked onto the first AMD PDU within a queue or not. If SN of the AMD PDU is smaller than VT(MS) and it has enogh space for piggyback, this procedure returns "YES".
FPAR
IN/OUT qu Queue,
IN/OUT re_qu Queue,
IN/OUT stat_pdu StatPdu,
IN vt_ms SequenceNumberType,
IN/OUT pos IndicatorType;
Place_in_mui_queue
This procedure places a message identifier in the mui queue.
FPAR
IN/OUT qu Queue,
IN mui MuiType;
Place_in_transmitted_queue
This procedure stores the individual pu:s within the transmitted queue.
FPAR
IN/OUT qu Queue,
IN/OUT pdu AmPdu;

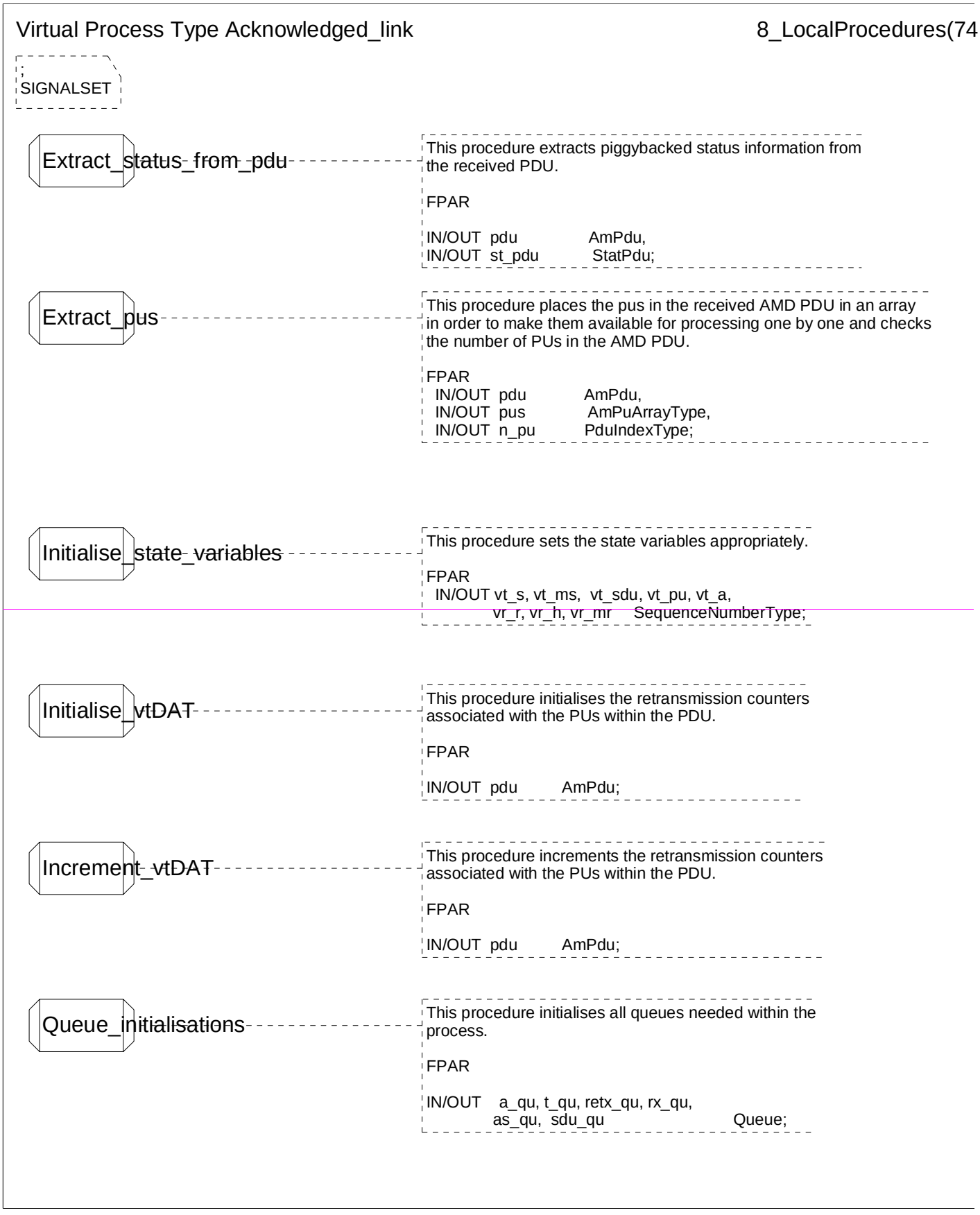

Virtual Process Type Acknowledged_link
3_LocalProcedures(74
SIGNALSET
Place_in_receiving_side_queue
This procedure places a PU in one of the receiving side queues.
FPAR
IN/OUT qu Queue,
IN/OUT pu AmPuStructType;
Place_in_retransmission_queue
This procedure places a PU in the retransmission queue.
FPAR
IN/OUT qu Queue,
IN/OUT pu AmPuStructType;
Remove_from_queue
This procedure removes the first PDU in the queue and returns the number of PUs within the removed PDU.
FPAR
IN/OUT qu Queue,
IN/OUT pdu AmPdu,
IN pdu_size OctetType,
IN pu_sze OctetType,
IN/OUT n_pu PduIndexType;
Remove_from_retransmission_queue
This procedure retrieves an Amd PDU from the retransmission queue.
FPAR
IN/OUT qu Queue,
IN/OUT pdu AmPdu,
IN pdu_s OctetType,
IN pu_s OctetType,
IN/OUT n_pu PduIndexType;

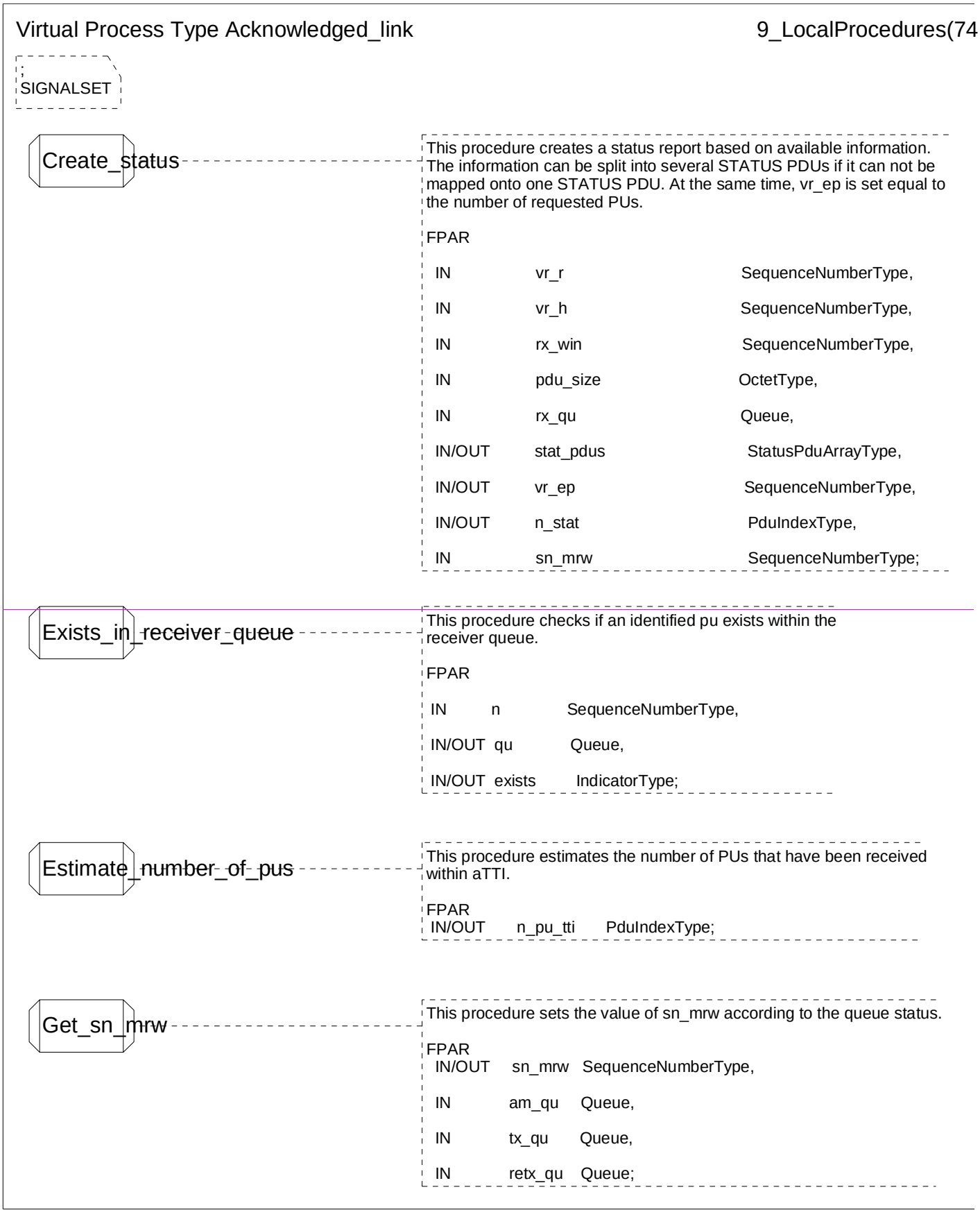

Virtual Process Type Acknowledged_link
4_LocalProcedures(74
SIGNALSET
Remove_identified_from_queue
This procedure removes a pu with a given sequence number from the queue identified.
FPAR
IN/OUT qu Queue,
IN sn SequenceNumberType,
IN/OUT pu AmPuStructType;
Remove_identified_from_mui_queue
This procedure removes a specific mui from the mui queue used to keep track of Timer_Discard instances.
FPAR
IN/OUT sdu_queue Queue,
IN mui MuiType;
Remove_list_from_queue
This procedure checks whether each sequence number of missing PU informed by LIST SUFI is within the value between vt_a and vt_s, and removes a list of pdus indicated by sequence numbersfrom the transmitted queue and retransmission_queue.
FPAR
IN/OUT qu Queue,
IN/OUT re_qu Queue,
IN sq SequenceNumberType,
IN/OUT no PduIndexType,
IN/OUT tot PduIndexType,
IN/OUT pus AmPuArrayStructType,
IN/OUT pos Indicator TYpe;

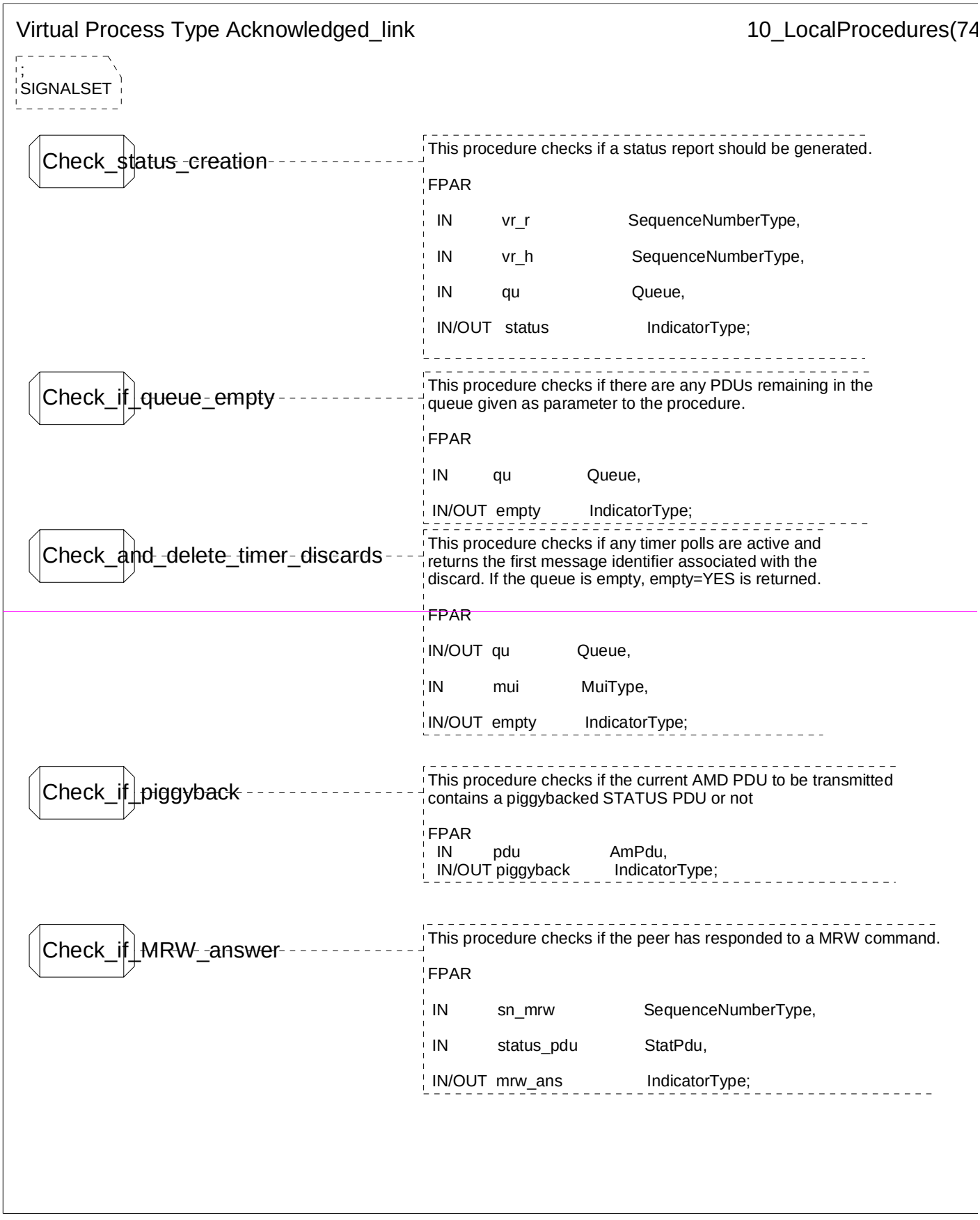

Virtual Process Type Acknowledged_link
5_LocalProcedures(74
SIGNALSET
Remove_bitmap_from_queue
This procedure checks whether each sequence number of missing PU informed by BITMAP SUFI is within the value between vt_a and vt_s, and removes a list of pdus in accordance with a bitmap from the transmitted queue and retranmission queue.
FPAR
IN/OUT qu Queue,
IN/OUT re_qu Queue,
IN sq SequenceNumberType,
IN/OUT no PduIndexType,
IN/OUT bmap IndicatorArrayType,
IN/OUT tot PduIndexType,
IN/OUT pus AmPuArrayStructType,
IN/OUT pos Indicator TYpe;
Remove_rlist_from_queue
This procedure checks whether each sequence number of missing PU informed by RLIST SUFI is within the value between vt_a and vt_s, and removes a list of pdus in accordance with a codewords from the transmissitted queue and retranmission queue.
FPAR
IN/OUT qu Queue,
IN/OUT re_qu Queue,
IN sq SequenceNumberType,
IN/OUT no PduIndexType,
IN/OUT cw IndicatorArrayType,
IN/OUT tot PduIndexType,
IN/OUT pus AmPuArrayType,
IN/OUT poss IndicatorType,
IN/OUT pos Indicator TYpe;

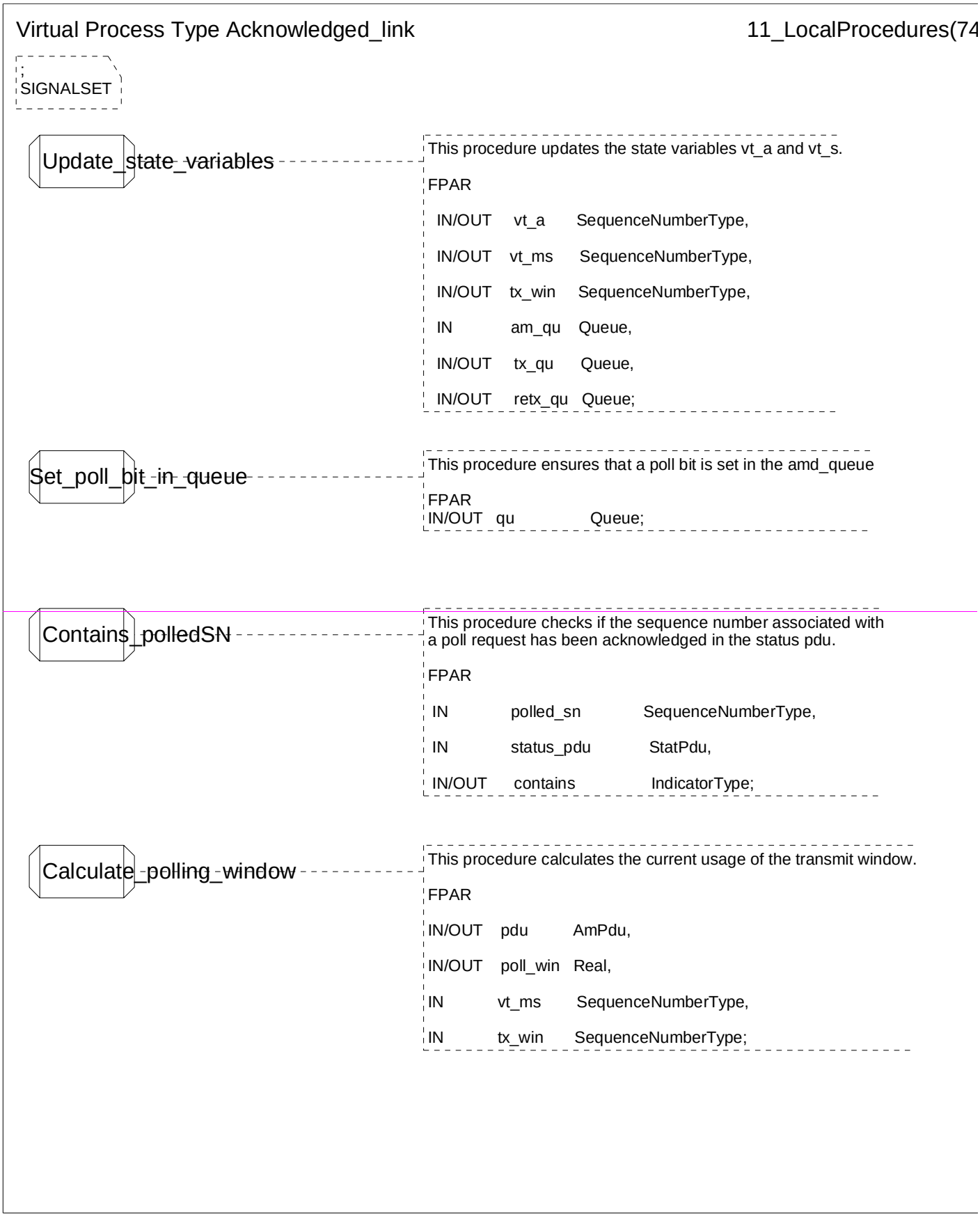

Virtual Process Type Acknowledged_link
6_LocalProcedures(74
SIGNALSET
Remove_mui_from_queue
This procedure removes all PUs associated with a given mui from the transmitted_queue.
FPAR
IN/OUT mui MuiType,
IN/OUT tx_qu Queue,
IN/OUT retx_qu Queue;
Remove_all_below_mrw_from_queue
This procedure removes all PUs below the move receiving window from all receiver queues.
FPAR
IN remove IndicatorType,
IN/OUT r_qu Queue,
IN/OUT a_qu Queue,
IN/OUT mrw SufiArrayType;
Remove_acks_and_get_muis
This procedure removes all pus that have been acknowledged from the indicated queue and stores the muis that are removed from the queue in a special array.
FPAR
IN/OUT tx_qu Queue,
IN re_qu Queue,
IN sn SequenceNumberType,
IN/OUT tot PduIndexType,
IN/OUT muis MuiArrayType,
IN/OUT poll_tot PduIndexType,
IN/OUT rem_poll SequenceNumberArrayType;

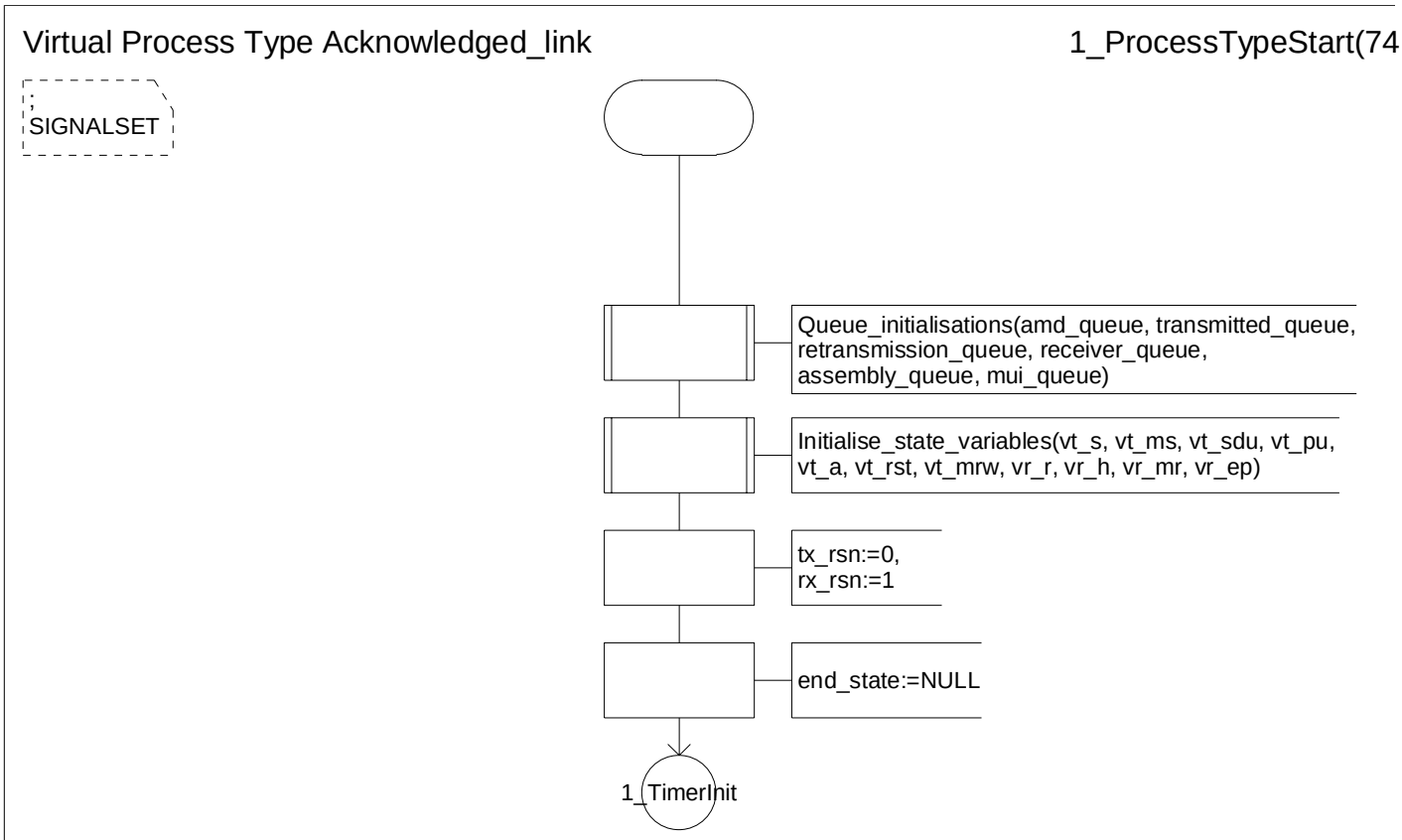

Virtual Process Type Acknowledged_link
7_LocalProcedures(74
SIGNALSET
Virtual Transmit_am_pdu
This procedure manages transmission of an AMD PDU across the proper SAP.
FPAR
IN pdu AmPdu,
IN ch LogicalChannelType;
Virtual Transmit_reset
This procedure transmits a RESET PDU on the correct logical channel.
FPAR
IN ch LogicalChannelType,
IN rsn PduIndexType;
Virtual Transmit_reset_ack
This procedure transmits a RESET ACK PDU on the correct logical channel.
FPAR
IN ch LogicalChannelType;
Virtual Transmit_status
This procedure transmits a STATUS PDU on the correct logical channel.
FPAR
IN pdu StatPdu,
IN ch LogicalChannelType;
Reassemble_am_pu
This procedure reassembles Rlc pdu contents into Sdu:s as they arrive.
FPAR
IN/OUT qu Queue,
IN/OUT comp IndicatorType,
IN/OUT sdus OctetArrayType,
IN/OUT n_sdu PduIndexType;

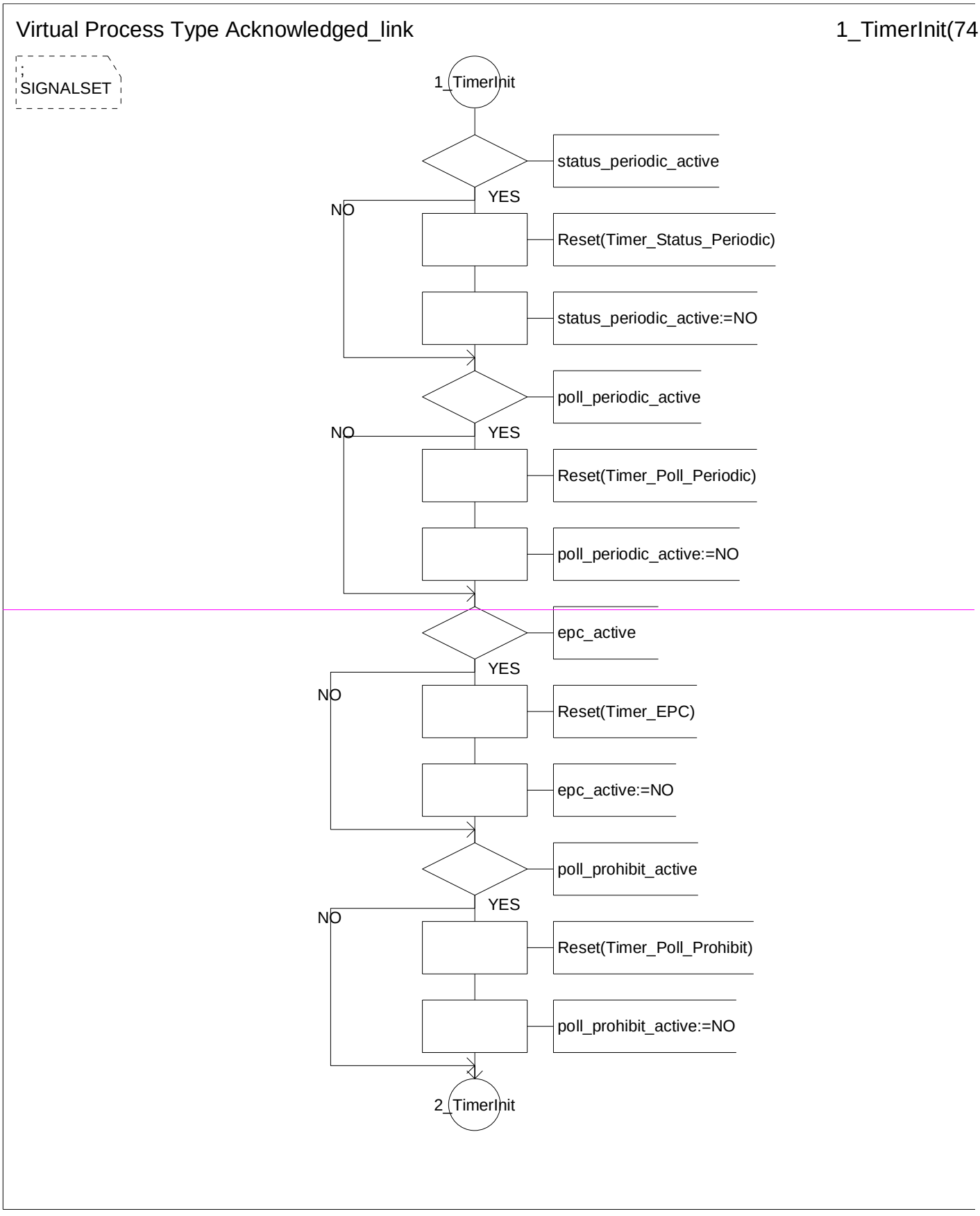

Virtual Process Type Acknowledged_link
8_LocalProcedures(74
SIGNALSET
Extract_status_from_pdu
This procedure extracts piggybacked status information from the received PDU.
FPAR
IN/OUT pdu AmPdu,
IN/OUT st_pdu StatPdu;
Extract_pus
This procedure places the pus in the received AMD PDU in an array in order to make them available for processing one by one and checks the number of PUs in the AMD PDU.
FPAR
IN/OUT pdu AmPdu,
IN/OUT pus AmPuArrayType,
IN/OUT n_pu PduIndexType;
Initialise_state_variables
This procedure sets the state variables appropriately.
FPAR
IN/OUT vt_s, vt_ms, vt_sdu, vt_pu, vt_a,
vr_r, vr_h, vr_mr SequenceNumberType;
Initialise_vtDAT
This procedure initialises the retransmission counters associated with the PUs within the PDU.
FPAR
IN/OUT pdu AmPdu;
Increment_vtDAT
This procedure increments the retransmission counters associated with the PUs within the PDU.
FPAR
IN/OUT pdu AmPdu;
Queue_initialisations
This procedure initialises all queues needed within the process.
FPAR
IN/OUT a_qu, t_qu, retx_qu, rx_qu,
as_qu, sdu_qu Queue;

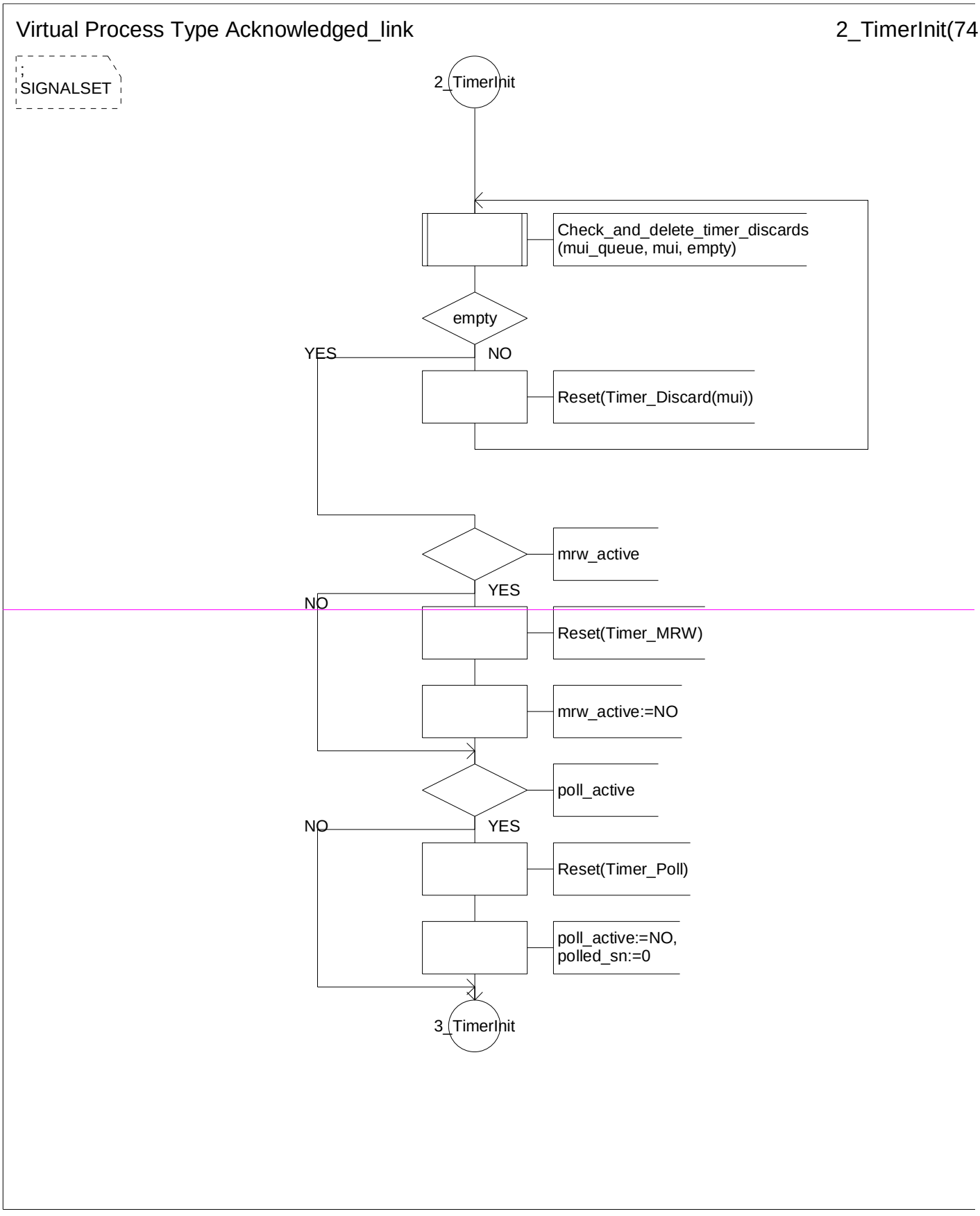

Virtual Process Type Acknowledged_link
9_LocalProcedures(74
SIGNALSET
Create_status
This procedure creates a status report based on available information. The information can be split into several STATUS PDUs if it can not be mapped onto one STATUS PDU. At the same time, vr_ep is set equal to the number of requested PUs.
FPAR
IN vr_r SequenceNumberType,
IN vr_h SequenceNumberType,
IN rx_win SequenceNumberType,
IN pdu_size OctetType,
IN rx_qu Queue,
IN/OUT stat_pdus StatusPduArrayType,
IN/OUT vr_ep SequenceNumberType,
IN/OUT n_stat PduIndexType,
IN sn_mrw SequenceNumberType;
Exists_in_receiver_queue
This procedure checks if an identified pu exists within the receiver queue.
FPAR
IN n SequenceNumberType,
IN/OUT qu Queue,
IN/OUT exists IndicatorType;
Estimate_number_of_pus
This procedure estimates the number of PUs that have been received within aTTI.
FPAR
IN/OUT n_pu_tti PduIndexType;
Get_sn_mrw
This procedure sets the value of sn_mrw according to the queue status.
FPAR
IN/OUT sn_mrw SequenceNumberType,
IN am_qu Queue,
IN tx_qu Queue,
IN retx_qu Queue;

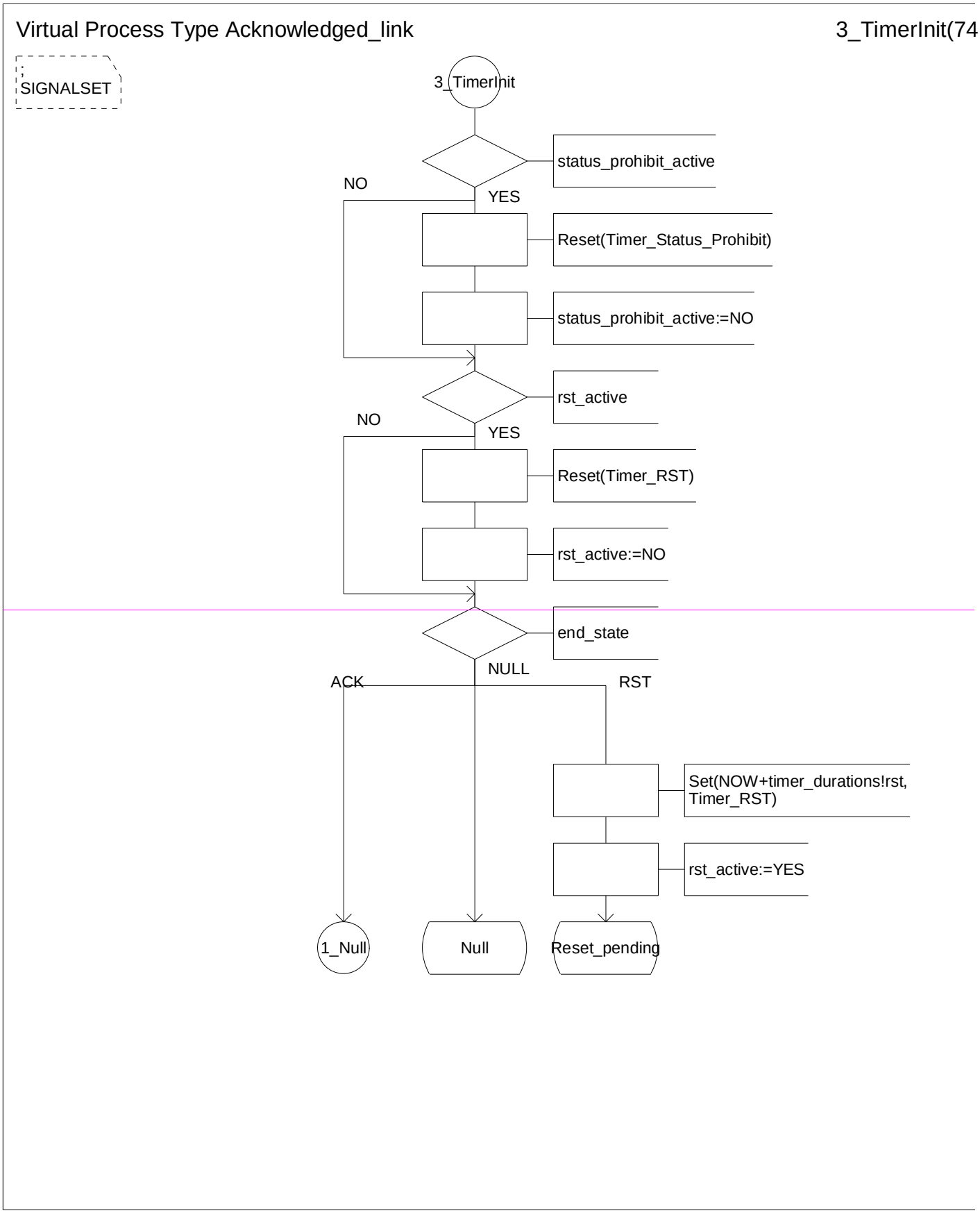

Virtual Process Type Acknowledged_link
10_LocalProcedures(74
SIGNALSET
Check_status_creation
This procedure checks if a status report should be generated.
FPAR
IN vr_r SequenceNumberType,
IN vr_h SequenceNumberType,
IN qu Queue,
IN/OUT status IndicatorType;
Check_if_queue_empty
This procedure checks if there are any PDUs remaining in the queue given as parameter to the procedure.
FPAR
IN qu Queue,
IN/OUT empty IndicatorType;
Check_and_delete_timer_discards
This procedure checks if any timer polls are active and returns the first message identifier associated with the discard. If the queue is empty, empty=YES is returned.
FPAR
IN/OUT qu Queue,
IN mui MuiType,
IN/OUT empty IndicatorType;
Check_if_piggyback
This procedure checks if the current AMD PDU to be transmitted contains a piggybacked STATUS PDU or not
FPAR
IN pdu AmPdu,
IN/OUT piggyback IndicatorType;
Check_if_MRW_answer
This procedure checks if the peer has responded to a MRW command.
FPAR
IN sn_mrw SequenceNumberType,
IN status_pdu StatPdu,
IN/OUT mrw_ans IndicatorType;

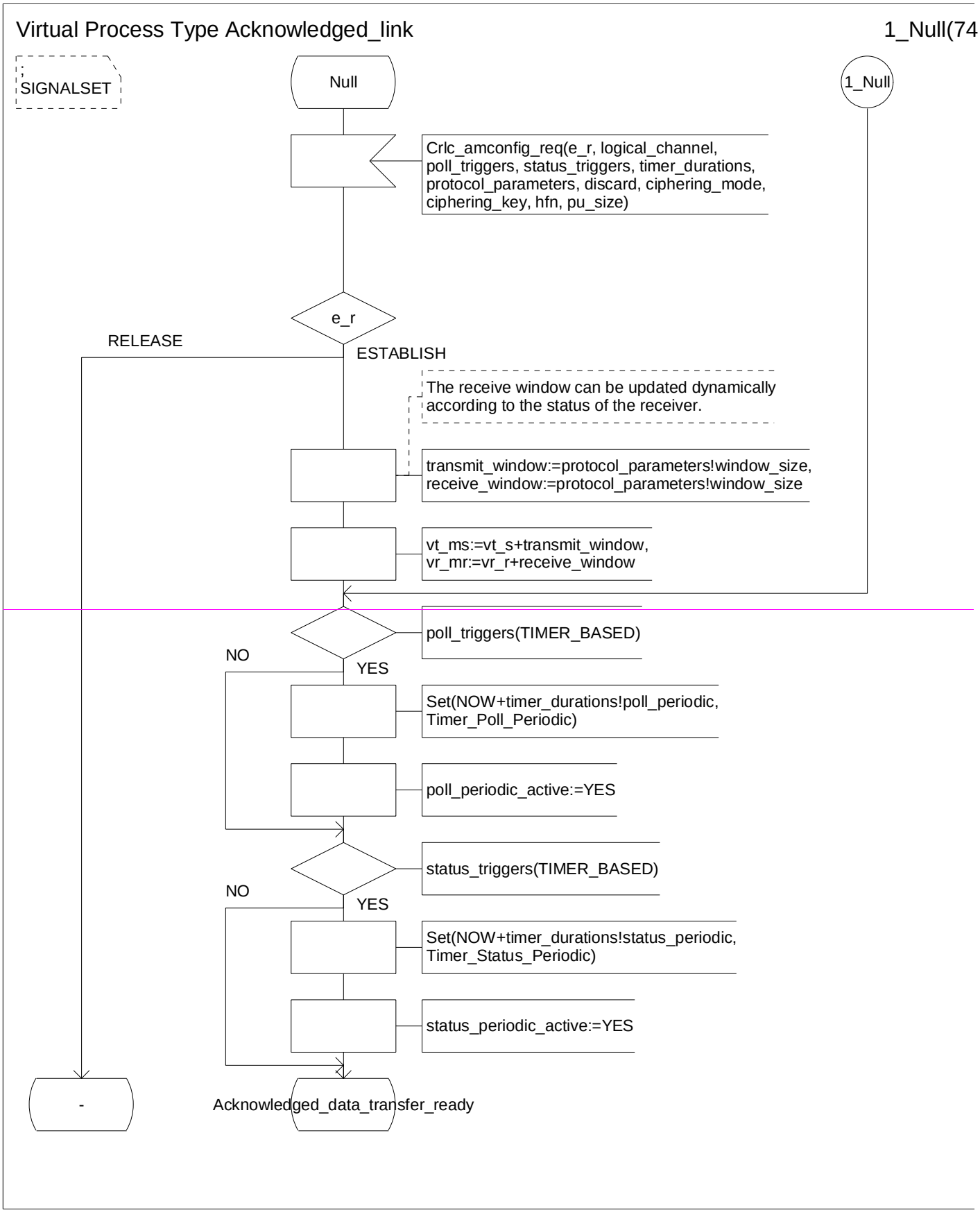

Virtual Process Type Acknowledged_link
11_LocalProcedures(74
SIGNALSET
Update_state_variables
This procedure updates the state variables vt_a and vt_s.
FPAR
IN/OUT vt_a SequenceNumberType,
IN/OUT vt_ms SequenceNumberType,
IN/OUT tx_win SequenceNumberType,
IN am_qu Queue,
IN/OUT tx_qu Queue,
IN/OUT retx_qu Queue;
Set_poll_bit_in_queue
This procedure ensures that a poll bit is set in the amd_queue
FPAR
IN/OUT qu Queue;
Contains_polledSN
This procedure checks if the sequence number associated with
a poll request has been acknowledged in the status pdu.
FPAR
IN polled_sn SequenceNumberType,
IN status_pdu StatPdu,
IN/OUT contains IndicatorType;
Calculate_polling_window
This procedure calculates the current usage of the transmit window.
FPAR
IN/OUT pdu AmPdu,
IN/OUT poll_win Real,
IN vt_ms SequenceNumberType,
IN tx_win SequenceNumberType;

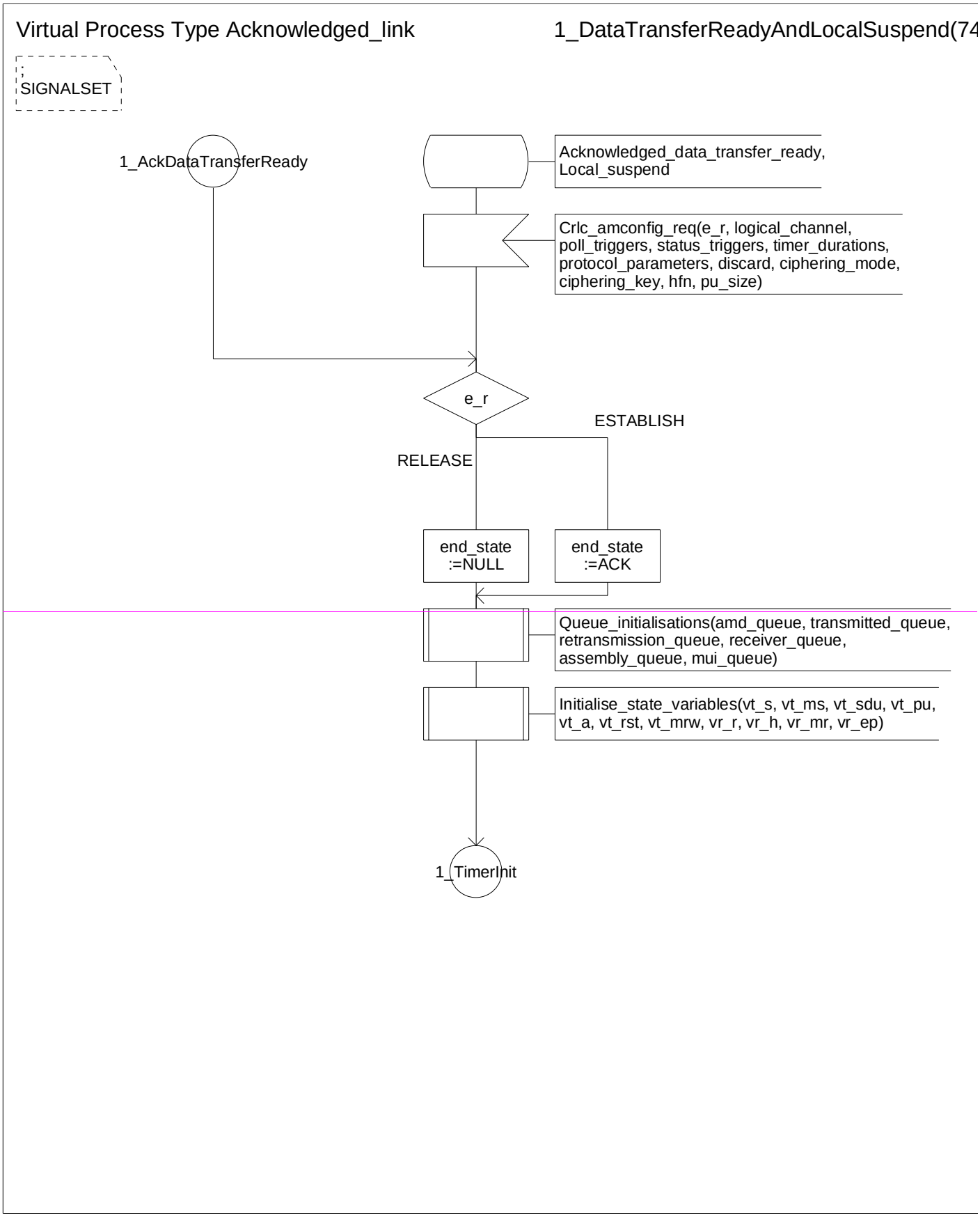

Virtual Process Type Acknowledged_link
1_ProcessTypeStart(74
SIGNALSET
Queue_initialisations(amd_queue, transmitted_queue, retransmission_queue, receiver_queue, assembly_queue, mui_queue)
Initialise_state_variables(vt_s, vt_ms, vt_sdu, vt_pu, vt_a, vt_rst, vt_mrw, vr_r, vr_h, vr_mr, vr_ep)
tx_rsn:=0, rx_rsn:=1
end_state:=NULL
1_TimerInit

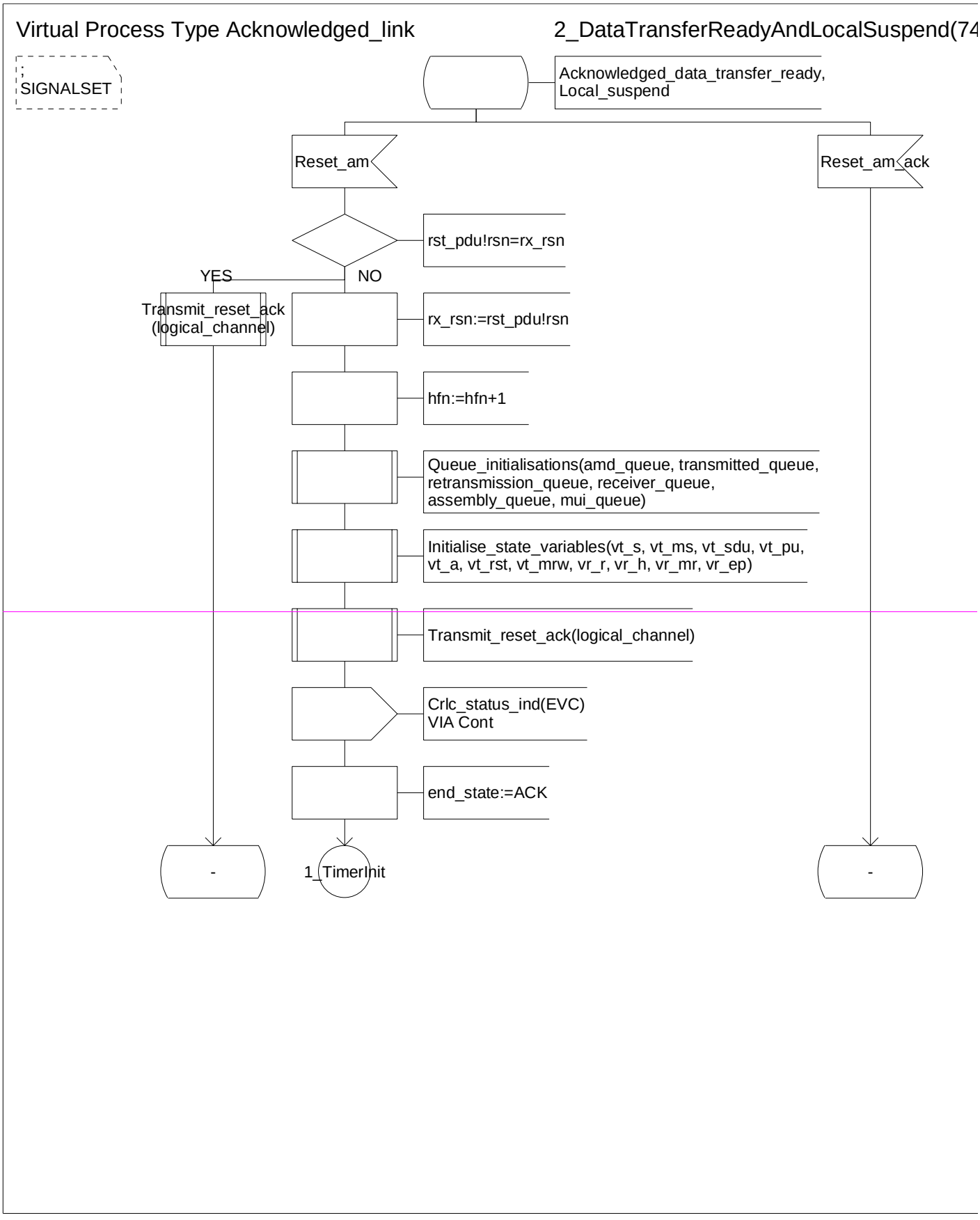

Virtual Process Type Acknowledged_link
1_TimerInit(74
SIGNALSET
1_TimerInit
status_periodic_active
YES
NO
Reset(Timer_Status_Periodic)
status_periodic_active:=NO
poll_periodic_active
NO
YES
Reset(Timer_Poll_Periodic)
poll_periodic_active:=NO
epc_active
YES
NO
Reset(Timer_EPC)
epc_active:=NO
poll_prohibit_active
YES
NO
Reset(Timer_Poll_Prohibit)
poll_prohibit_active:=NO
2_TimerInit

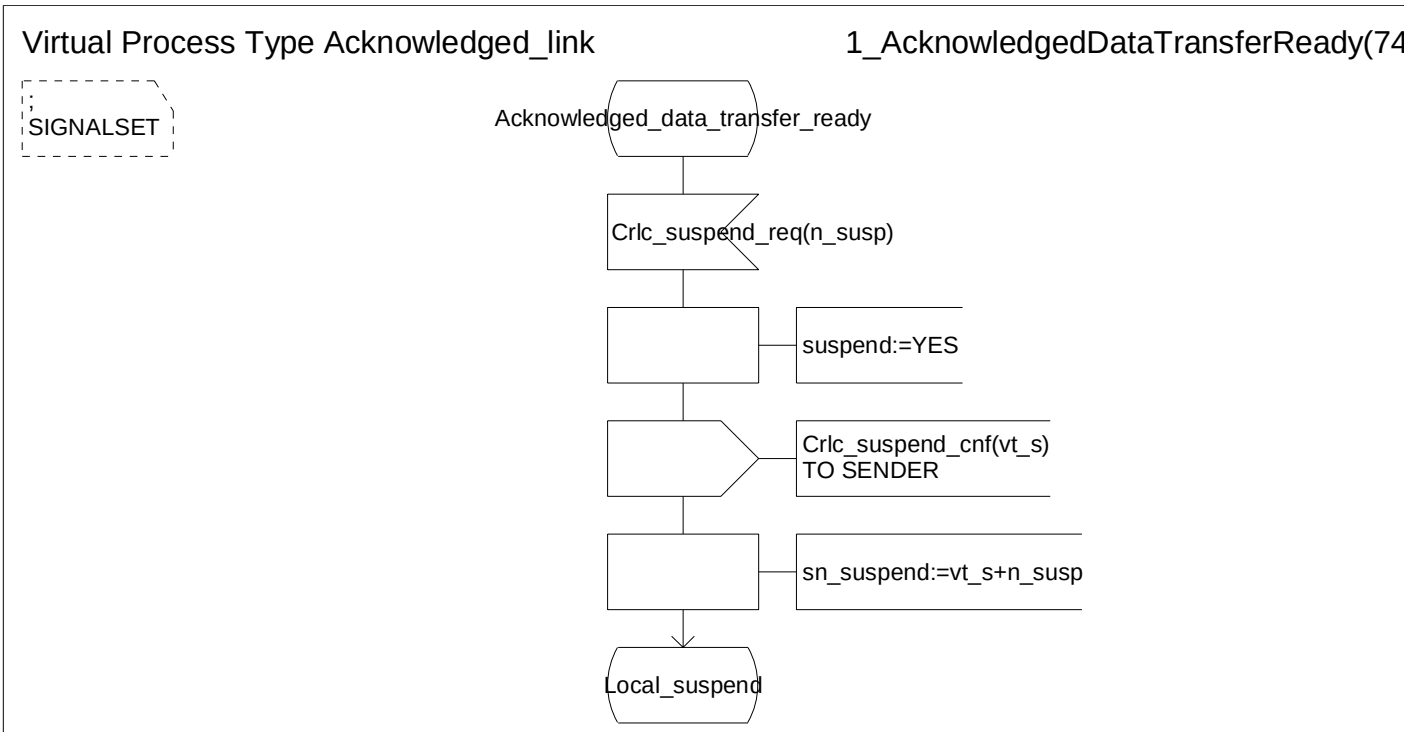

Virtual Process Type Acknowledged_link
2_TimerInit(74
SIGNALSET
2_TimerInit
Check_and_delete_timer_discards
(mui_queue, mui, empty)
empty
YES
NO
Reset(Timer_Discard(mui))
mrw_active
YES
NO
Reset(Timer_MRW)
mrw_active:=NO
poll_active
NO
YES
Reset(Timer_Poll)
poll_active:=NO,
polled_sn:=0
3_TimerInit

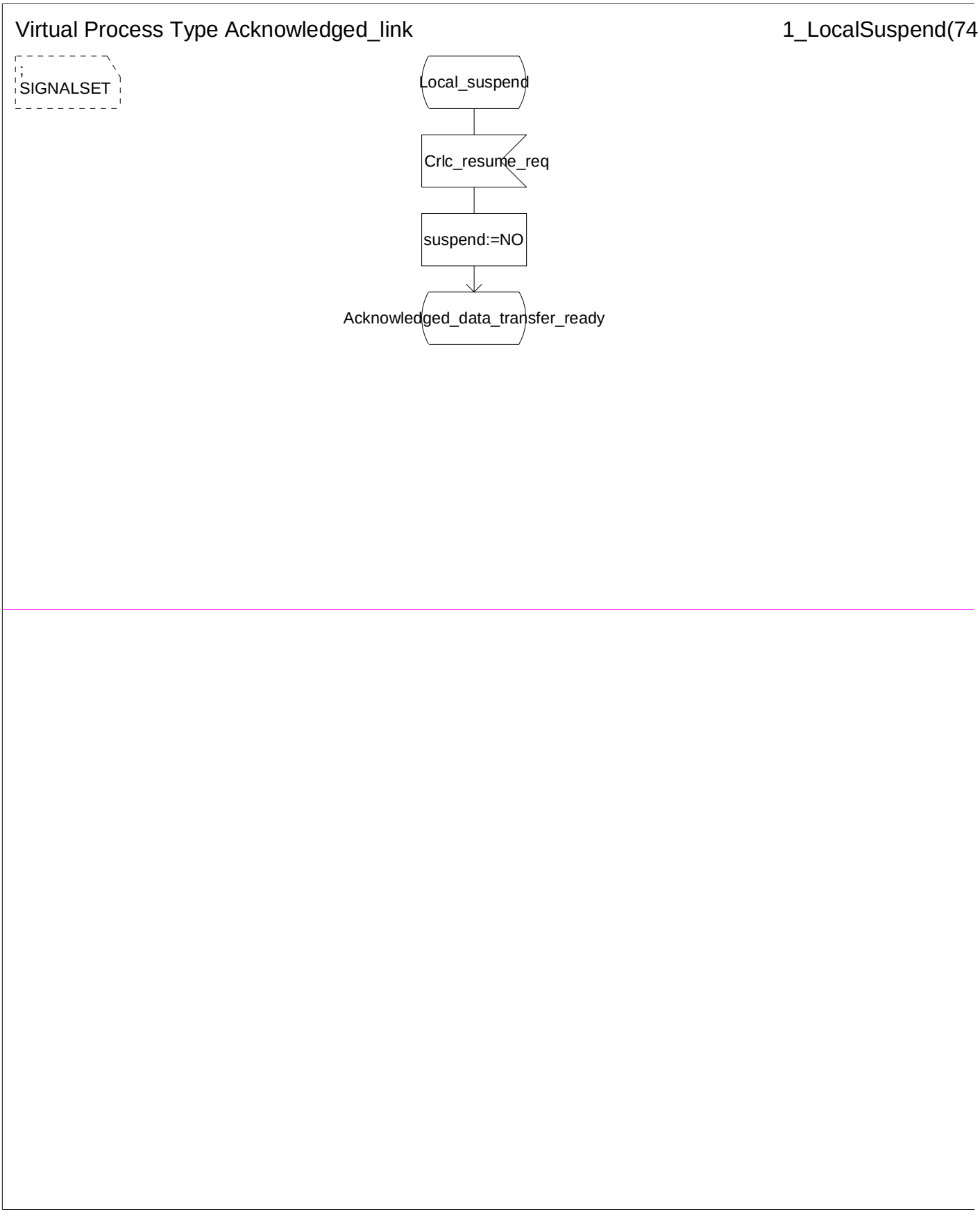

Virtual Process Type Acknowledged_link
3_TimerInit(74
SIGNALSET
3_TimerInit
status_prohibit_active
NO
YES
Reset(Timer_Status_Prohibit)
status_prohibit_active:=NO
rst_active
NO
YES
Reset(Timer_RST)
rst_active:=NO
end_state
NULL
ACK
RST
Set(NOW+timer_durations!rst,
Timer_RST)
rst_active:=YES
1_Null
Null
Reset_pending

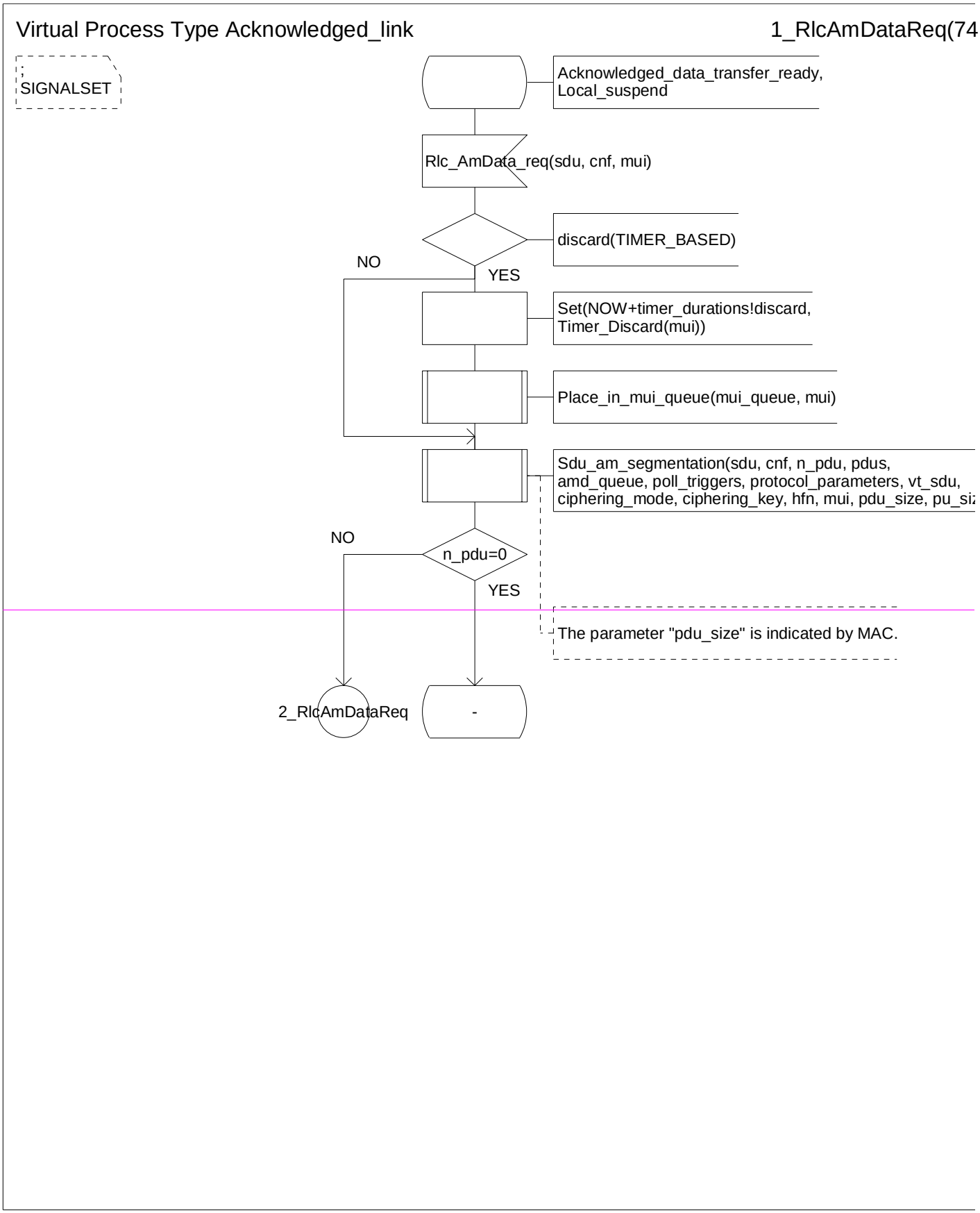

Virtual Process Type Acknowledged_link
1_Null(74
SIGNALSET
Null
1_Null
Crlc_amconfig_req(e_r, logical_channel, poll_triggers, status_triggers, timer_durations, protocol_parameters, discard, ciphering_mode, ciphering_key, hfn, pu_size)
e_r
RELEASE
ESTABLISH
The receive window can be updated dynamically according to the status of the receiver.
transmit_window:=protocol_parameters!window_size, receive_window:=protocol_parameters!window_size
vt_ms:=vt_s+transmit_window, vr_mr:=vr_r+receive_window
poll_triggers(TIMER_BASED)
NO
YES
Set(NOW+timer_durations!poll_periodic, Timer_Poll_Periodic)
poll_periodic_active:=YES
status_triggers(TIMER_BASED)
NO
YES
Set(NOW+timer_durations!status_periodic, Timer_Status_Periodic)
status_periodic_active:=YES
-
Acknowledged_data_transfer_ready

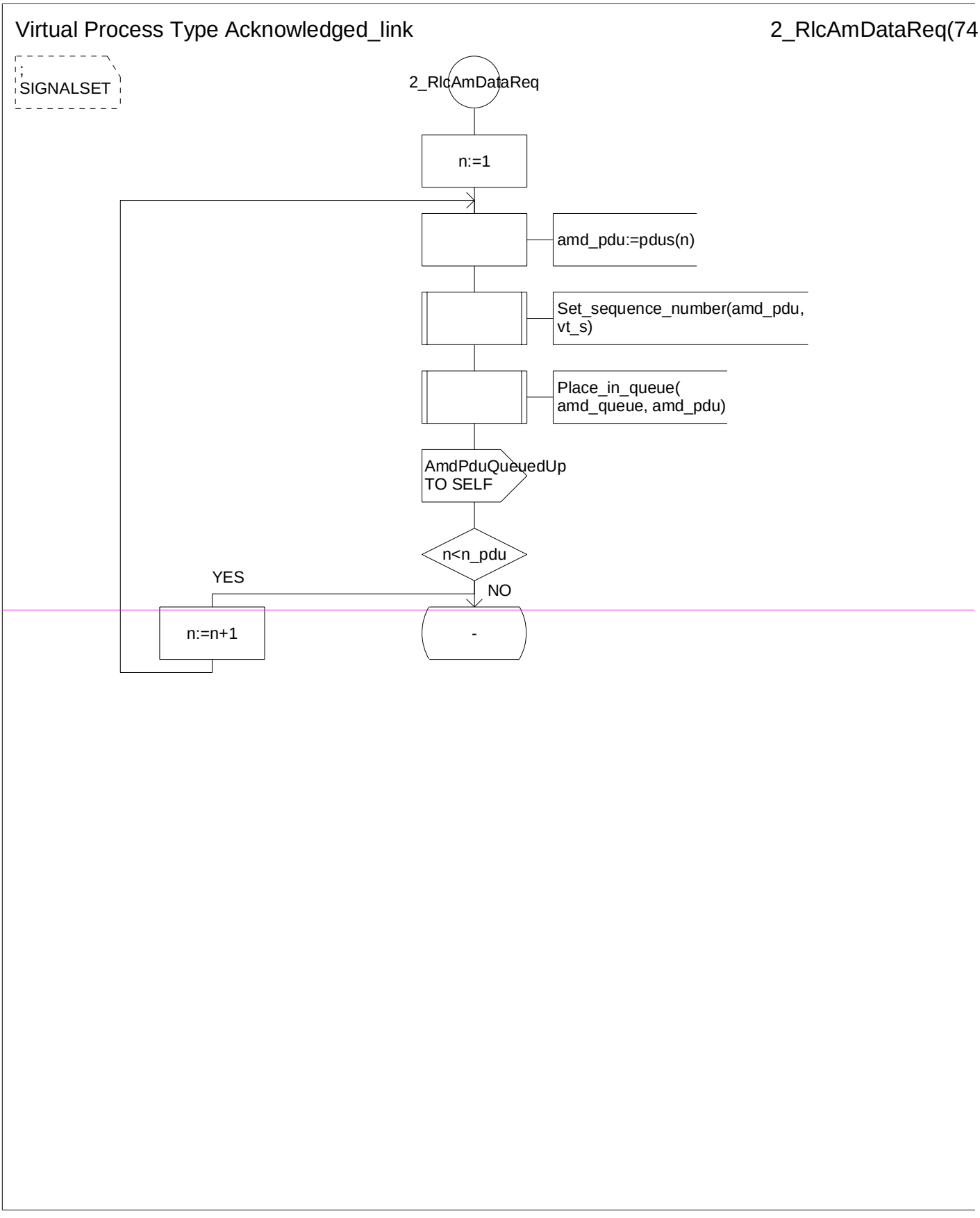

Virtual Process Type Acknowledged_link
1_DataTransferReadyAndLocalSuspend(74
SIGNALSET
1_AckDataTransferReady
Acknowledged_data_transfer_ready, Local_suspend
Crlc_amconfig_req(e_r, logical_channel, poll_triggers, status_triggers, timer_durations, protocol_parameters, discard, ciphering_mode, ciphering_key, hfn, pu_size)
e_r
ESTABLISH
RELEASE
end_state :=NULL
end_state :=ACK
Queue_initialisations(amd_queue, transmitted_queue, retransmission_queue, receiver_queue, assembly_queue, mui_queue)
Initialise_state_variables(vt_s, vt_ms, vt_sdu, vt_pu, vt_a, vt_rst, vt_mrw, vr_r, vr_h, vr_mr, vr_ep)
1_TimerInit

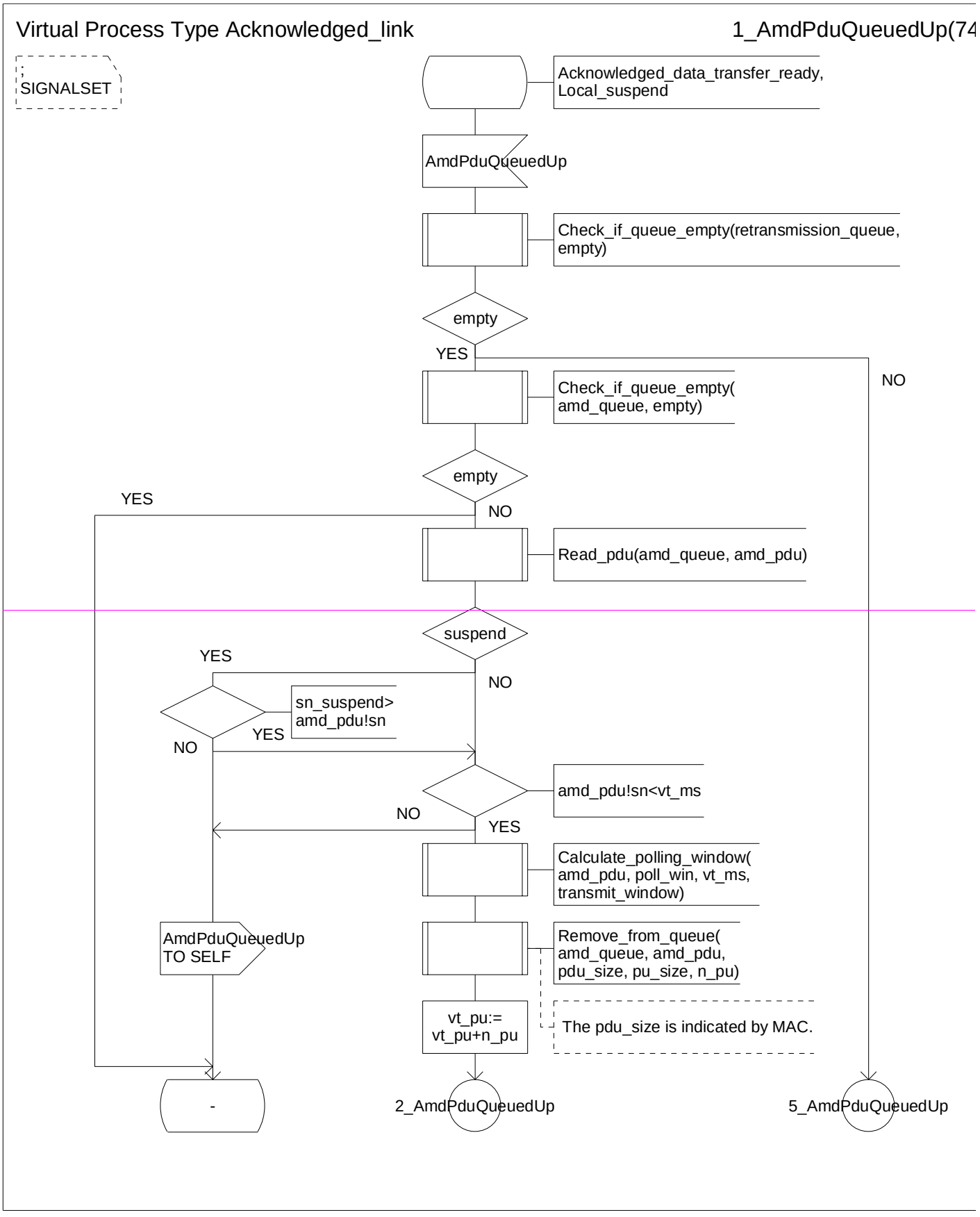

Virtual Process Type Acknowledged_link
2_DataTransferReadyAndLocalSuspend(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Reset_am
Reset_am_ack
rst_pdu!rsn=rx_rsn
YES
NO
Transmit_reset_ack
(logical_channel)
rx_rsn:=rst_pdu!rsn
hfn:=hfn+1
Queue_initialisations(amd_queue, transmitted_queue,
retransmission_queue, receiver_queue,
assembly_queue, mui_queue)
Initialise_state_variables(vt_s, vt_ms, vt_sdu, vt_pu,
vt_a, vt_rst, vt_mrw, vr_r, vr_h, vr_mr, vr_ep)
Transmit_reset_ack(logical_channel)
Crlc_status_ind(EVC)
VIA Cont
end_state:=ACK
-
1_TimerInit
-

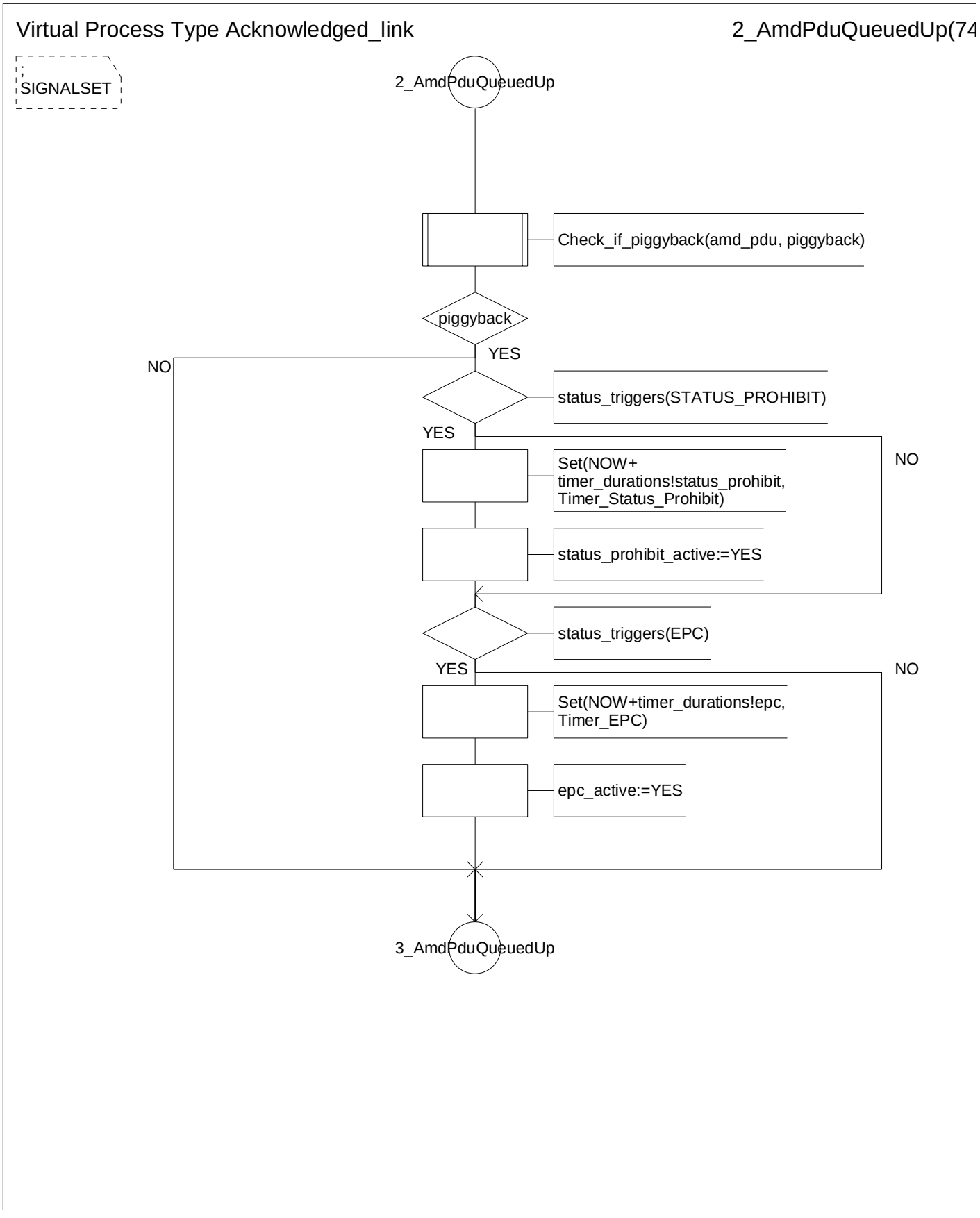

Virtual Process Type Acknowledged_link
1_AcknowledgedDataTransferReady(74
SIGNALSET
Acknowledged_data_transfer_ready
Crlc_suspend_req(n_susp)
suspend:=YES
Crlc_suspend_cnf(vt_s)
TO SENDER
sn_suspend:=vt_s+n_susp
Local_suspend

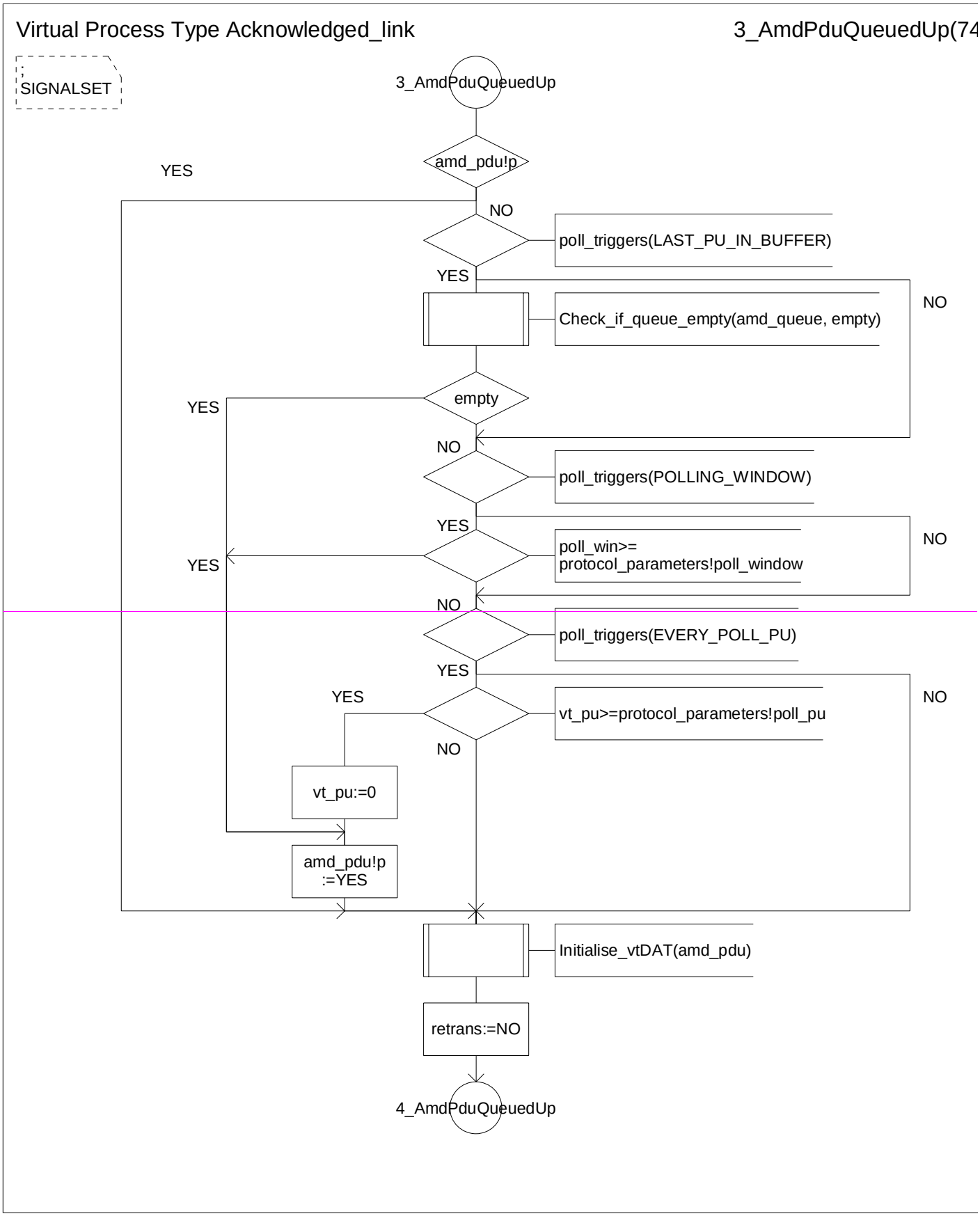

Virtual Process Type Acknowledged_link
1_LocalSuspend(74
SIGNALSET
Local_suspend
Crlc_resume_req
suspend:=NO
Acknowledged_data_transfer_ready

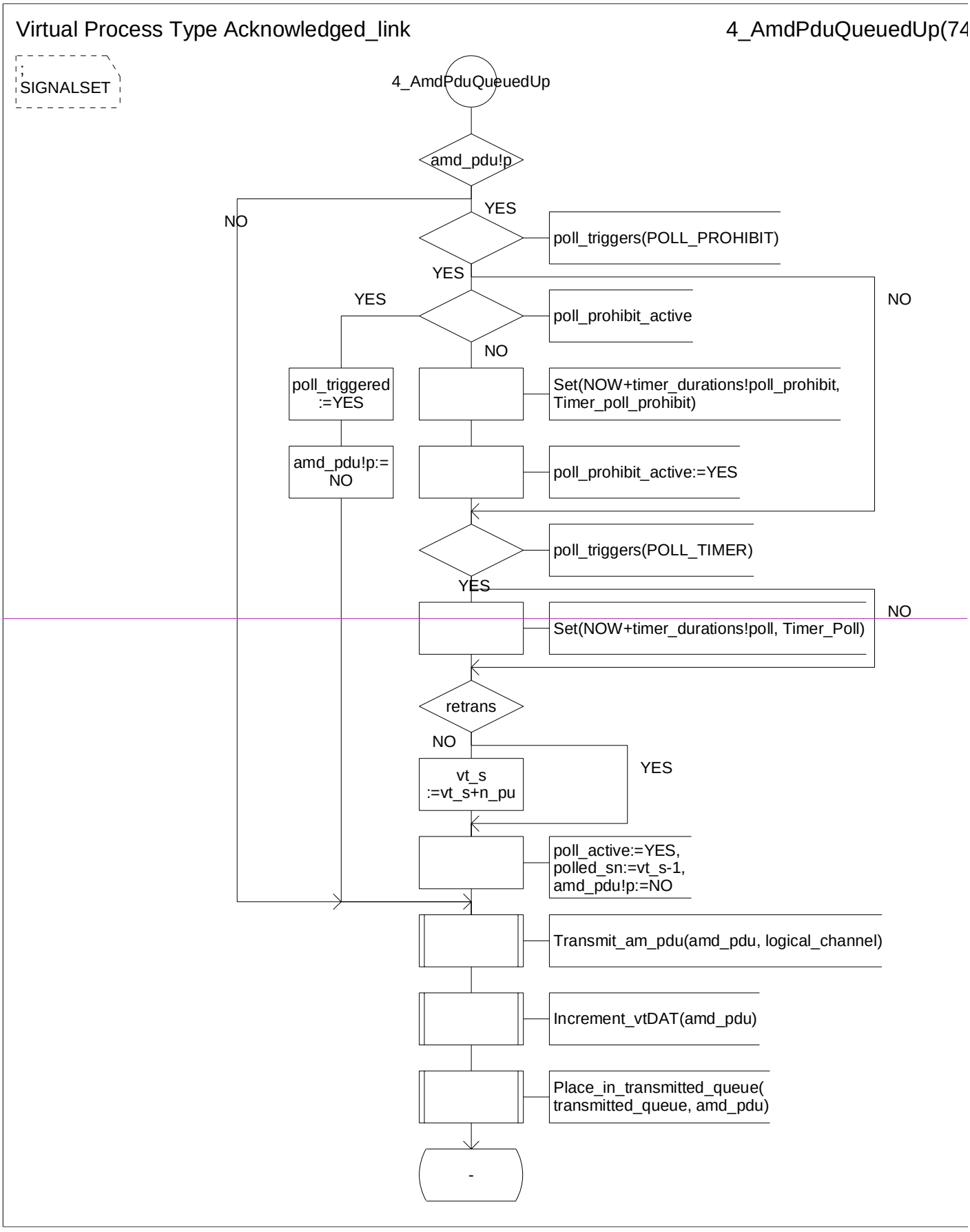

Virtual Process Type Acknowledged_link
1_RlcAmDataReq(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Rlc_AmData_req(sdu, cnf, mui)
discard(TIMER_BASED)
NO
YES
Set(NOW+timer_durations!discard,
Timer_Discard(mui))
Place_in_mui_queue(mui_queue, mui)
Sdu_am_segmentation(sdu, cnf, n_pdu, pdus,
amd_queue, poll_triggers, protocol_parameters, vt_sdu,
ciphering_mode, ciphering_key, hfn, mui, pdu_size, pu_siz
NO
n_pdu=0
YES
The parameter "pdu_size" is indicated by MAC.
2_RlcAmDataReq
-

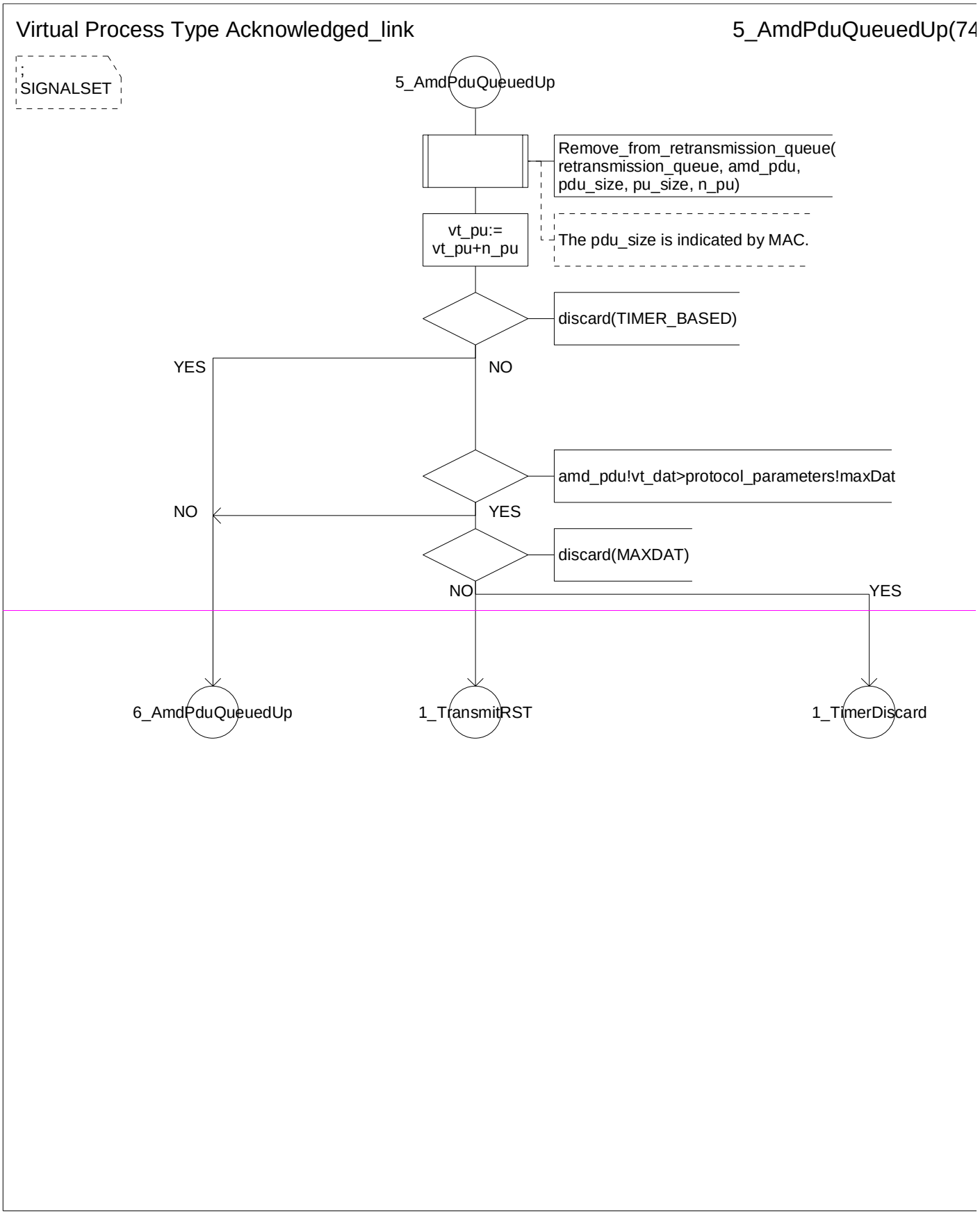

Virtual Process Type Acknowledged_link
2_RlcAmDataReq(74
SIGNALSET
2_RlcAmDataReq
n:=1
amd_pdu:=pdus(n)
Set_sequence_number(amd_pdu, vt_s)
Place_in_queue(amd_queue, amd_pdu)
AmdPduQueuedUp TO SELF
n<n_pdu
YES
NO
n:=n+1
-


Virtual Process Type Acknowledged_link
1_AmdPduQueuedUp(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
AmdPduQueuedUp
Check_if_queue_empty(retransmission_queue,
empty)
empty
YES
NO
Check_if_queue_empty(
amd_queue, empty)
empty
YES
NO
Read_pdu(amd_queue, amd_pdu)
suspend
YES
NO
sn_suspend>
amd_pdu!sn
YES
NO
amd_pdu!sn<vt_ms
NO
YES
Calculate_polling_window(
amd_pdu, poll_win, vt_ms,
transmit_window)
AmdPduQueuedUp
TO SELF
Remove_from_queue(
amd_queue, amd_pdu,
pdu_size, pu_size, n_pu)
vt_pu:=
vt_pu+n_pu
The pdu_size is indicated by MAC.
-
2_AmdPduQueuedUp
5_AmdPduQueuedUp

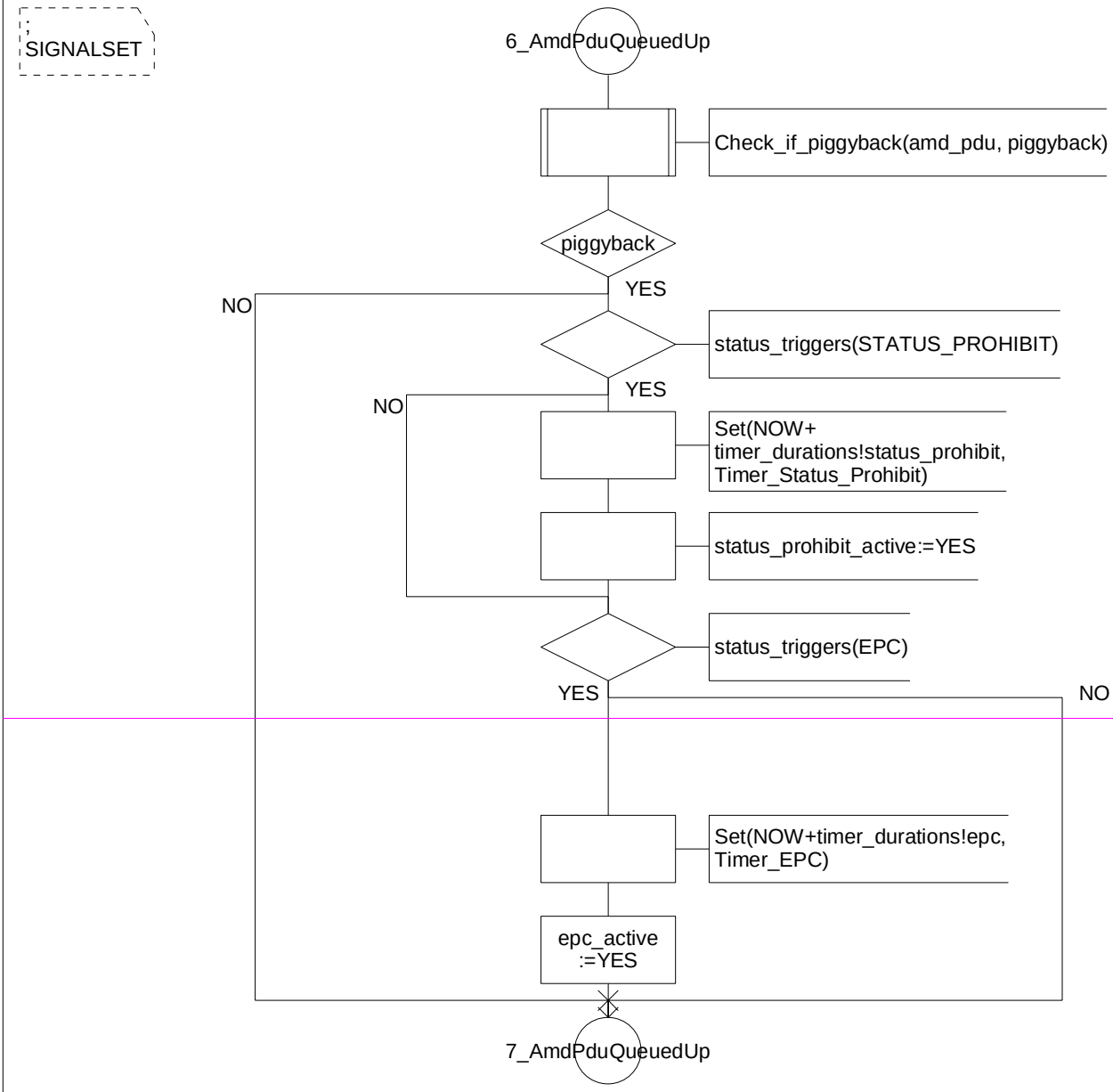

Virtual Process Type Acknowledged_link
2_AmdPduQueuedUp(74
SIGNALSET
2_AmdPduQueuedUp
Check_if_piggyback(amd_pdu, piggyback)
piggyback
YES
NO
status_triggers(STATUS_PROHIBIT)
YES
NO
Set(NOW+
timer_durations!status_prohibit,
Timer_Status_Prohibit)
status_prohibit_active:=YES
status_triggers(EPC)
YES
NO
Set(NOW+timer_durations!epc,
Timer_EPC)
epc_active:=YES
3_AmdPduQueuedUp


Virtual Process Type Acknowledged_link
3_AmdPduQueuedUp(74
SIGNALSET
3_AmdPduQueuedUp
amd_pdu!p
YES
NO
poll_triggers(LAST_PU_IN_BUFFER)
YES
NO
Check_if_queue_empty(amd_queue, empty)
empty
YES
NO
poll_triggers(POLLING_WINDOW)
YES
NO
poll_win>=
protocol_parameters!poll_window
YES
NO
poll_triggers(EVERY_POLL_PU)
YES
NO
vt_pu>=protocol_parameters!poll_pu
YES
NO
vt_pu:=0
amd_pdu!p
:=YES
Initialise_vtDAT(amd_pdu)
retrans:=NO
4_AmdPduQueuedUp

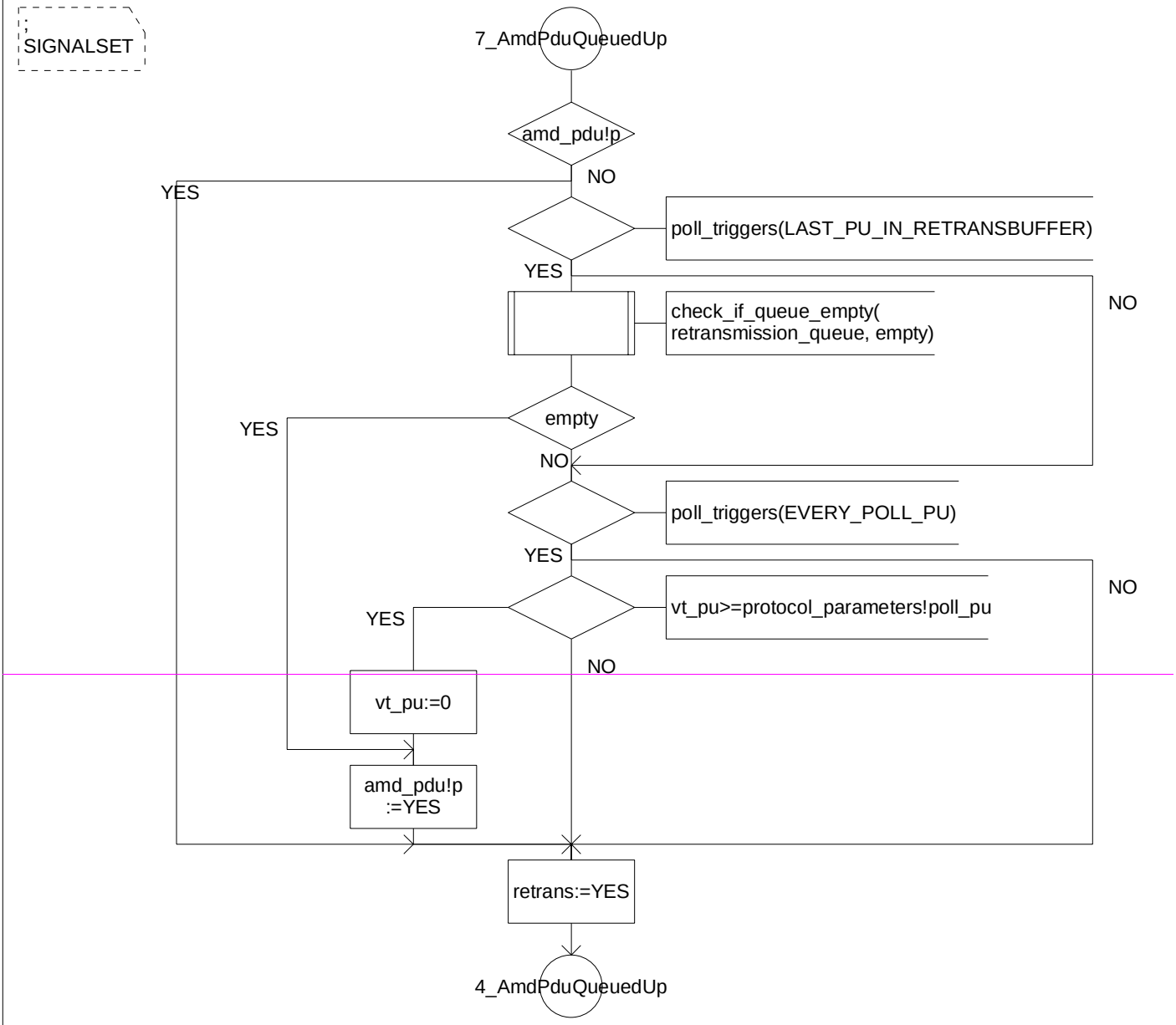

Virtual Process Type Acknowledged_link
4_AmdPduQueuedUp(74
SIGNALSET
4_AmdPduQueuedUp
amd_pdu!p
YES
NO
poll_triggers(POLL_PROHIBIT)
YES
YES
poll_prohibit_active
NO
NO
poll_triggered :=YES
Set(NOW+timer_durations!poll_prohibit, Timer_poll_prohibit)
amd_pdu!p:= NO
poll_prohibit_active:=YES
poll_triggers(POLL_TIMER)
YES
NO
Set(NOW+timer_durations!poll, Timer_Poll)
retrans
NO
YES
vt_s :=vt_s+n_pu
poll_active:=YES, polled_sn:=vt_s-1, amd_pdu!p:=NO
Transmit_am_pdu(amd_pdu, logical_channel)
Increment_vtDAT(amd_pdu)
Place_in_transmitted_queue(transmitted_queue, amd_pdu)
-

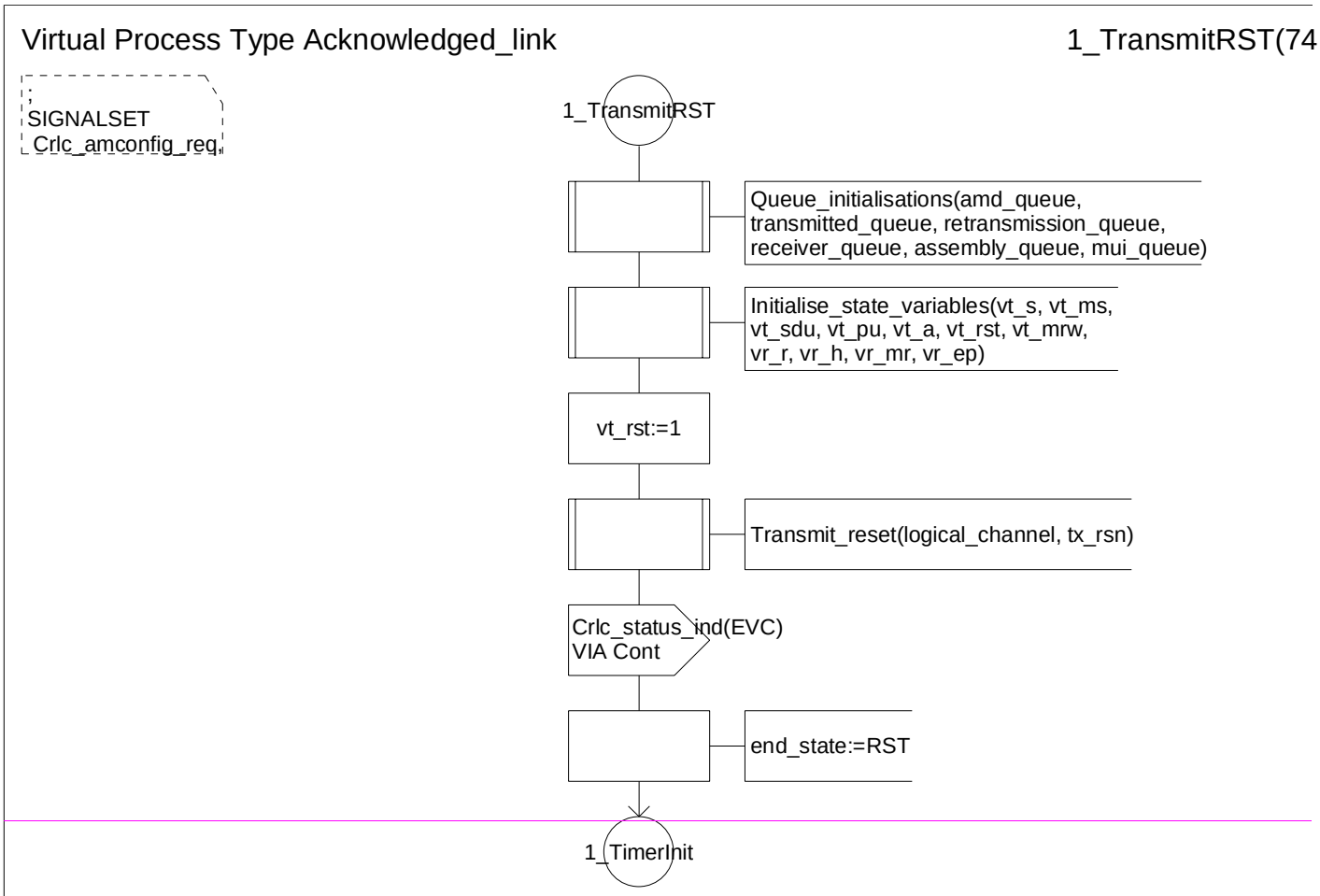

Virtual Process Type Acknowledged_link
5_AmdPduQueuedUp(74
SIGNALSET
5_AmdPduQueuedUp
Remove_from_retransmission_queue(
retransmission_queue, amd_pdu,
pdu_size, pu_size, n_pu)
vt_pu:=
vt_pu+n_pu
The pdu_size is indicated by MAC.
discard(TIMER_BASED)
YES
NO
amd_pdu!vt_dat>protocol_parameters!maxDat
NO
YES
discard(MAXDAT)
NO
YES
6_AmdPduQueuedUp
1_TransmitRST
1_TimerDiscard

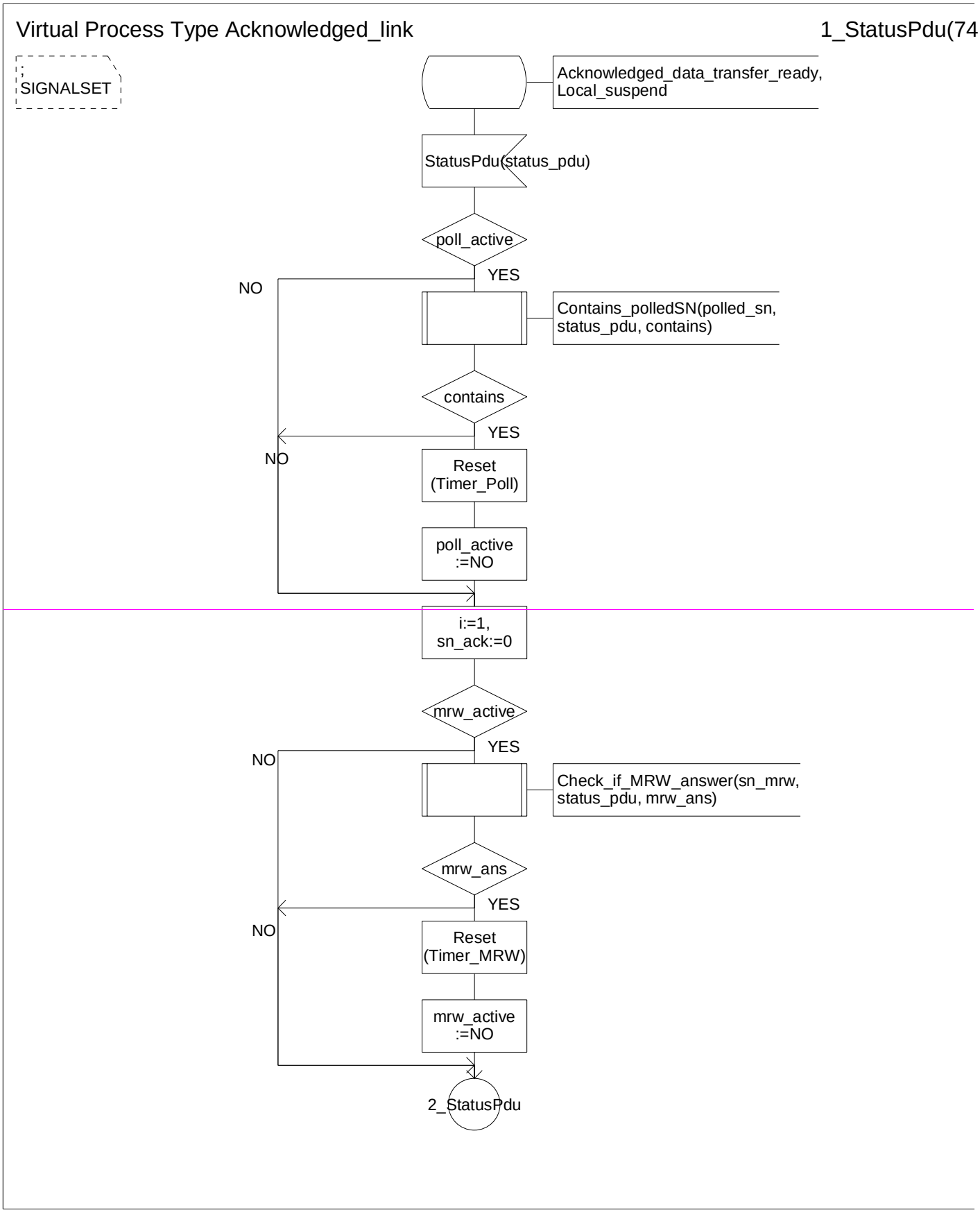
Virtual Process Type Acknowledged_link 6_AmdPduQueuedUp(74

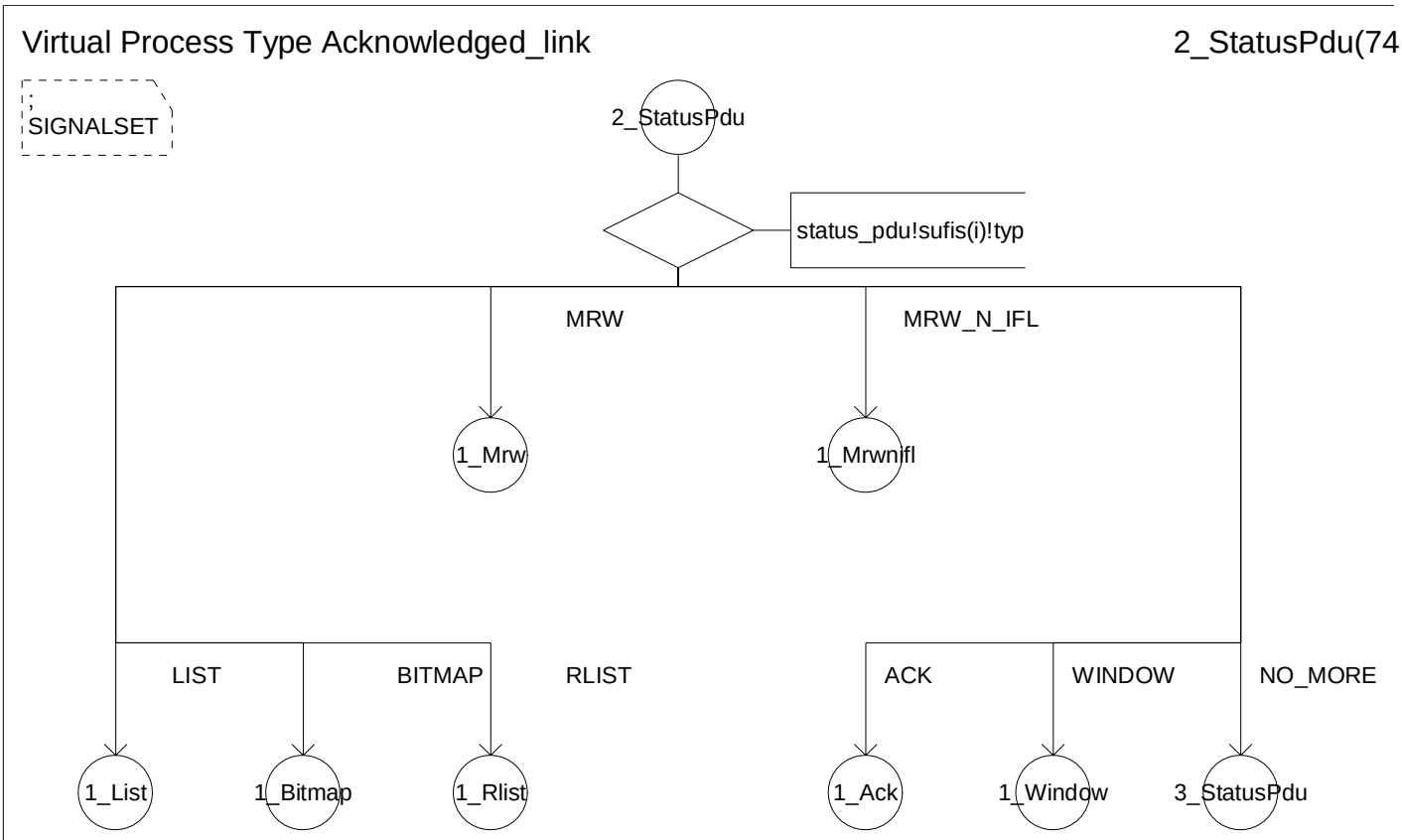


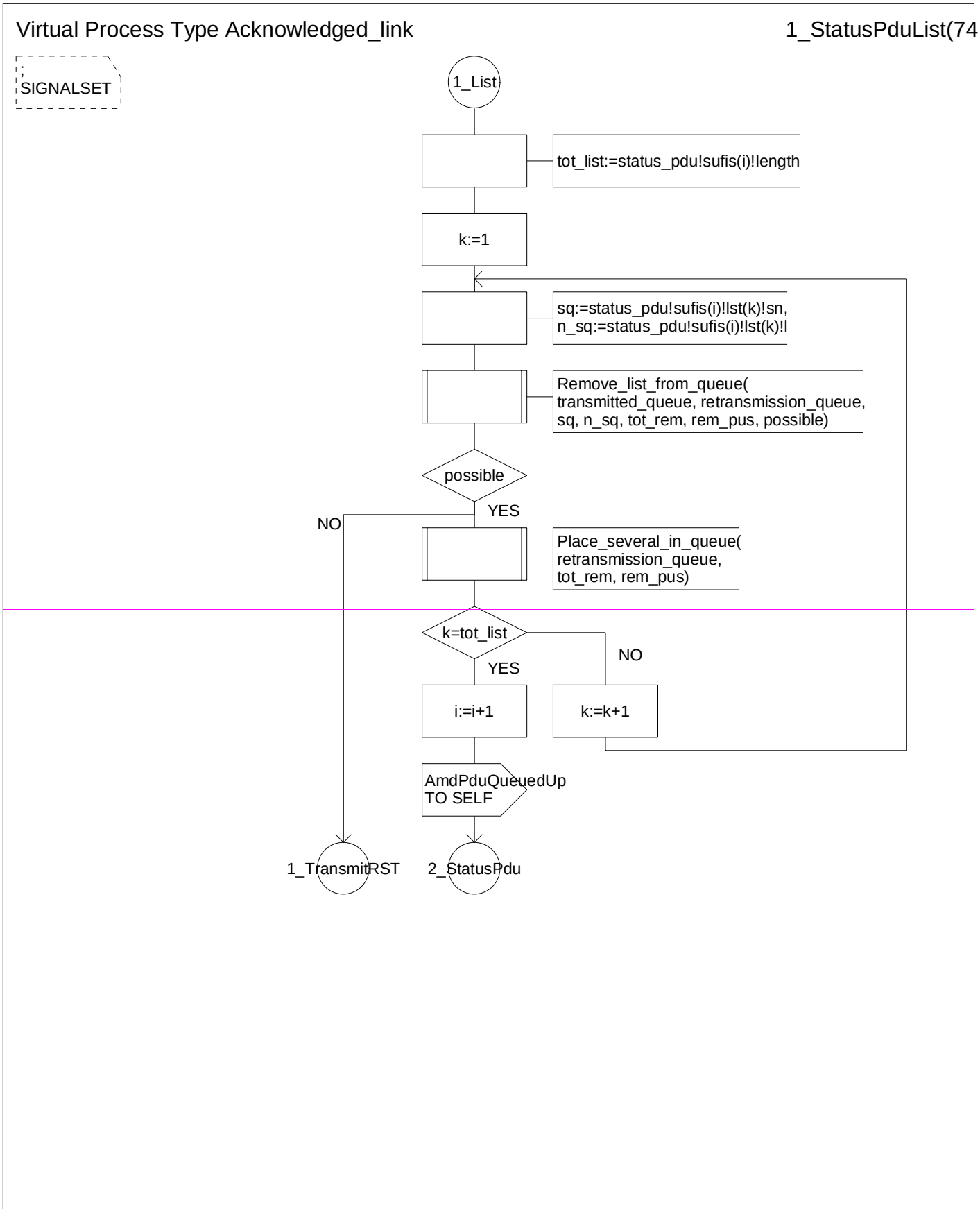
Virtual Process Type Acknowledged_link 7_AmdPduQueuedUp(74

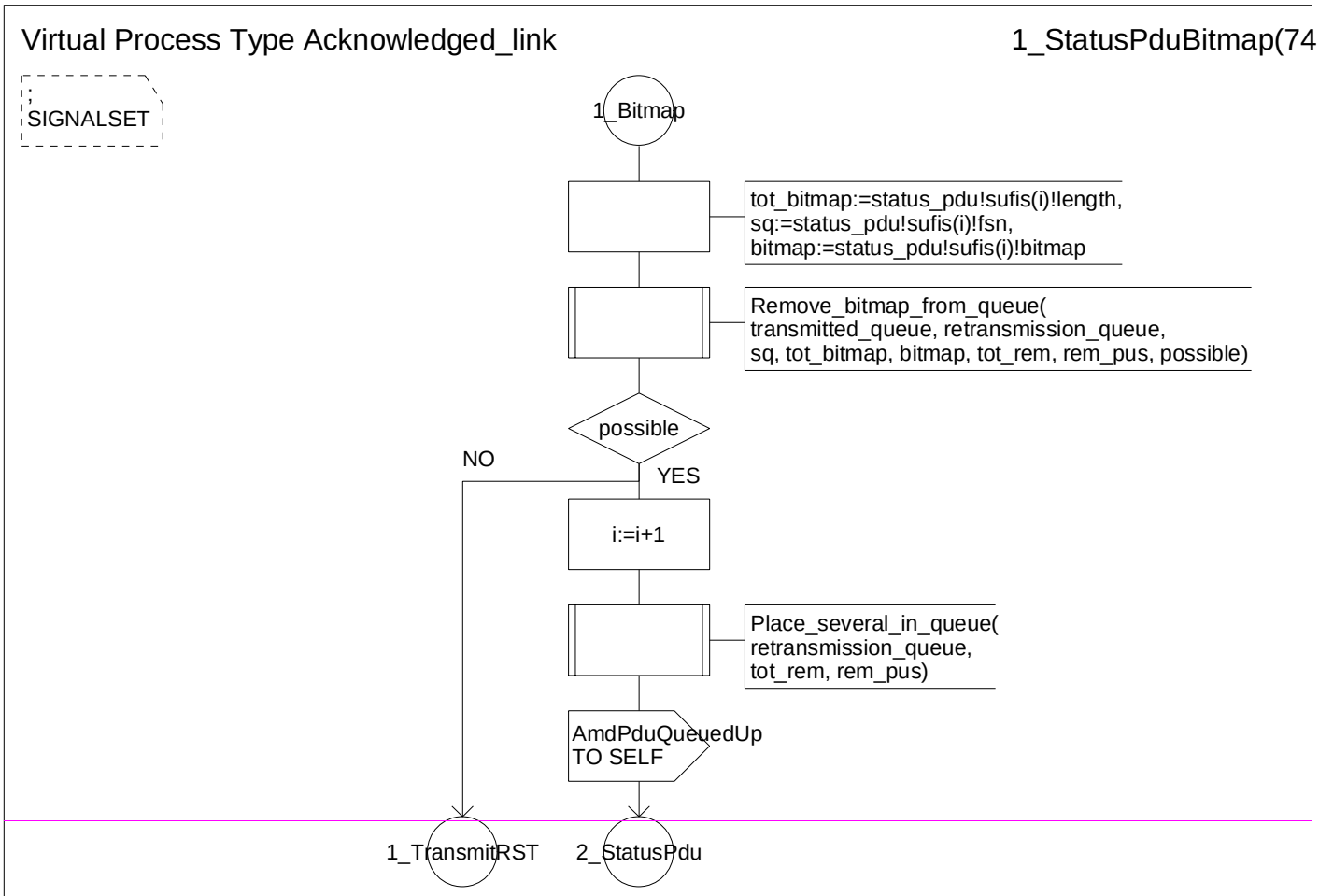


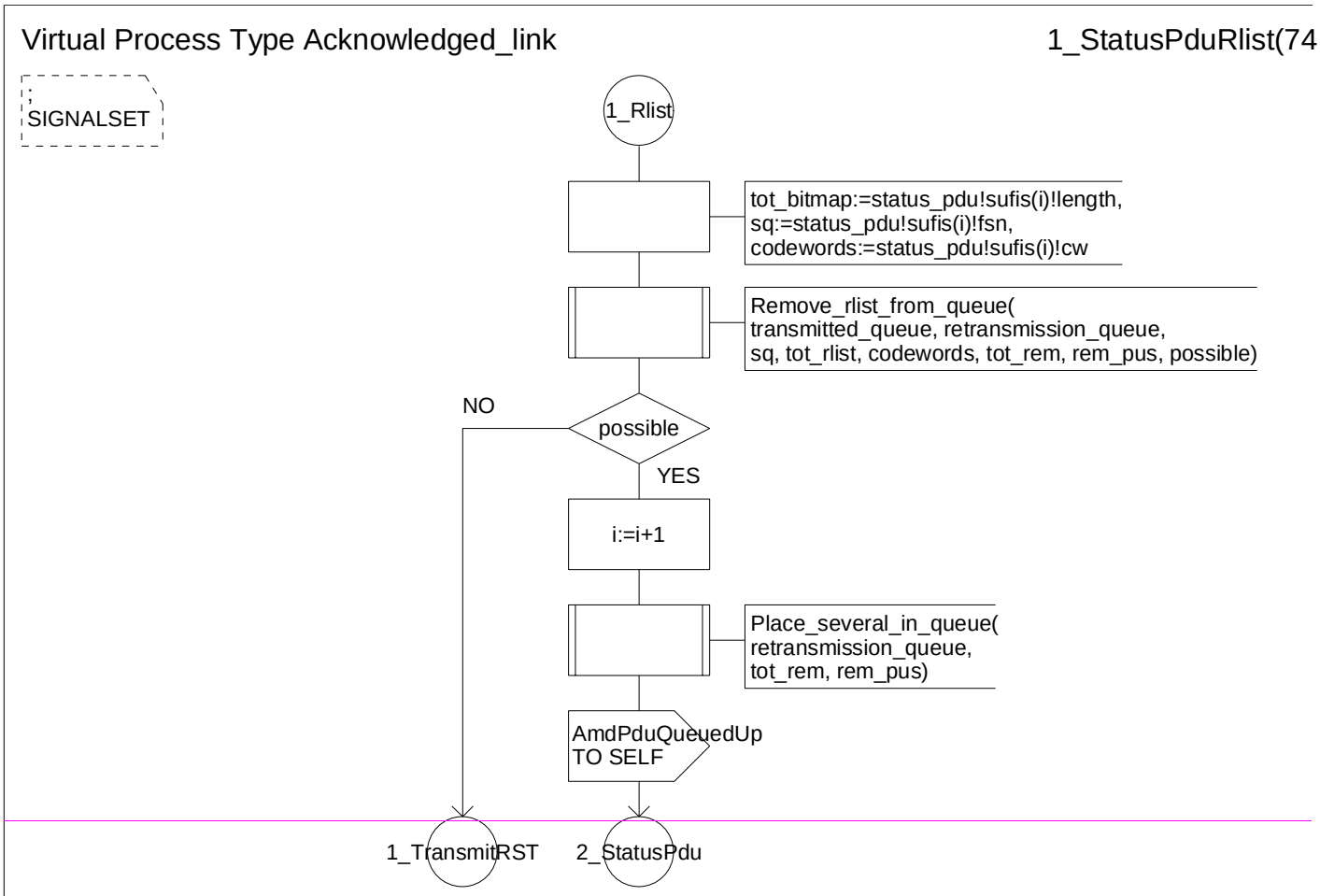

Virtual Process Type Acknowledged_link
1_TransmitRST(74
SIGNALSET
Crlc_amconfig_req,
1_TransmitRST
Queue_initialisations(amd_queue, transmitted_queue, retransmission_queue, receiver_queue, assembly_queue, mui_queue)
Initialise_state_variables(vt_s, vt_ms, vt_sdu, vt_pu, vt_a, vt_rst, vt_mrw, vr_r, vr_h, vr_mr, vr_ep)
vt_rst:=1
Transmit_reset(logical_channel, tx_rsn)
Crlc_status_ind(EVC)
VIA Cont
end_state:=RST
1_TimerInit


Virtual Process Type Acknowledged_link
1_StatusPdu(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
StatusPdu(status_pdu)
poll_active
YES
NO
Contains_polledSN(polled_sn,
status_pdu, contains)
contains
YES
NO
Reset
(Timer_Poll)
poll_active
:=NO
i:=1,
sn_ack:=0
mrw_active
YES
NO
Check_if_MRW_answer(sn_mrw,
status_pdu, mrw_ans)
mrw_ans
YES
NO
Reset
(Timer_MRW)
mrw_active
:=NO
2_StatusPdu

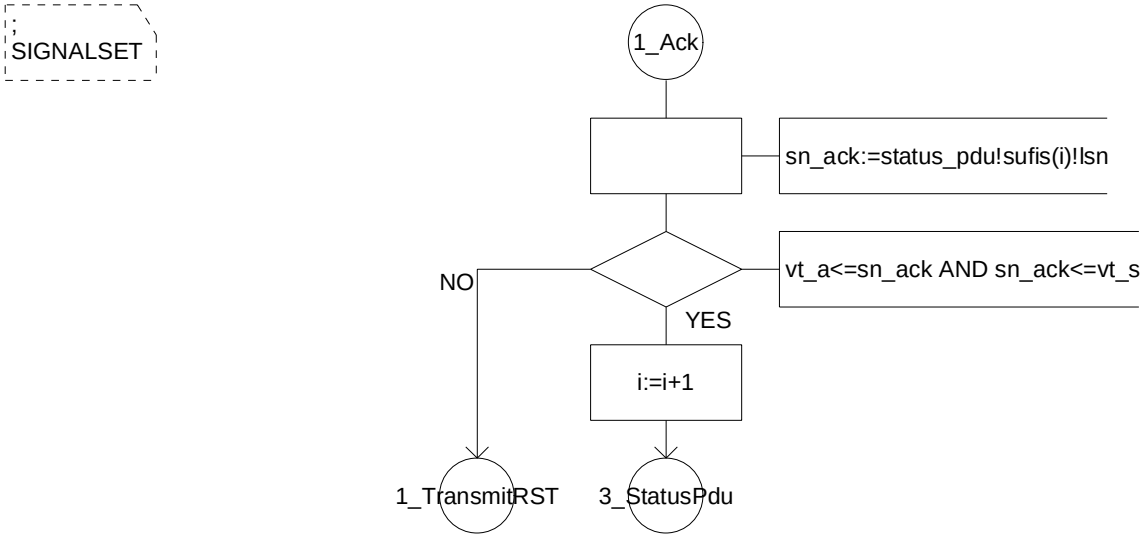

Virtual Process Type Acknowledged_link
2_StatusPdu(74
SIGNALSET
2_StatusPdu
status_pdu!sufis(i)!typ
MRW
MRW_N_IFL
1_Mrw
1_Mrwnifl
LIST
BITMAP
RLIST
ACK
WINDOW
NO_MORE
1_List
1_Bitmap
1_Rlist
1_Ack
1_Window
3_StatusPdu

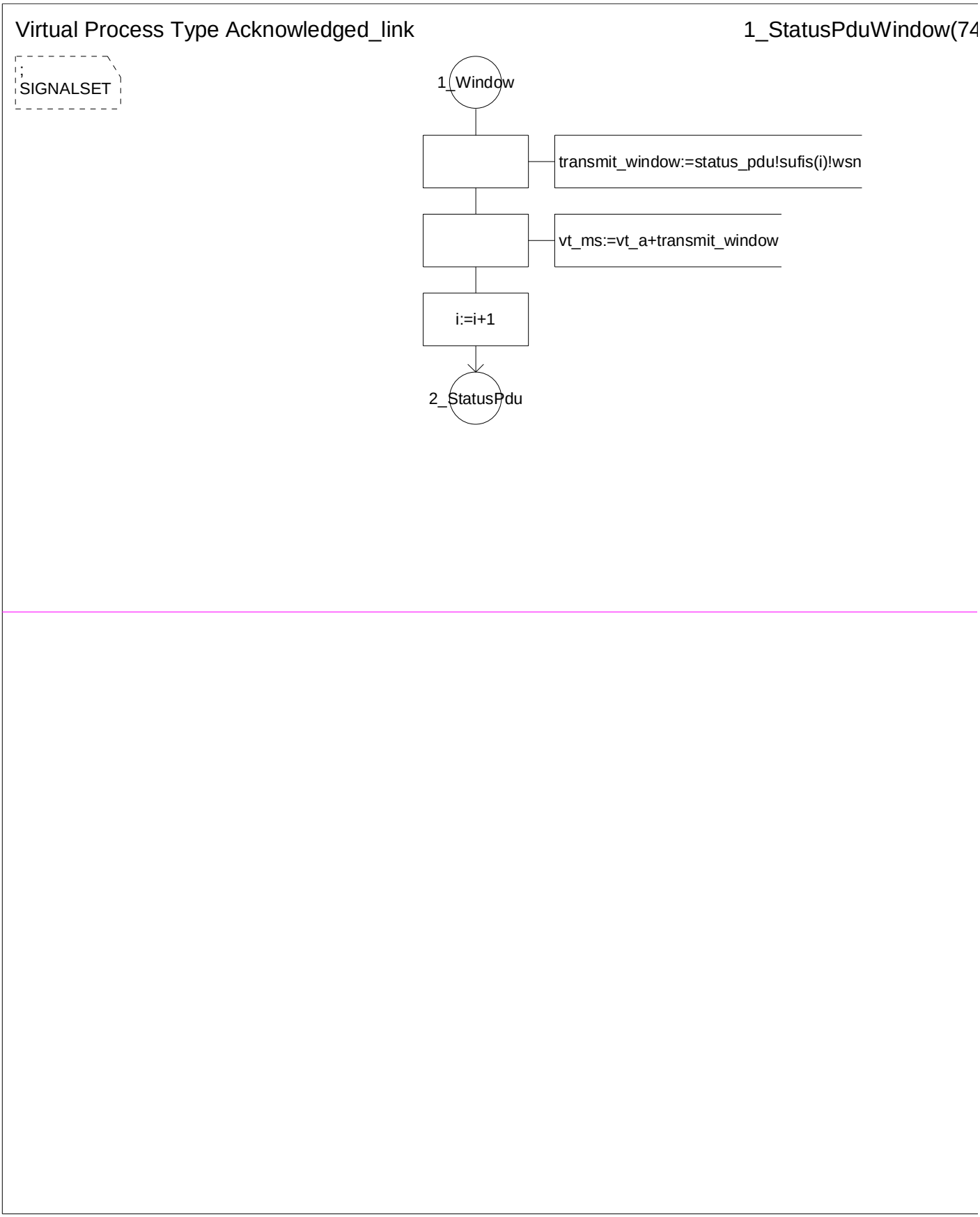

Virtual Process Type Acknowledged_link
1_StatusPduList(74
SIGNALSET
1_List
tot_list:=status_pdu!sufis(i)!length
k:=1
sq:=status_pdu!sufis(i)!lst(k)!sn,
n_sq:=status_pdu!sufis(i)!lst(k)!l
Remove_list_from_queue(
transmitted_queue, retransmission_queue,
sq, n_sq, tot_rem, rem_pus, possible)
possible
YES
NO
Place_several_in_queue(
retransmission_queue,
tot_rem, rem_pus)
k=tot_list
YES
NO
i:=i+1
k:=k+1
AmdPduQueuedUp
TO SELF
1_TransmitRST
2_StatusPdu

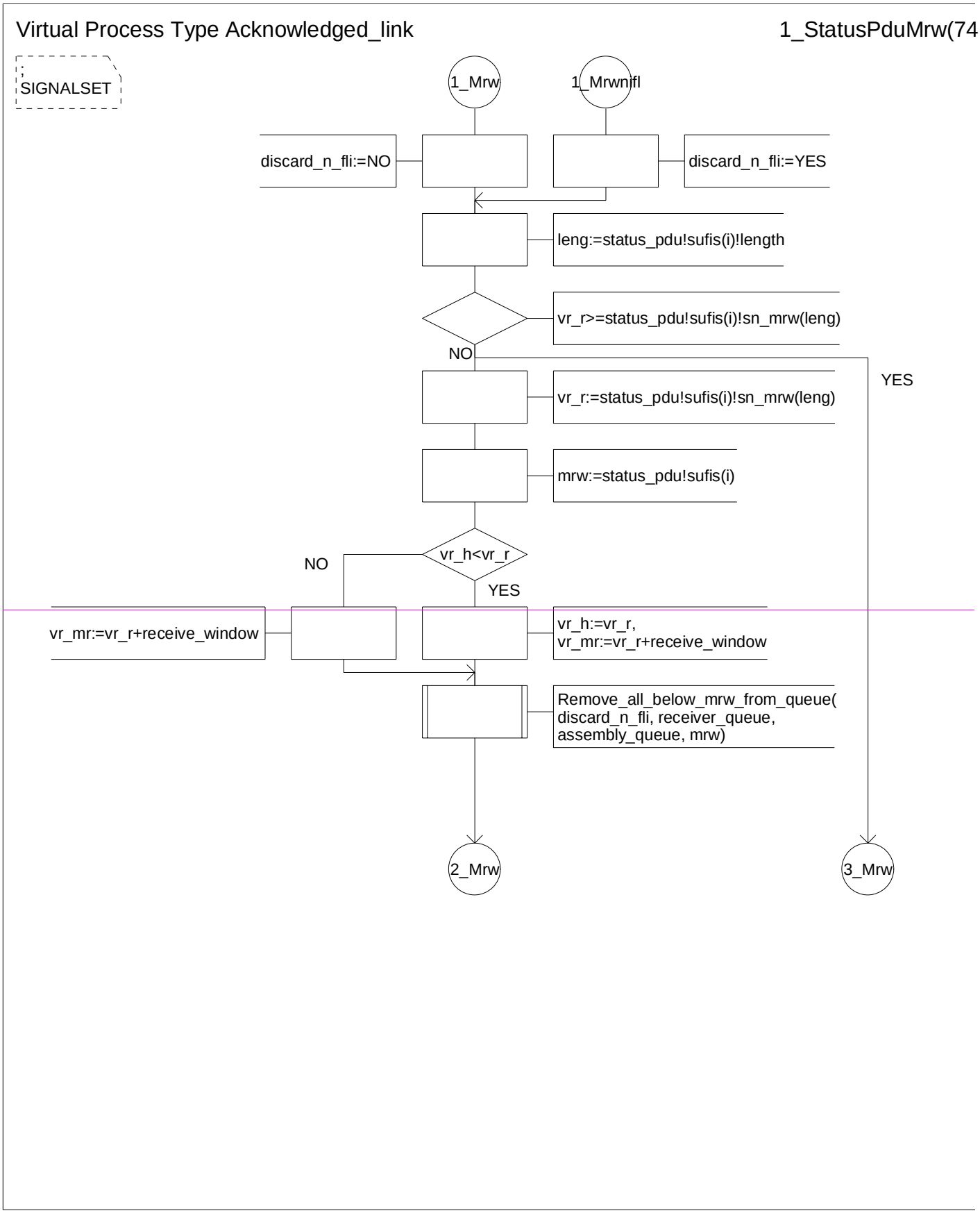

Virtual Process Type Acknowledged_link
1_StatusPduBitmap(74
SIGNALSET
1_Bitmap
tot_bitmap:=status_pdu!sufis(i)!length,
sq:=status_pdu!sufis(i)!fsn,
bitmap:=status_pdu!sufis(i)!bitmap
Remove_bitmap_from_queue(
transmitted_queue, retransmission_queue,
sq, tot_bitmap, bitmap, tot_rem, rem_pus, possible)
possible
NO
YES
i:=i+1
Place_several_in_queue(
retransmission_queue,
tot_rem, rem_pus)
AmdPduQueuedUp
TO SELF
1_TransmitRST
2_StatusPdu

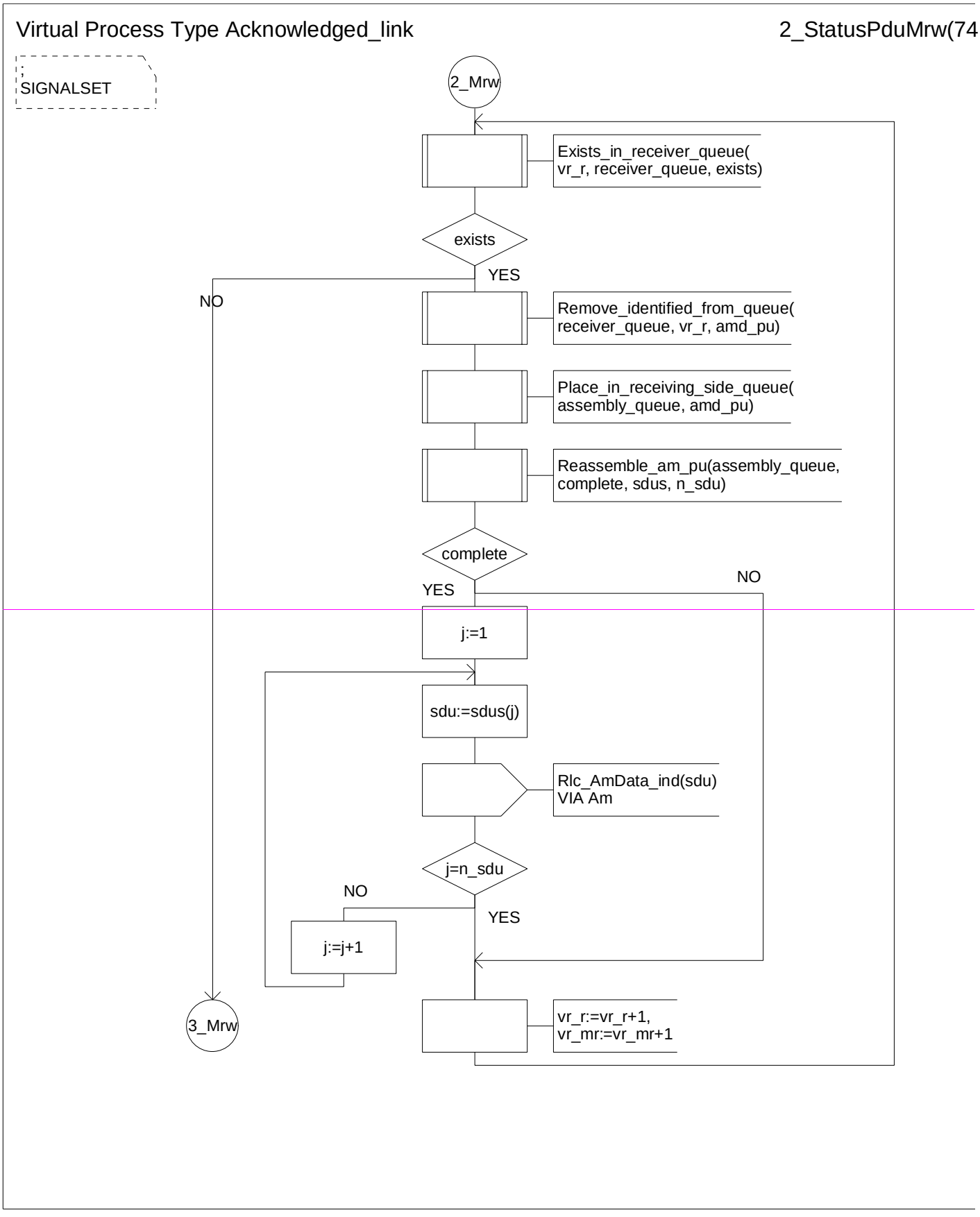

Virtual Process Type Acknowledged_link
1_StatusPduRlist(74
SIGNALSET
1_Rlist
tot_bitmap:=status_pdu!sufis(i)!length,
sq:=status_pdu!sufis(i)!fsn,
codewords:=status_pdu!sufis(i)!cw
Remove_rlist_from_queue(
transmitted_queue, retransmission_queue,
sq, tot_rlist, codewords, tot_rem, rem_pus, possible)
NO
possible
YES
i:=i+1
Place_several_in_queue(
retransmission_queue,
tot_rem, rem_pus)
AmdPduQueuedUp
TO SELF
1_TransmitRST
2_StatusPdu

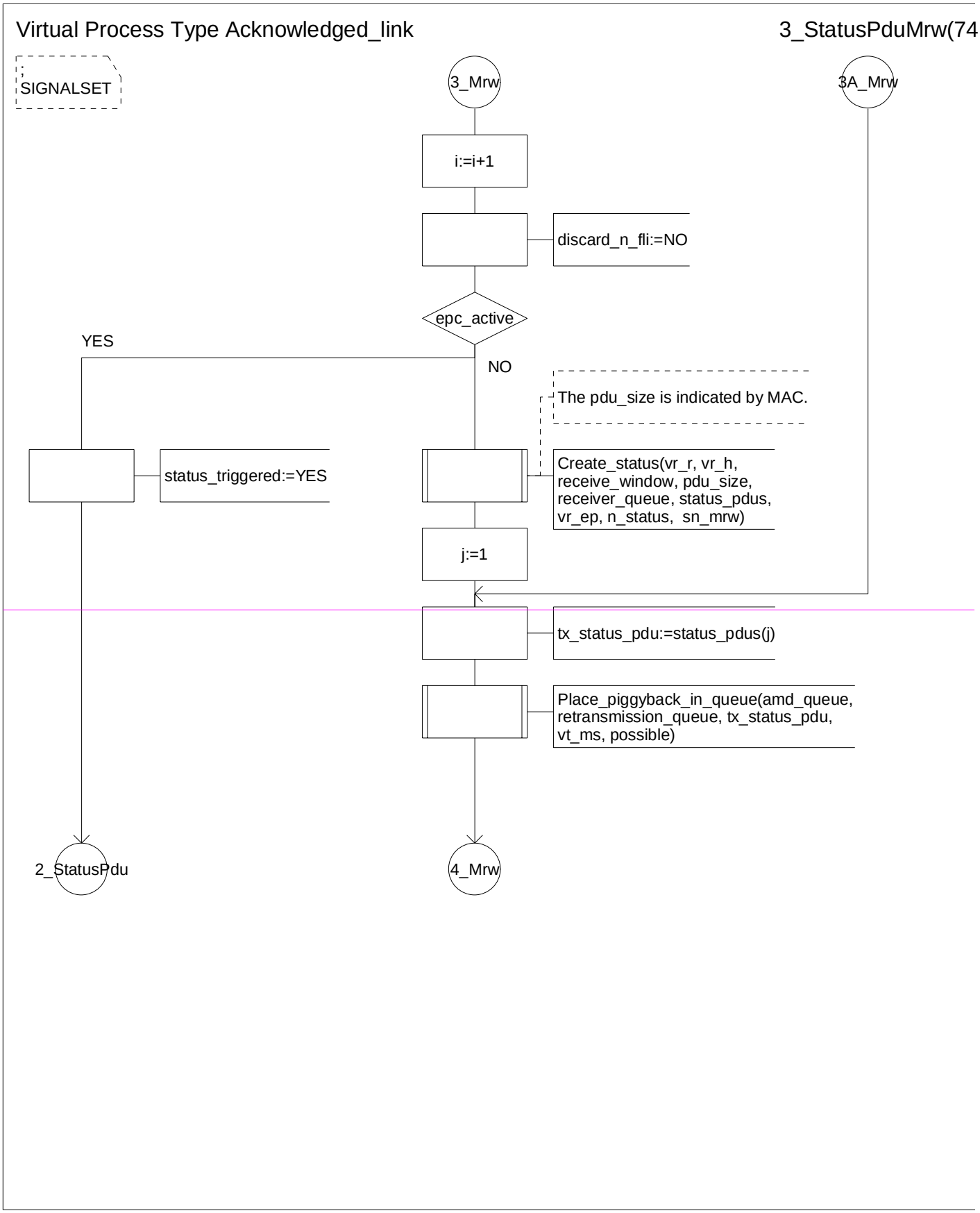
Virtual Process Type Acknowledged_link 1_StatusPduAck(74



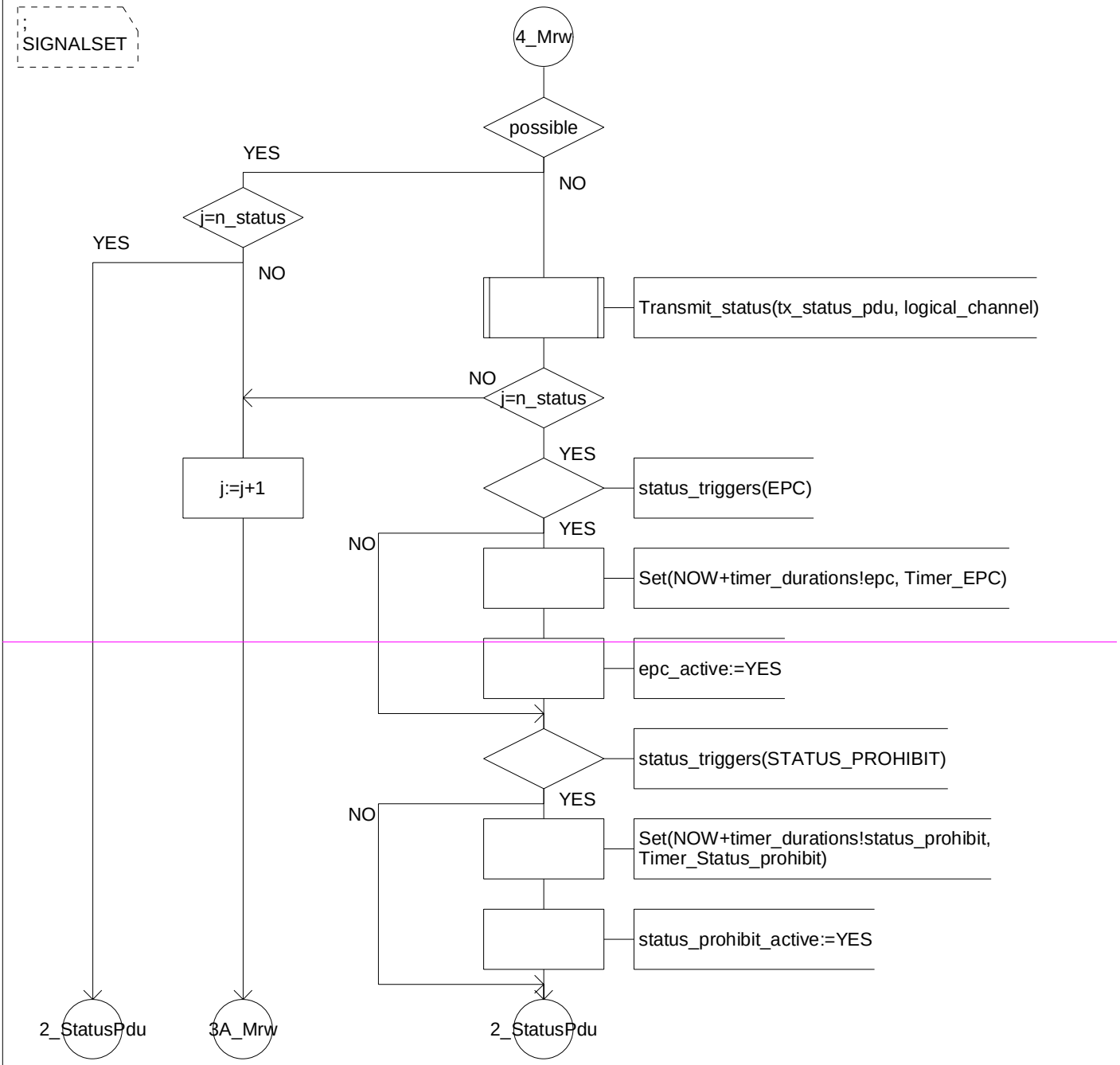

Virtual Process Type Acknowledged_link
1_StatusPduWindow(74
SIGNALSET
1_Window
transmit_window:=status_pdu!sufis(i)!wsn
vt_ms:=vt_a+transmit_window
i:=i+1
2_StatusPdu


Virtual Process Type Acknowledged_link
1_StatusPduMrw(74
SIGNALSET
1_Mrw
1_Mrwnfl
discard_n_fli:=NO
discard_n_fli:=YES
leng:=status_pdu!sufis(i)!length
vr_r>=status_pdu!sufis(i)!sn_mrw(leng)
NO
YES
vr_r:=status_pdu!sufis(i)!sn_mrw(leng)
mrw:=status_pdu!sufis(i)
vr_h<vr_r
NO
YES
vr_mr:=vr_r+receive_window
vr_h:=vr_r,
vr_mr:=vr_r+receive_window
Remove_all_below_mrw_from_queue(
discard_n_fli, receiver_queue,
assembly_queue, mrw)
2_Mrw
3_Mrw

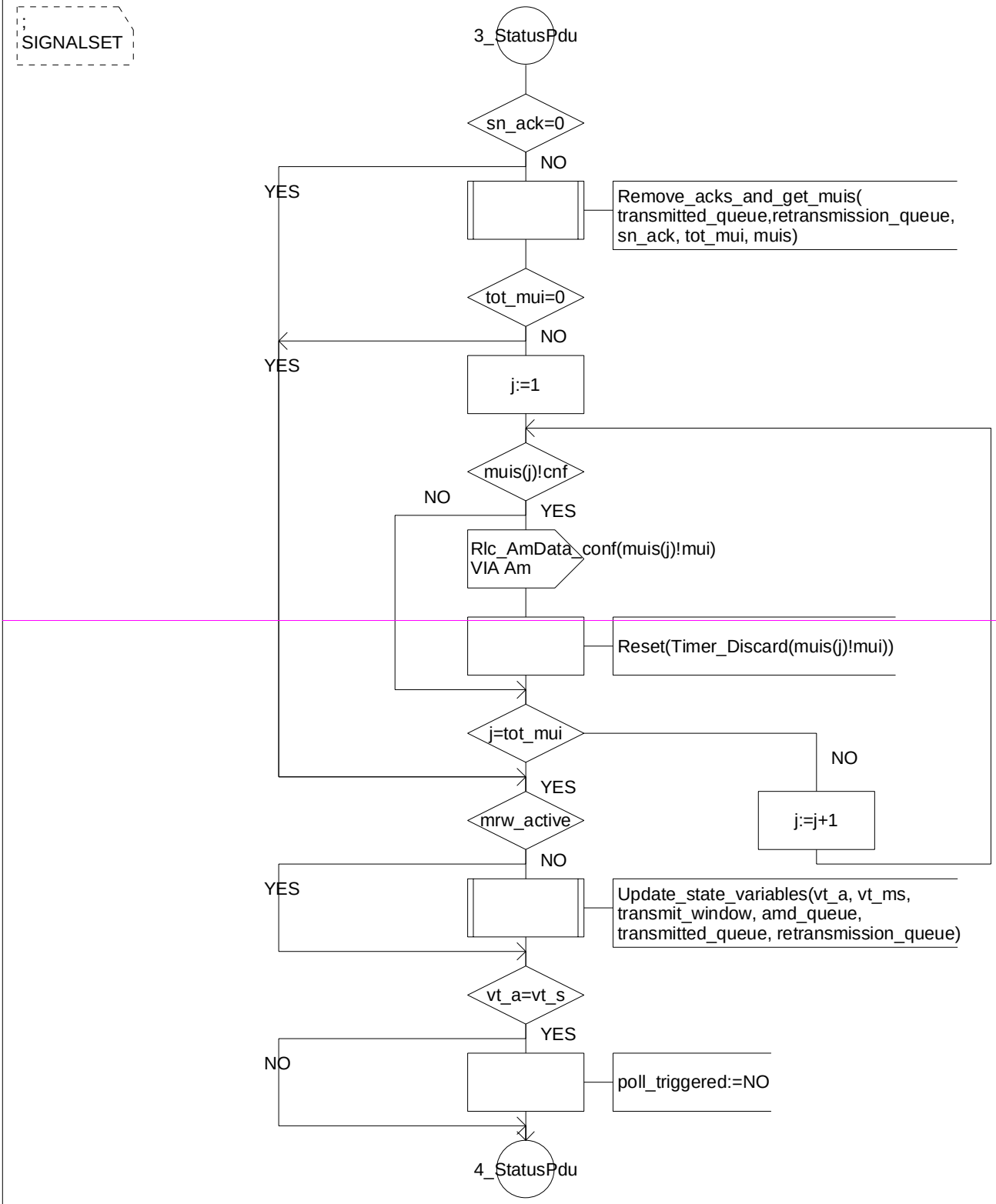

Virtual Process Type Acknowledged_link
2_StatusPduMrw(74
SIGNALSET
2_Mrw
Exists_in_receiver_queue(
vr_r, receiver_queue, exists)
exists
YES
NO
Remove_identified_from_queue(
receiver_queue, vr_r, amd_pu)
Place_in_receiving_side_queue(
assembly_queue, amd_pu)
Reassemble_am_pu(assembly_queue,
complete, sdus, n_sdu)
complete
NO
YES
j:=1
sdu:=sdus(j)
Rlc_AmData_ind(sdu)
VIA Am
j=n_sdu
NO
YES
j:=j+1
3_Mrw
vr_r:=vr_r+1,
vr_mr:=vr_mr+1


Virtual Process Type Acknowledged_link
3_StatusPduMrw(74
SIGNALSET
3_Mrw
3A_Mrw
i:=i+1
discard_n_fli:=NO
epc_active
YES
NO
The pdu_size is indicated by MAC.
status_triggered:=YES
Create_status(vr_r, vr_h, receive_window, pdu_size, receiver_queue, status_pdus, vr_ep, n_status, sn_mrw)
j:=1
tx_status_pdu:=status_pdus(j)
Place_piggyback_in_queue(amd_queue, retransmission_queue, tx_status_pdu, vt_ms, possible)
2_StatusPdu
4_Mrw

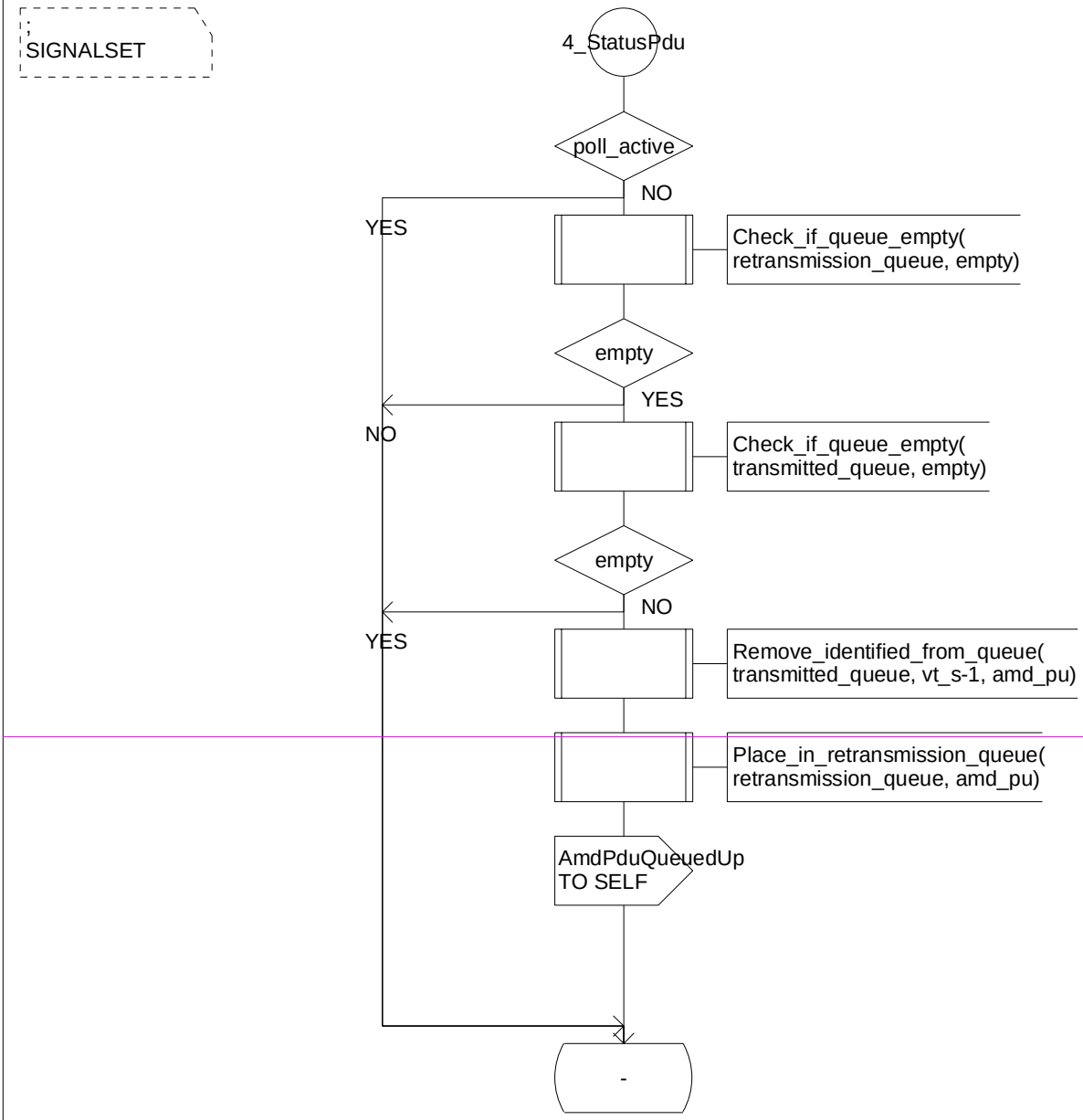
Virtual Process Type Acknowledged_link 4_StatusPduMrw(74

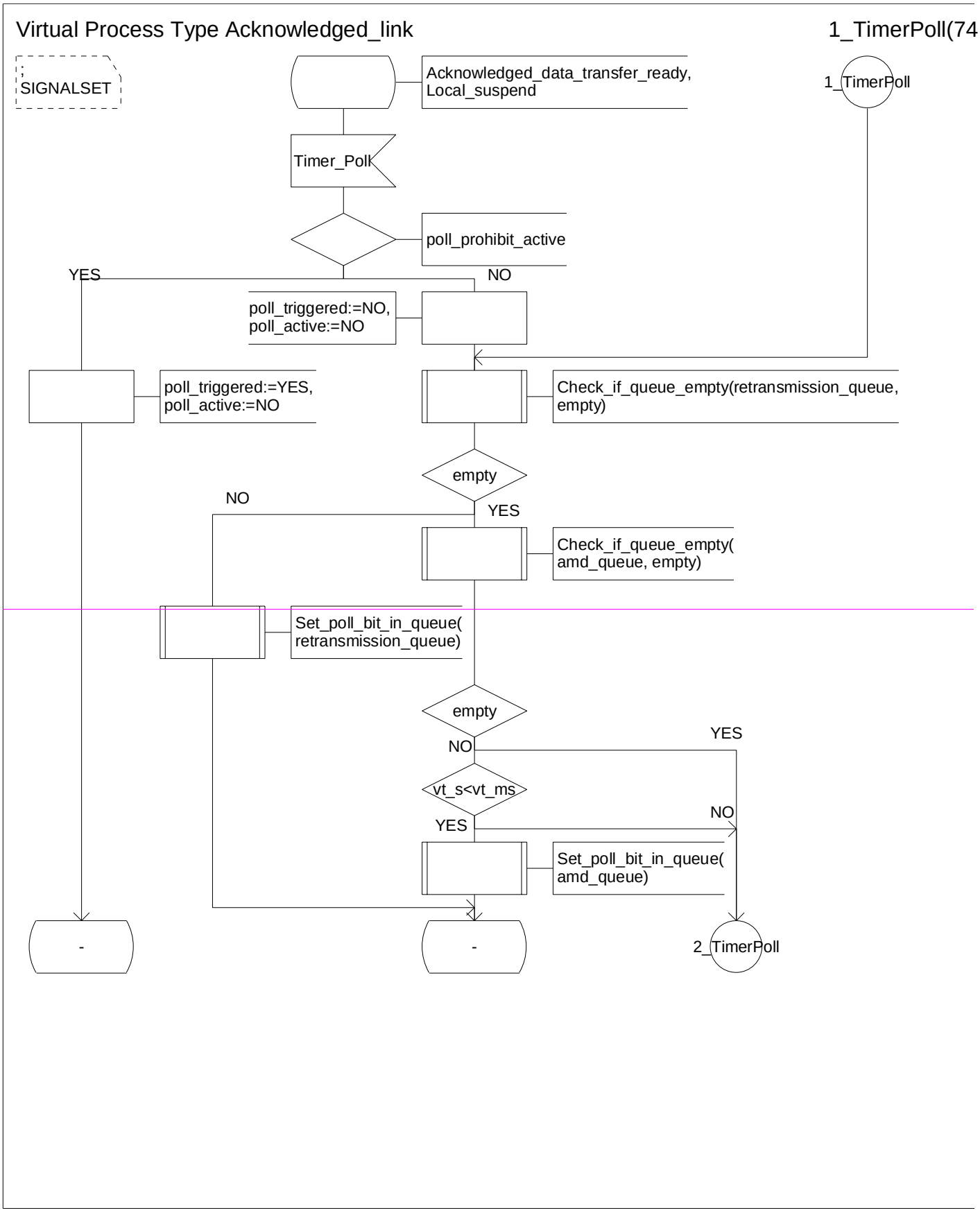


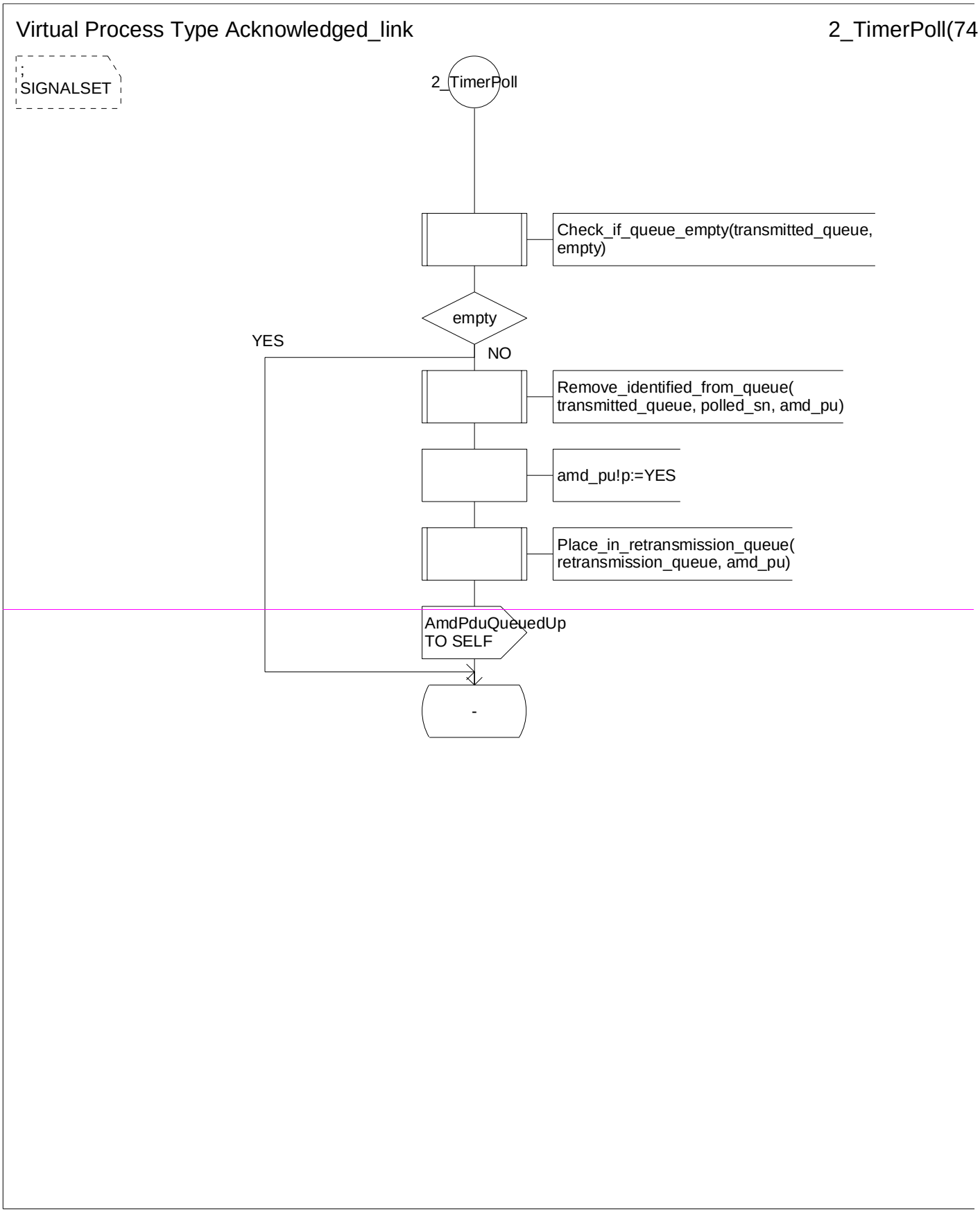
Virtual Process Type Acknowledged_link 3_StatusPdu(74



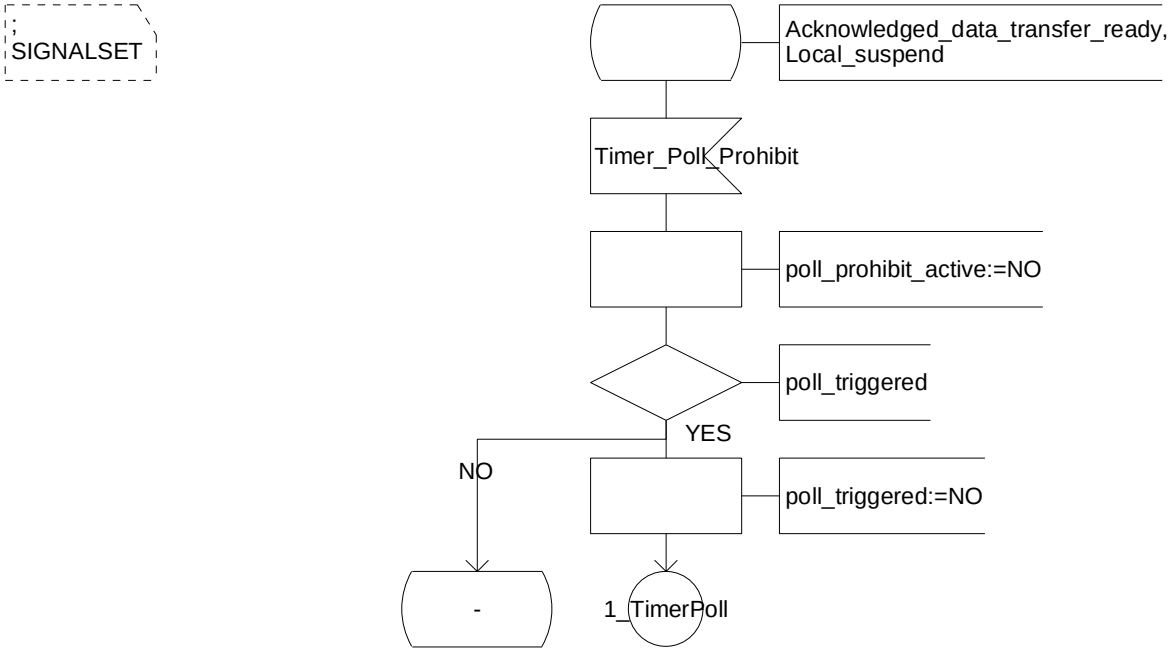
Virtual Process Type Acknowledged_link 4_StatusPdu(74

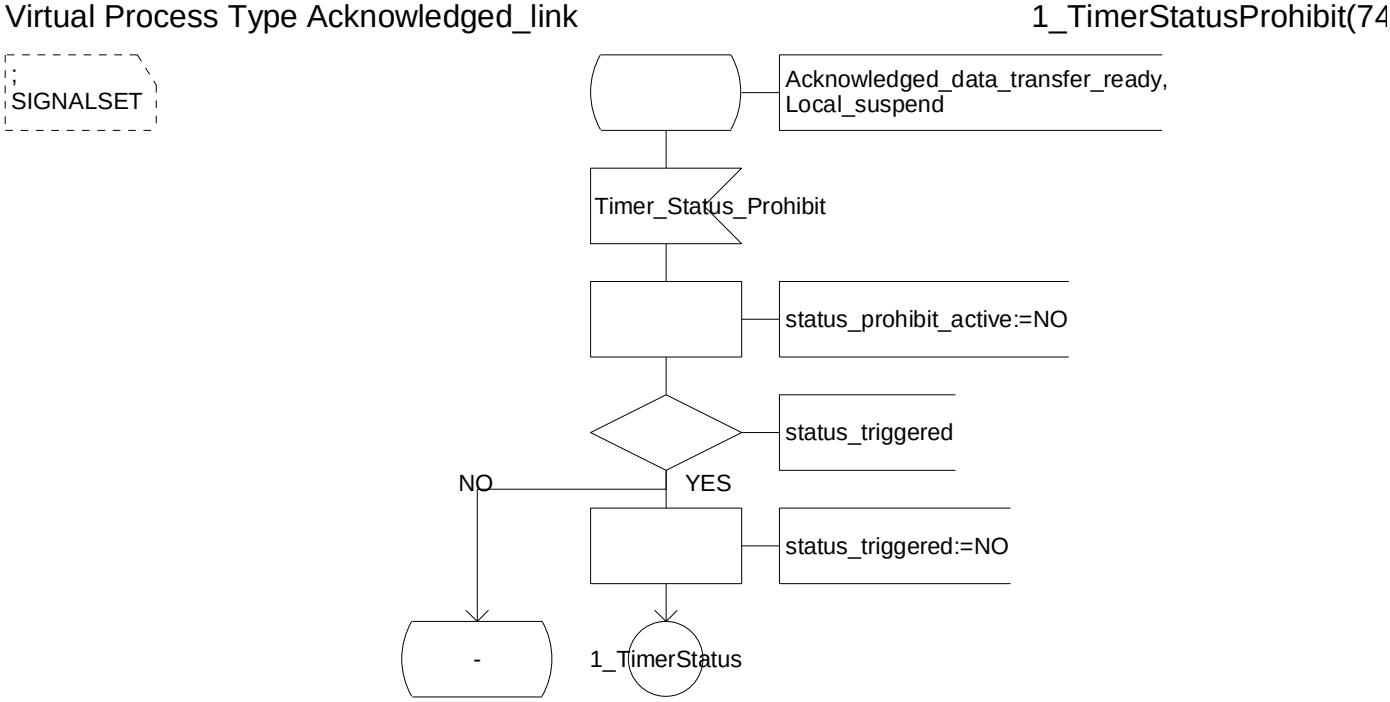


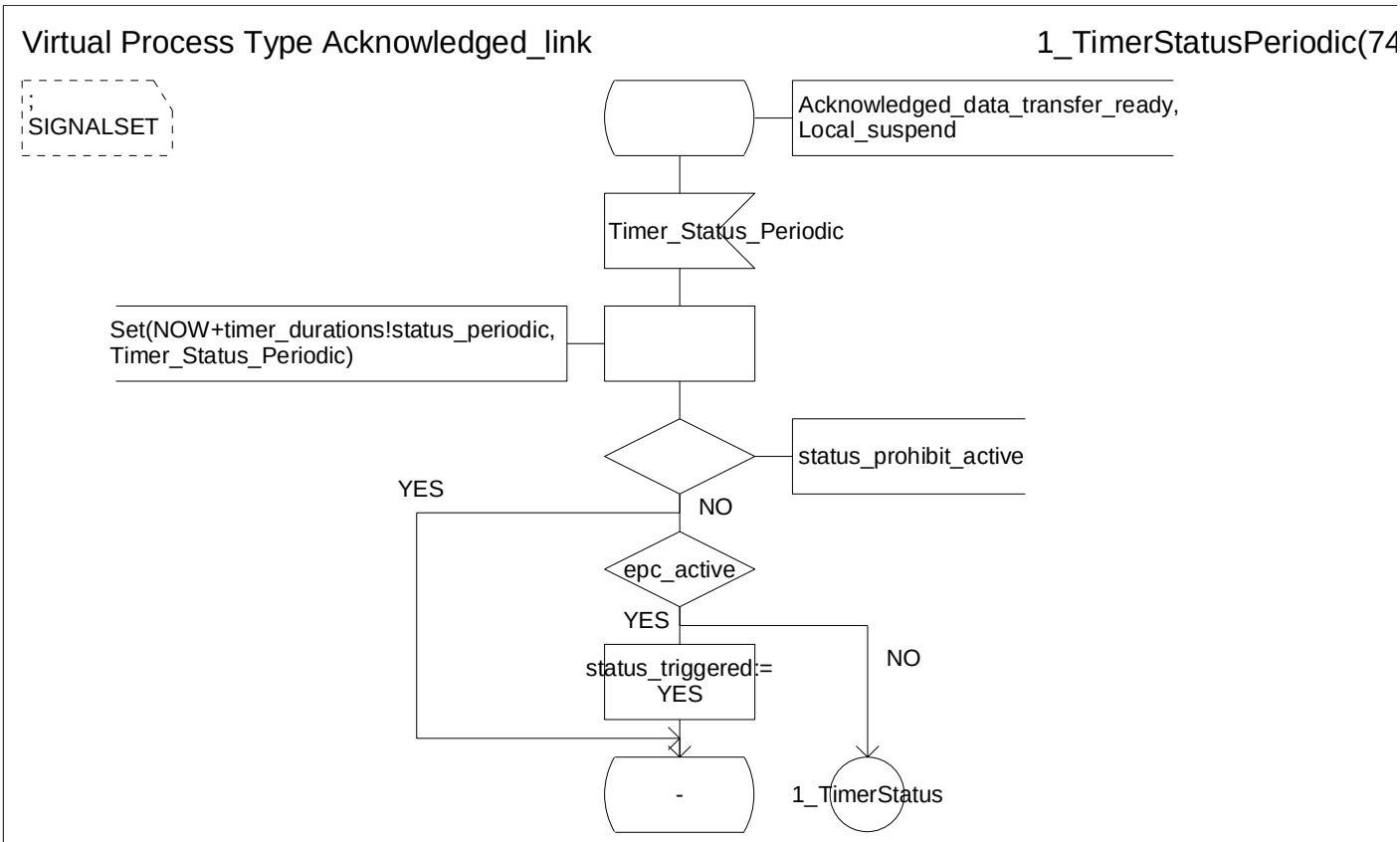

Virtual Process Type Acknowledged_link
1_TimerPoll(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
1_TimerPoll
Timer_Poll
poll_prohibit_active
YES
NO
poll_triggered:=NO,
poll_active:=NO
poll_triggered:=YES,
poll_active:=NO
Check_if_queue_empty(retransmission_queue,
empty)
empty
NO
YES
Check_if_queue_empty(
amd_queue, empty)
Set_poll_bit_in_queue(
retransmission_queue)
empty
YES
NO
vt_s<vt_ms
NO
YES
Set_poll_bit_in_queue(
amd_queue)
-
-
2_TimerPoll

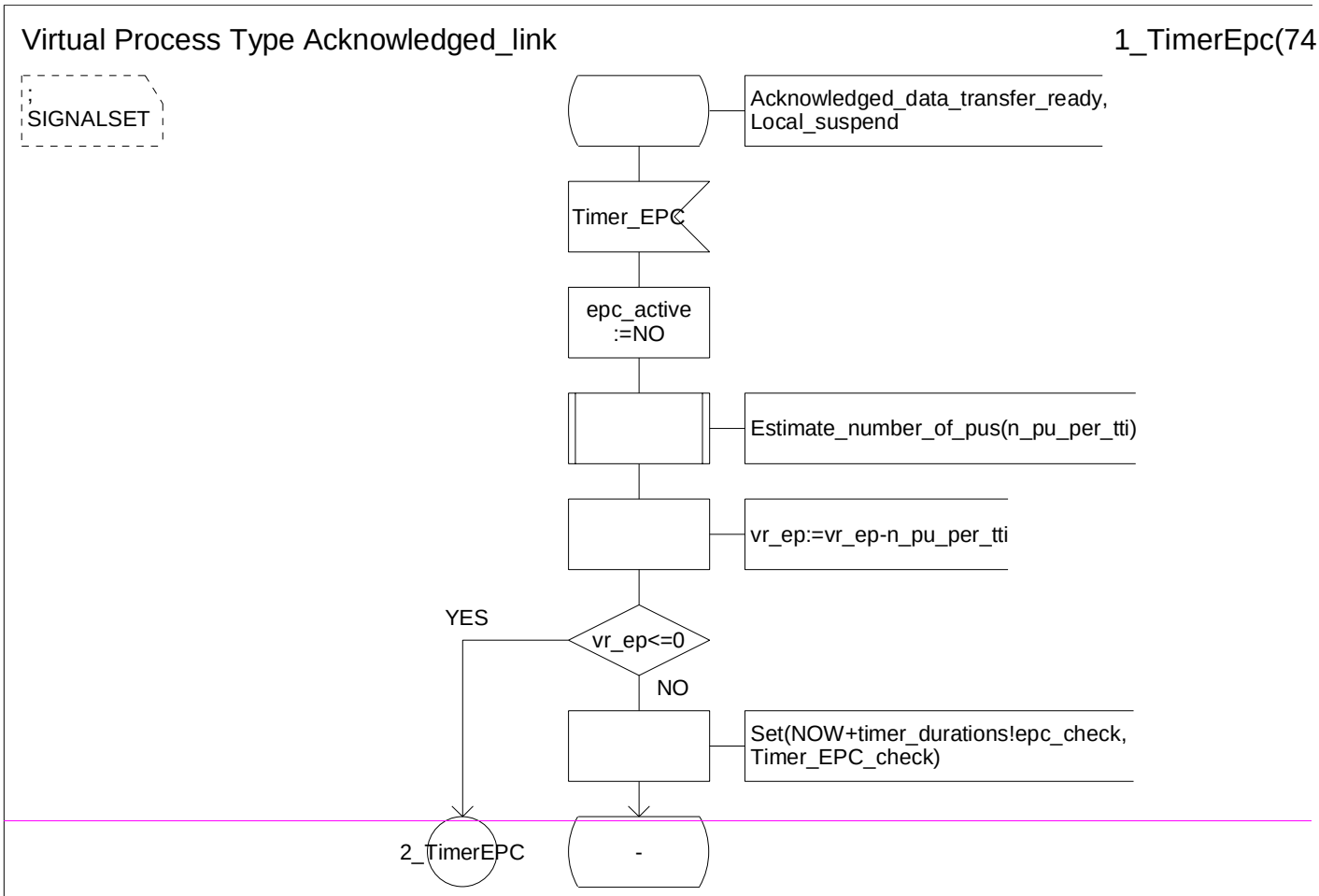

Virtual Process Type Acknowledged_link
2_TimerPoll(74
SIGNALSET
2_TimerPoll
Check_if_queue_empty(transmitted_queue, empty)
empty
YES
NO
Remove_identified_from_queue(transmitted_queue, polled_sn, amd_pu)
amd_pu!p:=YES
Place_in_retransmission_queue(retransmission_queue, amd_pu)
AmdPduQueuedUp TO SELF
-

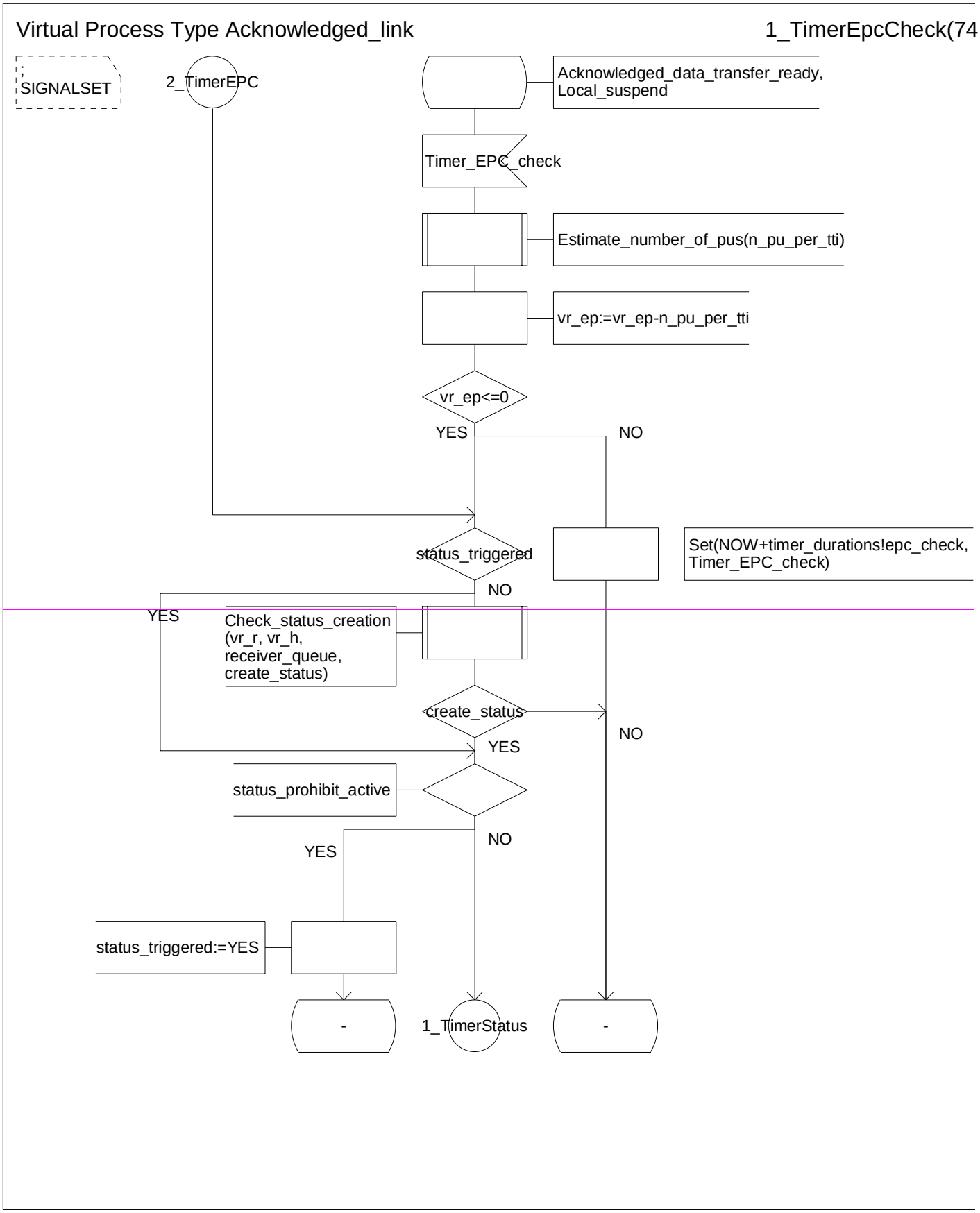
Virtual Process Type Acknowledged_link 1_TimerPollProhibit(74

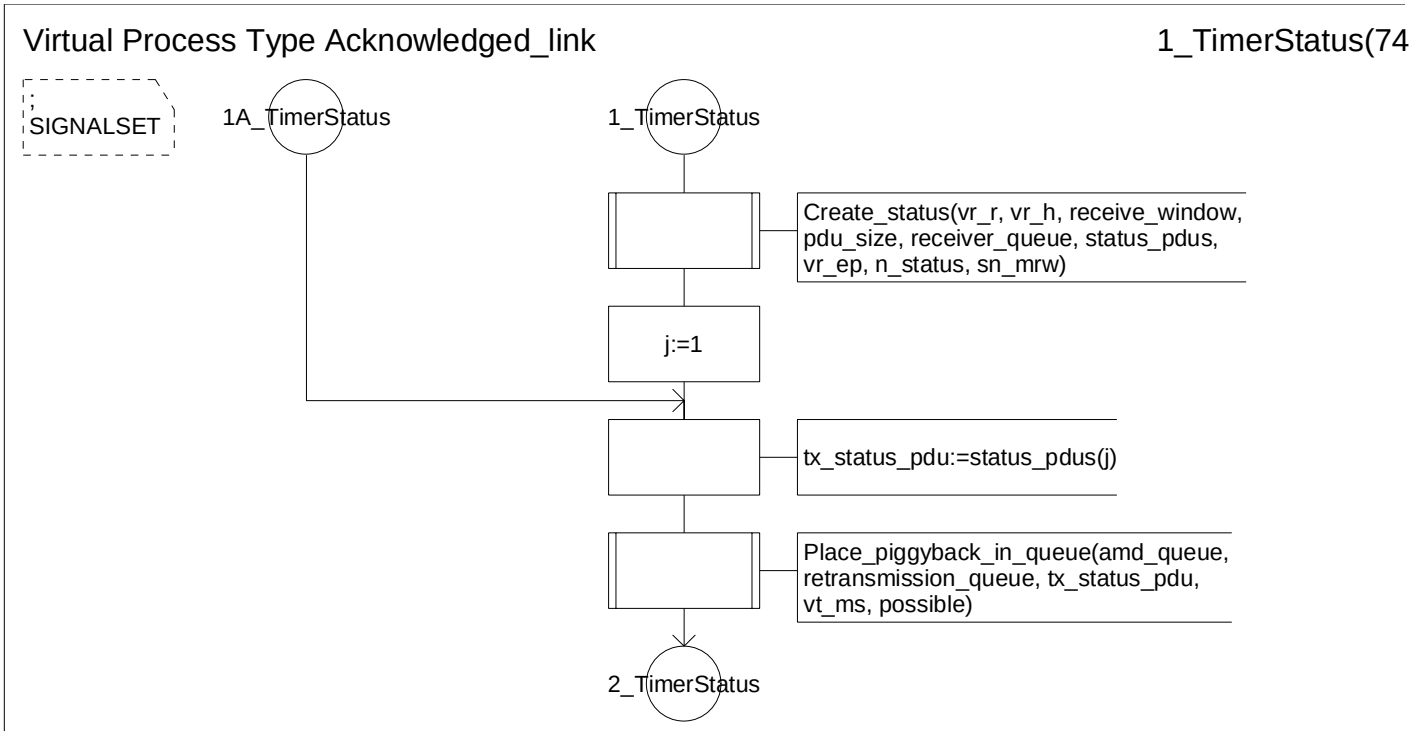


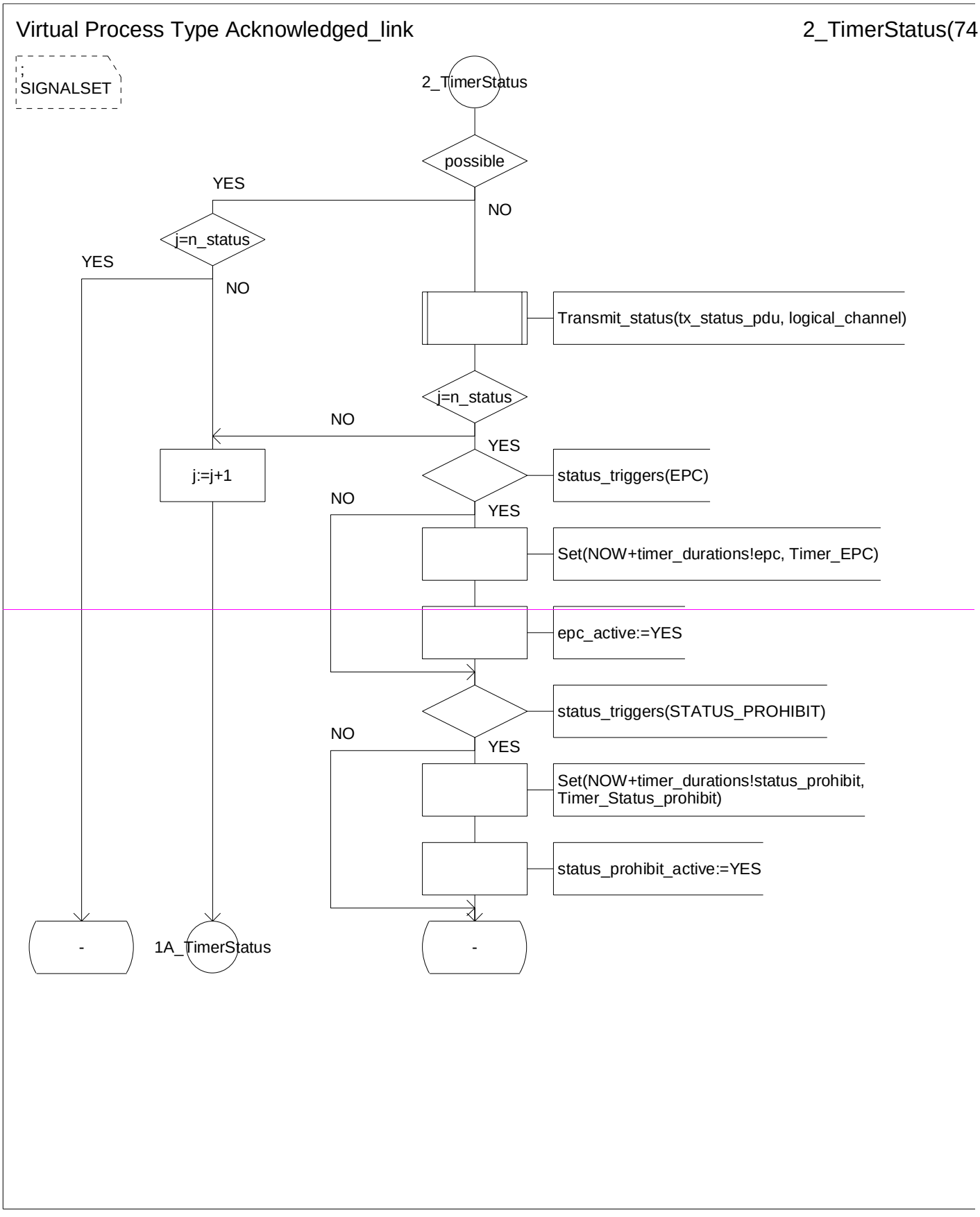

Virtual Process Type Acknowledged_link
1_TimerStatusProhibit(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Timer_Status_Prohibit
status_prohibit_active:=NO
status_triggered
NO
YES
status_triggered:=NO
-
1_TimerStatus

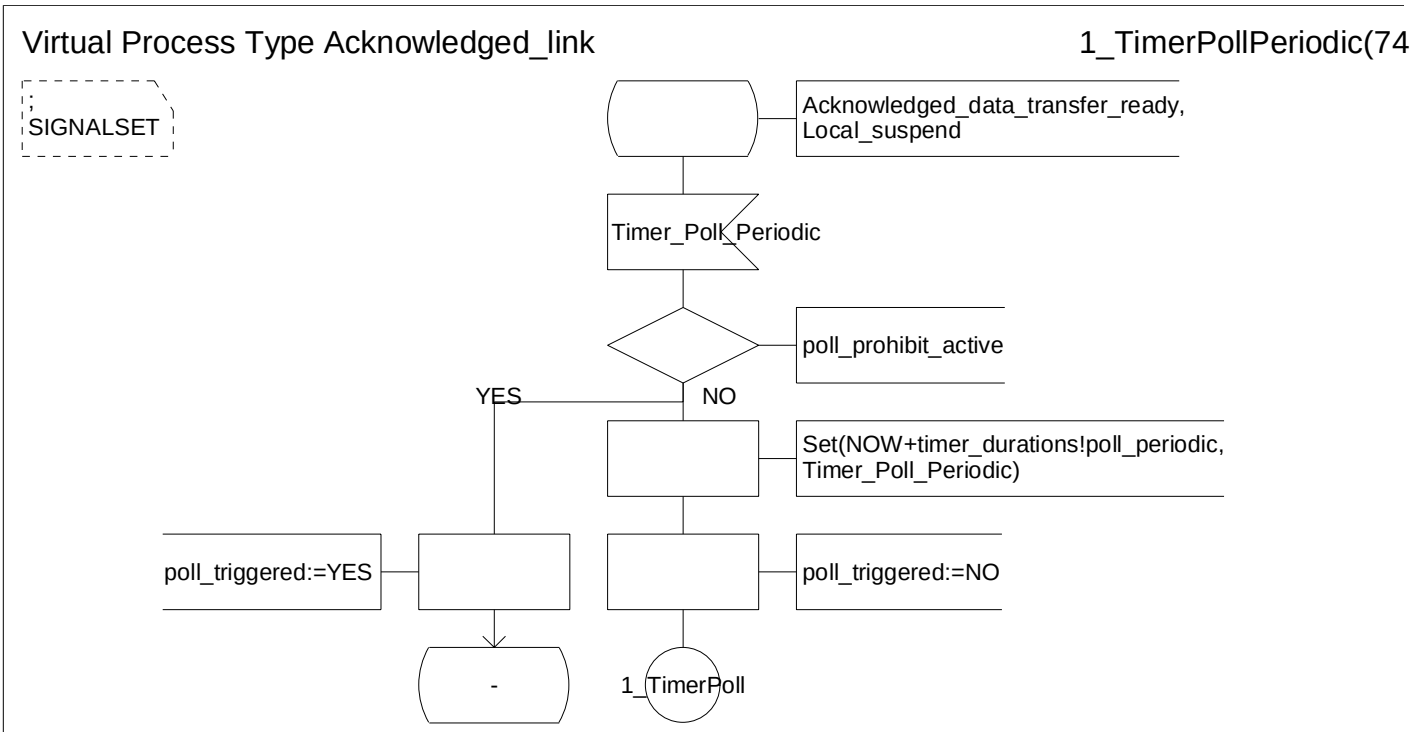

Virtual Process Type Acknowledged_link
1_TimerStatusPeriodic(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Timer_Status_Periodic
Set(NOW+timer_durations!status_periodic,
Timer_Status_Periodic)
status_prohibit_active
YES
NO
epc_active
YES
NO
status_triggered:=
YES
-
1_TimerStatus

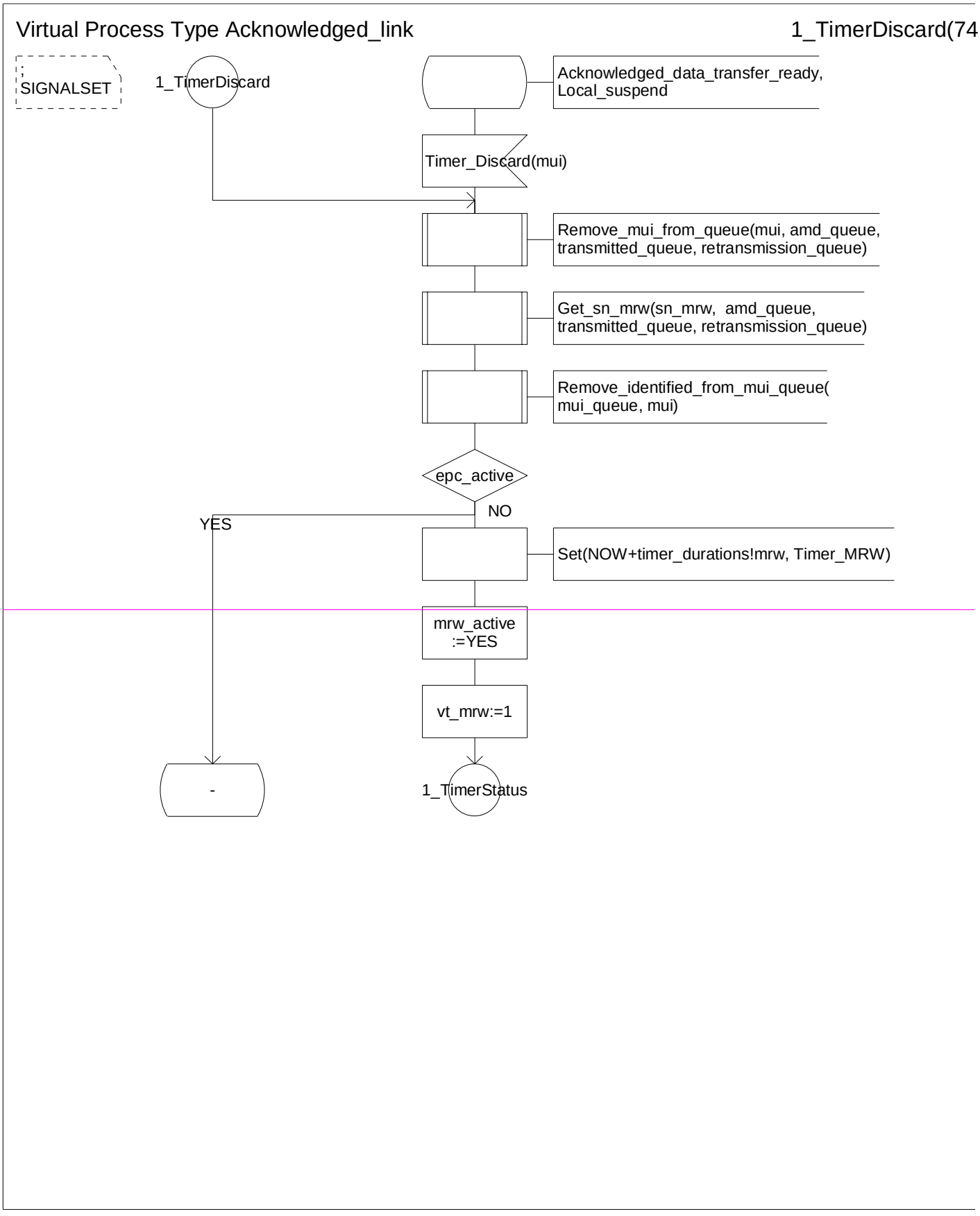

Virtual Process Type Acknowledged_link
1_TimerEpc(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Timer_EPC
epc_active
:=NO
Estimate_number_of_pus(n_pu_per_tti)
vr_ep:=vr_ep-n_pu_per_tti
YES
vr_ep<=0
NO
Set(NOW+timer_durations!epc_check,
Timer_EPC_check)
2_TimerEPC
-

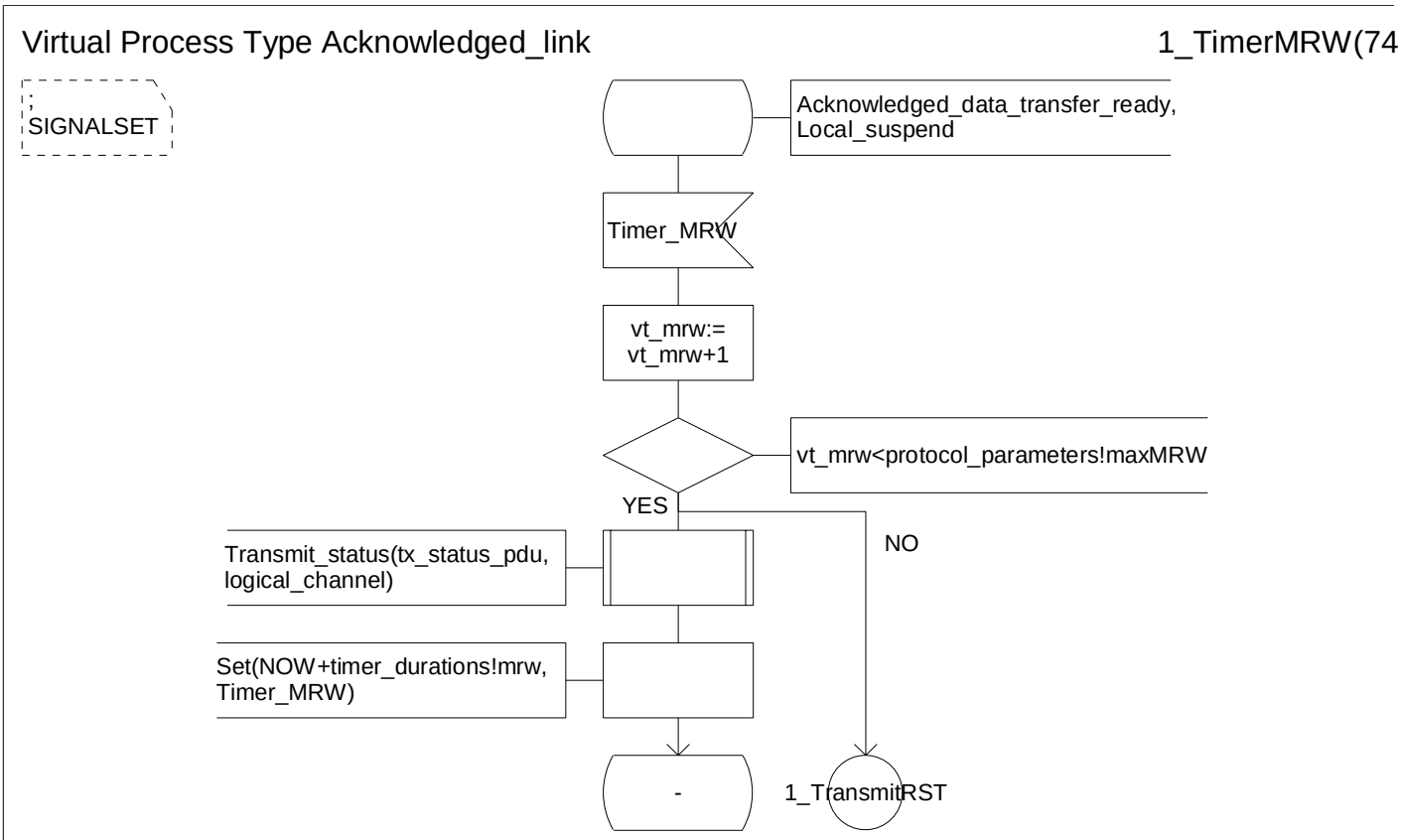

Virtual Process Type Acknowledged_link
1_TimerEpcCheck(74
SIGNALSET
2_TimerEPC
Acknowledged_data_transfer_ready,
Local_suspend
Timer_EPC_check
Estimate_number_of_pus(n_pu_per_tti)
vr_ep:=vr_ep-n_pu_per_tti
vr_ep<=0
YES
NO
status_triggered
Set(NOW+timer_durations!epc_check,
Timer_EPC_check)
NO
YES
Check_status_creation
(vr_r, vr_h,
receiver_queue,
create_status)
create_status
NO
YES
status_prohibit_active
NO
YES
status_triggered:=YES
-
1_TimerStatus
-

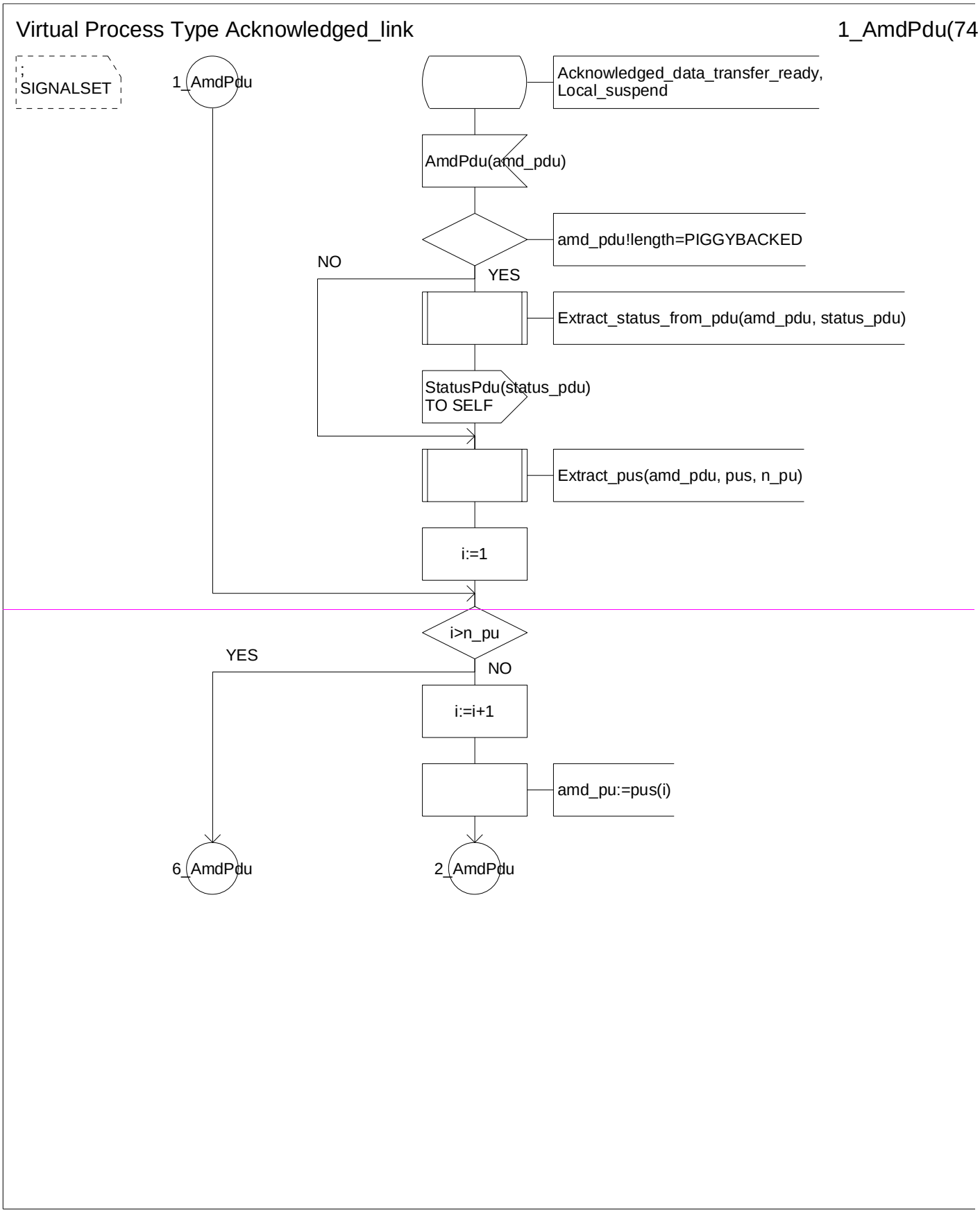

Virtual Process Type Acknowledged_link
1_TimerStatus(74
SIGNALSET
1A_TimerStatus
1_TimerStatus
Create_status(vr_r, vr_h, receive_window, pdu_size, receiver_queue, status_pdus, vr_ep, n_status, sn_mrw)
j:=1
tx_status_pdu:=status_pdus(j)
Place_piggyback_in_queue(amd_queue, retransmission_queue, tx_status_pdu, vt_ms, possible)
2_TimerStatus

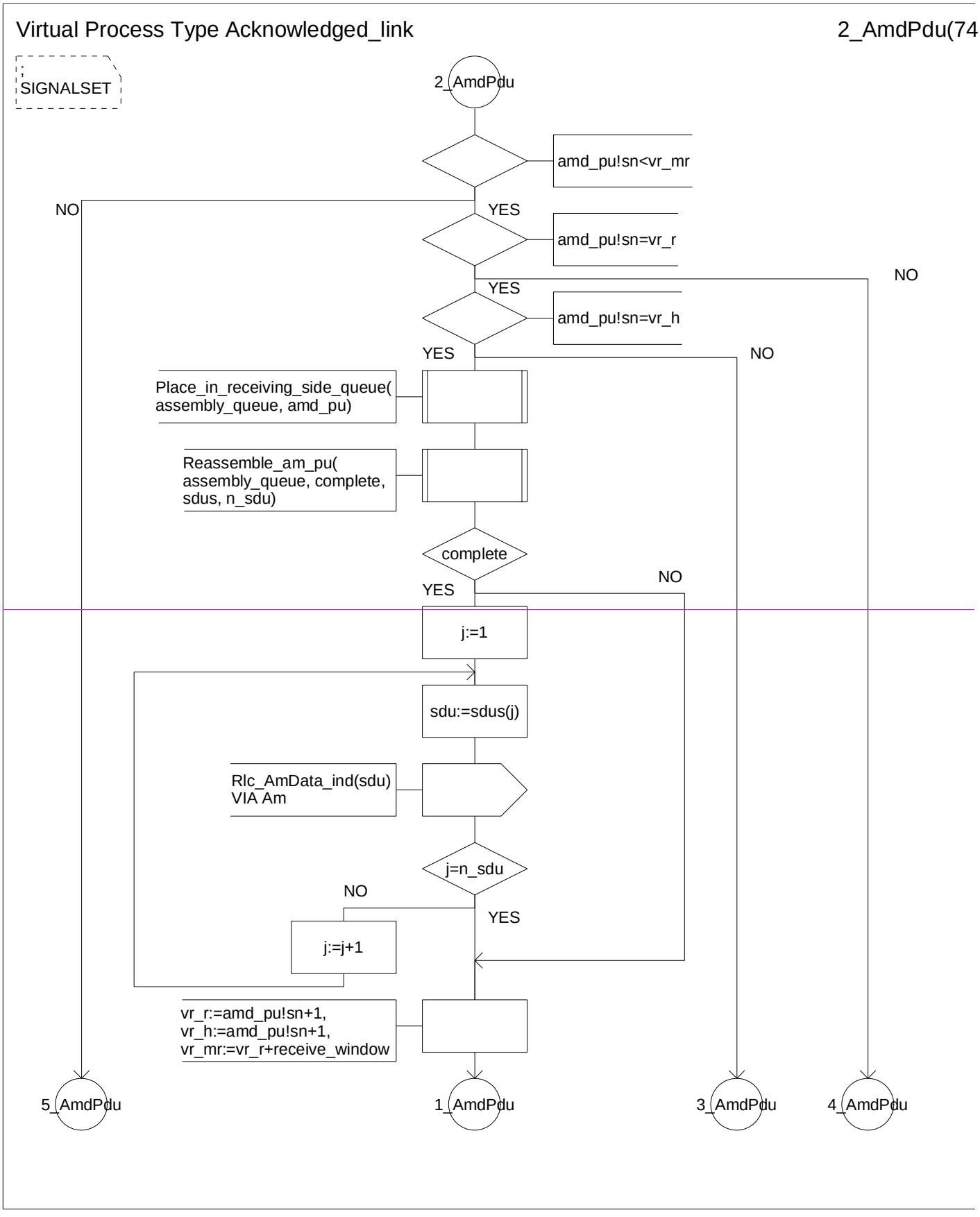

Virtual Process Type Acknowledged_link
2_TimerStatus(74
SIGNALSET
2_TimerStatus
possible
YES
NO
j=n_status
YES
NO
Transmit_status(tx_status_pdu, logical_channel)
j=n_status
NO
YES
j:=j+1
status_triggers(EPC)
NO
YES
Set(NOW+timer_durations!epc, Timer_EPC)
epc_active:=YES
status_triggers(STATUS_PROHIBIT)
NO
YES
Set(NOW+timer_durations!status_prohibit,
Timer_Status_prohibit)
status_prohibit_active:=YES
-
1A_TimerStatus
-

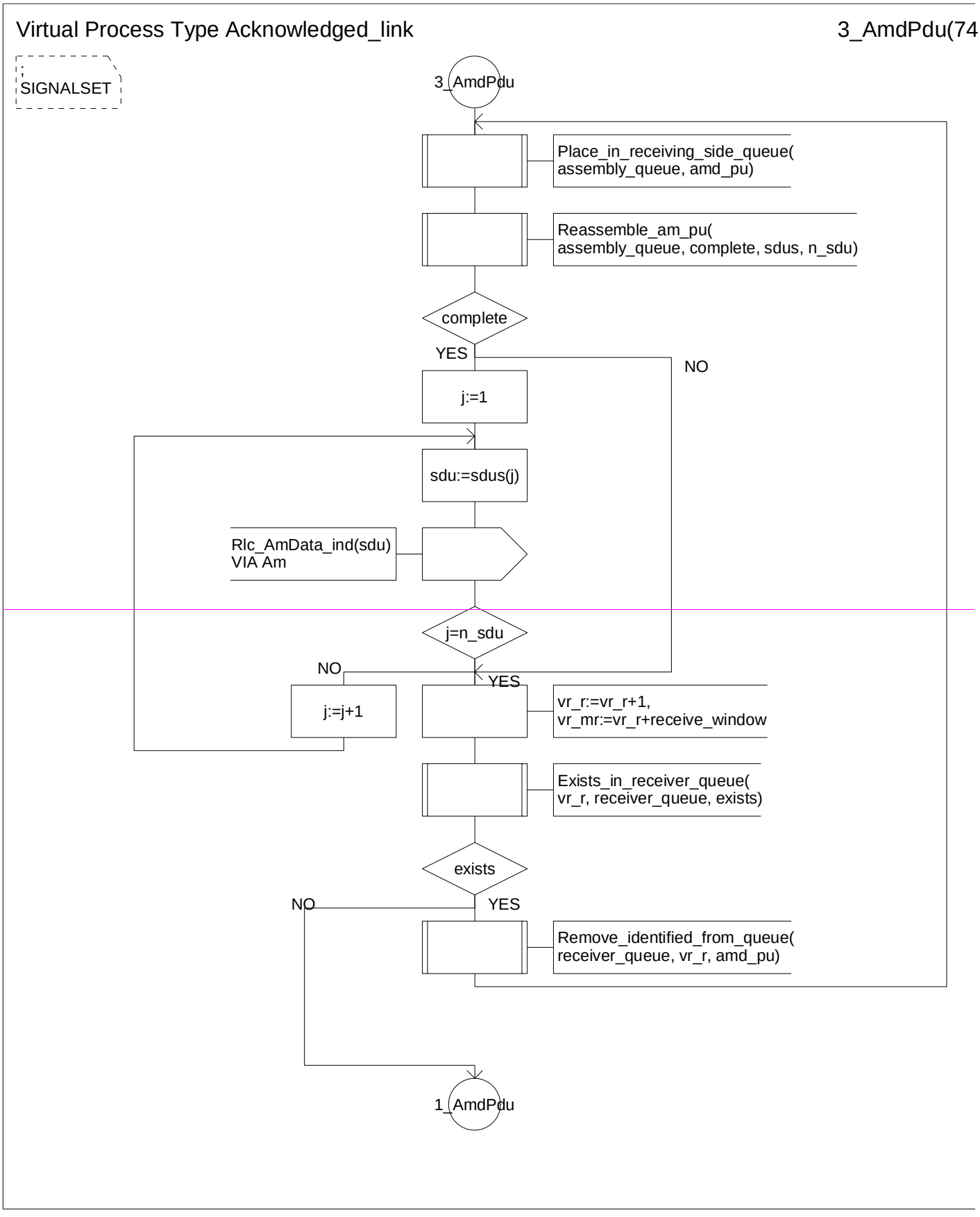

Virtual Process Type Acknowledged_link
1_TimerPollPeriodic(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Timer_Poll_Periodic
poll_prohibit_active
YES
NO
Set(NOW+timer_durations!poll_periodic,
Timer_Poll_Periodic)
poll_triggered:=YES
poll_triggered:=NO
-
1_TimerPoll

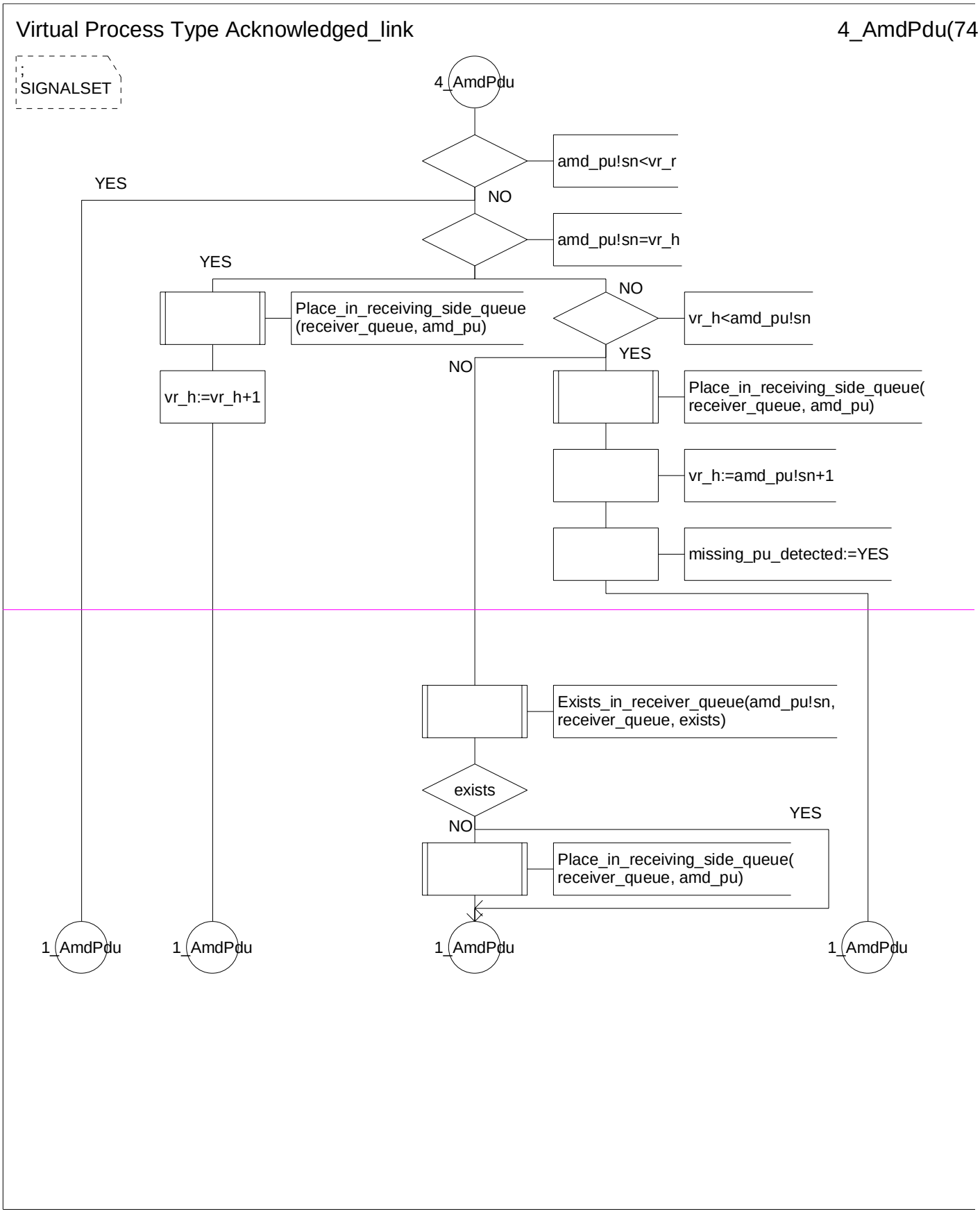

Virtual Process Type Acknowledged_link
1_TimerDiscard(74
SIGNALSET
1_TimerDiscard
Acknowledged_data_transfer_ready,
Local_suspend
Timer_Discard(mui)
Remove_mui_from_queue(mui, amd_queue,
transmitted_queue, retransmission_queue)
Get_sn_mrw(sn_mrw, amd_queue,
transmitted_queue, retransmission_queue)
Remove_identified_from_mui_queue(
mui_queue, mui)
epc_active
NO
YES
Set(NOW+timer_durations!mrw, Timer_MRW)
mrw_active
:=YES
vt_mrw:=1
-
1_TimerStatus


Virtual Process Type Acknowledged_link
1_TimerMRW(74
SIGNALSET
Acknowledged_data_transfer_ready,
Local_suspend
Timer_MRW
vt_mrw:=
vt_mrw+1
vt_mrw<protocol_parameters!maxMRW
YES
NO
Transmit_status(tx_status_pdu,
logical_channel)
Set(NOW+timer_durations!mrw,
Timer_MRW)
-
1_TransmitRST

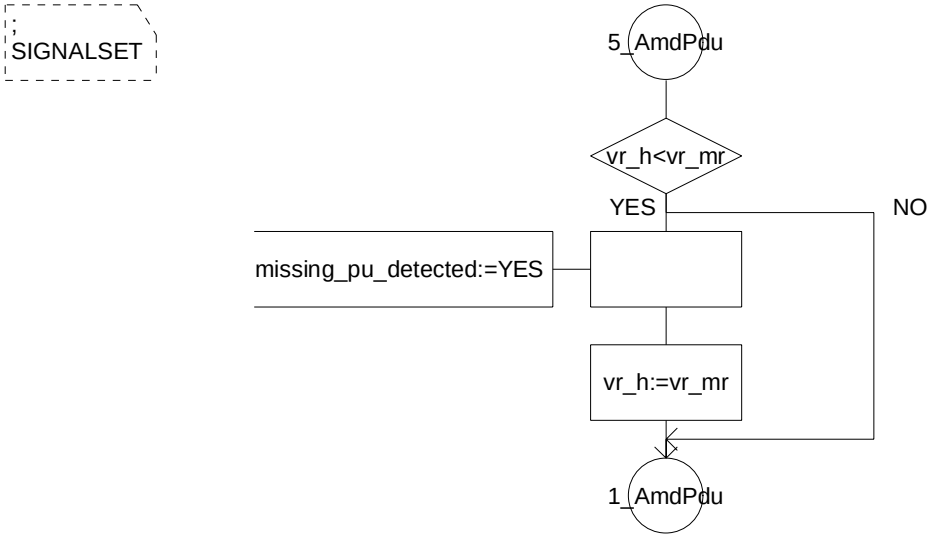

Virtual Process Type Acknowledged_link
1_AmdPdu(74
SIGNALSET
1_AmdPdu
Acknowledged_data_transfer_ready,
Local_suspend
AmdPdu(amd_pdu)
amd_pdu!length=PIGGYBACKED
NO
YES
Extract_status_from_pdu(amd_pdu, status_pdu)
StatusPdu(status_pdu)
TO SELF
Extract_pus(amd_pdu, pus, n_pu)
i:=1
i>n_pu
YES
NO
i:=i+1
amd_pu:=pus(i)
6_AmdPdu
2_AmdPdu


Virtual Process Type Acknowledged_link
2_AmdPdu(74
SIGNALSET
2_AmdPdu
amd_pu!sn<vr_mr
NO
YES
amd_pu!sn=vr_r
NO
YES
amd_pu!sn=vr_h
YES
NO
Place_in_receiving_side_queue(
assembly_queue, amd_pu)
Reassemble_am_pu(
assembly_queue, complete,
sdus, n_sdu)
complete
NO
YES
j:=1
sdu:=sdus(j)
Rlc_AmData_ind(sdu)
VIA Am
j=n_sdu
NO
YES
j:=j+1
vr_r:=amd_pu!sn+1,
vr_h:=amd_pu!sn+1,
vr_mr:=vr_r+receive_window
5_AmdPdu
1_AmdPdu
3_AmdPdu
4_AmdPdu

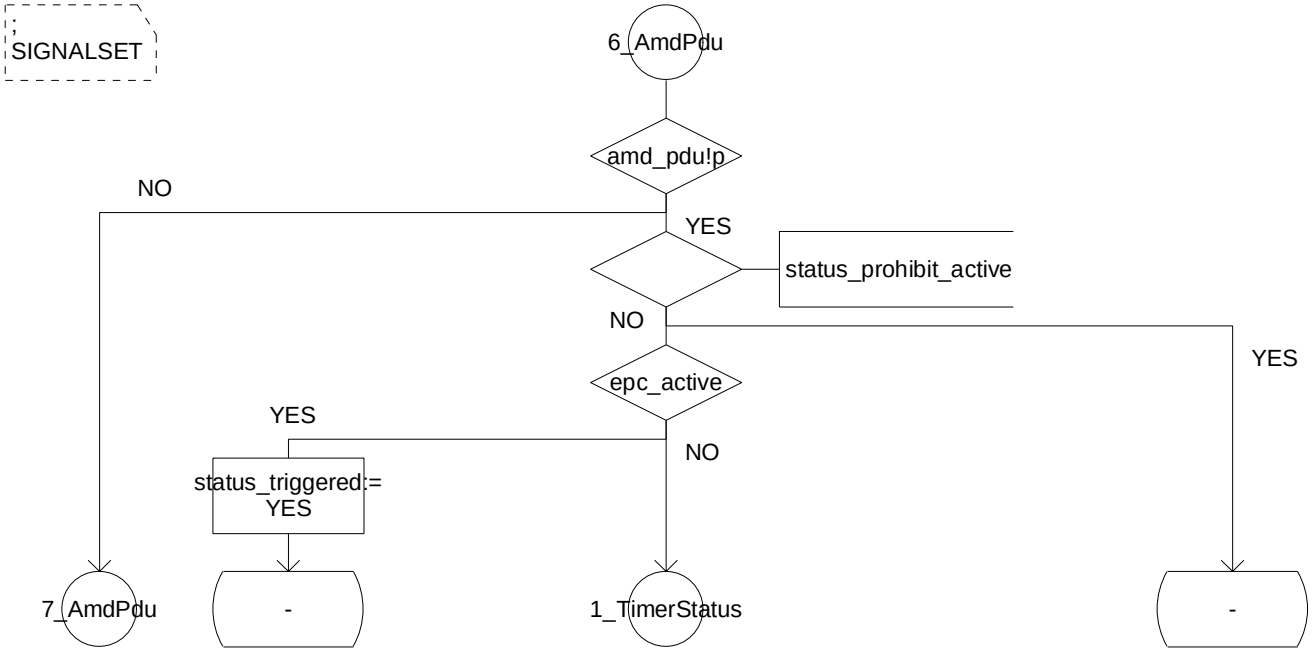

Virtual Process Type Acknowledged_link
3_AmdPdu(74
SIGNALSET
3_AmdPdu
Place_in_receiving_side_queue(
assembly_queue, amd_pu)
Reassemble_am_pu(
assembly_queue, complete, sdus, n_sdu)
complete
YES
NO
j:=1
sdu:=sdus(j)
Rlc_AmData_ind(sdu)
VIA Am
j=n_sdu
NO
YES
j:=j+1
vr_r:=vr_r+1,
vr_mr:=vr_r+receive_window
Exists_in_receiver_queue(
vr_r, receiver_queue, exists)
exists
NO
YES
Remove_identified_from_queue(
receiver_queue, vr_r, amd_pu)
1_AmdPdu

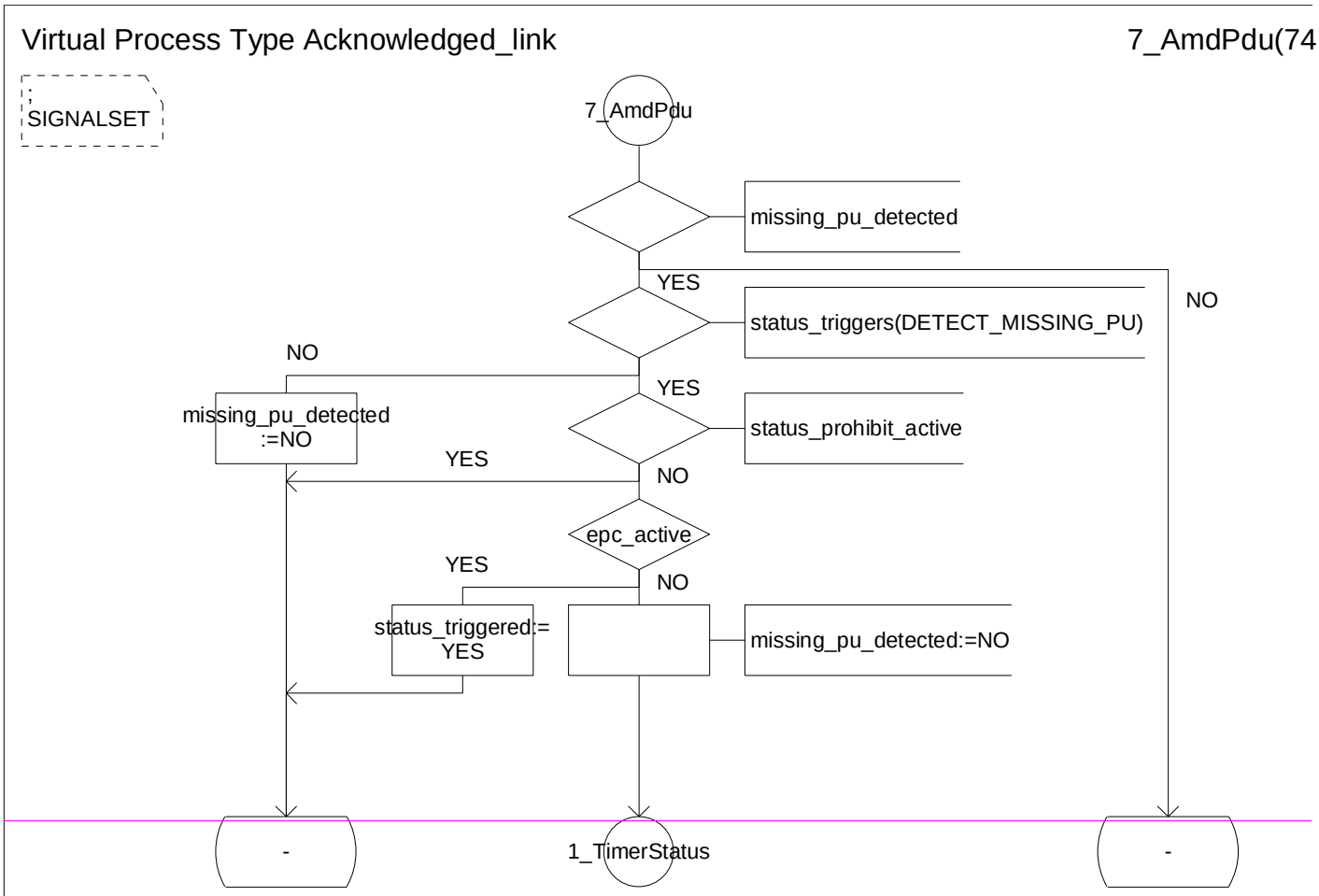

Virtual Process Type Acknowledged_link
4_AmdPdu(74
SIGNALSET
4_AmdPdu
amd_pu!sn<vr_r
YES
NO
amd_pu!sn=vr_h
YES
NO
Place_in_receiving_side_queue
(receiver_queue, amd_pu)
vr_h<amd_pu!sn
NO
YES
vr_h:=vr_h+1
Place_in_receiving_side_queue(
receiver_queue, amd_pu)
vr_h:=amd_pu!sn+1
missing_pu_detected:=YES
Exists_in_receiver_queue(amd_pu!sn,
receiver_queue, exists)
exists
YES
NO
Place_in_receiving_side_queue(
receiver_queue, amd_pu)
1_AmdPdu
1_AmdPdu
1_AmdPdu
1_AmdPdu

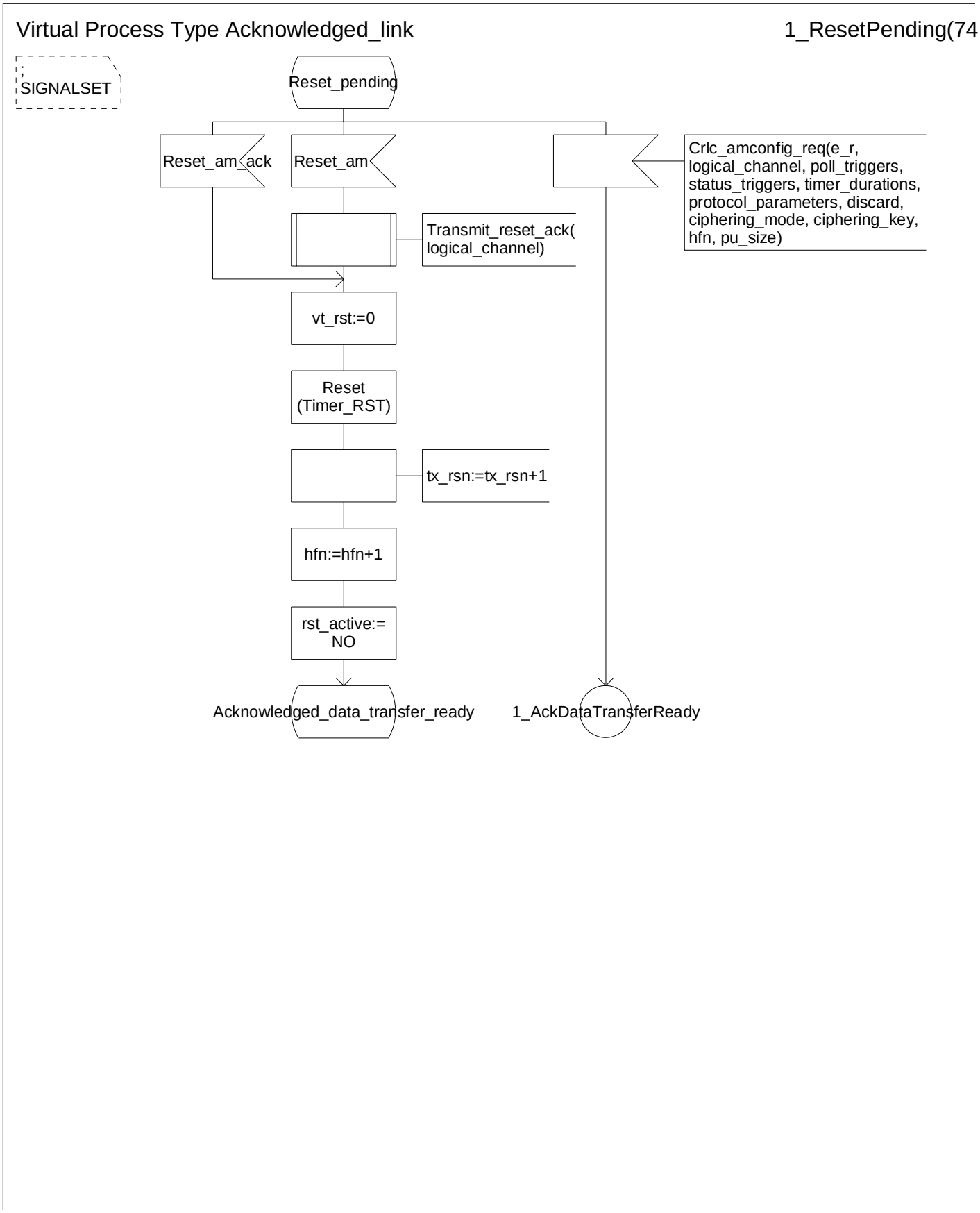
Virtual Process Type Acknowledged_link 5_AmdPdu(74

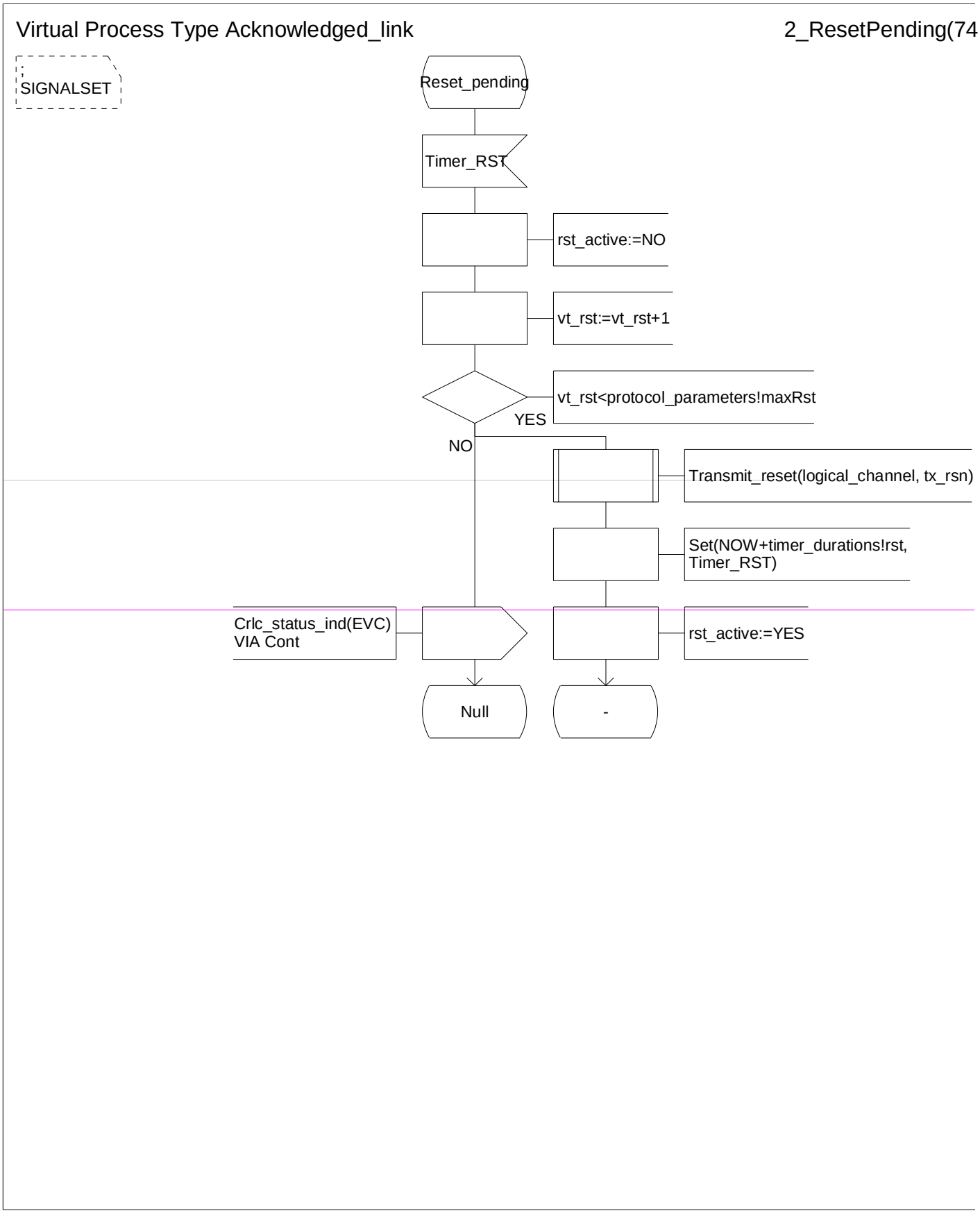


Virtual Process Type Acknowledged_link 6_AmdPdu(74




Virtual Process Type Acknowledged_link
7_AmdPdu(74
SIGNALSET
7_AmdPdu
missing_pu_detected
YES
NO
status_triggers(DETECT_MISSING_PU)
NO
YES
missing_pu_detected
:=NO
status_prohibit_active
YES
NO
epc_active
YES
NO
status_triggered:=
YES
missing_pu_detected:=NO
-
1_TimerStatus
-


Virtual Process Type Acknowledged_link
1_ResetPending(74
SIGNALSET
Reset_pending
Reset_am_ack
Reset_am
Crlc_amconfig_req(e_r, logical_channel, poll_triggers, status_triggers, timer_durations, protocol_parameters, discard, ciphering_mode, ciphering_key, hfn, pu_size)
Transmit_reset_ack(logical_channel)
vt_rst:=0
Reset (Timer_RST)
tx_rsn:=tx_rsn+1
hfn:=hfn+1
rst_active:= NO
Acknowledged_data_transfer_ready
1_AckDataTransferReady


Virtual Process Type Acknowledged_link
2_ResetPending(74
SIGNALSET
Reset_pending
Timer_RST
rst_active:=NO
vt_rst:=vt_rst+1
vt_rst<protocol_parameters!maxRst
YES
NO
Transmit_reset(logical_channel, tx_rsn)
Set(NOW+timer_durations!rst,
Timer_RST)
Crlc_status_ind(EVC)
VIA Cont
rst_active:=YES
Null
-

Annex B (informative): Pseudo code describing AMD PDU header Compression

The following Pseudo-Code is an example of algorithm to describe the exact Header Compression Operation that takes place when several PUs are packed into one RLC PDU.

```
/* Prior to calling this procedure it must be checked that <pus_in_pdu> consecutive PU:s
 are to be transmitted (or there is padding in the end)*/

Compress_PDU (pus_in_pdu, pu_size) {

 li_addition = 0; // reset the variable that counts data in full pu:s

 Loop through pus_in_pdu {

 d_e_flag = E-flag for this PU;

 If (d_e_flag == FALSE) {

 Append PU data to PDU data; // complete PU is SDU-data
 li_addition += pu_size; // to be added to the next LI

 } else { // E-flag is TRUE, so LI-field(s) exist

 Previous E-flag in PDU = TRUE; // Either in PDU header or pdu_li_vector;

 j = 0; // reset LI-counter for this PU
 pu_data_size = 0; // reset data size counter for this PU

 Loop until (d_e_flag == FALSE) {

 d_li = next LI; // in octet j of PU;
 d_e_flag = next E_FLAG; // in octet j of PU;

 if (d_li is not PADDING) {

 pu_data_size += d_li; // to keep track of data segment size in this PU);
 d_li += li_addition; // to add data from previous PU:s to LI-value);
 li_addition = 0; // reset li_addition;

 }

 Append (d_li + d_e_flag) to pdu_li_vector;

 j++; // go to next li_octet, if d_e_flag is TRUE);

 } /* end-of-loop (exit when d_e_flag is TRUE) */

 Append pu_data_size segments starting from j to RLC-PDU data;

 } /* end-of e-flag == TRUE */

 } /* end-of loop through PU:s in PDU */

} /* end-of Compress_PDU */
```

Annex ~~C~~ A (informative): Change history

Change history							
Date	TSG #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
10/99	RP-05	RP-99465	-		Approved at TSG-RAN #5 and placed under Change Control	-	3.0.0
12/99	RP-06	RP-99641	001		RLC: Editorial corrections	3.0.0	3.1.0
	RP-06	RP-99641	002	1	Editorial changes on RLC protocol specification	3.0.0	3.1.0
	RP-06	RP-99643	003	1	MRW procedure	3.0.0	3.1.0
	RP-06	RP-99643	004		SDU Discard Functionality	3.0.0	3.1.0
	RP-06	RP-99643	005	2	Change in RLC control PDU format	3.0.0	3.1.0
	RP-06	RP-99642	006	1	Editorial corrections regarding CTCH	3.0.0	3.1.0
	RP-06	RP-99641	007		Updated RLC SDL	3.0.0	3.1.0
	RP-06	RP-99642	011		RLC Editorial Changes	3.0.0	3.1.0
	RP-06	RP-99642	013		Editorial Modification on RLC specification	3.0.0	3.1.0
	RP-06	RP-99641	014		Editorial changes	3.0.0	3.1.0
	RP-06	RP-99642	015		Change to one PU in a AMD PDU	3.0.0	3.1.0
	RP-06	RP-99643	016	1	Introduction of RLC suspend state	3.0.0	3.1.0
	RP-06	RP-99641	017	1	RLC editorial corrections	3.0.0	3.1.0
01/00	-	-	-		Editorial corrections in title and Annex A (SDL)	3.1.0	3.1.1
	-	-	-		Correction of persistent error regarding SDL in Table of Contents	3.1.1	3.1.2
03/00	RP-07	RP-000040	018	1	RLC editorial changes	3.1.2	3.2.0
	RP-07	RP-000040	021	1	Corrections to RLC	3.1.2	3.2.0
	RP-07	RP-000040	025	2	Corrections to RLC	3.1.2	3.2.0
	RP-07	RP-000040	026	1	STATUS PDUs	3.1.2	3.2.0
	RP-07	RP-000040	027	1	Clarification of RLC AMD Model	3.1.2	3.2.0
	RP-07	RP-000040	028		Corrections to Timer_discard procedures	3.1.2	3.2.0
	RP-07	RP-000040	029	1	Segmentation of RLC SDUs	3.1.2	3.2.0
	RP-07	RP-000040	030	2	Modification of SDU discard to support virtual PDCP sequence numbers	3.1.2	3.2.0
	RP-07	RP-000040	031		Removal of SCCH	3.1.2	3.2.0
	RP-07	RP-000040	032		Updated RLC SDL	3.1.2	3.2.0
	RP-07	RP-000040	033	1	RLC Editorial Changes	3.1.2	3.2.0
	RP-07	RP-000040	034		Order of bit transmission for RLC PDUs	3.1.2	3.2.0
06/00	RP-08	RP-000220	038		(06/00) Corrections to RLC	3.2.0	3.3.0
	RP-08	RP-000220	039		Correction to the description of the MRW SUFI fields	3.2.0	3.3.0
	RP-08	RP-000220	040	1	Editorial corrections to length indicators and local suspend rate	3.2.0	3.3.0
	RP-08	RP-000220	041	4	Clarification of the RESET PDU	3.2.0	3.3.0
	RP-08	RP-000220	043	1	Clarification of RLC/MAC interaction	3.2.0	3.3.0
	RP-08	RP-000220	044	2	General RLC corrections	3.2.0	3.3.0
	RP-08	RP-000220	045		Clarification of RLC Transparent Mode operation	3.2.0	3.3.0
	RP-08	RP-000220	048		Editorial corrections to abbreviations, SCCH, BCCH	3.2.0	3.3.0
	RP-08	RP-000220	052		Updated RLC SDL	3.2.0	3.3.0
	RP-08	RP-000220	053		Correction to RLC	3.2.0	3.3.0
	RP-08	RP-000220	055		RLC Logical Channel mapping	3.2.0	3.3.0
	RP-08	RP-000220	057		Correction of EPC timer mechanism	3.2.0	3.3.0
09/00	RP-09	RP-000358	059	1	State variables after window change	3.3.0	3.4.0
	RP-09	RP-000358	060	4	SDU discard	3.3.0	3.4.0
	RP-09	RP-000358	061	5	General RLC corrections	3.3.0	3.4.0
	RP-09	RP-000358	066		Editorial changes to RLC	3.3.0	3.4.0
	RP-09	RP-000358	067	4	Correction to RLC window size range	3.3.0	3.4.0
	RP-09	RP-000358	068	2	Window based polling	3.3.0	3.4.0
	RP-09	RP-000358	070	2	General corrections to RLC	3.3.0	3.4.0
	RP-09	RP-000358	071		State Transition in RLC Acknowledged Mode	3.3.0	3.4.0
	RP-09	RP-000358	073		Clarification of the Length Indicators	3.3.0	3.4.0
	RP-09	RP-000358	076	1	RLC corrections	3.3.0	3.4.0
	RP-09	RP-000358	077	1	Corrections to reset procedure and length indicator definitions	3.3.0	3.4.0
	RP-09	RP-000358	078		RLC Modes for SHCCH	3.3.0	3.4.0
	RP-09	RP-000358	079		CCCH in UM RLC	3.3.0	3.4.0
12/00	RP-10	RP-000568	080	1	Length Indicator and PDU formats	3.4.0	3.5.0
	RP-10	RP-000568	083	3	Clarification to the Estimated PDU Counter	3.4.0	3.5.0
	RP-10	RP-000568	084	2	Model of UM and AM entities	3.4.0	3.5.0
	RP-10	RP-000568	085	1	General RLC corrections	3.4.0	3.5.0
	RP-10	RP-000568	086	1	General RLC corrections	3.4.0	3.5.0
	RP-10	RP-000568	087	5	RLC timers	3.4.0	3.5.0
	RP-10	RP-000568	088	1	Reset procedure	3.4.0	3.5.0

	RP-10	RP-000568	089	1	Editorial corrections to RLC	3.4.0	3.5.0
	RP-10	RP-000568	090	2	RLC UM protocol	3.4.0	3.5.0
	RP-10	RP-000568	092	2	Clarification to window size parameters, MRW SUFI and window based polling	3.4.0	3.5.0
	RP-10	RP-000568	093	3	General RLC Corrections	3.4.0	3.5.0
	RP-10	RP-000568	094	1	RLC Reset handling	3.4.0	3.5.0
	RP-10	RP-000568	095		Inclusion of stage 3 for ciphering	3.4.0	3.5.0

3GPP TSG-RAN2 Meeting #19
Sophia Antipolis, France, 19 - 23 February 2001

Tdoc R2-010351

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **103** ⌘ rev **-** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarification on MRW SUFI and SDU discard procedure
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 12, 2001
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change: ⌘
1. R2-010136 points out that Discard procedure may fail when several SDUs are discarded in the same MRW SUFI and the first SDU ends below the Rx window and the second ends above the Rx window.
2. R2-010137 points out that it can not be decided from the MRW_ACK SUFI or other discard termination conditions whether the SDUs indicated in the MRW SUFI are discarded by the receiver. Thus, it should be clarified that the SDUs that are requested to be discarded by a MRW SUFI shall not be confirmed to higher layer.

Summary of change: ⌘
1. The SN_MRW_i fields in the MRW SUFI shall be listed in sequential order.
2. Upon reception of the MRW SUFI, the receiver shall explain the SN_MRW_i list as follows:
   (1) If LENGTH=0 (“0000”), the receiver shall consider SN_MRW_1 to be equal to or greater than VR(R).
   (2) Else, the receiver shall consider SN_MRW_1 to be below VR(MR). All the other SN_MRW_i’s are considered to be after SN_MRW_1 and are considered to be listed in sequential order.
3. The SDUs that are requested to be discarded by a MRW SUFI shall not be confirmed to higher layer.

Consequences if not approved: ⌘
1. The discard procedure may fail in some cases.

Clauses affected: ⌘ 9.2.2.11.8, 11.6.3, 11.6.4

Other specs affected: ⌘
☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm.
Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3)With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

9.2.2.11.8 The Move Receiving Window (MRW) super-field

The 'Move Receiving Window' super-field is used to request the RLC receiver to move its receiving window and to indicate the amount of discarded SDUs, as a result of a SDU discard in the RLC transmitter. The format is given in the figure below.

Type = **MRW**
LENGTH
SN_MRW_1
SN_MRW_2
...
SN_MRW_{LENGTH}
N_{LENGTH}

Figure 9.15: The MRW fields in a STATUS PDU

LENGTH

Length: 4 bits

The number of SN_MRW_i fields in the super-field of type MRW**.** The values "0001" through "1111" indicate 1 through 15 SN_MRW_i respectively. The value "0000" indicates that one SN_MRW_i field is present and that the discarded SDU extends above the configured Tx window in the transmitter.

SN_MRW_i

Length: 12 bits

SN_MRW_i is used to indicate the end of each discarded SDU. SN_MRW_i is the sequence number of the PU that contains the LI of the i:th discarded SDU (except when $N_{LENGTH} = 0$, see definition of N_{LENGTH}). The order of the SN_MRW_i shall be in the same sequential order as the SDUs that they refer to.

Additionally SN_MRW_{LENGTH} requests the RLC receiver to discard all PUs with sequence number < SN_MRW_{LENGTH}, and to move the receiving window accordingly. In addition, the receiver has to discard the first N_{LENGTH} LIs and the corresponding data octets ~~bytes~~ in the PU with sequence number SN_MRW_{LENGTH}.

N_{LENGTH}

Length: 4 bits

N_{LENGTH} is used together with SN_MRW_{LENGTH} to indicate the end of the last discarded SDU.

N_{LENGTH} indicates which LI in the PU with sequence number SN_MRW_{LENGTH} corresponds to the last discarded SDU. $N_{LENGTH} = 0$ indicates that the last SDU ended in the PU with sequence number $SN\_MRW_{LENGTH} - 1$ and that the first data octet ~~byte~~ in the PU with sequence number SN_MRW_{LENGTH} is the first data octet ~~byte~~ to be reassembled next.

11.6.3 Reception of the STATUS PDU by the receiver

The receiver shall upon reception of the STATUS PDU/piggybacked STATUS PDU discard PUs and update the state variables VR(R), VR(H) and VR(MR) according to the received STATUS PDU/piggybacked STATUS PDU. Additionally the receiver should indicate the higher layers of all of the discarded SDUs.

The receiver shall initiate the transmission of a status report containing an MRW_ACK SUFI.

In the MRW_ACK SUFI, SN_ACK shall be set to the new value of VR(R), updated after reception of the MRW SUFI. The N field in the MRW_ACK SUFI shall be set to N_{LENGTH} field in the received MRW SUFI if the SN_ACK field is equal to SN_MRW_{LENGTH}. Otherwise N shall be set to 0.

The last discarded data octet ~~byte~~ is the octet ~~byte~~ indicated by the N_{LENGTH}:th LI field of the PU with sequence number SN_MRW_{LENGTH} and the succeeding data octet ~~byte~~ is the first data octet ~~byte~~ to be reassembled after the discard. When N_{LENGTH} = 0, the first data octet ~~byte~~ of the PU with sequence number SN_MRW_{LENGTH} is the first data octet ~~byte~~ to be reassembled after the discard.

~~If the MRW SUFI indicates an SN_MRW_i outside the interval VR(R)≤ SN_MRW_i <VR(MR) , the Rx shall consider the sequence number to be below VR(R), unless LENGTH="0000" or at least the first indicated SN_MRW_i in the MRW SUFI is within the interval VR(R)≤ SN_MRW_i <VR(MR), in which case the sequence number shall be considered to be above or equal to VR(MR).~~If LENGTH="0000", the sequence number SN_MRW_1 is considered to be above or equal to VR(R). Else, the sequence number SN_MRW_1 is considered to be less than VR(MR). All the SN_MRW_i's other than SN_MRW_1 are considered to be in sequential order within the list and sequentially above or equal to SN_MRW_{i-1}.

11.6.4 Termination

The procedure is terminated in the sender in the following cases:

1. On the reception of a STATUS PDU which contains an MRW_ACK SUFI with SN_ACK > SN_MRW_{LENGTH} and the N field is equal to zero.
2. On the reception of a STATUS PDU which contains an ACK SUFI indicating VR(R) > SN_MRW_{LENGTH}
3. On reception of a STATUS PDU which contains an MRW_ACK with SN_ACK = SN_MRW_{LENGTH} and N is equal to the N_{LENGTH} indicated in the transmitted MRW SUFI.

If one of the termination criteria above is fulfilled, Timer_MRW is stopped and the discard procedure is terminated. The SDUs that are requested to be discarded shall not be confirmed to higher layer.

When VT(MRW) reaches MaxMRW, the procedure is terminated and an RLC reset is performed.

3GPP TSG-RAN2 Meeting #19 **R2-010618**
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **104** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	General clarification on SN arithmetic comparison
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 21, 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change: ⌘

1. The current basic rule of arithmetic comparison is confined to state variables and VT(A) or VR(R) is always implied to be the lower bound of the compared variables.
2. However, the arithmetic comparisons are not confined to state variables. In the SDU discard procedure, as an example, there are comparisons between SN_MRW_i and SN_ACK, which are SN values and are not state variables. Thus, the comparison rule should apply to any SN values.
3. There are some occasions that VR(R) is compared at the sender. See 11.6.4, 'On the reception of a STATUS PDU which contains an ACK SUFI indicating VR(R) > SN_MRW_{LENGTH}'. If apply the current rule, the condition VR(R) > SN_MRW_{LENGTH} will never be true since VR(R) is assumed to be the base.
4. The arithmetic comparison rule for UM is not given explicitly.

Summary of change: ⌘

1. The arithmetic comparison rule is modified to include both state variables and sequence number.
2. The arithmetic comparison rule for UM is added.

Consequences if not approved: ⌘ Arithmetic comparisons will be unclarified.

Clauses affected: ⌘ 9.3.2.3, 9.3.3.4, 9.4, 9.7.6, 11.2.3, 11.3.4.2, 11.6.2, 11.6.4

Other specs affected: ⌘
☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘	

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3)With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

9.3.2 State model for unacknowledged mode entities

Figure 9.17 illustrates the state model for unacknowledged mode RLC entities (both transmitting and receiving). An unacknowledged mode entity can be in one of following states.

9.3.2.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is created and unacknowledged data transfer ready state is entered.

9.3.2.2 Unacknowledged Data Transfer Ready State

In the unacknowledged data transfer ready, unacknowledged mode data can be exchanged between the entities. Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

9.3.2.3 Local Suspend State

Upon reception of CRLC-SUSPEND-Req from higher layer (RRC) the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send RLC-PDUs with SN≥~~>=~~VT(US)+N. Upon reception of CRLC-RESUME-Req from higher layer (RRC) the RLC entity is resumed and the Data Transfer Ready state is entered.

9.3.3 State model for acknowledged mode entities

9.3.3.4 Local Suspend State

Upon reception of CRLC-SUSPEND-Req from higher layer (RRC) in Acknowledge Data Transfer Ready State the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send ~~a~~ RLC-PDUs with ~~a~~ SN≥~~>=~~VT(S)+N. Upon reception of CRLC-RESUME-Req from higher layer (RRC) in this state, the RLC entity is resumed and the Data Transfer Ready state is entered.

9.4 State variables

This sub-clause describes the state variables used in the specification of the peer-to-peer protocol. All state variables are non-negative integers. PUs are sequentially and independently numbered and may have the value 0 through n minus 1 (where n is the modulus of the sequence numbers). The modulus equals 2^{12} for AM and 2^{7} for UM; the sequence numbers cycle through the entire range: 0 through $2^{12} - 1$ for AM and 0 through $2^{7} - 1$ for UM. All arithmetic operations on the following state variables and sequence numbers contained in this specification are affected by the modulus: VT(S), VT(A), VT(MS), VR(R), VR(H), VR(MR), VT(US) and VR(US). When performing arithmetic comparisons of ~~transmitter~~ variables or SN values at the sender, VT(A) and VT(US) are~~is~~ assumed to be the base in AM and UM respectively. When performing arithmetic comparisons of ~~receiver~~ variables or SN values at the receiver , VR(R) and VR(US) are ~~is~~ assumed to be the base in AM and UM respectively.

The RLC maintains the following state variables at the transmitter.

a) VT(S) - Send state variable.

   The sequence number of the next PU to be transmitted for the first time (i.e. excluding retransmission). It is updated after transmission of a PDU, which includes not earlier transmitted PUs and after transmission of a MRW SUFI which includes SN_MRW_{LENGTH} ≥~~>~~VT(S). The initial value of this variable is 0.

b) VT(A) - Acknowledge state variable.

   The sequence number of the next in-sequence PU expected to be acknowledged, which forms the lower edge of the window of acceptable acknowledgements. VT(A) is updated based on receipt of a STATUS PDU including an ACK and/or MRW_ACK super-field. The initial value of this variable is 0.

c) VT(DAT).

This state variable counts the number of times a PU has been transmitted. There is one VT(DAT) for each PU and it is incremented each time the PU is transmitted. The initial value of this variable is 0.

d) VT(MS) - Maximum Send state variable.

The sequence number of the first PU not allowed by the peer receiver [i.e. the receiver will allow up to VT(MS) – 1], VT(MS) = VT(A) + VT(WS). This value represents the upper edge of the transmit window. The transmitter shall not transmit a PU with SN $\geq$ VT(MS). VT(MS) is updated when either VT(A) or VT(WS) is updated. The PU with SN VT(S)-1 can be transmitted also when VT(S) $\geq$ VT(MS).

The RLC maintains the following state variables at the receiver:

a) VR(R) - Receive state variable.

The sequence number of the next in-sequence PU expected to be received. It is set equal to SNmax+1 upon receipt of the next in-sequence PU, where SNmax is the sequence number of the highest received in-sequence PU. The initial value of this variable is 0.

b) VR(H) - Highest expected state variable.

The sequence number of the highest expected PU. This state variable is set equal to SN+1 only when a new PU is received with VR(MR)>SN$\geq$VR(H). The initial value of this variable is 0.

c) VR(MR) - Maximum acceptable Receive state variable.

The sequence number of the first PU not allowed by the receiver [i.e. the receiver will allow up to VR(MR) – 1], VR(MR) = VR(R) + Configured_Rx_Window_Size. The receiver shall discard PUs with SN $\geq$ VR(MR).

d) VR(US) - Receiver Send Sequence state variable.

The sequence number of the next PDU to be received. It shall set equal to SN + 1 upon reception of a PDU. The initial value of this variable is 0.

e) VR(EP) - Estimated PDU Counter state variable.

The number of PUs that should be received yet as a consequence of the transmission of the latest status report. In acknowledged mode, this state variable is updated at the end of each transmission time interval. It is decremented by the number of PUs that should have been received during the transmission time interval. If VR(EP) is equal to zero, then check if all PUs requested for retransmission in the latest status report have been received.

9.7.6 Local Suspend function for acknowledged and unacknowledged mode

The higher layer (RRC) may suspend the RLC entity. The CRLC-SUSPEND-Req indicates this request. The RLC entity shall, when receiving this request, not send RLC PDUs with SN≥>=VT(S)+N for AM and SN≥>=VT(US)+N for UM ,where N is given by the CRLC_SUSPEND-Req primitive. The RLC entity shall acknowledge the CRLC-SUSPEND-Req ordering a suspend with a CRLC-SUSPEND-Conf with the current value of VT(S) for AM and VT(US) for UM. The suspend state is left when a CRLC-RESUME-Req primitive indicating resume is received.

11.2.3 Reception of UMD PDU

Upon reception of a UMD PDU, the receiver shall update VR(US) state variable according to the received PDU(s).

The PDUs are reassembled into RLC SDUs. If the updating step of the variable VR(US) is greater than one, ~~a~~ one or more SPDUs are ~~PDU with sequence number < VR(US) is~~ missing. ~~then all~~ The SDUs that have segments in ~~this~~ these missing PDUs shall be discarded. RLC delivers the RLC SDUs to the higher layer through the UM-SAP.

11.3.4 Abnormal cases

11.3.4.1 Timer_Poll timeout

Upon expiry of the Timer_Poll, the sender shall retransmit the poll. The poll can be retransmitted in either a new PDU or a retransmitted PDU.

11.3.4.2 Receiving a PU outside the receiving window

Upon reception of a PU with ~~SN<VR(R) or SN≥VR(MR)~~ ~~SN~~sequence number outside the interval VR(R)≤SN<VR(MR), the receiver shall discard the PU. The poll bit shall be considered even if a complete PDU is discarded.

11.6 SDU discard with explicit signalling procedure

11.6.2 Initiation

This procedure is initiated by the sender when the following conditions are fulfilled:

1) Timer based SDU discard with explicit signalling is used, and Timer_Discard expires for an SDU.

2) SDU discard after MaxDAT number of retransmissions is used, and MaxDAT number of retransmissions is reached for an SDU.

The sender shall discard all PUs that contain segments of the associated SDUs. If the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded. VT(A) shall be updated when the procedure is terminated, and VT(S) shall be updated when a new MRW SUFI which includes SN_MRW_{LENGTH} ≥ VT(S) is transmitted.

The sender shall transmit a status report on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

This status report is sent even if the 'STATUS prohibit' is used and the timer 'Timer_Status_Prohibit' or 'Timer_EPC' is active.

The STATUS PDUs have higher priority than data PDUs.

The sender shall start timer Timer_MRW. If a new SDU discard procedure is triggered when Timer_MRW is running, no new MRW SUFIs shall be sent before the current SDU discard procedure is terminated by one of the termination criteria.

11.6.2.1 Piggybacked STATUS PDU

It is possible to piggyback a STATUS PDU on an AMD PDU. If a PDU includes padding a piggybacked STATUS PDU can be inserted instead of the padding.

11.6.2.2 STATUS PDU contents to set

The size of the STATUS PDU shall be equal to one of the allowed PDU sizes. The discard information shall not be split into several MRW SUFIs.

The status report shall include the MRW SUFI, other SUFI fields can be used additionally. MRW SUFI shall convey information about the discarded SDU(s) to the receiver.

In order to discard a single SDU that ends in a PDU with SN≥ VT(A)+Configured_Tx_Window_Size, the LENGTH field in the MRW SUFI shall be set to "0000". If more then one SDU are discarded with the same MRW SUFI, at least the first discarded SDUs must end (i.e. the LI must be located) in a PDU with SN in the interval VT(A)≤ SN <VT(A)+Configured_Tx_Window_Size.

Padding shall be inserted if the SUFI fields do not fill the entire STATUS PDU. If the STATUS PDU contains padding the last SUFI field shall be either an ACK SUFI or a NO_MORE SUFI. If there is no padding in the STATUS PDU, NO_MORE SUFI or ACK SUFI does not need to be included in the STATUS PDU.

11.6.3 Reception of the STATUS PDU by the receiver

The receiver shall upon reception of the STATUS PDU/piggybacked STATUS PDU discard PUs and update the state variables VR(R), VR(H) and VR(MR) according to the received STATUS PDU/piggybacked STATUS PDU. Additionally the receiver should indicate the higher layers of all of the discarded SDUs.

The receiver shall initiate the transmission of a status report containing an MRW_ACK SUFI.

In the MRW_ACK SUFI, SN_ACK shall be set to the new value of VR(R), updated after reception of the MRW SUFI. The N field in the MRW_ACK SUFI shall be set to N_{LENGTH} field in the received MRW SUFI if the SN_ACK field is equal to SN_MRW_{LENGTH}. Otherwise N shall be set to 0.

The last discarded data byte is the byte indicated by the N_{LENGTH}:th LI field of the PU with sequence number SN_MRW_{LENGTH} and the succeeding data byte is the first data byte to be reassembled after the discard. When N_{LENGTH} = 0, the first data byte of the PU with sequence number SN_MRW_{LENGTH} is the first data byte to be reassembled after the discard.

If the MRW SUFI indicates an SN_MRW_i outside the interval $VR(R) \leq SN\_MRW_i < VR(MR)$, the Rx shall consider the sequence number to be below VR(R), unless LENGTH="0000" or at least the first indicated SN_MRW_i in the MRW SUFI is within the interval $VR(R) \leq SN\_MRW_i < VR(MR)$, in which case the sequence number shall be considered to be above or equal to VR(MR).

11.6.4 Termination

The procedure is terminated in the sender in the following cases:

1. On the reception of a STATUS PDU which contains an MRW_ACK SUFI with $SN\_ACK > SN\_MRW_{LENGTH}$ and the N field is equal to zero.
2. On the reception of a STATUS PDU which contains an ACK SUFI indicating $VR(R-) > SN\_MRW_{LENGTH}$
3. On reception of a STATUS PDU which contains an MRW_ACK with $SN\_ACK = SN\_MRW_{LENGTH}$ and N is equal to the N_{LENGTH} indicated in the transmitted MRW SUFI.

If one of the termination criteria above is fulfilled, Timer_MRW is stopped and the discard procedure is terminated.

When VT(MRW) reaches MaxMRW, the procedure is terminated and an RLC reset is performed.

11.6.5 Expiration of timer Timer_MRW

If Timer_MRW expires before the discard procedure is terminated, the MRW SUFI shall be retransmitted, VT(MRW) is incremented by one and Timer_MRW restarted. MRW SUFI shall be exactly the same as previously transmitted even though some new SDUs would have been discarded during the running of the Timer_MRW. If the retransmitted STATUS PDU contains other SUFIs than the MRW SUFI, the status information indicated by these SUFIs shall be updated.

11.6.6 Abnormal cases

11.6.6.1 Obsolete/corrupted MRW command

If the MRW command contains outdated information about the receiver window (receiver window already moved further than MRW command is indicating), the MRW command shall be discarded and a status report containing SUFI MRW_ACK shall be transmitted indicating the value of VR(R) and the N field shall be set to zero.

11.6.6.2 VT(MRW) equals MaxMRW

If the number of retransmission of a MRW command (i.e. VT(MRW)) reaches MaxMRW, an error indication shall be passed to RRC and RESET procedure shall be performed.

11.6.6.3 Reception of obsolete MRW_ACK

The received MRW_ACK shall be discarded in the following cases.

1. If timer Timer_MRW is not active.
2. If the SN_ACK field in the received $MRW\_ACK < SN\_MRW_{LENGTH}$ in the transmitted MRW SUFI.
3. If the SN_ACK field in the received MRW_ACK is equal to the SN_MRW_{LENGTH} in the transmitted MRW SUFI and the N field in the received MRW_ACK is not equal to the N_{LENGTH} field in the transmitted MRW SUFI

4. If the SN_ACK field in the received MRW_ACK > SN_MRW_{LENGTH} in the transmitted MRW SUFI and the N field in the received MRW_ACK is not equal to zero.

3GPP TSG-RAN2 Meeting #19 ***R2-010682***
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **105** ⌘ rev **r2** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	General clarification on RLC header and PDU header
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 22, 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change:	⌘	1. The PDU header is 'defined' in the very last subclause and is in an informal 'i.e.' description. The RLC header is not defined explicitly at all in the specification. 2. Headers are used in the specification in an inconsistent way and may cause confusion and unclarification.
Summary of change:	⌘	1. Explicit formal definition of the two types of header are added in the PDU format subclauses. 2. Inconsistent places of headers are corrected.
Consequences if not approved:	⌘	Inconsistent header definitions may cause confusion.

Clauses affected:	⌘	4.2.1.3, 9.2.1.3, 9.2.1.4, 9.2.2.8, 11.2.2.1, 11.2.4.2, 11.3.2.1, 11.3.4.5, 11.7
Other specs affected:	⌘	☐ Other core specifications ⌘ ☐ Test specifications ☐ O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be

downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3)With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

4.2.1.2 Unacknowledged mode entities

The transmitting UM-entity receives SDUs from the higher layers. RLC might segment the SDUs into RLC PDUs of appropriate size. The SDU might also be concatenated with other SDUs.. RLC delivers the RLC PDUs to MAC through either a DCCH, CTCH or a DTCH. The CCCH and SHCCH also uses unacknowledged mode, but only for the downlink. Which type of logical channel depends on if the higher layer is located in the control plane (CCCH, DCCH, SHCCH) or user plane (CTCH, DTCH).

The receiving UM-entity receives PDUs through one of the logical channels from the MAC sublayer. RLC removes header from the PDUs and reassembles the PDUs (if segmentation has been performed) into RLC SDUs. The RLC SDUs are delivered to the higher layer.

4.2.1.3 Acknowledged mode entity

The transmitting side of the AM-entity receives SDUs from the higher layers. The SDUs are segmented and/or concatenated to PUs of fixed length. PU length is a semi-static value that is decided in bearer setup and can only be changed through bearer reconfiguration by RRC.

For purposes of RLC buffering and retransmission handling, the operation is the same as if there would be one PU per PDU. For concatenation or padding purposes, bits of information on the length and extension are inserted into the beginning of the last PU where data from an SDU is included. Padding can be replaced by piggybacked status information. This includes setting the poll bit.

If several SDUs fit into one PU, they are concatenated and the appropriate length indicators are inserted into the beginning of the PU. After that the PUs are placed in the retransmission buffer and the transmission buffer. One PU is included in one RLC PDU.

The MUX then decides which PDUs and when the PDUs are submitted to lower layer. The PDUs are submitted via a function that completes the ~~RLC~~ AMD PDU header and potentially replaces padding with piggybacked status information. The RLC entity shall assume a PDU to be transmitted when the PDU is submitted to lower layer.

The ciphering is applied only for AMD PDUs. The fixed 2 octet AMD PDU header is not ciphered. Piggybacked and Padding parts of AMD PDU when existing are ciphered. The other Control PDUs (e.g, STATUS, RESET, and RESET ACK PDU) shall not be ciphered.

When Piggybacking mechanism is applied the padding is replaced by control information, in order to increase the transmission efficiency and making possible a faster message exchange between the peer to peer RLC entities. The piggybacked control information is not saved in any retransmission buffer. The piggybacked control information is contained in the piggybacked STATUS PDU, which is in turn included into the AMD-PDU. The piggybacked STATUS PDUs will be of variable size in order to match with the amount of free space in the AMD PDU.

The retransmission buffer also receives acknowledgements from the receiving side, which are used to indicate retransmissions of PUs and when to delete a PU from the retransmission buffer.

The Receiving Side of the AM-entity receives PDUs through one of the logical channels from the MAC sublayer. The RLC-PDUs are expanded into separate PUs and potential piggybacked status information are extracted. The PUs are placed in the receiver buffer until a complete SDU has been received. The receiver buffer requests retransmissions of PUs by sending negative acknowledgements to the peer entity. After that the RLC headers are removed from the PDUs and the PDUs are reassembled into a SDU. Finally the SDU is delivered to the higher layer. The receiving side also receives acknowledgements from the peer entity. The acknowledgements are passed to the retransmission buffer on the transmitting side.

9.2.1.3 UMD PDU

The UMD PDU transfers user data when RLC is operating in unacknowledged mode. The length of the data part shall be an integer number of octets. The UMD PDU header ~~contains~~consists of the first octet, which contains the sequence number. The RLC header ~~contains~~consists of the first octet and all the octets that contain ~~L~~length indicators.

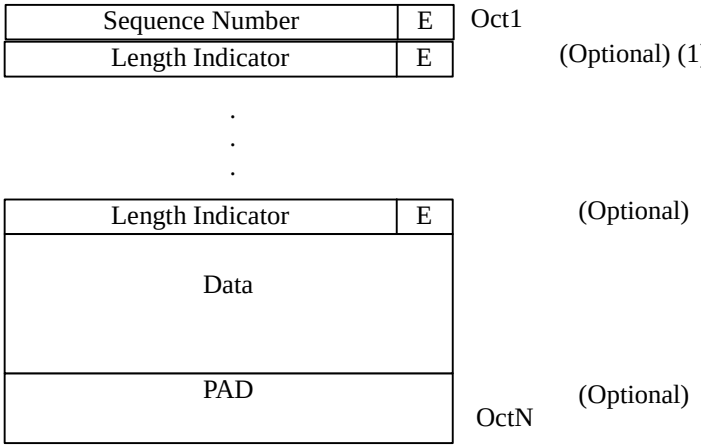
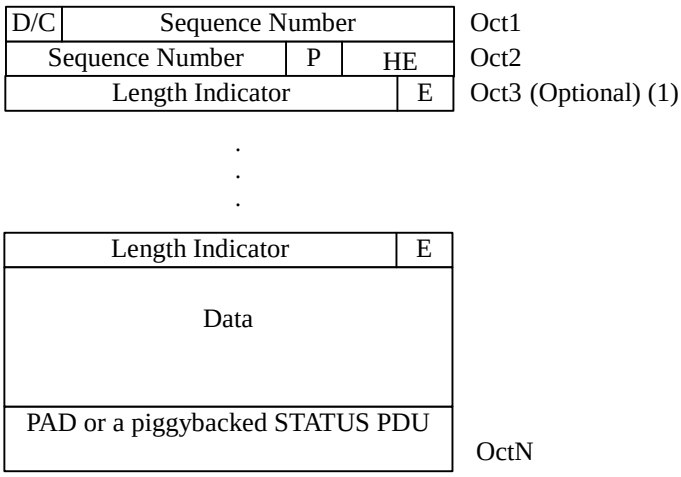


Figure 9.2: UMD PDU

NOTE (1): The Length Indicator may be 15 bits.

9.2.1.4 AMD PDU

The AMD PDU transfers user data and piggybacked status information and requests status report by setting Poll bit when RLC is operating in acknowledged mode. The length of the data part shall be an integer number of octets. The AMD PDU header ~~contains~~consists of the first two octets, which contain the sequence number. The RLC header ~~contains~~consists of the first two octets and all the octets that contain ~~L~~length indicators.



NOTE (1): The Length Indicator may be 15 bits.

Figure 9.3: AMD PDU

9.2.2.8 Length Indicator (LI)

The Length Indicator is used to indicate, each time, the end of an SDU occurs in the PU. The Length Indicator points out the number of octets between the end of the last Length Indicator field and up to and including the octet at the end of an SDU segment. Length Indicators are included in the PUs that they refer to. The size of the Length Indicator may be either 7bits or 15bits. The ~~maximum~~ value of a Length Indicator ~~in AM will be no greater than~~ shall not exceed the ~~calculated values of RLC PDU size – AMD PDU Header~~RLC header size ~~– PADDING~~ size. ~~(See~~ specified in subclauses 11.2.4.2 and 11.3.4.5.~~)~~ ~~The maximum value of a Length Indicator in UM will be no greater than the RLC PDU size – UMD PDU Header – PADDING.~~

A Length Indicator group is a set of Length Indicators that refer to a PU. Length Indicators that are part of a Length Indicator group must never be reordered within the Length Indicator group or removed from the Length Indicator group.

11.1.2.1 TrD PDU contents to set

The TrD PDU includes a complete SDU or a segment of an SDU. How to perform the segmentation is decided upon when the service is established. No overhead or header is added, instead segmentation is done based on which of the transport formats of the transport channel that will be used. A particular transport format informs the receiver how the segmentation was performed.

11.2.2.1 UMD PDU contents to set

The Sequence Number field shall be set equal to VT(US).

The Extension bit shall be set to 1 if the next field is a length indicator field, otherwise it shall be set to zero.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the RLC header fields and the end of the segment. If padding is needed, another LI field set to only 1's shall be added unless the padding size is one octet for PDUs with 15-bit LIs. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

11.2.3 Reception of UMD PDU

Upon reception of a UMD PDU, the receiver shall update VR(US) state variable according to the received PDU(s).

The PDUs are reassembled into RLC SDUs. If a PDU with sequence number < VR(US) is missing then all SDUs that have segments in this PDU shall be discarded. RLC delivers the RLC SDUs to the higher layer through the UM-SAP.

11.2.4 Abnormal cases

11.2.4.1 Length Indicator value 1111110

Upon reception of an UMD PDU that contains Length Indicator value 1111110 or 111111111111110 ("piggybacked STATUS PDU", in case 7bit or 15 bit Length Indicator field is used, respectively) the receiver shall discard that UMD PDU. This Length Indicator value is not used in unacknowledged mode data transfer.

11.2.4.2 Invalid length indicator value

If the length indicator of a PDU has a value that is larger than the PDU size – ~~the number of octets containing LIs in the PDU – 1~~RLC header size and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PDU shall be discarded and treated as a missing PDU.

11.3 Acknowledged mode data transfer procedure

11.3.2.1 AMD PDU contents to set

If the PDU is transmitted for the first time, the Sequence Number field shall be set equal to VT(S) and VT(S) shall be updated.

The setting of the Polling bit is specified in subclause 11.3.2.1.1.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the RLC header fields and the end of the segment. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

How to perform the segmentation of a SDU is specified in subclause 11.3.2.1.2.

11.3.3 Reception of AMD PDU by the receiver

Upon reception of a AMD PDU, the receiver shall update VR(R), VR(H) and VR(MR) state variables according to the received PU(s).

If any of the PUs includes a Polling bit set to 1, the STATUS PDU transfer procedure shall be initiated.

If the detection of missing PU(s) shall be used and the receiver detects that a PU is missing, the receiver shall initiate the STATUS PDU transfer procedure.

11.3.4 Abnormal cases

11.3.4.5 Invalid length indicator value

If the length indicator of a PU has a value that is larger than the PDU size – ~~the number of octets containing LIs in the PU~~RLC header size and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PU shall be discarded and treated as a missing PU.

11.7 Ciphering

The ciphering function is performed in RLC, according to the following rules if a radio bearer is using a non-transparent RLC mode (AM or UM). The data unit that is ciphered, depends on the transmission mode as described below.

- For RLC UM mode, the ciphering unit is the UMD PDU excluding the first octet, i.e. excluding the RLC UMD PDU header. This is shown below in Figure 11.7.1.

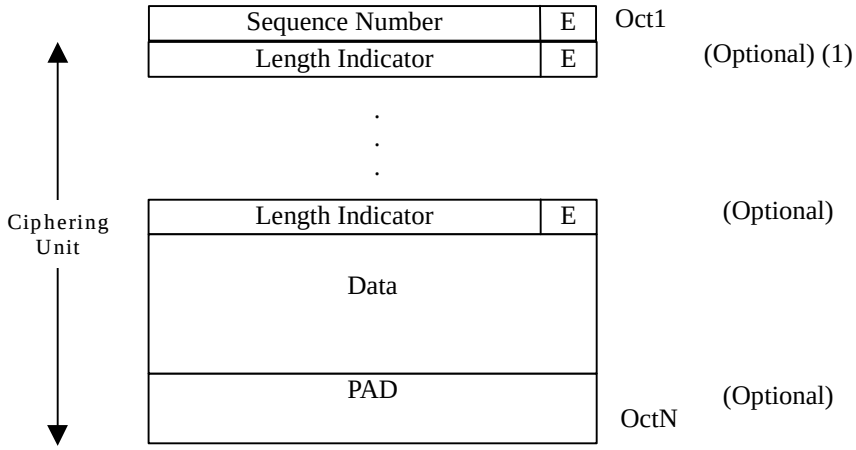


Figure 11.7.1: Ciphering unit for a UMD PDU

- For RLC AM mode, the ciphering unit is the AMD PDU excluding the first two ~~first~~ octets, i.e. excluding the RLC AMD PDU header. This is shown below in Figure 11.7.2.

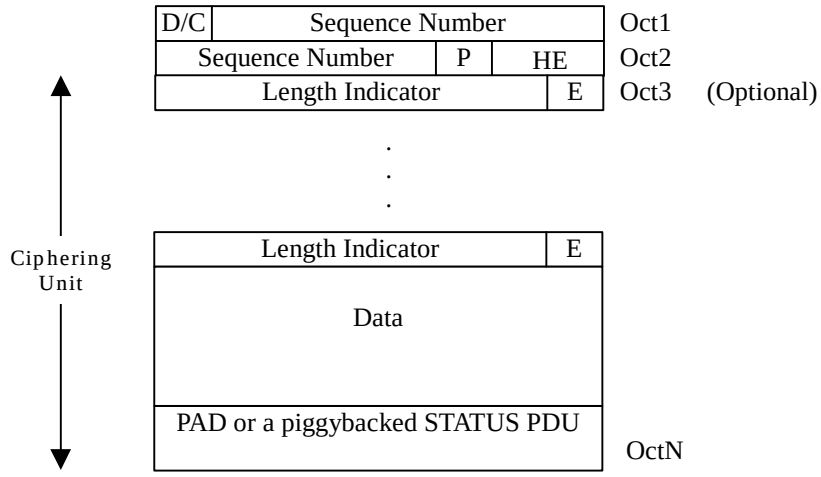


Figure 11.7.1: Ciphering unit for a AMD PDU

The ciphering algorithm and key to be used are configured by upper layers [8] and the ciphering method shall be applied as specified in [10].

The parameters that are required by RLC for ciphering are defined in [10] and are input to the ciphering algorithm. The parameters required by RLC which are provided by upper layers [8] are listed below:

- RLC AM HFN (Hyper frame number for radio bearers that are mapped onto RLC AM)
- RLC UM HFN (Hyper frame number for radio bearers that are mapped onto RLC UM)
- BEARER (Radio Bearer ID)
- CK (Ciphering Key)

is not equal to zero.

3GPP TSG-RAN2 Meeting #19 **R2-010615**
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **106** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarification on the primitives between RLC and higher layers
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 21, 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change: ⌘ 1. Some parameters needed for RLC functions are missing :

(1) Segmentation indication (Booling, TM only)

(2) In-sequence delivery indication (Booling, AM only)

(3) Activation time should be differentiate for DL and UL.

(4) Timer_Discard for TM and UM

~~2.~~PU size should be deleted from the AM_parameters since there is only one PU in one PDU for Release 99 and the PDU size is already given by MAC. Else, the two values might conflict to each other.

~~3.~~2. The No_discard mode is missing from the SDU discard mode parameter in subclause 9.7.3.

Summary of change: ⌘ 1. The missing primitive paramters are included. ~~PU size is deleted from the AM_parameters.~~

2. No_discard mode is included to the SDU discard mode parameter in subclause 9.7.3.

Consequences if not approved: ⌘ The Specification is incomplete.

Clauses affected: ⌘ 8.1, 8.2, 9.7.3, 9.7.3.4 (new), 11.3.4.4

Other specs affected: ⌘
☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘	

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3)With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

8 Elements for layer-to-layer communication

The interaction between the RLC layer and other layers are described in terms of primitives where the primitives represent the logical exchange of information and control between the RLC layer and other layers. The primitives shall not specify or constrain implementations.

8.1 Primitives between RLC and higher layers

The primitives between RLC and upper layers are shown in Table 8.1.

Table 8.1: Primitives between RLC and upper layers

Generic Name	Parameter			
	Req.	**Ind.**	**Resp.**	**Conf.**
RLC-AM-DATA	Data, CNF, MUI	Data, DiscardInfo	Not Defined	MUI
RLC-UM-DATA	Data, Use special LI	Data	Not Defined	Not Defined
RLC-TR-DATA	Data	Data	Not Defined	Not Defined
CRLC-CONFIG	E/R, Stop, Continue, Ciphering Elements (UM/AM only), TM_parameters (TM only), UM_parameters (UM only), AM_parameters (AM only)	Not Defined	Not Defined	Not Defined
CRLC-SUSPEND (UM/AM only)	N	Not Defined	Not Defined	VT(US) (UM only), VT(S) (AM only)
CRLC-RESUME (UM/AM only)	No Parameter	Not Defined	Not Defined	Not Defined
CRLC-STATUS	Not Defined	EVC	Not Defined	Not Defined

Each Primitive is defined as follows:

RLC-AM-DATA-Req/Ind/Conf

- RLC-AM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in acknowledged mode.
- RLC-AM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in acknowledged mode and to indicate higher layers of the discarded RLC SDU in the receiving RLC.
- RLC-AM-DATA-Conf is used by RLC to confirm to higher layers the transmission of a RLC SDU.

RLC-UM-DATA-Req/Ind

- RLC-UM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in unacknowledged mode.
- RLC-UM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in unacknowledged mode.

RLC-TR-DATA-Req/Ind

- RLC-TR-DATA-Req is used by higher layers to request transmission of a higher layer PDU in transparent mode.
- RLC-TR-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in transparent mode.

CRLC-CONFIG-Req

This primitive is used by RRC to establish, re-establish, release, stop, continue or reconfigure the RLC. Ciphering elements are included for UM and AM operation.

CRLC-SUSPEND-Req/Conf

This primitive is used by RRC to suspend the RLC. The N parameter indicates that RLC shall not send a PDU with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM, where N is an integer. RLC informs RRC of the VT(S) for AM and VT(US) for UM in the confirm primitive.

CRLC-RESUME-Req

This primitive is used by RRC to resume RLC when RLC has been suspended.

CRLC-STATUS-Ind

It is used by the RLC to send status information to RRC.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the correct transmission of the RLC SDU.

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC If it indicates (re-)establishment, the state variables in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state. If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the PU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (i.e., unrecoverable errors such as data link layer loss or recoverable status events such as reset, etc.).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Transmitting Activation Time (SN to activate a new ciphering configuration at the transmitter), Receiving Activation Time (SN to activate a new ciphering configuration at the receiver) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains PDU size, ~~PU size,~~ In-sequence Delivery Indication (indicating that SDUs shall be deliver to the upper layers in sequence or out of sequence), Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), SDU discard mode (see subclause 9.7.3) and Minimum WSN (see subclause 9.2.2.11.3). The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set.

8) The parameter DiscardInfo indicates the upper layer of each of the discarded RLC SDU. It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly in the end or one octet short (only when 15 bit LI is used) ~~in the end~~ of the previous RLC PDU is present, the LI shall not be used.

11) The UM_parameters are only applicable for UM operation. It contains Timer_Discard value (see subclause 9.5).

12) The TM_parameters are only applicable for TM operation. It contains Segmentation indication (see subclauses 9.2.2.9 and 11.1.2.1) and Timer_Discard value (see subclause 9.5).

9.7.3 SDU discard function for acknowledged, unacknowledged, and transparent mode

The SDU discard function allows to discharge RLC PDU from the buffer on the transmitter side, when the transmission of the RLC PDU does not success for a long time. The SDU discard function allows to avoid buffer overflow. There will be several alternative operation modes of the RLC SDU discard function, and which discard function to use will be given by the QoS requirements of the Radio Access Bearer.

The following is a list of operation modes for the RLC SDU discard function.

Table 9.2: List of criteria's that control when to perform SDU discard

Operation mode	Presence
Timer based discard, with explicit signalling	Network controlled
Timer based discard, without explicit signalling	Network controlled
SDU discard after MaxDAT number of retransmissions	Network controlled
No_discard after MaxDAT number of retransmissions	Network controlled

9.7.3.1 Timer based discard, with explicit signalling

This alternative uses a timer based triggering of SDU discard (Timer_Discard). This makes the SDU discard function insensitive to variations in the channel rate and provides means for exact definition of maximum delay. However, the SDU loss rate of the connection is increased as SDUs are discarded.

For every SDU received from a higher layer, timer monitoring of the transmission time of the SDU is started. If the transmission time exceeds a predefined value for a SDU in acknowledged mode RLC, this SDU is discarded in the transmitter and a Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded.

The MRW command is defined as a super-field in the RLC STATUS PDU (see subclause 9.2), and piggybacked to status information of transmissions in the opposite direction. If the MRW command has not been acknowledged by receiver, it will be retransmitted. Therefore, SDU discard variants requiring peer-to-peer signalling are only possible for full duplex connections.

9.7.3.2 Timer based discard, without explicit signalling

This alternative uses the same timer based trigger for SDU discard (Timer_Discard) as the one described in the subclause 9.7.3.1. The difference is that this discard method does not use any peer-to-peer signalling. This function is applied only for unacknowledged and transparent mode RLC and peer-to-peer signalling is never needed. The SDUs are simply discarded in the transmitter, once the transmission time is exceeded.

9.7.3.3 SDU discard after MaxDAT number of retransmissions

This alternative uses the number of retransmissions as a trigger for SDU discard, and is therefore only applicable for acknowledged mode RLC. This makes the SDU discard function dependent of the channel rate. Also, this variant of the SDU discard function strives to keep the SDU loss rate constant for the connection, on the cost of a variable delay. SDU discard is triggered at the transmitter, and a MRW command is necessary to convey the discard information to the receiver, like in the timer based discard with explicit signalling.

9.7.3.4 No_discard after MaxDAT number of retransmissions

This alternative uses the number of retransmissions, and is therefore only applicable for acknowledged mode RLC. Reset procedure ~~instead of SDU discard procedure~~ shall be ~~done~~ initiated after MaxDAT number of retransmissions of an~~y~~ AMD PDU (see subclause 11.3.4.4).

9.7.5 Multiple payload units in an RLC PDU for acknowledged mode

The possibility to include multiple payload units (PU) into one RLC AMD PDU is part of the service capabilities of a UE in acknowledged mode. For Release 99, there shall be only one PU per AMD PDU.

A payload unit is the smallest unit that can be separately addressed for retransmission and is of fixed size, containing data and optionally, length indicators and/or padding. The padding space of a PU can be used to piggyback STATUS PDUs.

The size of the PU is set by the RRC.

11.3.4 Abnormal cases

11.3.4.1 Timer_Poll timeout

Upon expiry of the Timer_Poll, the sender shall retransmit the poll. The poll can be retransmitted in either a new PDU or a retransmitted PDU.

11.3.4.2 Receiving a PU outside the receiving window

Upon reception of a PU with SN<VR(R) or SN≥VR(MR), the receiver shall discard the PU. The poll bit shall be considered even if a complete PDU is discarded.

11.3.4.3 Timer_Discard timeout

11.3.4.3.1 SDU discard with explicit signalling

Upon expiry of Timer_Discard, the sender shall initiate the SDU discard with explicit signalling procedure.

11.3.4.4 VT(DAT) ≥ MaxDAT

If SDU discard after MaxDAT number of retransmission is used and VT(DAT) ≥ MaxDAT for any PU, the sender shall initiate the SDU discard with explicit signalling procedure for the SDUs to which the PU with VT(DAT) ≥MaxDAT belongs.

If ~~the SDU discard is not~~ No_discard after MaxDAT number of retransmissions is used, the sender shall initiate the RLC reset procedure when VT(DAT) ≥ MaxDAT.

11.3.4.5 Invalid length indicator value

If the length indicator of a PU has a value that is larger than the PU size – the number of octets containing LIs in the PU and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PU shall be discarded and treated as a missing PU.

3GPP TSG-RAN2 Meeting #19 **R2-010616**
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **107** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarification on the model of AM entity
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 21, 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change: ⌘ In subclause 4.2.1.3 (Acknowledged mode entity):

~~1.There are steps for multiple PUs in one PDU and should be deleted for Release 99.~~

~~2.At the receiving side, behaviours when receiving RESET PDU, RESET ACK PDU and the WINDOW, MRW and MRW_ACK SUFIs are not described.~~

~~3.~~1. The route from Extract Piggybacked Information to RLC Control Unit for piggybacked WINDOW, MRW and MRW_ACK SUFIs is missing in the model entity figure.

Summary of change: ⌘ ~~1.The description of Acknowledged mode entity is cleaned up.~~

~~2.~~1. The figure of AM entity model (Figure 4.4) is modified.

Consequences if not approved: ⌘ ~~The Specification is inconsistent.~~ The AM entity model is incomplete.

Clauses affected: ⌘ 4.2.1.3

Other specs affected: ⌘ ☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm.
Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

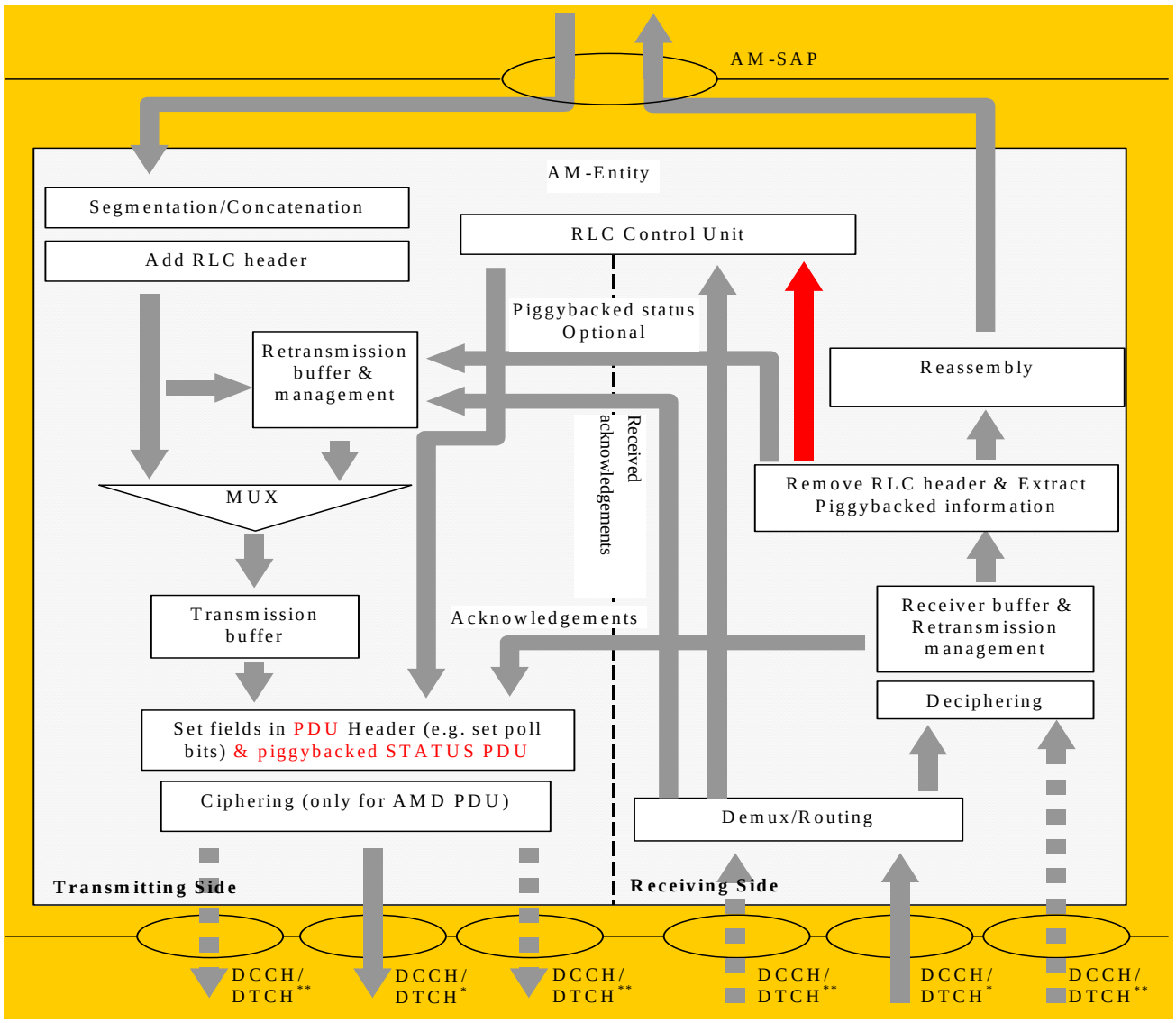
2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3)With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

4.2.1.3 Acknowledged mode entity

Figure 4.4 below shows the model of an acknowledged mode entity, when one logical channel (shown as a solid line) and when two logical channels (shown as dashed lines) are used.

In case two logical channels are used in the uplink the first logical channel shall be used for data PDUs and the second logical channel shall be used for control PDUs. In case one logical channel is used, the RLC PDU size shall be the same for AMD PDUs and control PDUs.


AM-SAP
AM-Entity
Segmentation/Concatenation
Add RLC header
RLC Control Unit
Piggybacked status Optional
Retransmission buffer & management
Reassembly
Received acknowledgements
MUX
Remove RLC header & Extract Piggybacked information
Transmission buffer
Acknowledgements
Receiver buffer & Retransmission management
Deciphering
Set fields in PDU Header (e.g. set poll bits) & piggybacked STATUS PDU
Ciphering (only for AMD PDU)
Demux/Routing
Transmitting Side
Receiving Side
DCCH/ DTCH**
DCCH/ DTCH*
DCCH/ DTCH**
DCCH/ DTCH**
DCCH/ DTCH*
DCCH/ DTCH**

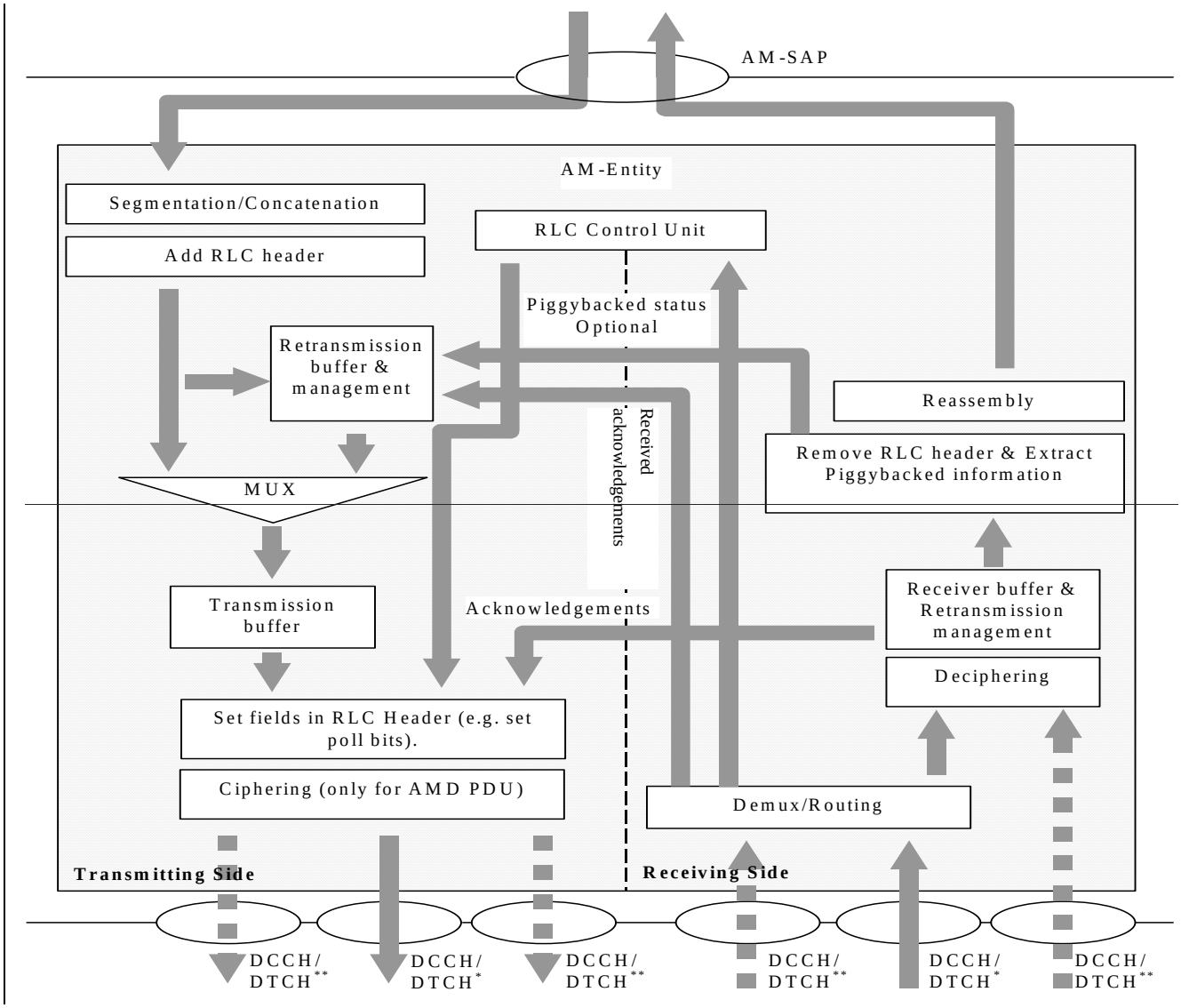


Figure 4.4: Model of an acknowledged mode entity

3GPP TSG-RAN2 Meeting #19 **R2-010683**
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322 CR 109** ⌘ rev **r2** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarification on UMD transfer procedure
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ February 22, 2001
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change: ⌘ 1. There are potential COUNT-C (HFN) synchronization problems for UMD transfer procedure.

~~2.In the case where a PDU contains a 15-bit LI indicating that an SDU ends with one octet left in the PDU, the last octet of this PDU shall be padded and shall not be concatenated with the next SDU. This should be described in the UMD and AMD transfer procedures.~~

~~3.~~2. The abnormal cases for UM transfer procedure are not complete.

Summary of change: ⌘ 1. ~~A discard LI is defined and used for UM SDU discard. The SN must be consecutively sent by the sender. The receiver decided the correctness of the PDUs by their receiving sequence or 'position' in each TTI. The UM receiving window size is equal to No_PDU, number of PDUs within a TTI.~~The procedure of timer based discard without explicit signalling for UM is clarified.

2. The UM and AM transfer procedures are cleaned up.

Consequences if not approved: ⌘ 1. COUNT-C might be unsynchronized between the sender and the receiver.

2. UM and AM transfer procedures are incomplete.

Clauses affected: ⌘ 9.6, 9.7.3.2, 11.2.2, 11.2.2.1, 11.2.4.1, 11.2.4.3, 11.3.2.1, 11.3.2.1.1, 11.3.2.1.2, 11.3.3, 11.3.4.2, 11.3.4.x(new)

Other specs affected: ⌘ ☐ Other core specifications ⌘
☐ Test specifications
☐ O&M Specifications

Other comments: ⌘	

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

9.6 Protocol Parameters

The values of the protocol parameters in this section are signalled by RRC.

a) MaxDAT.

It is the maximum value for the number of retransmissions of a PU. This parameter is an upper limit of counter VT(DAT). When the value of VT(DAT) comes to MaxDAT, error recovery procedure will be performed.

b) Poll_PU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_PU PU. This is an upper limit for the VT(PU) state variable, when VT(PU) reaches Poll_PU a poll is transmitted to the peer entity.

c) Poll_SDU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_SDU SDU. This is an upper limit for the VT(SDU) state variable, when VT(SDU) reaches Poll_SDU a poll is transmitted to the peer entity.

d) Poll_Window.

This parameter indicates when the transmitter should poll the receiver in case of performing window-based polling. The range of values of this parameter shall be $0 \leq$ Poll_Window ≤ 100. A poll is triggered for each PU when $J \geq$ Poll_Window, where J is the window transmission percentage defined by

$$J = \frac{(4096+VT(S) - VT(A)) \textbf{ mod } 4096}{VT(WS)} * 100 ,$$

where the constant 4096 is the modulus for AM described in Subclause 9.4.

e) MaxRST.

It is the maximum value for the number of retransmission of RESET PDU. This parameter is an upper limit of counter VT(RST). When the value of VT(RST) comes to MaxRST, the higher layer (RRC) is notified.

f) Configured_Tx_Window_Size.

The maximum allowed transmitter window size.

g) Configured_Rx_Window_Size.

The allowed receiver window size.

h) MaxMRW.

It is the maximum value for the number of retransmissions of a MRW command. This parameter is an upper limit of counter VT(MRW). When the value of VT(MRW) comes to MaxMRW, error recovery procedure will be performed.

9.7.3.1 Timer based discard, with explicit signalling

This alternative uses a timer based triggering of SDU discard (Timer_Discard). This makes the SDU discard function insensitive to variations in the channel rate and provides means for exact definition of maximum delay. However, the SDU loss rate of the connection is increased as SDUs are discarded.

For every SDU received from a higher layer, timer monitoring of the transmission time of the SDU is started. If the transmission time exceeds a predefined value for a SDU in acknowledged mode RLC, this SDU is discarded in the transmitter and a Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded.

The MRW command is defined as a super-field in the RLC STATUS PDU (see subclause 9.2), and piggybacked to status information of transmissions in the opposite direction. If the MRW command has not been acknowledged by receiver, it will be retransmitted. Therefore, SDU discard variants requiring peer-to-peer signalling are only possible for full duplex connections.

9.7.3.2 Timer based discard, without explicit signalling

This alternative uses the same timer based trigger for SDU discard (Timer_Discard) as the one described in the subclause 9.7.3.1. The difference is that this discard method does not use any peer-to-peer signalling. This function is applied only for unacknowledged and transparent mode RLC and peer-to-peer signalling is never needed. The SDUs are simply discarded in the transmitter, once the transmission time is exceeded. For UM RLC, how to update the sequence number is specified in subclause 11.2.4.3.

9.7.3.3 SDU discard after MaxDAT number of retransmissions

This alternative uses the number of retransmissions as a trigger for SDU discard, and is therefore only applicable for acknowledged mode RLC. This makes the SDU discard function dependent of the channel rate. Also, this variant of the SDU discard function strives to keep the SDU loss rate constant for the connection, on the cost of a variable delay. SDU discard is triggered at the transmitter, and a MRW command is necessary to convey the discard information to the receiver, like in the timer based discard with explicit signalling.

11.2 Unacknowledged mode data transfer procedure

11.2.1 Purpose

The unacknowledged mode data transfer procedure is used for transferring data between two RLC peer entities, which are operating in unacknowledged mode. Figure 11.2 below illustrates the elementary procedure for unacknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.2: Unacknowledged mode data transfer procedure

11.2.2 Initiation

The sender initiates this procedure upon a request of unacknowledged mode data transfer from higher layer.

When the sender is in data transfer ready state it shall segment and, if possible, concatenate the data received from the higher layer into PDUs.

Channels that can be used are DTCH, DCCH, CCCH (downlink only), CTCH, SHCCH (downlink only). The type of logical channel depends on if the RLC entity is located in the user plane (DTCH, CTCH) or in the control plane (DCCH/CCCH(downlink only)/SHCCH(downlink only)). One or several PDUs may be transmitted in each transmission time interval (TTI). For each TTI, MAC decides which PDU size shall be used and how many PDUs shall be transmitted.

The VT(US) state variable shall be updated for each UMD PDU that is transmitted.

11.2.2.1 UMD PDU contents to set

The Sequence Number field shall be set equal to VT(US).

The Extension bit shall be set to 1 if the next field is a length indicator field, otherwise it shall be set to zero.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI fields shall be set as specified in subclause 9.2.2.8.~~equal to the number of octets between the end of the header fields and the end of the segment. If padding is needed, another LI field set to only 1's shall be added unless the padding size is one octet for PDUs with 15-bit LIs. In the case where a PDU contains a 15-bit LI indicating that an SDU ends with one octet left in the PDU, the last octet of this PDU shall be padded with no additional LI in the next PDU and shall not be concatenated with the next SDU. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.~~

11.2.3 Reception of UMD PDU

Upon reception of a UMD PDU, the receiver shall update VR(US) state variable according to the received PDU(s).

The PDUs are reassembled into RLC SDUs. If a PDU with sequence number < VR(US) is missing then all SDUs that have segments in this PDU shall be discarded. RLC delivers the RLC SDUs to the higher layer through the UM-SAP.

11.2.4 Abnormal cases

11.2.4.1 Length Indicator value reserved for UMD PDU~~1111110~~

Upon reception of an UMD PDU that contains Length Indicator value reserved for UMD PDU, ~~1111110 or 11111111111110 ("piggybacked STATUS PDU", in case 7bit or 15 bit Length Indicator field is used, respectively)~~ the receiver shall discard that UMD PDU. ~~This Length Indicator value is not used in unacknowledged mode data transfer.~~

11.2.4.2 Invalid length indicator value

If the length indicator of a PDU has a value that is larger than the PDU size – the number of octets containing LIs in the PDU – 1 and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PDU shall be discarded and treated as a missing PDU.

11.2.4.3 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side, the sender shall discard the associated SDU~~-~~. For UM RLC, SN of the UMD PDUs shall be incremented by a step of 2 for the first PDU transmitted after a Discard Operation to indicate that there were some RLC SDUs discarded before this RLC PDU. The first data octet in this RLC PDU shall be the first octet of a RLC SDU. ~~The next UMD PDU shall carry the first segment of the oldest SDU not discarded. The state variable VT(US) shall be updated so that the receiver can detect at least one missing PDUs.~~ To avoid that the receiver should discard one extra SDU, a LI field shall be added in the first PDU transmitted after a Discard Operation. The value of the LI field shall be either the value indicating that the previous SDU filled exactly the previous RLC PDU or the value indicating that the first data octet in this RLC PDU is the first octet of a RLC SDU.

11.3 Acknowledged mode data transfer procedure

11.3.1 Purpose

The acknowledged mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in acknowledged mode. Figure 11.3 below illustrates the elementary procedure for acknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

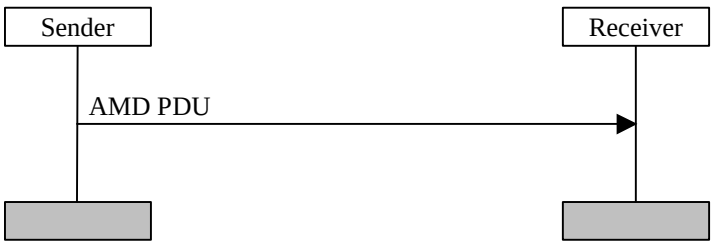


Figure 11.3: Acknowledged mode data transfer procedure

11.3.2 Initiation

The sender initiates this procedure upon a request of acknowledged mode data transfer from higher layer or upon retransmission of PUs. Retransmitted PUs have higher priority than PUs transmitted for the first time.

The sender is only allowed to retransmit PUs that have been indicated missing by the receiver. An exception is the PU with SN VT(S)-1 which can be retransmitted. In addition, a PU that has not yet been acknowledged, may be retransmitted if Configured_Tx_Window_Size is less than 2048.

RLC shall segment the data received from the higher layer into PUs. When the sender is in data transfer ready state one or several PUs are included in one AMD PDU, which is sent to the receiver. The PDUs shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane. One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can not be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

The VT(DAT) state variables shall be updated for each AMD PDU that is transmitted. The PDU shall not include any PU with Sequence Number ≥ VT(MS), except the PU with sequence number VT(S)-1 which may be included also when VT(S) ≥ VT(MS).

If the poll bit is set in any of the AMD PDUs and the timer Timer_Poll shall be used, the sender shall start the timer Timer_Poll when the successful or unsuccessful transmission of a PDU with the set poll bit is indicated by lower layer (in UE) or submitted to lower layer (in UTRAN).

If timer based SDU discard is used, the timer Timer_Discard shall be started when the RLC entity receives an SDU from higher layer. One timer is used for each SDU that is received from higher layer.

If the trigger for polling, "Every Poll_PU PU", is used, the VT(PU) shall be increased by 1 for each PU that is transmitted.

If the trigger for polling, "Every Poll_SDU SDU", is used, the VT(SDU) shall be increased by 1 for each SDU that is transmitted.

11.3.2.1 AMD PDU contents to set

If the PDU is transmitted for the first time, the Sequence Number field shall be set equal to VT(S) and VT(S) shall be updated.

The setting of the Polling bit is specified in subclause 11.3.2.1.1.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI fields shall be set as specified in subclause 9.2.2.8.~~equal to the number of octets between the end of the header fields and the end of the segment. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU. In the case where a PDU contains a 15-bit LI indicating that an SDU ends with one octet left in the PDU, the last octet of this PDU shall be padded with no additional LI in the next PDU and shall not be concatenated with the next SDU.~~

How to perform the segmentation and concatenation of ~~a~~ SDUs is specified in subclause 11.3.2.1.2.

11.3.2.1.1 Setting of the Polling bit

The Polling bit shall be set to 1 if any of following conditions are fulfilled except when the poll prohibit function is used and the timer Timer_Poll_Prohibit is active (the different triggers are described in 9.7.1):

1) Last PU in buffer is used and the last PU available for transmission is transmitted.

2) Last PU in retransmission buffer is used and the last PU to be retransmitted is transmitted.

3) Poll timer is used and timer Timer_Poll has expired.

4) Every Poll_PU PU is used and when VT(PU)=Poll_PU.

5) Every Poll_SDU is used and VT(SDU)=Poll_SDU and the PDU contains the last segment of that SDU.

6) Window based polling is used, , and J≥Poll_Window, where J is defined in subclause 9.6.

.

7) Timer based polling is used and Timer_Poll_Periodic has expired.

8) Poll prohibit shall be used, the timer Timer_Poll_Prohibit has expired and one or several polls were prohibited during the time Timer_Poll_Prohibit was active.

11.3.2.1.2 Segmentation and concatenation of ~~a~~ SDUs

Upon reception of a SDU, RLC shall segment the SDU to fit into the fixed size of a PU. The segments are inserted in the data field of a PU. A length indicator shall be added to each PU that includes a border of an SDU, i.e. if a PU does not contain an LI, the SDU continues in the next PU. The length indicator indicates where the border occurs in the PU.

The data after the indicated border can be either a new SDU, padding or piggybacked information. If padding or piggybacking is added another LI shall be added unless the padding size is one octet for PDUs with 15-bit LIs, see subclauses 9.2.2.8 and 9.2.2.9.

11.3.3 Reception of AMD PDU by the receiver

Upon reception of a AMD PDU, the receiver shall update VR(R), VR(H) and VR(MR) state variables according to the received PU(s).

If any of the PDUs includes a Polling bit set to 1, the STATUS PDU transfer procedure shall be initiated.

If the detection of missing PU(s) shall be used and the receiver detects that a PU is missing, the receiver shall initiate the STATUS PDU transfer procedure.

11.3.4 Abnormal cases

11.3.4.1 Timer_Poll timeout

Upon expiry of the Timer_Poll, the sender shall retransmit the poll. The poll can be retransmitted in either a new PDU or a retransmitted PDU.

11.3.4.2 Receiving a PDU outside the receiving window

Upon reception of a PDU with SN<VR(R) or SN≥VR(MR), the receiver shall discard the PDU. The poll bit shall be considered even if a complete PDU is discarded.

11.3.4.3 Timer_Discard timeout

11.3.4.3.1 SDU discard with explicit signalling

Upon expiry of Timer_Discard, the sender shall initiate the SDU discard with explicit signalling procedure.

11.3.4.4 VT(DAT) ≥ MaxDAT

If SDU discard after MaxDAT number of retransmission is used and VT(DAT) ≥ MaxDAT for any PU, the sender shall initiate the SDU discard with explicit signalling procedure for the SDUs to which the PU with VT(DAT) ≥MaxDAT belongs.

If the SDU discard is not used, the sender shall initiate the RLC reset procedure when VT(DAT) ≥ MaxDAT.

11.3.4.5 Invalid length indicator value

If the length indicator of a PU has a value that is larger than the PU size – the number of octets containing LIs in the PU and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PU shall be discarded and treated as a missing PU.

11.3.4.X Length Indicator value reserved for AMD PDU

Upon reception of an AMD PDU that contains Length Indicator value reserved for AMD PDU, the receiver shall discard that AMD PDU.

3GPP TSG-RAN WG2 Meeting #19 ***Tdoc R2-010617***
Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **110** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [ ] ME/UE [X] Radio Access Network [X] Core Network [ ]

Title:	⌘	RLC status transmission in CELL_PCH and URA_PCH
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 2001-02-23
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	
Summary of change:	⌘	The trigger "periodical status transmission" can be blocked by RRC. This is needed in CELL_PCH and URA_PCH to avoid that the UE enters CELL_FACH state at regular time intervals to transmit the status message Editorial changes in r1 after comments.
Consequences if not approved:	⌘	When periodical status transmission is configured, the UE will leave CELL_PCH and enter CELL_FACH at regular (short) time intervals in order to transmit the RLC status.

Clauses affected:	⌘	8.2, 9.5, 9.7.2
Other specs affected:	⌘	[X] Other core specifications ⌘ 25.331 [ ] Test specifications [ ] O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm.
Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded

from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the correct transmission of the RLC SDU.

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC If it indicates (re-)establishment, the state variables in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state. If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the PU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (i.e., unrecoverable errors such as data link layer loss or recoverable status events such as reset, etc.).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Activation Time (SN to activate a new ciphering configuration) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains PU size, Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), Periodical Status blocking configuration (see subclause 9.7.2~~9.5~~) ,SDU discard mode (see subclause 9.7.3) and Minimum WSN (see subclause 9.2.2.11.3). The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set.

8) The parameter DiscardInfo indicates the upper layer of each of the discarded RLC SDU. It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly or one octet short (only when 15 bit LI is used) in the end of the previous RLC PDU is present, the LI shall not be used.

9.5 Timers

a) Timer_Poll.

This timer is only used when the poll timer trigger is used. It is started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The timer is stopped when receiving a STATUS PDU that contains an acknowledgement of all AMD PDUs with SN up to and including VT(S)-1 at the time the poll was submitted to lower layer,or when a negative acknowledgement of the same PU is received. The value of the timer is signalled by RRC.

If the timer expires and no STATUS PDU fulfilling the criteria above has been received, the receiver is polled once more (either by the transmission of a PDU which was not yet sent, or by a retransmission) and the timer is restarted at the time specified above, with a new value of VT(S)-1.

If a new poll is sent when the timer is running the timer is restarted at the time specified above, with a new value of VT(S)-1.

b) Timer_Poll_Prohibit.

This timer is only used when the poll prohibit function is used. It is used to prohibit transmission of polls within a certain period. The timer shall be started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time a PDU containing a poll is submitted to lower layer until the timer has expired. A poll shall be delayed until the prohibit time expires if a poll is triggered during the prohibit time. Only one poll shall be transmitted when the prohibit time expires even if several polls were triggered during the prohibit time. This timer will not be stopped by a received STATUS PDU. The value of the timer is signalled by RRC.

c) Timer_EPC.

This timer is only used when the EPC function is used and it accounts for the roundtrip delay, i.e. the time when the first retransmitted PU should be received after a status report has been sent. The timer is started when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN) and when it expires VR(EP) can start its counting-down process (see subclause 9.7.4). The value of the timer is signalled by RRC.

d) Timer_Discard.

This timer is used for the SDU discard function. In the transmitter, the timer is activated upon reception of a SDU from higher layer. One timer is used for each SDU that is received from higher layer. If the SDU has not been acknowledged and/or transmitted when the timer expires, the SDU is discarded. Following which, if the SDU discard function uses explicit signalling, a Move Receiving Window request is sent to the receiver. The value of the timer is signalled by RRC.

e) Timer_Poll_Periodic.

This timer is only used when the timer based polling is used. The timer is started when the RLC entity is created. Each time the timer expires, the timer is restarted and a poll is triggered (either by the transmission of a PDU which was not yet sent, or by a retransmission). If there is no PU to be transmitted and all PUs have already been acknowledged, a poll shall not be triggered and the timer shall only be restarted. The value of the timer is signalled by RRC.

f) Timer_Status_Prohibit.

This timer is only used when the STATUS prohibit function is used. It prohibits the receiving side from sending status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. The timer is started when the successful or unsuccessful transmission of the last STATUS PDU in a status report is indicated by lower layer (in UE) or the last STATUS PDU in a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired and no new status report containing the mentioned SUFIs can be transmitted during the prohibit time.

The timer does not prohibit transmission of the SUFIs MRW, MRW_ACK, WINDOW or NO_MORE. The value of the timer is signalled by RRC.

g) Timer_Status_Periodic.

This timer is only used when timer based status report sending is used. The timer is started when the RLC entity is created. Each time the timer expires the transmission of a status report is triggered and the timer is restarted. The value of the timer is signalled by RRC. This timer can be blocked by RRC. In this case, the timer shall not be active. The timer shall be reset and restarted when it is unblocked by RRC.

h) Timer_RST.

This timer is used to detect the loss of RESET ACK PDU from the peer RLC entity. This timer is started when the successful or unsuccessful transmission of a RESET PDU is indicated by lower layer (in UE) or a RESET PDU is submitted to lower layer (in UTRAN). It will only be stopped upon reception of RESET ACK PDU, i.e. this timer is not stopped when an RLC reset occurs which was initiated from the peer RLC entity. If it expires, RESET PDU will be retransmitted. The value of the timer is signalled by RRC.

i) Timer_MRW.

This timer is used as part of the Move Receiving Window protocol. It is used to trigger the retransmission of a status report containing an MRW SUFI field. The timer is started when the successful or unsuccessful transmission of a STATUS PDU containing the MRW SUFI is indicated by lower layer (in UE) or a STATUS PDU containing the MRW SUFI is submitted to lower layer (in UTRAN). Each time the timer expires the MRW SUFI is retransmitted and the timer is restarted (at the time specified above). It shall be stopped when one of the termination criteria for the SDU discard is fulfilled. The value of the timer is signalled by RRC.

9.7.2 STATUS transmission for acknowledged mode

The receiver of AMD PDUs transmits status reports (each status report consists of one or several STATUS PDUs) to the sender in order to inform about which PUs that have been received and not received. There are several triggers for sending a status report. The network (RRC) controls which triggers should be used for each RLC entity, except for one, which is always present. The receiver shall always send a status report when receiving a poll request. Except for that trigger following triggers are configurable:

1) Detection of missing PU(s).

   If the receiver detects one or several missing PUs it shall trigger the transmission of a status report to the sender.

2) Timer based STATUS transfer.

   The receiver triggers the transmission of a status report periodically to the sender. The timer Timer_Status_Periodic controls the time period. When Periodical Status blocking is configured by higher layer, the trigger shall not be active.

3) The EPC mechanism.

   The timer Timer_EPC is started and the state variable VR(EP) is set when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN). If not all PUs requested for retransmission have been received before the variable VR(EP) has reached zero, a new status report is transmitted to the peer entity. A more detailed description of the EPC mechanism is given in subclause 9.7.4.

There are two functions that can prohibit the receiver from sending a status report. The network (RRC) controls which functions should be used for each RLC entity. If any of the following functions is used the sending of the status report shall be delayed, even if any of the triggering conditions above are fulfilled:

1) STATUS prohibit.

   The Timer_Status_Prohibit is started when the successful or unsuccessful transmission of the last STATUS PDU of a status report is indicated by lower layer (in UE) or the last STATUS PDU of a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired. The receiving side is not allowed to transmit a status report during the prohibit time. If a status report was triggered during the prohibit time, the status report is transmitted after the prohibit time has expired. The receiver shall only send one status report, even if there are several triggers during the prohibit time. This timer only prohibits the transmission of status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. Status reports containing other SUFIs are not prohibited.

2) The EPC mechanism.

   If the EPC mechanism is active and the sending of a status report is triggered it shall be delayed until the EPC mechanism has ended. The receiver shall only send one status report, even if there are several triggers when the timer is active or the counter is counting down. This mechanism only prohibits the transmission of status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. Status reports containing other SUFIs are not prohibited.

3GPP TSG-RAN WG2 Meeting #19
Sophia Antipolis, France, 19 - 23 February 2001

Tdoc R2-010376

CR-Form-v3

CHANGE REQUEST

⌘ **25.322 CR 111** ⌘ rev **-** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Re-establishment description
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 2001-02-14
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	The behaviour when RLC is reset by RRC is not described in the normative sections, only mentioned in the primitive description. The description of unrecoverable error is unclear and inconsistent.
Summary of change:	⌘	1. A description of RLC re-establishment (RLC reset requested by RRC) is added. The same behaviour as for RLC reset is described with the following exceptions: For RLC re-establishment, RLC RESET PDUs are not transmitted and the HFN numbers are configured by RRC after the re-establishment. 2. The description of unrecoverable error and error causes is clarified.
Consequences if not approved:	⌘	Unclear behaviour. Inconsistency between specs.

Clauses affected:	⌘	8.2, 9.6, 10, 11.4.5.2, 11.4.5.3, 11.x (new)		
Other specs affected:	⌘	**X** Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	25.331
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be

downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

8 Elements for layer-to-layer communication

The interaction between the RLC layer and other layers are described in terms of primitives where the primitives represent the logical exchange of information and control between the RLC layer and other layers. The primitives shall not specify or constrain implementations.

8.1 Primitives between RLC and higher layers

The primitives between RLC and upper layers are shown in Table 8.1.

Table 8.1: Primitives between RLC and upper layers

Generic Name	Parameter			
	Req.	**Ind.**	**Resp.**	**Conf.**
RLC-AM-DATA	Data, CNF, MUI	Data, DiscardInfo	Not Defined	MUI
RLC-UM-DATA	Data, Use special LI	Data	Not Defined	Not Defined
RLC-TR-DATA	Data	Data	Not Defined	Not Defined
CRLC-CONFIG	E/R, Stop,Continue, Ciphering Elements (UM/AM only), AM_parameters (AM only)	Not Defined	Not Defined	Not Defined
CRLC-SUSPEND (UM/AM only)	N	Not Defined	Not Defined	VT(US) (UM only), VT(S) (AM only)
CRLC-RESUME (UM/AM only)	No Parameter	Not Defined	Not Defined	Not Defined
CRLC-STATUS	Not Defined	EVC	Not Defined	Not Defined

Each Primitive is defined as follows:

RLC-AM-DATA-Req/Ind/Conf

- RLC-AM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in acknowledged mode.
- RLC-AM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in acknowledged mode and to indicate higher layers of the discarded RLC SDU in the receiving RLC.
- RLC-AM-DATA-Conf is used by RLC to confirm to higher layers the transmission of a RLC SDU.

RLC-UM-DATA-Req/Ind

- RLC-UM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in unacknowledged mode.
- RLC-UM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in unacknowledged mode.

RLC-TR-DATA-Req/Ind

- RLC-TR-DATA-Req is used by higher layers to request transmission of a higher layer PDU in transparent mode.
- RLC-TR-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in transparent mode.

CRLC-CONFIG-Req

This primitive is used by RRC to establish, re-establish, release, stop, continue or reconfigure the RLC. Ciphering elements are included for UM and AM operation.

CRLC-SUSPEND-Req/Conf

This primitive is used by RRC to suspend the RLC. The N parameter indicates that RLC shall not send a PDU with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM, where N is an integer. RLC informs RRC of the VT(S) for AM and VT(US) for UM in the confirm primitive.

CRLC-RESUME-Req

This primitive is used by RRC to resume RLC when RLC has been suspended.

CRLC-STATUS-Ind

It is used by the RLC to send status information to RRC.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the correct transmission of the RLC SDU.

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC If it indicates (re-)establishment, the state variables and configurable parameters are initialised according to subclause 11.8~~in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state.~~ If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the PU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (i.e., unrecoverable errors such as data link layer loss or recoverable status events such as reset, etc.).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Activation Time (SN to activate a new ciphering configuration) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains PU size, Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), SDU discard mode (see subclause 9.7.3) and Minimum WSN (see subclause 9.2.2.11.3). The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set.

8) The parameter DiscardInfo indicates the upper layer of each of the discarded RLC SDU. It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly or one octet short (only when 15 bit LI is used) in the end of the previous RLC PDU is present, the LI shall not be used.

9.6 Protocol Parameters

The values of the protocol parameters in this section are signalled by RRC.

a) MaxDAT.

It is the maximum value for the number of retransmissions of a PU. This parameter is an upper limit of counter VT(DAT). When the value of VT(DAT) comes to MaxDAT, either RLC RESET procedure or SDU discard procedure shall be initiated according to configuration by higher layer. ~~error recovery procedure will be performed.~~

b) Poll_PU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_PU PU. This is an upper limit for the VT(PU) state variable, when VT(PU) reaches Poll_PU a poll is transmitted to the peer entity.

c) Poll_SDU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_SDU SDU. This is an upper limit for the VT(SDU) state variable, when VT(SDU) reaches Poll_SDU a poll is transmitted to the peer entity.

d) Poll_Window.

This parameter indicates when the transmitter should poll the receiver in case of performing window-based polling. The range of values of this parameter shall be $0 \leq$ Poll_Window ≤ 100. A poll is triggered for each PU when :$J \geq$ Poll_Window, where J is the window transmission percentage defined by

$$J = \frac{(4096+VT(S) - VT(A)) \textbf{ mod } 4096}{VT(WS)} * 100 ,$$

where the constant 4096 is the modulus for AM described in Subclause 9.4.

.

e) MaxRST.

It is the maximum value for the number of retransmission of RESET PDU. This parameter is an upper limit of counter VT(RST). When the value of VT(RST) comes to MaxRST, unrecoverable error shall be indicated to higher layer. ~~the higher layer (RRC) is notified.~~

f) Configured_Tx_Window_Size.

The maximum allowed transmitter window size.

g) Configured_Rx_Window_Size.

The allowed receiver window size.

h) MaxMRW.

It is the maximum value for the number of retransmissions of a MRW command. This parameter is an upper limit of counter VT(MRW). When the value of VT(MRW) comes to MaxMRW, ~~error recovery procedure will be performed.~~RLC RESET procedure shall be initiated.

10 Handling of unknown, unforeseen and erroneous protocol data

In case of error situations the following actions are foreseen:

1) RLC entity shall initiate RESET procedure in case of a protocol error.

2) RLC entity shall discard invalid PDUs.

3) RLC entity shall notify upper layer of unrecoverable error occurrence (see subclause 11.4.5.2).

The list of protocol error cases is reported below:

~~a)~~ Inconsistent state variables~~:~~.

If the RLC entity receives a PDU including "erroneous Sequence Number", state variables between peer entities may be inconsistent. Following shows "erroneous Sequence Number" examples:

- Each Sequence Number of missing PU informed by SUFI LIST, BITMAP or RLIST is not within the value between "Acknowledge state variable(VT(A))" and "Send state variable(VT(S)) – 1", and

- LSN of SUFI ACK is not within the value between "Acknowledge state variable(VT(A))" and "Send state variable(VT(S))".

~~In case of error situations the following actions are foreseen:~~

~~1) RLC entity should use RESET procedure in case of an unrecoverable error.~~

~~2) RLC entity should discard invalid PDU.~~

~~3) RLC entity should notify upper layer of unrecoverable error occurrence in case of failed retransmission.~~

~~b)~~ Inconsistent status indication of a PU;

- If a received STATUS PDU indicates different status for the same PU, then the transmitter shall discard the STATUS PDU.

Invalid PDU format;

- If the RLC PDU format contains reserved or invalid values, the RLC PDU shall be discarded.

11.4 RLC reset procedure

11.4.1 Purpose

The RLC reset procedure is used to reset two RLC peer entities, which are operating in acknowledged mode. Figure 11.4 below illustrates the elementary procedure for a RLC reset. The sender can be either the UE or the network and the receiver is either the network or the UE. During the reset procedure the hyper frame numbers (HFN) in UTRAN and UE are synchronised. Two HFNs used for ciphering needs to be synchronised, DL HFN in downlink and UL HFN in uplink. In the reset procedure, the highest UL HFN and DL HFN used by the RLC entity are exchanged between UE and UTRAN. After the reset procedure is terminated, the UL HFN and DL HFN shall be increased with one in both UE and UTRAN, and the updated HFN values shall be used after the reset procedure.

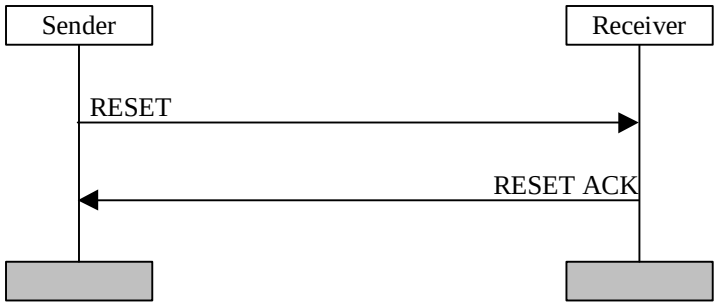


Figure 11.4: RLC reset procedure

11.4.2 Initiation

The procedure shall be initiated when a protocol error occurs.

The sender sends the RESET PDU when it is in data transfer ready state and enters reset pending state. The sender shall start the timer Timer_RST and increase VT(RST) with 1. The RESET PDU shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

The RESET PDU has higher priority than data PDUs.

When a reset procedure has been initiated it can only be ended upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU, i.e., a reset procedure is not interrupted by the reception of a RESET PDU from the peer entity.

11.4.2.1 RESET PDU contents to set

The size of the RESET PDU shall be equal to one of the allowed PDU sizes. The hyper frame number indicator field (HFNI) shall be set equal to the currently used HFN (DL HFN when the RESET is sent by UTRAN or UL HFN when the RESET is sent by the UE). The RSN field shall indicate the sequence number of the RESET PDU. This sequence number is incremented every time a new RESET PDU is transmitted, but not when a RESET PDU is retransmitted.

11.4.3 Reception of the RESET PDU by the receiver

Upon reception of a RESET PDU the receiver shall respond with a RESET ACK PDU. The receiver resets the state variables in 9.4 to their initial value and resets configurable parameters to their configured value. Both the transmitter and receiver side of the AM RLC entity are reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded.

When a RESET PDU is received, the receiver shall set the HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) equal to the HFNI field in the received RESET PDU.

The RESET ACK PDU shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

The RESET ACK PDU has higher priority than data PDUs.

11.4.3.1 RESET ACK PDU contents to set

The size of the RESET ACK PDU shall be equal to one of the allowed PDU sizes. The RSN field shall always be set to the same value as in the corresponding RESET PDU. The hyper frame number indicator field (HFNI) shall be set equal to the currently used HFN (DL HFN when the RESET ACK is sent by UTRAN or UL HFN when the RESET ACK is sent by the UE).

11.4.4 Reception of the RESET ACK PDU by the sender

When the sender is in reset pending state and receives a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU the Timer_RST shall be stopped and the value of the HFN (DL HFN when the RESET ACK is received in UE or UL HFN when the RESET ACK is received in UTRAN) shall be set equal to the HFNI field in the received RESET ACK PDU. The sender resets the state variables in 9.4 to their initial value and resets configurable parameters to their configured value. Both the transmitter and receiver side of the AM RLC entity is reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded.

The sender shall enter data transfer ready state.

Upon reception of a RESET ACK PDU with a different RSN value as in the corresponding RESET PDU the RESET ACK PDU is discarded.

Upon reception of a RESET ACK PDU in data transfer ready state the RESET ACK PDU is discarded.

11.4.5 Abnormal cases

11.4.5.1 Timer_RST timeout

Upon expiry of Timer_RST the sender shall retransmit the RESET PDU and increase VT(RST) with 1. In the retransmitted RESET PDU the value of the RSN field shall not be incremented.

11.4.5.2 Unrecoverable error (VT(RST) ≥ MaxRST)

If VT(RST) becomes larger or equal to MaxRST, unrecoverable error shall be indicated to higher layer. ~~the RRC layer shall be informed.~~

11.4.5.3 Reception of the RESET PDU by the sender

Upon reception of a RESET PDU in reset pending state, the sender shall respond with a RESET ACK PDU. The sender resets the state variables in 9.4 to their initial value, resets configurable parameters to their configured value. However, VT(RST) and Timer_RST are not reset. Both the transmitter and receiver side of the AM RLC entity are reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded. The hyper frame number, HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) is set equal to the HFNI field in the received RESET PDU. The sender shall stay in the reset pending state. The sender shall enter data transfer ready state only upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU.

11.x RLC re-establishment procedure

The RLC re-establishment procedure is used when higher layer request the RLC entity to be re-established.

When an RLC entity is re-established, the state variables in the RLC entity (see 9.4) shall be reset to their initial value and the configurable parameters shall be set to their configured value. All RLC PDUs in the RLC receiver and transmitter shall be discarded. The hyper frame number (HFN) in UL and DL shall be set to the value configured by higher layer. After the re-establishment, RLC shall enter the data transfer ready state.

3GPP TSG-RAN WG2 Meeting #19 ***Tdoc R2-010619***

Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **112** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarifications on the RESET and RESET ACK PDU sizes
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 2001-02-13
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
- ***F*** *(essential correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(Addition of feature),*
- ***C*** *(Functional modification of feature)*
- ***D*** *(Editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	No description of the sizes allowed for the RESET and RESET ACK PDUs is given. In case of a RLC AM entity with 2 logical channels, the PDUs sizes that can be used for the control PDUs can be different that the those used in the entire RLC entity.
Summary of change:	⌘	1. Addition of a description of the sizes allowed for the RESET and RESET ACK PDUs 2. Clarification in 9.2.1.5: PDU sizes shall correspond to what is used by the logical channel on which the control PDUs are sent.
Consequences if not approved:	⌘	Unclear specification.

Clauses affected:	⌘	9.2.1.5, 9.2.1.7		
Other specs affected:	⌘	☐ Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

9.2.1.5 STATUS PDU

The STATUS PDU is used to report the status between two RLC AM entities. Both receiver and transmitter status information may be included in the same STATUS PDU.

The format of the STATUS PDU is given in Figure 9.4 below. The Figure shows an example and the length of each SUFI is dependent on the SUFI type.

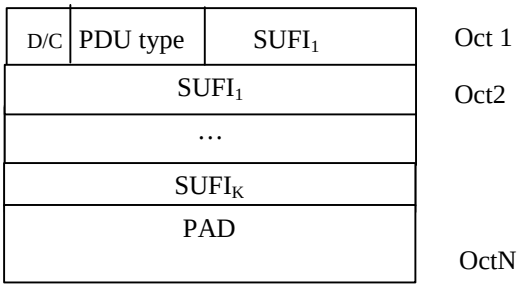


Figure 9.4: Status Information Control PDU (STATUS PDU)

Up to K super-fields ($SUFI_1$-$SUFI_K$) can be included into one STATUS PDU, in which each super-field can be of different type. The size of a STATUS PDU is variable and upper bounded by the maximum RLC PDU size used by the logical channel on which the control PDUs are sent~~an RLC entity~~. Padding shall be included to exactly fit one of the PDU sizes used by the logical channel on which the control PDUs are sent.~~the entity~~. The length of the STATUS PDU shall be an integer number of octets.

9.2.1.7 RESET, RESET ACK PDU

The RESET PDU and RESET ACK PDU has a one-bit sequence number field (RSN). With the aid of this field the Receiver can define whether the received RESET PDU is transmitted by the Sender for the first time or whether it is a retransmission of a previous RESET PDU.

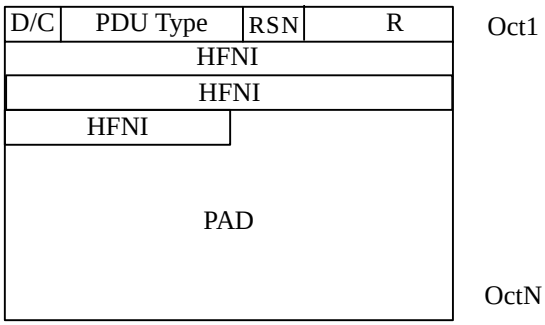


Figure 9.6: RESET, RESET ACK PDU

The size of a RESET or RESET ACK PDU is variable and upper bounded by the maximum RLC PDU size used by the logical channel on which the control PDUs are sent~~an RLC entity~~. Padding shall be included to exactly fit one of the PDU sizes used by the logical channel on which the control PDUs are sent. The length of the RESET or RESET ACK PDU shall be an integer number of octets.

3GPP TSG-RAN WG2 Meeting #19 ***Tdoc R2-010620***

Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **113** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Editorial corrections and clarifications
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 2001-02-13
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
- ***F*** *(essential correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(Addition of feature),*
- ***C*** *(Functional modification of feature)*
- ***D*** *(Editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change: ⌘ Editorial corrections and clarifications have been made.

Summary of change: ⌘
1. Various editorial corrections
2. 'Release 99' replaced by 'current release' to avoid future CRs (when a new release of the protocol is defined).
3. 'MAC' replaced by 'lower layer' in some places
4. Distinction between the 2 Reserved fields.

Consequences if not approved: ⌘ CR will have to be generated to update Release 99.

Unclarity of the specification.

Clauses affected: ⌘ 1, 4.2.1.1, 4.2.1.2, 4.2.1.3, 5, 9.1.2, 9.2.1.6, 9.2.1.7, 9.2.2.4, 9.2.2.6, 9.2.2.8, 9.2.2.11.6, 9.2.2.11.7, 9.2.2.11.8, 9.2.2.12, 9.3.1.1, 9.3.1.2, 9.3.2.1, 9.3.2.2, 9.3.2.3, 9.3.3.1, 9.3.3.3, 9.3.3.4, 9.7.3, 9.7.6, 9.7.7, 11.4.2.1, 11.5.2.2, 11.6.2, 11.6.4

Other specs affected: ⌘
- ☐ Other core specifications ⌘
- ☐ Test specifications
- ☐ O&M Specifications

Other comments: ⌘

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

1 Scope

The present document specifies the RLC protocol.

~~Release '99 features~~Features for the current Release:

- Transparent mode.
- Unacknowledged mode.
- Acknowledged mode.

~~Features for future Releases:~~

- ~~Hybrid ARQ.~~

4.2.1.1 Transparent mode entities

Figure 4.2 below shows the model of two transparent mode peer entities.

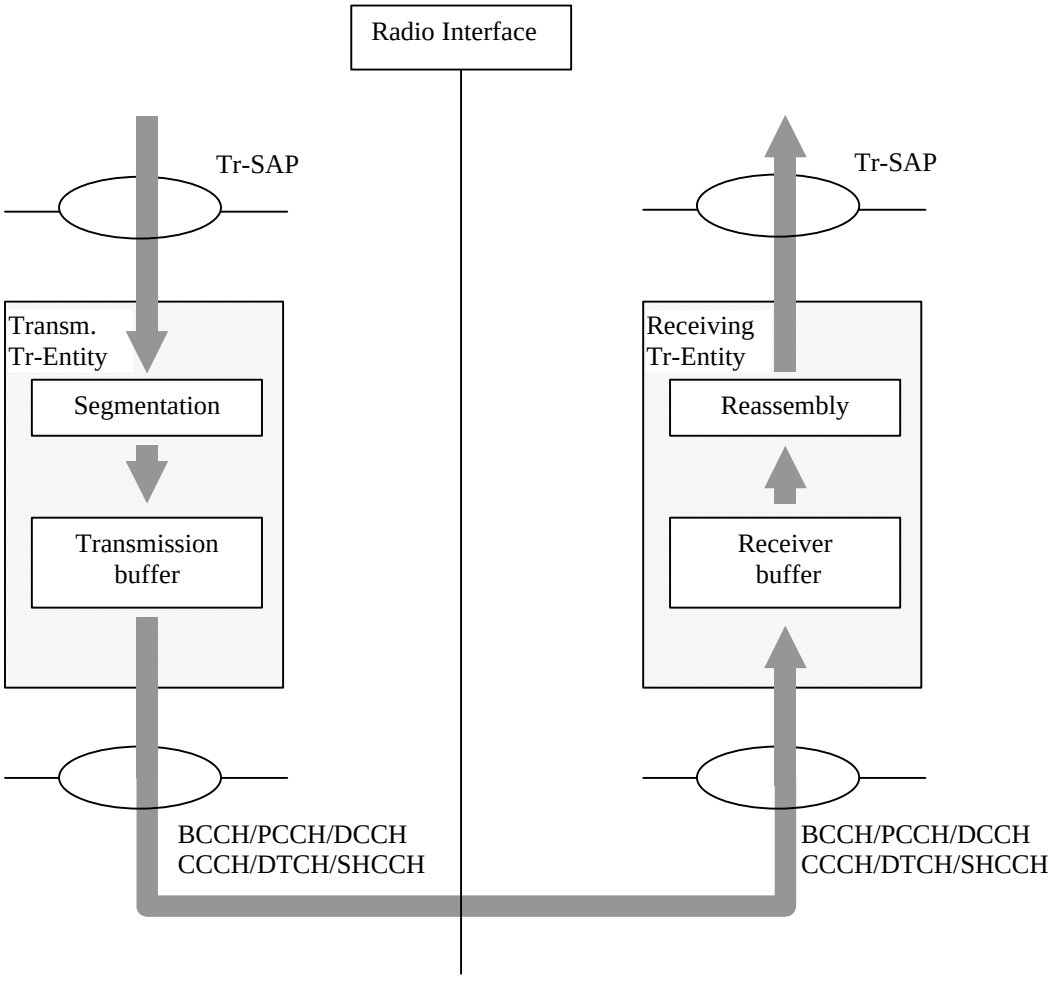


Figure 4.2: Model of two transparent mode peer entities

The transmitting Tr-entity receives SDUs from the higher layers through the Tr-SAP. RLC might segment the SDUs into appropriate RLC PDUs without adding any overhead. How to perform the segmentation is decided upon when the

service is established. RLC delivers the RLC PDUs to ~~MAC~~ lower layer through either a BCCH, DCCH, PCCH, CCCH, SHCCH or a DTCH. The CCCH and SHCCH ~~also~~ uses transparent mode~~, but~~ only for the uplink. Which type of logical channel depends on if the higher layer is located in the control plane (BCCH, DCCH, PCCH, CCCH, SHCCH) or user plane (DTCH).

The receiving Tr-entity receives PDUs through one of the logical channels from lower layer ~~the MAC sublayer~~. RLC reassembles (if segmentation has been performed) the PDUs into RLC SDUs. How to perform the reassembling is decided upon when the service is established. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

4.2.1.2 Unacknowledged mode entities

Figure 4.3 below shows the model of two unacknowledged mode peer entities.

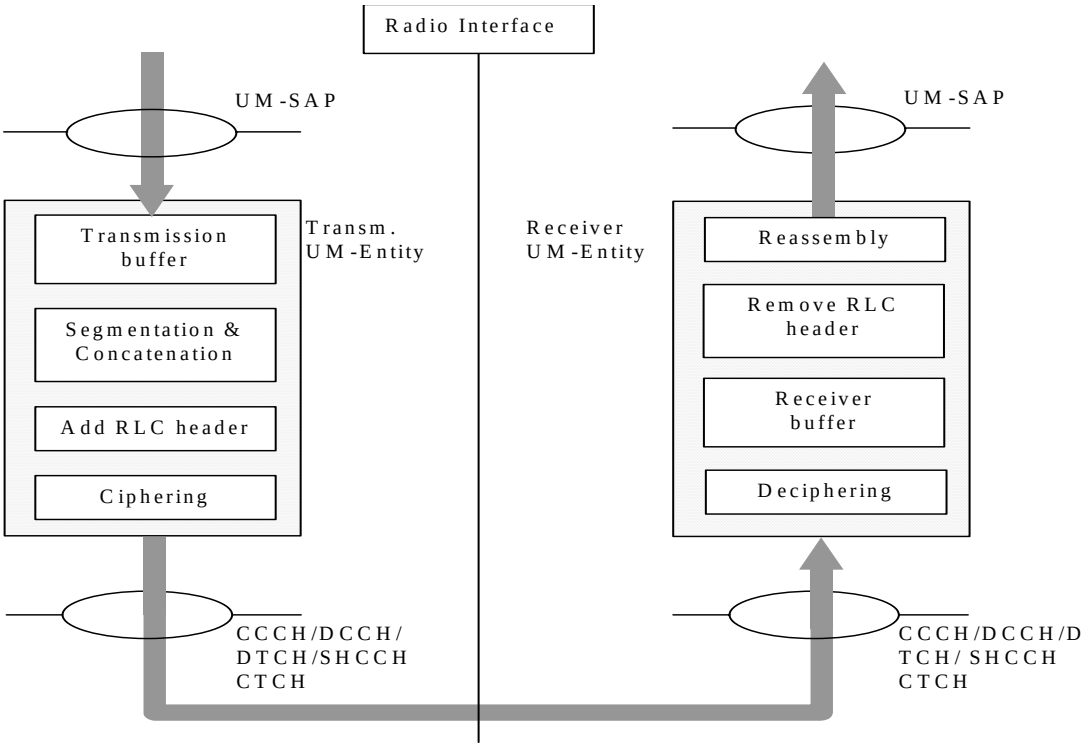


Figure 4.3: Model of two unacknowledged mode peer entities

The transmitting UM-entity receives SDUs from the higher layers. RLC might segment the SDUs into RLC PDUs of appropriate size. The SDU might also be concatenated with other SDUs.~~.~~ RLC delivers the RLC PDUs to lower layer ~~MAC~~ through either a CCCH, SHCCH, DCCH, CTCH or a DTCH. The CCCH and SHCCH ~~also~~ uses unacknowledged mode~~, but~~ only for the downlink. Which type of logical channel depends on if the higher layer is located in the control plane (CCCH, DCCH, SHCCH) or user plane (CTCH, DTCH).

The receiving UM-entity receives PDUs through one of the logical channels from the MAC sublayer. RLC removes header from the PDUs and reassembles the PDUs (if segmentation has been performed) into RLC SDUs. The RLC SDUs are delivered to the higher layer.

4.2.1.3 Acknowledged mode entity

Figure 4.4 below shows the model of an acknowledged mode entity, when one logical channel (shown as a solid line) and when two logical channels (shown as dashed lines) are used.

In case two logical channels are used in the uplink the first logical channel shall be used for data PDUs and the second logical channel shall be used for control PDUs. In case one logical channel is used, the RLC PDU size shall be the same for AMD PDUs and control PDUs.

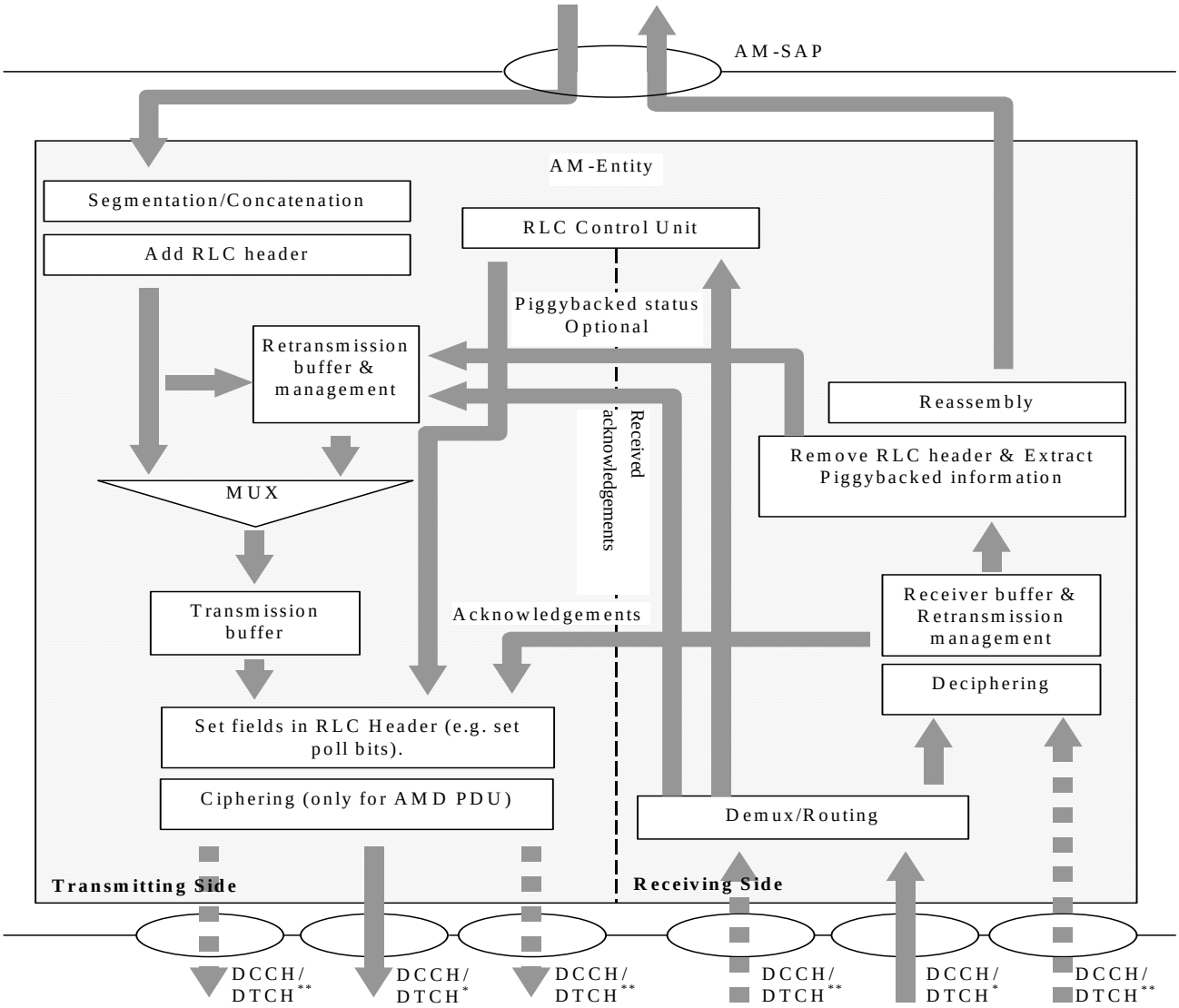


Figure 4.4: Model of an acknowledged mode entity

The transmitting side of the AM-entity receives SDUs from the higher layers. The SDUs are segmented and/or concatenated to PUs of fixed length. PU length is a semi-static value that is decided in bearer setup and can only be changed through bearer reconfiguration by RRC.

For purposes of RLC buffering and retransmission handling, the operation is the same as if there would be one PU per PDU. For concatenation or padding purposes, bits of information on the length and extension are inserted into the beginning of the last PU where data from an SDU is included. Padding can be replaced by piggybacked status information. This includes setting the poll bit.

If several SDUs fit into one PU, they are concatenated and the appropriate length indicators are inserted into the beginning of the PU. After that the PUs are placed in the retransmission buffer and the transmission buffer. One PU is included in one RLC PDU.

The MUX then decides which PDUs and when the PDUs are submitted to lower layer. The PDUs are submitted via a function that completes the RLC-PDU header and potentially replaces padding with piggybacked status information. The RLC entity shall assume a PDU to be transmitted when the PDU is submitted to lower layer.

The ciphering is applied only for AMD PDUs. The fixed 2 ~~octet~~octets AMD PDU header is not ciphered. Piggybacked STATUS PDU and Padding parts of AMD PDU when existing are ciphered. The other Control PDUs (~~e.g.~~i.e. STATUS PDU, RESET PDU, and RESET ACK PDU) shall not be ciphered.

When Piggybacking mechanism is applied the padding is replaced by control information, in order to increase the transmission efficiency and making possible a faster message exchange between the peer to peer RLC entities. The piggybacked control information is not saved in any retransmission buffer. The piggybacked control information is contained in the piggybacked STATUS PDU, which is in turn included into the AMD-PDU. The piggybacked STATUS PDUs will be of variable size in order to match with the amount of free space in the AMD PDU.

The retransmission buffer also receives acknowledgements from the receiving side, which are used to indicate retransmissions of PUs and when to delete a PU from the retransmission buffer.

The ~~R~~receiving ~~S~~side of the AM-entity receives PDUs through one of the logical channels from lower layer ~~the MAC sublayer~~. The ~~RLC~~ AMD PDUs are expanded into separate PUs and potential piggybacked status information are extracted. The PUs are placed in the receiver buffer until a complete SDU has been received. The receiver buffer requests retransmissions of PUs by sending negative acknowledgements to the peer entity. After that the headers are removed from the PDUs and the PDUs are reassembled into an SDU. Finally the SDU is delivered to the higher layer. The receiving side also receives acknowledgements from the peer entity. The acknowledgements are passed to the retransmission buffer on the transmitting side.

5 Functions

The following functions are supported by RLC. For a detailed description of the following functions see [3]:

- Segmentation and reassembly.
- Concatenation.
- Padding.
- Transfer of user data.
- Error correction.
- In-sequence delivery of higher layer PDUs.
- Duplicate d~~D~~etection.
- Flow control.
- Sequence number check ~~(Unacknowledged data transfer mode)~~.
- Protocol error detection and recovery.
- Ciphering.
- Suspend/resume function.

9.1.2 Control PDUs

a) STATUS PDU and Piggybacked STATUS PDU

The STATUS PDU and the Piggybacked STATUS PDU are used in acknowledged mode:

- by the receiving entity to inform the transmitting entity about missing PUs at the receiving entity;
- by the receiving entity to inform the transmitting entity about the size of the allowed transmission window;
- and by the transmitting entity to request the receiving entity to move the receiving window.

b) RESET PDU

The RESET PDU is used in acknowledged mode to reset all protocol states, protocol variables and protocol timers of the peer RLC entity in order to synchronise the two peer entities.

c) RESET ACK PDU

The RESET ACK PDU is an acknowledgement to the RESET PDU.

Table 9.1: RLC PDU names and descriptions

Data Transfer Mode	PDU name	Description
Transparent	TrD	Transparent mode data
Unacknowledged	UMD	Sequenced unacknowledged mode data
Acknowledged	AMD	Sequenced acknowledged mode data
	STATUS	Solicited or Unsolicited Status Report
	Piggybacked STATUS	Piggybacked Solicited or Unsolicited Status Report
	RESET	Reset Command
	RESET ACK	Reset Acknowledgement

9.2.1.6 Piggybacked STATUS PDU

The format of the piggybacked STATUS PDU is the same as the ordinary Control PDU except that the D/C field is replaced by a reserved bit (R2). This PDU can be used to piggyback STATUS PDU in an AMD PDU if the data does not fill the complete AMD PDU. The PDU Type field is set to zero and all other values are invalid for this version of the protocol and the PDU is discarded.

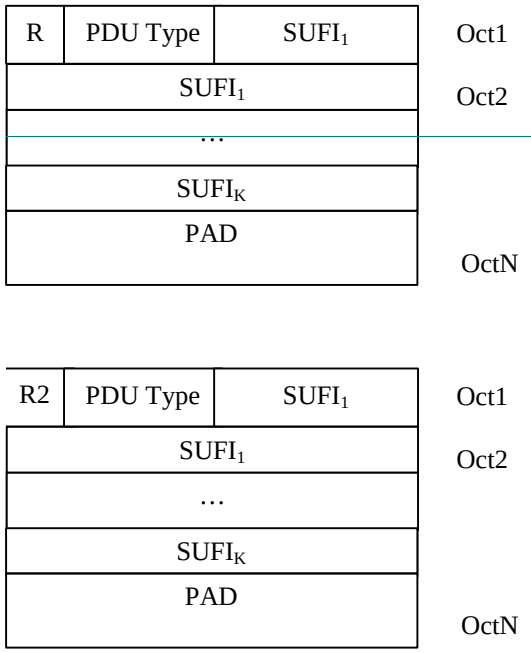


Figure 9.5: Piggybacked STATUS PDU

9.2.1.7 RESET, RESET ACK PDU

The RESET PDU and RESET ACK PDU have~~has~~ a one-bit sequence number field (RSN). With the aid of this field the Receiver can define whether the received RESET PDU is transmitted by the Sender for the first time or whether it is a retransmission of a previous RESET PDU.

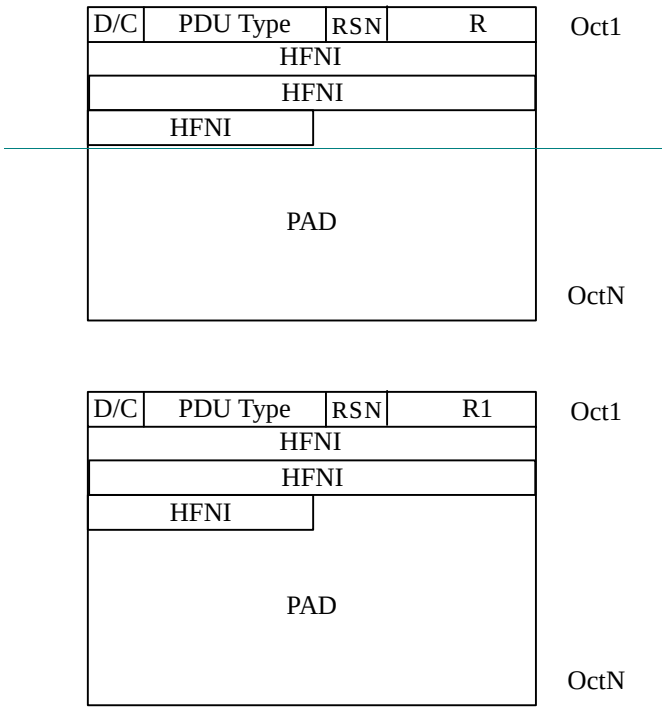


Figure 9.6: RESET, RESET ACK PDU

9.2.2.4 Polling bit (P)

Length: 1bit.

This field is used to request a status report (one or several STATUS PDUs) from the receiver RLC.

Bit	Description
0	–Status report not requested
1	Request a status report

9.2.2.6 Reserved 1 (R1)

Length: 3 bits.

This field in the RESET PDU and RESET ACK PDU is used to achieve octet alignment and for this purpose it is coded as 000. Other functions of it are left for future releases.

9.2.2.8 Length Indicator (LI)

The Length Indicator is used to indicate, each time, the end of an SDU occurs in the PU. The Length Indicator points out the number of octets between the end of the last Length Indicator field and up to and including the octet at the end of an SDU segment. Length Indicators are included in the PUs that they refer to. The size of the Length Indicator may be either 7bits or 15bits. The maximum value of a Length Indicator in AM will be no greater than the RLC PDU size – AMD PDU Header – PADDING. The maximum value of a Length Indicator in UM will be no greater than the RLC PDU size – UMD PDU Header – PADDING.

A Length Indicator group is a set of Length Indicators that refer to a PU. Length Indicators that are part of a Length Indicator group must never be reordered within the Length Indicator group or removed from the Length Indicator group.

If there can be more than one Length Indicator, each specifying the end of an SDU in a PU, the order of these Length Indicators must be in the same order as the SDUs that they refer to.

In the case where the end of last segment of an SDU exactly ends at the end of a PDU and there is no LI that indicates the end of the SDU, the next Length Indicator, shall be placed as the first Length Indicator in the following PU and have value LI=0.

In the case where a PDU contains a 15-bit LI indicating that an SDU ends with one octet left in the PDU, the last octet of this PDU shall be ignored and shall not be filled with the first octet of the next SDU data.

In the case where the last segment of an RLC SDU is one octet short of exactly filling the previous RLC PU, and 15-bit Length Indicators are used, the Length Indicator shall be placed as the first Length Indicator in the following PU and have value LI=111 1111 1111 1011. The remaining one octet in the previous RLC PU shall be ignored.

A PU that has unused space, to be referred to as padding, shall use a Length Indicator to indicate that this space is used as padding unless the padding size is one octet for PDUs with 15-bit LIs. A padding Length Indicator must be placed after any Length Indicators for a PU.

All unused space in a PU must be located at the end of the PDU, be a homogeneous space and is referred to as padding. Predefined values of the Length Indicator are used to indicate this. The values that are reserved for special purposes are listed in the tables below depending on the size of the Length Indicator. Only predefined Length Indicator values can refer to the padding space.

STATUS PDUs can be piggybacked on the AMD PDU by using part or all of the padding space. A Length Indicator must be used to indicate the piggybacked STATUS PDU. This Length Indicator takes space from the padding space or piggybacked STATUS PDU and not the PDU data and will always be the last Length Indicator. Where only part of the padding space is used by a piggybacked STATUS PDU then the end of the piggybacked STATUS PDU is determined by one of the SUFI fields NO_MORE or ACK, thus no additional Length Indicator is required to show that there is still padding in the PDU. The padding/piggybacked STATUS PDU predefined Length Indicators shall be added after the very last (i.e. there could be more than one SDU that end within a PDU) Length Indicator that indicates the end of the last SDU segment in the PU.

If SDU discard with explicit signalling is used an AMD PDU can contain a maximum number of 15 LIs indicating the end of an SDU and the rest of the AMD PDU space shall be used as padding/piggybacked STATUS PDU.

For AM, 7bit indicators shall be used if the AMD PDU size is ≤ l26 octets. Otherwise 15bit indicators shall be used. For UM, 7bit indicators shall be used if the UMD PDU size is ≤ l25 octets. Otherwise 15bit indicators shall be used.

The length of the Length Indicator only depends on the size of the largest RLC PDU. The length of the Length Indicator is always the same for all UMD PDUs or AMD PUs, for one RLC entity.

If the maximum RLC PDU size for an RLC entity is not explicitly configured (e.g. on FACH), the length of the Length Indicator is determined by the maximum configured TB size for the transport channel on which the logical channel is mapped.

For ~~Release 99~~the current release, there is one PU in an AMD PDU.

Length: 7bits

Bit	Description
0000000	The previous RLC PDU was exactly filled with the last segment of a RLC SDU and there is no LI that indicates the end of the SDU in the previous RLC PDU.
1111100	UMD PDU: The first data octet in this RLC PDU is the first octet of a RLC SDU. AMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111101	Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111110	AMD PDU: The rest of the RLC PDU includes a piggybacked STATUS PDU. UMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111111	The rest of the RLC PDU is padding. The padding length can be zero.

Length: 15bits

Bit	Description
000000000000000	The previous RLC PDU was exactly filled with the last segment of an RLC SDU and there is no LI that indicates the end of the SDU in the previous RLC PDU.
111111111111011	The last segment of an RLC SDU was one octet short of exactly filling the previous RLC PDU and there is no LI that indicates the end of the SDU in the previous RLC PDU. The remaining one octet in the previous RLC PDU is ignored.
111111111111100	UMD PDU: The first data octet in this RLC PDU is the first octet of a RLC SDU. AMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111101	Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111110	AMD PDU: The rest of the RLC PDU includes a piggybacked STATUS PDU. UMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111111	The rest of the RLC PDU is padding. The padding length can be zero.

9.2.2.11.6 The Relative List super-field

The Relative List super-field consists of a type identifier field (RLIST), a list length field (LENGTH), the first sequence number (FSN) and a list of LENGTH number of codewords (CW) as shown in Figure 9.134 below.

Type = **RLIST**
LENGTH
FSN
CW_1
CW_2
…
CW_{LENGTH}

Figure 9.13: The RList fields in a STATUS PDU

LENGTH

Length: 4 bits

The number of codewords (CW) in the super-field of type RLIST.

FSN

Length: 12 bits

The sequence number for the first erroneous PU in the RLIST, i.e. LENGTH="0000" means that only FSN is present in the SUFI.

CW

Length: 4 bits

The CW consists of 4 bits where the three first bits are part of a number and the last bit is a status indicator and it shall be interpreted as follows:

Code Word	Description
$X_1X_2X_3$ 0	Next 3 bits of the number are $X_1X_2X_3$ and the number continues in the next CW. The most significant bit within this CW is X_1.
$X_1X_2X_3$ 1	Next 3 bits of the number are $X_1X_2X_3$ and the number is terminated. The most significant bit within this CW is X_1. This is the most significant CW within the number.

By default, the number given by the CWs represents a distance between the previous indicated erroneous PU up to and including the next erroneous PU.

One special value of CW is defined:

`000 1` 'Error burst indicator'.

The error burst indicator means that the next CWs will represent the number of subsequent erroneous PUs (not counting the already indicated error position). After the number of errors in a burst is terminated with XXX 1, the next codeword will again by default be the least significant bits (LSB) of the distance to the next error.

9.2.2.11.7 The Move Receiving Window Acknowledgement super-field

The 'Move Receiving Window Acknowledgement' super-field acknowledges the reception of a MRW SUFI. The format is given in ~~the~~ F~~f~~igure 9.14 below.

Type = **MRW_ACK**
N
SN_ACK

Figure 9.14: The MRW-ACK fields in a STATUS PDU

N

Length: 4 bits

The N field shall be set equal to the N_{LENGTH} field in the received MRW SUFI if the SN_ACK field is equal to the SN_MRW_{LENGTH} field. Otherwise N shall be set to 0.

With the aid of this field in combination with the SN_ACK field, it can be determined if the MRW_ACK corresponds to a previously transmitted MRW SUFI.

SN_ACK

Length: 12 bits

The SN_ACK field indicates the updated value of VR(R) after the reception of the MRW SUFI. With the aid of this field in combination with the N field, it can be determined if the MRW_ACK corresponds to a previously transmitted MRW SUFI.

9.2.2.11.8 The Move Receiving Window (MRW) super-field

The 'Move Receiving Window' super-field is used to request the RLC receiver to move its receiving window and to indicate the amount of discarded SDUs, as a result of an SDU discard in the RLC transmitter. The format is given in ~~the~~ F~~f~~igure 9.15 below.

Type = **MRW**
LENGTH
SN_MRW_1
SN_MRW_2
...
SN_MRW_{LENGTH}
N_{LENGTH}

Figure 9.15: The MRW fields in a STATUS PDU

LENGTH

Length: 4 bits

The number of SN_MRW_i fields in the super-field of type MRW**.** The values "0001" through "1111" indicate 1 through 15 SN_MRW_i respectively. The value "0000" indicates that one SN_MRW_i field is present and that the discarded SDU extends above the configured Tx window in the transmitter.

SN_MRW_i

Length: 12 bits

SN_MRW_i is used to indicate the end of each discarded SDU. SN_MRW_i is the sequence number of the PU that contains the LI of the i:th discarded SDU (except when $N_{LENGTH} = 0$, see definition of N_{LENGTH}).

Additionally SN_MRW_{LENGTH} requests the RLC receiver to discard all PUs with sequence number < SN_MRW_{LENGTH}, and to move the receiving window accordingly. In addition, the receiver has to discard the first N_{LENGTH} LIs and the corresponding data bytes in the PU with sequence number SN_MRW_{LENGTH}.

N_{LENGTH}

Length: 4 bits

N_{LENGTH} is used together with SN_MRW_{LENGTH} to indicate the end of the last discarded SDU.

N_{LENGTH} indicates which LI in the PU with sequence number SN_MRW_{LENGTH} corresponds to the last discarded SDU. $N_{LENGTH} = 0$ indicates that the last SDU ended in the PU with sequence number $SN\_MRW_{LENGTH} - 1$ and that the first data byte in the PU with sequence number SN_MRW_{LENGTH} is the first data byte to be reassembled next.

9.2.2.12 Reserved 2 (R2)

Length: 1 bit

This bit in the Piggybacked STATUS PDU is used to achieve octet alignment and for this purpose it is coded as 0. Otherwise the PDU is treated as invalid and hence shall be discarded by this version of the protocol.

9.3.1.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of a~~n~~ CRLC-CONFIG-Req from higher layer the RLC entity is created and transparent data transfer ready state is entered.

9.3.1.2 Transparent Data Transfer Ready State

In the transparent data transfer ready, transparent mode data can be exchanged between the entities. Upon reception of a~~n~~ CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

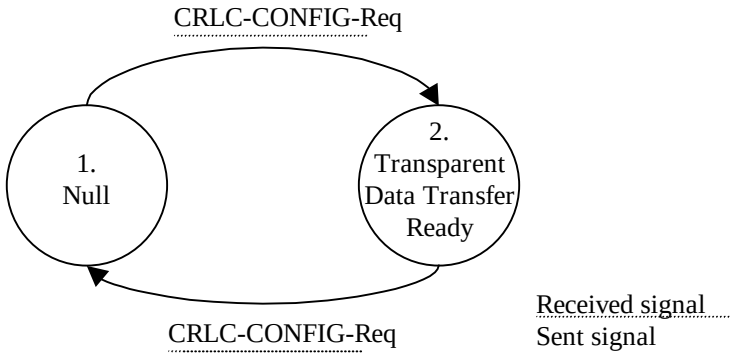


Figure 9.16: The state model for transparent mode entities

9.3.2.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of a~~n~~ CRLC-CONFIG-Req from higher layer the RLC entity is created and unacknowledged data transfer ready state is entered.

9.3.2.2 Unacknowledged Data Transfer Ready State

In the unacknowledged data transfer ready, unacknowledged mode data can be exchanged between the entities. Upon reception of a~~n~~ CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

9.3.2.3 Local Suspend State

Upon reception of a CRLC-SUSPEND-Req from higher layer (RRC) the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send RLC-PDUs with SN>=VT(US)+N. Upon reception of a CRLC-RESUME-Req from higher layer (RRC) the RLC entity is resumed and the Data Transfer Ready state is entered.

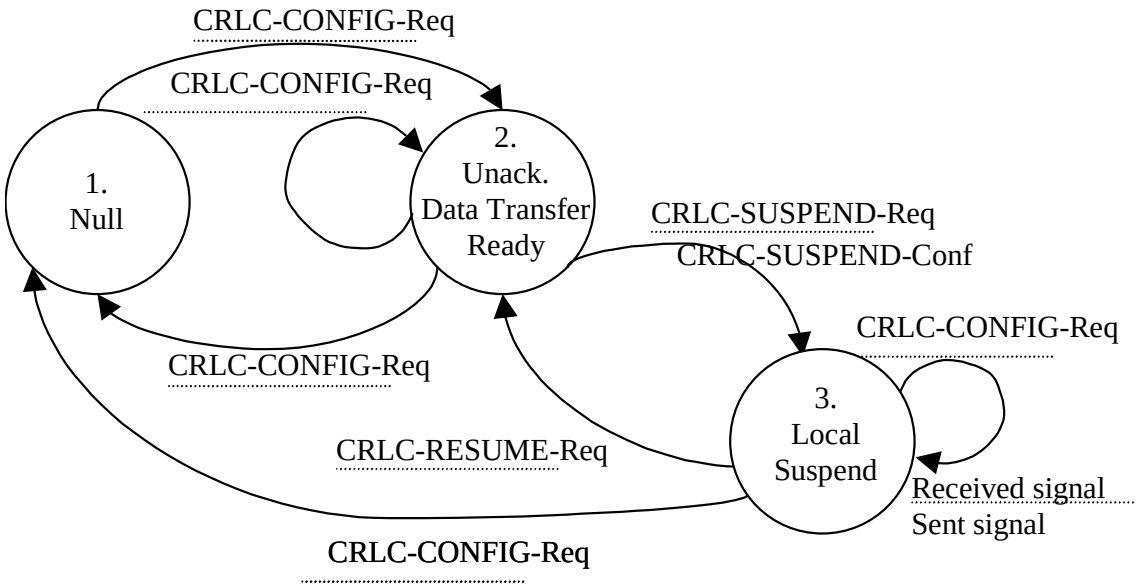


Figure 9.17: The state model for unacknowledged mode entities

---

9.3.3.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of a~~n~~ CRLC-CONFIG-Req from higher layer the RLC entity is created and acknowledged data transfer ready state is entered.

9.3.3.3 *Reset* Pending State

In the reset pending state the entity waits for a response from its peer entity and no data can be exchanged between the entities. Upon reception of a CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

Upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU, the RLC entity resets the protocol (see subclause 11.4.4), sets the hyper frame number HFN (DL HFN when the RESET ACK is received in UE or UL HFN when the RESET ACK is received in UTRAN) equal to the HFNI field in the RESET ACK PDU and one of the following state transitions take place.

The RLC entity enters the acknowledged data transfer ready state if Reset Pending State was entered from Acknowledged Data Transfer Ready State or if Reset Pending State was entered from Local Suspend State and a CRLC-RESUME–Req was received in Reset Pending State.

The RLC entity enters into Local Suspend State if Reset Pending State was entered from Local Suspend State or if Reset Pending State was entered from Acknowledged Data Transfer Ready State and a CRLC-SUSPEND-Req was received in Reset Pending State.

Upon reception of a RESET ACK PDU with a different RSN value as in the corresponding RESET PDU the RESET ACK PDU is discarded.

Upon reception of a RESET PDU, the RLC entity resets the protocol (see subclause 11.4.3), sets the hyper frame number HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) equal to the HFNI field in the RESET PDU, sends a RESET ACK PDU and stays in the reset pending state.

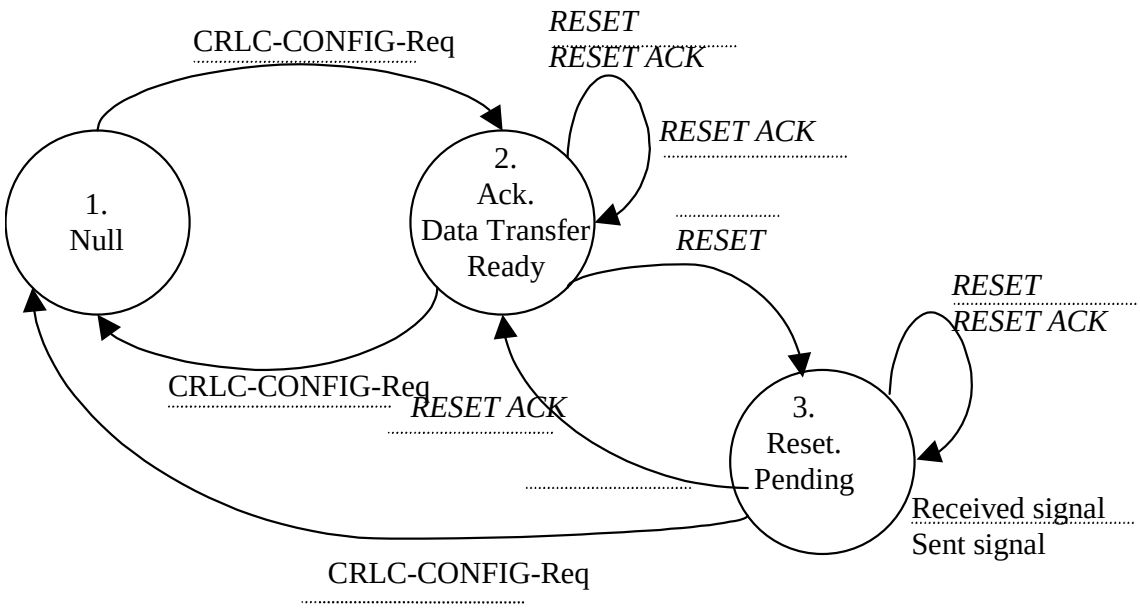


Figure 9.18: The state model for the acknowledged mode entities when reset is performed

9.3.3.4 Local Suspend State

Upon reception of a CRLC-SUSPEND-Req from higher layer (RRC) in Acknowledge Data Transfer Ready State the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send a RLC-PDUs with a SN>=VT(S)+N. Upon reception of CRLC-RESUME-Req from higher layer (RRC) in this state, the RLC entity is resumed and the Data Transfer Ready state is entered.

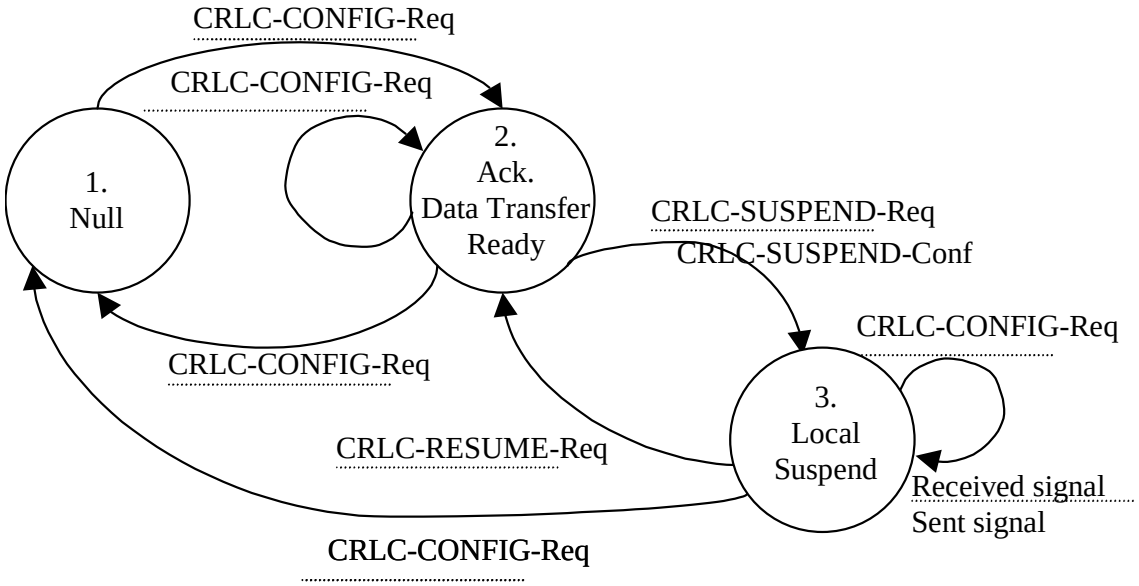


Figure 9.19: The state model for the acknowledged mode entities when local suspend is performed

9.7.3 SDU discard function for acknowledged, unacknowledged, and transparent mode

The SDU discard function allows to discharge RLC PDU from the buffer on the transmitter side, when the transmission of the RLC PDU does not success for a long time. The SDU discard function allows to avoid buffer overflow. There will be several alternative operation modes of the RLC SDU discard function.~~., and which discard function to use will be given by the QoS requirements of the Radio Access Bearer.~~ The network (RRC) controls, which discard function shall~~should~~ be used for each RLC entity.

The following is a list of operation modes for the RLC SDU discard function.

Table 9.2: List of criteria~~'~~s that control when to perform SDU discard

Operation mode	Presence
Timer based discard, with explicit signalling	Network controlled
Timer based discard, without explicit signalling	Network controlled
SDU discard after MaxDAT number of retransmissions	Network controlled

---

9.7.7 RLC stop, RLC Continue function

The higher layer may stop the RLC entity. The stop parameter in the CRLC-CONFIG-Req primitive indicates this request. The RLC entity shall, when receiving this request, not submit any RLC PDUs to lower layer or receive any RLC PDUs. The data transmission and reception is continued when the continue parameter in the CRLC-CONFIG-Req primitive is received. If the continue parameter is received when the RLC entity is not stopped, no action shall be taken.

When the RLC entity is stopped, the RLC timers are not affected. T~~t~~riggered polls and status transmissions are delayed until the RLC entity is continued.

11.4.2.1 RESET PDU contents to set

The size of the RESET PDU shall be equal to one of the allowed PDU sizes. The hyper frame number indicator field (HFNI) shall be set equal to the currently used HFN (DL HFN when the RESET is sent by UTRAN or UL HFN when the RESET is sent by the UE~~-~~). The RSN field shall indicate the sequence number of the RESET PDU. This sequence number is incremented every time a new RESET PDU is transmitted, but not when a RESET PDU is retransmitted.

11.5.2.2 STATUS PDU contents to set

The size of the STATUS PDU shall be equal to one of the allowed PDU sizes. The information that needs to be transmitted in a status report can be split into several STATUS PDUs if one STATUS PDU does not accommodate all the information. A SUFI ~~can not~~cannot be split into several STATUS PDUs. Indication of the same PU shall not be given in more than one STATUS PDU of a STATUS report, but the ACK SUFI can be present in more than one STATUS PDU of a status report.

Which SUFI fields to use is implementation dependent, but the status report shall include information about PUs that have been received and information about all PUs detected as missing. No information shall be given for PUs with SN≥VR(H), i.e. PUs that have not yet reached the receiver.

Padding shall be inserted if the SUFI fields do not fill an entire STATUS PDU. If the PDU contains padding the last SUFI field shall be either an ACK SUFI or a NO_MORE SUFI. If there is no padding in the STATUS PDU, NO_MORE SUFI or ACK SUFI does not need to be included in the STATUS PDU.

11.6.2 Initiation

This procedure is initiated by the sender when the following conditions are fulfilled:

1) Timer based SDU discard with explicit signalling is used, and Timer_Discard expires for an SDU.

2) SDU discard after MaxDAT number of retransmissions is used, and MaxDAT number of retransmissions is reached for an SDU.

The sender shall discard all PUs that contain segments of the associated SDUs. If the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded. VT(A) shall be updated when the procedure is terminated, and VT(S) shall be updated when a new MRW SUFI which includes SN_MRW_{LENGTH} ≥VT(S) is transmitted.

The sender shall transmit a status report on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

This status report is sent even if the 'STATUS prohibit' is used and the timer 'Timer_Status_Prohibit' or 'Timer_EPC' are~~is~~ active.

The STATUS PDUs have higher priority than data PDUs.

The sender shall start timer Timer_MRW. If a new SDU discard procedure is triggered when Timer_MRW is running, no new MRW SUFIs shall be sent before the current SDU discard procedure is terminated by one of the termination criteria.

11.6.4 Termination

The procedure is terminated in the sender in the following cases:

1. On the reception of a STATUS PDU which contains an MRW_ACK SUFI with SN_ACK > SN_MRW_{LENGTH} and with the N field set~~is~~ equal to zero.

2. On the reception of a STATUS PDU which contains an ACK SUFI indicating VR(R) > SN_MRW_{LENGTH}

3. On reception of a STATUS PDU which contains an MRW_ACK with SN_ACK = SN_MRW_{LENGTH} and with the –N field set~~is~~ equal to the N_{LENGTH} indicated in the transmitted MRW SUFI.

If one of the termination criteria above is fulfilled, Timer_MRW shall be~~is~~ stopped and the discard procedure is terminated.

When VT(MRW) reaches MaxMRW, the procedure is terminated and an RLC reset shall be~~is~~ performed.

3GPP TSG-RAN WG2 Meeting #19 ***Tdoc R2-0100655***

Sophia Antipolis, France, 19 - 23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **114** ⌘ rev **r1** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Clarifications on the RLC-AM-DATA-Conf primitive
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 2001-02-13
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 *(GSM Phase 2)*
R96 *(Release 1996)*
R97 *(Release 1997)*
R98 *(Release 1998)*
R99 *(Release 1999)*
REL-4 *(Release 4)*
REL-5 *(Release 5)*

Reason for change:	⌘	The description of the RLC-AM-DATA-Conf primitive and the Confirmation request parameter are misleading.
Summary of change:	⌘	1. Correction in of the description of the RLC-AM-DATA-Conf primitive and the Confirmation request parameter.
Consequences if not approved:	⌘	Unclear and misleading specification.

Clauses affected:	⌘	8.1, 8.2		
Other specs affected:	⌘	☐ Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	
Other comments:	⌘			

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

8.1 Primitives between RLC and higher layers

The primitives between RLC and upper layers are shown in Table 8.1.

Table 8.1: Primitives between RLC and upper layers

Generic Name	Parameter			
	Req.	**Ind.**	**Resp.**	**Conf.**
RLC-AM-DATA	Data, CNF, MUI	Data, DiscardInfo	Not Defined	MUI
RLC-UM-DATA	Data, Use special LI	Data	Not Defined	Not Defined
RLC-TR-DATA	Data	Data	Not Defined	Not Defined
CRLC-CONFIG	E/R, Stop,Continue, Ciphering Elements (UM/AM only), AM_parameters (AM only)	Not Defined	Not Defined	Not Defined
CRLC-SUSPEND (UM/AM only)	N	Not Defined	Not Defined	VT(US) (UM only), VT(S) (AM only)
CRLC-RESUME (UM/AM only)	No Parameter	Not Defined	Not Defined	Not Defined
CRLC-STATUS	Not Defined	EVC	Not Defined	Not Defined

Each Primitive is defined as follows:

RLC-AM-DATA-Req/Ind/Conf

- RLC-AM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in acknowledged mode.
- RLC-AM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in acknowledged mode and to indicate higher layers of the discarded RLC SDU in the ~~receiving~~ peer-RLC AM entity.
- RLC-AM-DATA-Conf is used by RLC to confirm to higher layers reception of an RLC SDU by the peer-RLC AM entity ~~the transmission of a RLC SDU~~.

RLC-UM-DATA-Req/Ind

- RLC-UM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in unacknowledged mode.
- RLC-UM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in unacknowledged mode.

RLC-TR-DATA-Req/Ind

- RLC-TR-DATA-Req is used by higher layers to request transmission of a higher layer PDU in transparent mode.
- RLC-TR-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in transparent mode.

CRLC-CONFIG-Req

This primitive is used by RRC to establish, re-establish, release, stop, continue or reconfigure the RLC. Ciphering elements are included for UM and AM operation.

CRLC-SUSPEND-Req/Conf

This primitive is used by RRC to suspend the RLC. The N parameter indicates that RLC shall not send a PDU with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM, where N is an integer. RLC informs RRC of the VT(S) for AM and VT(US) for UM in the confirm primitive.

CRLC-RESUME-Req

This primitive is used by RRC to resume RLC when RLC has been suspended.

CRLC-STATUS-Ind

It is used by the RLC to send status information to RRC.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the reception ~~correct transmission~~ of the RLC SDU by the peer-RLC AM entity.~~.~~

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC. If it indicates (re-)establishment, the state variables in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state. If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the PU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (~~i.e.,~~e.g. unrecoverable errors such as data link layer loss or recoverable status events such as reset~~, etc.~~).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Activation Time (SN to activate a new ciphering configuration) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains PU size, Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), SDU discard mode (see subclause 9.7.3) and Minimum WSN (see subclause 9.2.2.11.3). The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set.

8) The parameter DiscardInfo indicates to upper layer the discarded RLC SDU in the peer-RLC AM entity. ~~the upper layer of each of the discarded RLC SDU.~~ It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly or one octet short (only when 15 bit LI is used) in the end of the previous RLC PDU is present, the LI shall not be used.

3GPP TSG-RAN WG2 Meeting #19 ***Tdoc R2-010449***

Sophia Antipolis, France, 19-23 February 2001

CR-Form-v3

CHANGE REQUEST

⌘ **25.322 CR 116** ⌘ rev ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE ☒ Radio Access Network ☐ Core Network ☐

Title:	⌘	Removal of the payload unit concept
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 14th February 01
Category:	⌘ **F**	***Release:*** ⌘ R99

Use one of the following categories:
- ***F*** *(essential correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(Addition of feature),*
- ***C*** *(Functional modification of feature)*
- ***D*** *(Editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
REL-4	*(Release 4)*
REL-5	*(Release 5)*

Reason for change:	⌘	For release 99, it was decided to limit the payload unit concept to the case that one PDU contains exactly only one payload unit (PU). However, there is no work item that deals with further refining the PU concept for later releases, and the proponent NOKIA explained that they are not planning to come up with such a refinement in the future.
Summary of change:	⌘	The term payload unit is replaced by PDU where appropriate, and text portions only refering to the payload unit concept are removed.
Consequences if not approved:	⌘	Inconsistent description. Not all parts of the current text seem to be valid if the PDU contains more than one PU.

Clauses affected:	⌘	3, 4.2.1.3, 8.2, 9.1.1, 9.1.2, 9.2.2.3, 9.2.2.8, 9.2.2.11, 9.2.2.11.2, 9.2.2.11.4, 9.2.2.11.6, 9.2.2.11.8, 9.4, 9.5, 9.6, 9.7.1, 9.7.2, 9.7.4, 9.7.5, 10, 11.3.2, 11.3.2.1.1, 11.3.2.1.2, 11.3.3, 11.3.4, 11.5.2, 11.5.2.2, 11.5.3, 11.5.4, 11.6.2, 11.6.3, Annex B
Other specs affected:	⌘	☐ Other core specifications ⌘ ☐ Test specifications ☐ O&M Specifications
Other comments:	⌘	

3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ARQ	Automatic Repeat Request
BCCH	Broadcast Control Channel
BCH	Broadcast Channel
C-	Control-
CCCH	Common Control Channel
CCH	Control Channel
CCTrCH	Coded Composite Transport Channel
CRC	Cyclic Redundancy Check
DCCH	Dedicated Control Channel
DCH	Dedicated Channel
DL	Downlink
DSCH	Downlink Shared Channel
DTCH	Dedicated Traffic Channel
FACH	Forward Link Access Channel
FDD	Frequency Division Duplex
L1	Layer 1 (physical layer)
L2	Layer 2 (data link layer)
L3	Layer 3 (network layer)
LI	Length Indicator
LSB	Least Significant Bit
MAC	Medium Access Control
MRW	Move Receiving Window
MSB	Most Significant Bit
PCCH	Paging Control Channel
PCH	Paging Channel
PDU	Protocol Data Unit
~~PU~~	~~Payload Unit.~~
PHY	Physical layer
PhyCH	Physical Channels
RACH	Random Access Channel
RLC	Radio Link Control
RRC	Radio Resource Control
SAP	Service Access Point
SDU	Service Data Unit
SHCCH	Shared Channel Control Channel
SN	Sequence Number
SUFI	Super Field
TCH	Traffic Channel
TDD	Time Division Duplex
TFI	Transport Format Indicator
TTI	Transmission Time Interval
U-	User-
UE	User Equipment
UL	Uplink
UMTS	Universal Mobile Telecommunications System
UTRA	UMTS Terrestrial Radio Access
UTRAN	UMTS Terrestrial Radio Access Network

4 General

4.2 Overview on sublayer architecture

The model presented in this section is not for implementation purposes.

4.2.1 Model of RLC

Figure 4.1 gives an overview model of the RLC layer. The figure illustrates the different RLC peer entities. There is one transmitting and one receiving entity for the transparent mode service and the unacknowledged mode service and one combined transmitting and receiving entity for the acknowledged mode service. In this specification the word transmitted is equivalent to "submitted to lower layer" unless otherwise explicitly stated. The dashed lines between the AM-Entities illustrate the possibility to send the RLC PDUs on separate logical channels, e.g. control PDUs on one and data PDUs on the other. More detailed descriptions of the different entities are given in subclauses 4.2.1.1, 4.2.1.2 and 4.2.1.3.

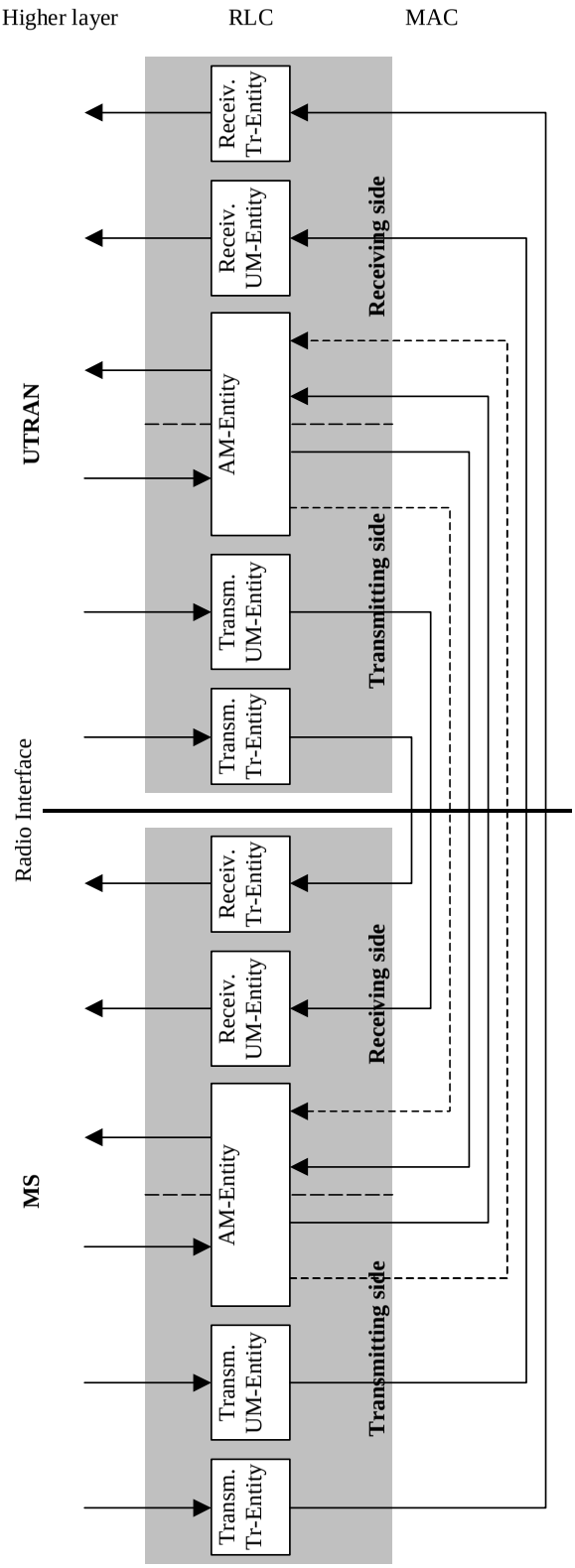


Figure 4.1: Overview model of RLC

4.2.1.1 Transparent mode entities

Figure 4.2 below shows the model of two transparent mode peer entities.

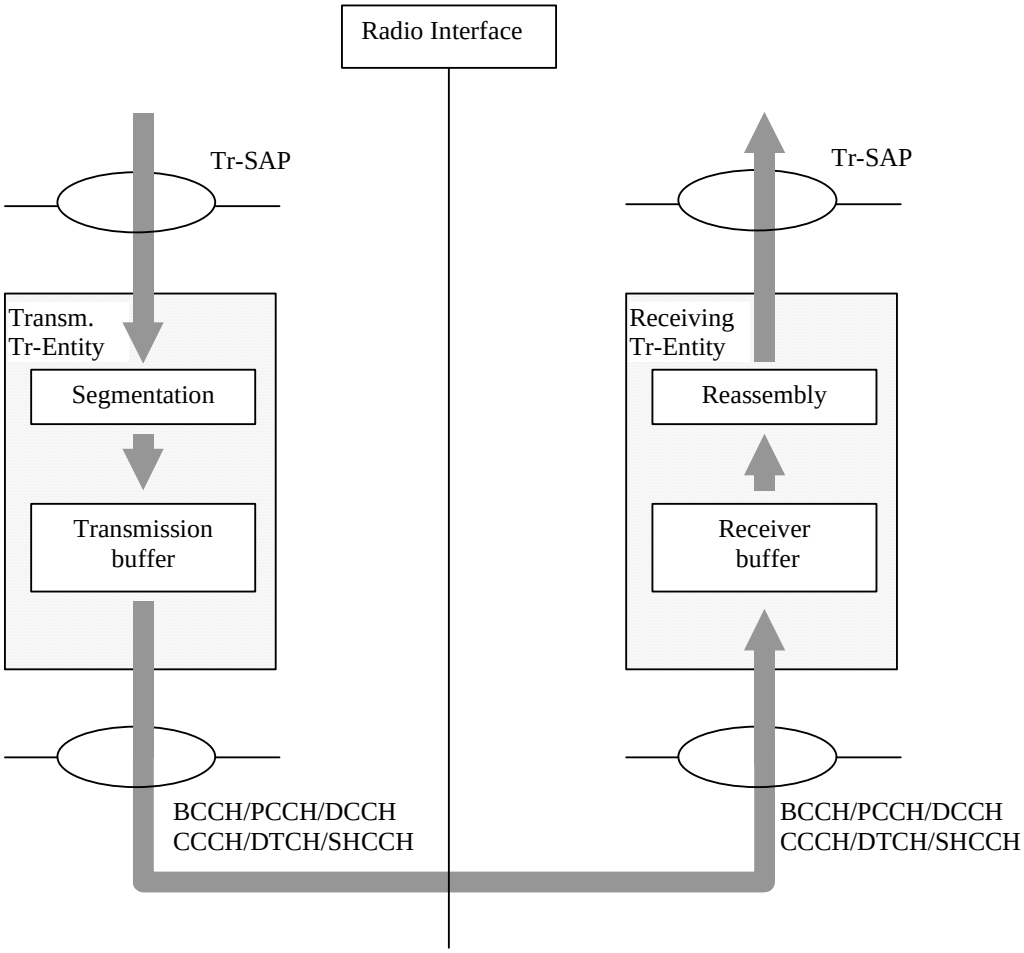


Figure 4.2: Model of two transparent mode peer entities

The transmitting Tr-entity receives SDUs from the higher layers through the Tr-SAP. RLC might segment the SDUs into appropriate RLC PDUs without adding any overhead. How to perform the segmentation is decided upon when the service is established. RLC delivers the RLC PDUs to MAC through either a BCCH, DCCH, PCCH, SHCCH or a DTCH. The CCCH and SHCCH also uses transparent mode, but only for the uplink. Which type of logical channel depends on if the higher layer is located in the control plane (BCCH, DCCH, PCCH, CCCH, SHCCH) or user plane (DTCH).

The Tr-entity receives PDUs through one of the logical channels from the MAC sublayer. RLC reassembles (if segmentation has been performed) the PDUs into RLC SDUs. How to perform the reassembling is decided upon when the service is established. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

4.2.1.2 Unacknowledged mode entities

Figure 4.3 below shows the model of two unacknowledged mode peer entities.

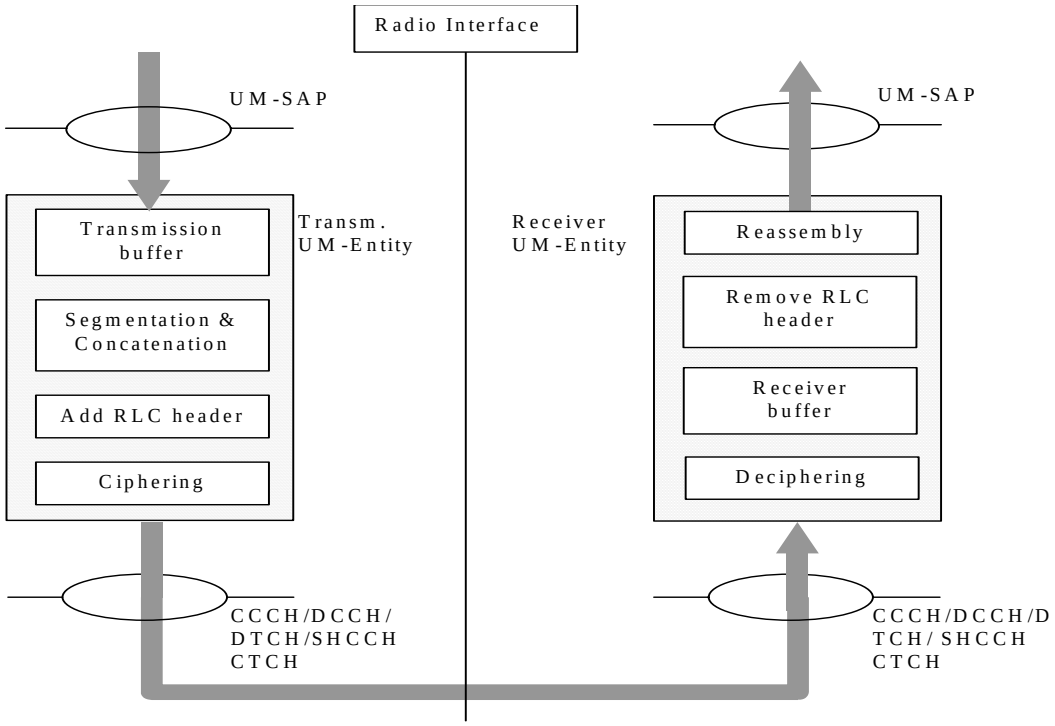


Figure 4.3: Model of two unacknowledged mode peer entities

The transmitting UM-entity receives SDUs from the higher layers. RLC might segment the SDUs into RLC PDUs of appropriate size. The SDU might also be concatenated with other SDUs.. RLC delivers the RLC PDUs to MAC through either a DCCH, CTCH or a DTCH. The CCCH and SHCCH also uses unacknowledged mode, but only for the downlink. Which type of logical channel depends on if the higher layer is located in the control plane (CCCH, DCCH, SHCCH) or user plane (CTCH, DTCH).

The receiving UM-entity receives PDUs through one of the logical channels from the MAC sublayer. RLC removes header from the PDUs and reassembles the PDUs (if segmentation has been performed) into RLC SDUs. The RLC SDUs are delivered to the higher layer.

4.2.1.3 Acknowledged mode entity

Figure 4.4 below shows the model of an acknowledged mode entity, when one logical channel (shown as a solid line) and when two logical channels (shown as dashed lines) are used.

In case two logical channels are used in the uplink the first logical channel shall be used for data PDUs and the second logical channel shall be used for control PDUs. In case one logical channel is used, the RLC PDU size shall be the same for AMD PDUs and control PDUs.

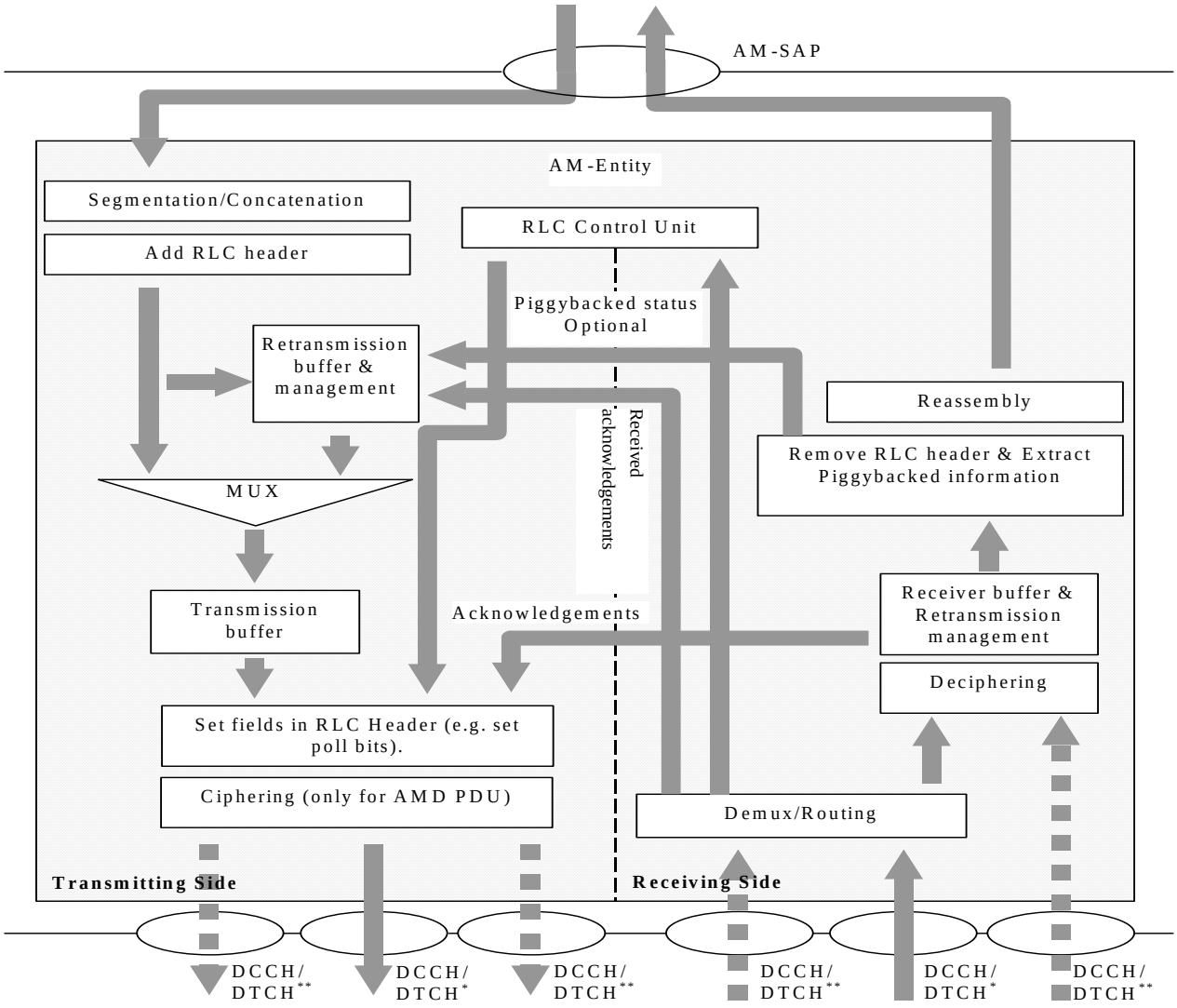


Figure 4.4: Model of an acknowledged mode entity

The transmitting side of the AM-entity receives SDUs from the higher layers. The SDUs are segmented and/or concatenated to ~~PU~~PDUs of fixed length. ~~PU~~PDU length is a semi-static value that is decided in bearer setup and can only be changed through bearer reconfiguration by RRC.

~~For purposes of RLC buffering and retransmission handling, the operation is the same as if there would be one PU per PDU.~~ For concatenation or padding purposes, bits of information on the length and extension are inserted into the beginning of the last ~~PU~~PDU where data from an SDU is included. Padding can be replaced by piggybacked status information. This includes setting the poll bit.

If several SDUs fit into one ~~PU~~PDU, they are concatenated and the appropriate length indicators are inserted into the beginning of the ~~PU~~PDU. After that the ~~PU~~PDUs are placed in the retransmission buffer and the transmission buffer. ~~One PU is included in one RLC PDU.~~

The MUX then decides which PDUs and when the PDUs are submitted to the lower layer. The PDUs are submitted via a function that completes the RLC-PDU header and potentially replaces padding with piggybacked status information. The RLC entity shall assume a PDU to be transmitted when the PDU is submitted to lower layer.

The ciphering is applied only for AMD PDUs. The fixed 2 octet AMD PDU header is not ciphered. Piggybacked and Padding parts of AMD PDU when existing are ciphered. The other Control PDUs (e.g, STATUS, RESET, and RESET ACK PDU) shall not be ciphered.

When Piggybacking mechanism is applied the padding is replaced by control information, in order to increase the transmission efficiency and making possible a faster message exchange between the peer to peer RLC entities. The piggybacked control information is not saved in any retransmission buffer. The piggybacked control information is contained in the piggybacked STATUS PDU, which is in turn included into the AMD-PDU. The piggybacked STATUS PDUs will be of variable size in order to match with the amount of free space in the AMD PDU.

The retransmission buffer also receives acknowledgements from the receiving side, which are used to indicate retransmissions of ~~PU~~PDUs and when to delete a ~~PU~~PDU from the retransmission buffer.

The Receiving Side of the AM-entity receives PDUs through one of the logical channels from the MAC sublayer. ~~The RLC-PDUs are expanded into separate PUs and potential P~~piggybacked status information ~~are~~ is extracted, if present. The ~~PU~~PDUs are placed in the receiver buffer until a complete SDU has been received. The receiver buffer requests retransmissions of ~~PU~~PDUs by sending negative acknowledgements to the peer entity. After that the headers are removed from the PDUs and the PDUs are reassembled into a SDU. Finally the SDU is delivered to the higher layer. The receiving side also receives acknowledgements from the peer entity. The acknowledgements are passed to the retransmission buffer on the transmitting side.

8 Elements for layer-to-layer communication

The interaction between the RLC layer and other layers are described in terms of primitives where the primitives represent the logical exchange of information and control between the RLC layer and other layers. The primitives shall not specify or constrain implementations.

8.1 Primitives between RLC and higher layers

The primitives between RLC and upper layers are shown in Table 8.1.

Table 8.1: Primitives between RLC and upper layers

Generic Name	Parameter			
	Req.	**Ind.**	**Resp.**	**Conf.**
RLC-AM-DATA	Data, CNF, MUI	Data, DiscardInfo	Not Defined	MUI
RLC-UM-DATA	Data, Use special LI	Data	Not Defined	Not Defined
RLC-TR-DATA	Data	Data	Not Defined	Not Defined
CRLC-CONFIG	E/R, Stop,Continue, Ciphering Elements (UM/AM only), AM_parameters (AM only)	Not Defined	Not Defined	Not Defined
CRLC-SUSPEND (UM/AM only)	N	Not Defined	Not Defined	VT(US) (UM only), VT(S) (AM only)
CRLC-RESUME (UM/AM only)	No Parameter	Not Defined	Not Defined	Not Defined
CRLC-STATUS	Not Defined	EVC	Not Defined	Not Defined

Each Primitive is defined as follows:

RLC-AM-DATA-Req/Ind/Conf

- RLC-AM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in acknowledged mode.

- RLC-AM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in acknowledged mode and to indicate higher layers of the discarded RLC SDU in the receiving RLC.

- RLC-AM-DATA-Conf is used by RLC to confirm to higher layers the transmission of a RLC SDU.

RLC-UM-DATA-Req/Ind

- RLC-UM-DATA-Req is used by higher layers to request transmission of a higher layer PDU in unacknowledged mode.

- RLC-UM-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in unacknowledged mode.

RLC-TR-DATA-Req/Ind

- RLC-TR-DATA-Req is used by higher layers to request transmission of a higher layer PDU in transparent mode.

- RLC-TR-DATA-Ind is used by RLC to deliver to higher layers RLC SDUs, that have been transmitted in transparent mode.

CRLC-CONFIG-Req

This primitive is used by RRC to establish, re-establish, release, stop, continue or reconfigure the RLC. Ciphering elements are included for UM and AM operation.

CRLC-SUSPEND-Req/Conf

This primitive is used by RRC to suspend the RLC. The N parameter indicates that RLC shall not send a PDU with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM, where N is an integer. RLC informs RRC of the VT(S) for AM and VT(US) for UM in the confirm primitive.

CRLC-RESUME-Req

This primitive is used by RRC to resume RLC when RLC has been suspended.

CRLC-STATUS-Ind

It is used by the RLC to send status information to RRC.

8.2 Primitive parameters

Following parameters are used in the primitives:

1) The parameter Data is the RLC SDU that is mapped onto the Data field in RLC PDUs. The Data parameter may be divided over several RLC PDUs. In case of a RLC-AM-DATA or a RLC-UM-DATA primitive the length of the Data parameter shall be octet-aligned.

2) The parameter Confirmation request (CNF) indicates whether the RLC needs to confirm the correct transmission of the RLC SDU.

3) The parameter Message Unit Identifier (MUI) is an identity of the RLC SDU, which is used to indicate which RLC SDU that is confirmed with the RLC-AM-DATA conf. primitive.

4) The parameter E/R indicates (re)establishment, release or modification of RLC If it indicates (re-)establishment, the state variables in 9.4 shall be set to their initial value, the configurable parameters shall be set to their configured value and RLC shall enter the data transfer ready state. If it indicates release, all protocol parameters, variables and timers shall be released and RLC shall exit the data transfer ready state. If it indicates modification, the protocol parameters indicated by RRC (e.g. ciphering parameters) shall only be modified with keeping the other protocol parameters, the protocol variables, the protocol timers and the protocol state. RLC shall always be re-established if the ~~PU~~PDU size is changed.

5) The parameter Event Code (EVC) indicates the reason for the CRLC-STATUS-ind (i.e., unrecoverable errors such as data link layer loss or recoverable status events such as reset, etc.).

6) The parameter ciphering elements are only applicable for UM and AM operation. These parameters are Ciphering Mode, Ciphering Key, Activation Time (SN to activate a new ciphering configuration) and HFN (Hyper Frame Number).

7) The AM_parameters are only applicable for AM operation. It contains ~~PU~~PDU size, Timer values (see subclause 9.5), Protocol parameter values (see subclause 9.6), Polling triggers (see subclause 9.7.1), Status triggers (see subclause 9.7.2), SDU discard mode (see subclause 9.7.3) and Minimum WSN (see subclause 9.2.2.11.3). The Minimum WSN shall always be greater than or equal to the number of transport blocks in the smallest transport block set.

8) The parameter DiscardInfo indicates the upper layer of each of the discarded RLC SDU. It is applicable only when in-sequence delivery is active and it is purposed to be used when the upper layer requires the reliable data transfer and especially the information of the discarded RLC SDU.

9) The Stop parameter indicates that the RLC entity shall not transmit or receive RLC PDUs. The Continue parameter indicates that the RLC entity shall continue transmission and reception of RLC PDUs.

10) The parameter Use special LI indicates that the LI indicating that a RLC SDU begins in the beginning of a RLC PDU (the first data octet of the PDU is the first octet of an SDU) shall be used. If the RLC SDU does not begin in the beginning of the RLC PDU, or if the LI indicating that a SDU ended exactly or one octet short (only when 15 bit LI is used) in the end of the previous RLC PDU is present, the LI shall not be used.

9 Elements for peer-to-peer communication

9.1 Protocol data units

9.1.1 Data PDUs

a) TrD PDU (Transparent Mode Data PDU).

The TrD PDU is used to convey RLC SDU data without adding any RLC overhead. The TrD PDU is used by RLC when it is in transparent mode.

b) UMD PDU (Unacknowledged Mode Data PDU).

The UMD PDU is used to convey sequentially numbered PDUs containing RLC SDU data. It is used by RLC when using unacknowledged data transfer.

c) AMD PDU (Acknowledged Mode Data PDU).

The AMD PDU is used to convey sequentially numbered ~~PU~~PDUs containing RLC SDU data. The AMD PDU is used by RLC when it is in acknowledged mode.

9.1.2 Control PDUs

a) STATUS PDU and Piggybacked STATUS PDU

The STATUS PDU and the Piggybacked STATUS PDU are used:

- by the receiving entity to inform the transmitting entity about missing ~~PU~~PDUs at the receiving entity;
- by the receiving entity to inform the transmitting entity about the size of the allowed transmission window;
- and by the transmitting entity to request the receiving entity to move the receiving window.

b) RESET PDU

The RESET PDU is used in acknowledged mode to reset all protocol states, protocol variables and protocol timers of the peer RLC entity in order to synchronise the two peer entities.

c) RESET ACK PDU

The RESET ACK PDU is an acknowledgement to the RESET PDU.

Table 9.1: RLC PDU names and descriptions

Data Transfer Mode	PDU name	Description
Transparent	TrD	Transparent mode data
Unacknowledged	UMD	Sequenced unacknowledged mode data
Acknowledged	AMD	Sequenced acknowledged mode data
	STATUS	Solicited or Unsolicited Status Report
	Piggybacked STATUS	Piggybacked Solicited or Unsolicited Status Report
	RESET	Reset Command
	RESET ACK	Reset Acknowledgement

9.2 Formats and parameters

9.2.1 Formats

This subclause specifies the format of the RLC PDUs. The parameters of each PDU are explained in subclause 9.2.2.

9.2.1.1 General

An RLC PDU is a bit string, with a length not necessarily a multiple of 8 bits. In the drawings in clause 9.2, bit strings are represented by tables in which the first bit is the leftmost one on the first line of the table, the last bit is the rightmost on the last line of the table, and more generally the bit string is to be read from left to right and then in the reading order of the lines.

Depending on the provided service, RLC SDUs are bit strings, with any nonnull length, or bit strings with an integer number of octets in length. An SDU is included into an RLC PDU from first bit onward.

9.2.1.2 TrD PDU

The TrD PDU transfers user data when RLC is operating in transparent mode. No overhead is added to the SDU by RLC. The data length is not constrained to be an integer number of octets.

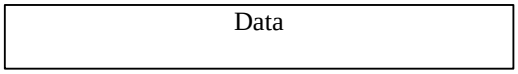


Figure 9.1: TrD PDU

9.2.1.3 UMD PDU

The UMD PDU transfers user data when RLC is operating in unacknowledged mode. The length of the data part shall be an integer number of octets.

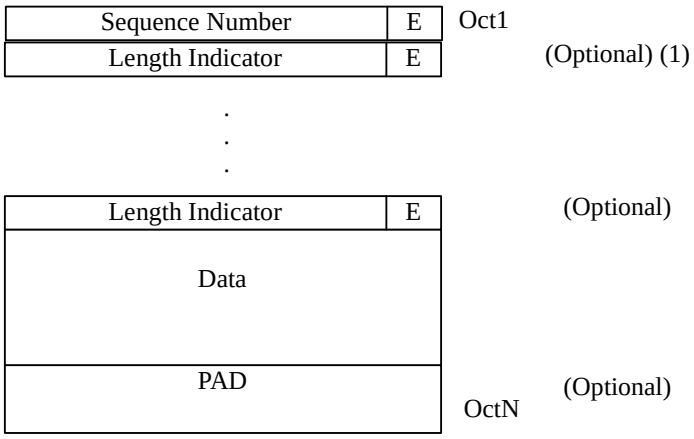
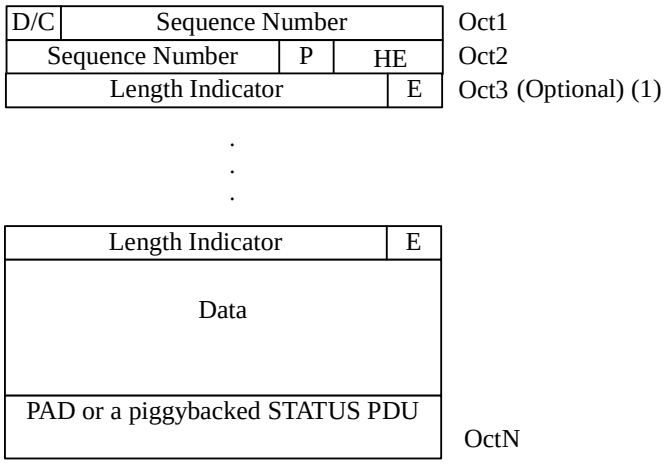


Figure 9.2: UMD PDU

NOTE (1): The Length Indicator may be 15 bits.

9.2.1.4 AMD PDU

The AMD PDU transfers user data and piggybacked status information and requests status report by setting Poll bit when RLC is operating in acknowledged mode. The length of the data part shall be an integer number of octets.



NOTE (1): The Length Indicator may be 15 bits.

Figure 9.3: AMD PDU

9.2.1.5 STATUS PDU

The STATUS PDU is used to report the status between two RLC AM entities. Both receiver and transmitter status information may be included in the same STATUS PDU.

The format of the STATUS PDU is given in Figure 9.4 below. The Figure shows an example and the length of each SUFI is dependent on the SUFI type.

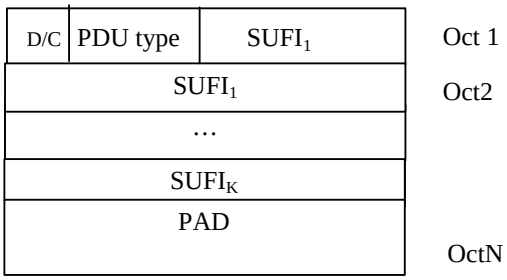


Figure 9.4: Status Information Control PDU (STATUS PDU)

Up to K super-fields ($SUFI_1$-$SUFI_K$) can be included into one STATUS PDU, in which each super-field can be of different type. The size of a STATUS PDU is variable and upper bounded by the maximum RLC PDU size used by an RLC entity. Padding shall be included to exactly fit one of the PDU sizes used by the entity. The length of the STATUS PDU shall be an integer number of octets.

9.2.1.6 Piggybacked STATUS PDU

The format of the piggybacked STATUS PDU is the same as the ordinary Control PDU except that the D/C field is replaced by a reserved bit (R). This PDU can be used to piggyback STATUS PDU in an AMD PDU if the data does not fill the complete AMD PDU. The PDU Type field is set to zero and all other values are invalid for this version of the protocol and the PDU is discarded.

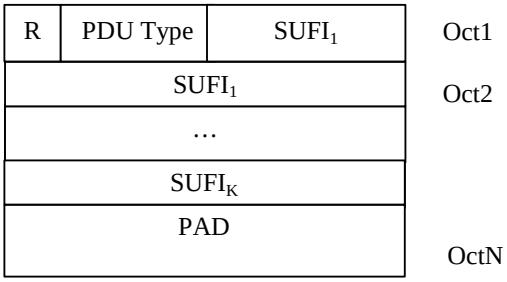


Figure 9.5: Piggybacked STATUS PDU

9.2.1.7 RESET, RESET ACK PDU

The RESET PDU and RESET ACK PDU has a one-bit sequence number field (RSN). With the aid of this field the Receiver can define whether the received RESET PDU is transmitted by the Sender for the first time or whether it is a retransmission of a previous RESET PDU.

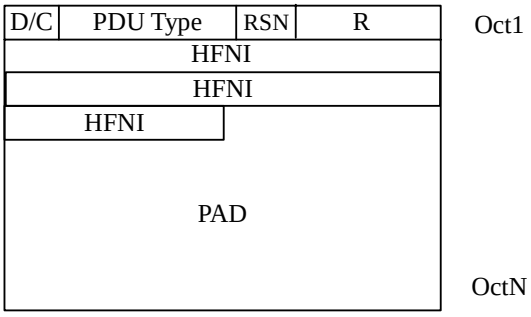


Figure 9.6: RESET, RESET ACK PDU

9.2.2 Parameters

If not otherwise mentioned in the definition of each field then the bits in the parameters shall be interpreted as follows: the left-most bit string is the first and most significant and the right most bit is the last and least significant bit.

Unless otherwise mentioned, integers are encoded in standard binary encoding for unsigned integers. In all cases, including when a value extends over more than one octet as shown in the tables, the bits appear ordered from MSB to LSB when read in the PDU.

9.2.2.1 D/C field

Length: 1bit.

The D/C field indicates the type of an acknowledged mode PDU. It can be either data or control PDU.

Bit	Description
0	Control PDU
1	Acknowledged mode data PDU

9.2.2.2 PDU Type

Length: 3 bit.

The PDU type field indicates the Control PDU type.

Bit	PDU Type
000	STATUS
001	RESET
010	RESET ACK
011-111	Reserved (PDUs with this coding will be discarded by this version of the protocol).

9.2.2.3 Sequence Number (SN)

This field indicates the sequence number of the ~~payload unit~~PDU, encoded in binary.

PDU type	Length	Notes
AMD PDU	12 bits	Used for retransmission and reassembly
UMD PDU	7 bits	Used for reassembly

9.2.2.4 Polling bit (P)

Length: 1bit.

This field is used to request a status report (one or several STATUS PDUs) from the receiver RLC.

Bit	Description
0	Status report not requested
1	Request a status report

9.2.2.5 Extension bit (E)

Length: 1bit.

This bit indicates if the next octet will be a length indicator and E bit.

Bit	Description
0	The next field is data
1	The next field is Length Indicator and E bit

9.2.2.6 Reserved (R)

Length: 3 bits.

This field in the RESET PDU and RESET ACK PDU is used to achieve octet alignment and for this purpose it is coded as 000. Other functions of it are left for future releases.

9.2.2.7 Header Extension Type (HE)

Length: 2 bits.

This two-bit field indicates if the next octet will be data or a length indicator and E bit.

Value	Description
00	The succeeding octet contains data
01	The succeeding octet contains a length indicator and E bit
10-11	Reserved (PDUs with this coding will be discarded by this version of the protocol).

9.2.2.8 Length Indicator (LI)

The Length Indicator is used to indicate, each time, the end of an SDU occurs in the ~~PU~~PDU. The Length Indicator points out the number of octets between the end of the last Length Indicator field and up to and including the octet at the end of an SDU segment. Length Indicators are included in the ~~PU~~PDUs that they refer to. The size of the Length Indicator may be either 7bits or 15bits. The maximum value of a Length Indicator in AM will be no greater than the RLC PDU size – AMD PDU Header – PADDING. The maximum value of a Length Indicator in UM will be no greater than the RLC PDU size – UMD PDU Header – PADDING.

A Length Indicator group is a set of Length Indicators that refer to a ~~PU~~PDU. Length Indicators that are part of a Length Indicator group must never be reordered within the Length Indicator group or removed from the Length Indicator group.

If there can be more than one Length Indicator, each specifying the end of an SDU in a ~~PU~~PDU, the order of these Length Indicators must be in the same order as the SDUs that they refer to.

In the case where the end of the last segment of an SDU exactly ends at the end of a PDU and there is no LI that indicates the end of the SDU, the next Length Indicator, shall be placed as the first Length Indicator in the following ~~PU~~PDU and have value LI=0.

In the case where a PDU contains a 15-bit LI indicating that an SDU ends with one octet left in the PDU, the last octet of this PDU shall be ignored and shall not be filled with the first octet of the next SDU data.

In the case where the last segment of an RLC SDU is one octet short of exactly filling the previous RLC ~~PU~~PDU, and 15-bit Length Indicators are used, the Length Indicator shall be placed as the first Length Indicator in the following ~~PU~~PDU and have value LI=111 1111 1111 1011. The remaining one octet in the previous RLC ~~PU~~PDU shall be ignored.

A ~~PU~~PDU that has unused space, to be referred to as padding, shall use a Length Indicator to indicate that this space is used as padding unless the padding size is one octet for PDUs with 15-bit LIs. A padding Length Indicator must be placed after any Length Indicators for a ~~PU~~PDU.

All unused space in a ~~PU~~PDU must be located at the end of the PDU, be a homogeneous space and is referred to as padding. Predefined values of the Length Indicator are used to indicate this. The values that are reserved for special purposes are listed in the tables below depending on the size of the Length Indicator. Only predefined Length Indicator values can refer to the padding space.

STATUS PDUs can be piggybacked on the AMD PDU by using part or all of the padding space. A Length Indicator must be used to indicate the piggybacked STATUS PDU. This Length Indicator takes space from the padding space or piggybacked STATUS PDU and not the PDU data and will always be the last Length Indicator. Where only part of the padding space is used by a piggybacked STATUS PDU then the end of the piggybacked STATUS PDU is determined by one of the SUFI fields NO_MORE or ACK, thus no additional Length Indicator is required to show that there is still padding in the PDU. The padding/piggybacked STATUS PDU predefined Length Indicators shall be added after the very last (i.e. there could be more than one SDU that end within a PDU) Length Indicator that indicates the end of the last SDU segment in the ~~PU~~PDU.

If SDU discard with explicit signalling is used an AMD PDU can contain a maximum number of 15 LIs indicating the end of an SDU and the rest of the AMD PDU space shall be used as padding/piggybacked STATUS PDU.

For AM, 7bit indicators shall be used if the AMD PDU size is ≤ l26 octets. Otherwise 15bit indicators shall be used. For UM, 7bit indicators shall be used if the UMD PDU size is ≤ l25 octets. Otherwise 15bit indicators shall be used.

The length of the Length Indicator only depends on the size of the largest RLC PDU. The length of the Length Indicator is always the same for all UMD PDUs or AMD ~~PU~~PDUs, for one RLC entity.

If the maximum RLC PDU size for an RLC entity is not explicitly configured (e.g. on FACH), the length of the Length Indicator is determined by the maximum configured TB size for the transport channel on which the logical channel is mapped.

~~For Release 99, there is one PU in an AMD PDU.~~

Length: 7bits

Bit	Description
0000000	The previous RLC PDU was exactly filled with the last segment of a RLC SDU and there is no LI that indicates the end of the SDU in the previous RLC PDU.
1111100	UMD PDU: The first data octet in this RLC PDU is the first octet of a RLC SDU. AMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111101	Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111110	AMD PDU: The rest of the RLC PDU includes a piggybacked STATUS PDU. UMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
1111111	The rest of the RLC PDU is padding. The padding length can be zero.

Length: 15bits

Bit	Description
000000000000000	The previous RLC PDU was exactly filled with the last segment of an RLC SDU and there is no LI that indicates the end of the SDU in the previous RLC PDU.
111111111111011	The last segment of an RLC SDU was one octet short of exactly filling the previous RLC PDU and there is no LI that indicates the end of the SDU in the previous RLC PDU. The remaining one octet in the previous RLC PDU is ignored.
111111111111100	UMD PDU: The first data octet in this RLC PDU is the first octet of a RLC SDU. AMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111101	Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111110	AMD PDU: The rest of the RLC PDU includes a piggybacked STATUS PDU. UMD PDU: Reserved (PDUs with this coding will be discarded by this version of the protocol).
111111111111111	The rest of the RLC PDU is padding. The padding length can be zero.

9.2.2.9 Data

RLC SDUs or segments of RLC SDUs are mapped to this field in transparent, unacknowledged and acknowledged mode.

Transparent mode data:

The length of RLC SDUs is not constrained to a multiple of 8 bits.

The RLC SDUs might be segmented. The allowed size for the segments shall be determined from the transport formats of the transport channel [4, 8]. All the RLC PDUs carrying one RLC SDU shall be sent in one transmission time interval. Only segments from one RLC SDU shall be sent in one transmission time interval.

NOTE: If segmentation is not used for the transparent mode RLC entity then more than one RLC SDU can be sent in one transmission time interval using one RLC PDU per RLC SDU. The RLC PDUs need, however, to be of the same size due to L1 limitations.

Unacknowledged mode data and Acknowledged mode data:

The length of RLC SDUs is constrained to a multiple of 8 bits.

RLC SDUs might be segmented. If possible, the last segment of an SDU shall be concatenated with the first segment of the next SDU in order to fill the data field completely and avoid unnecessary padding. The length indicator field is used to point the borders between SDUs.

For PDUs with 15-bit LIs, if an SDU ends with one octet left in a PDU whether the LI indicating the end of the SDU is contained in this PDU or in the next PDU, padding for the last octet of this PDU is necessary and the next SDU shall not be concatenated in this PDU. No LI shall be needed to indicate this kind of one-octet padding.

9.2.2.10 Padding (PAD)

Padding has a length such that the PDU has the required predefined total length.

Padding may have any value and the receiving entity shall disregard it.

9.2.2.11 SUFI

Which SUFI fields to use is implementation dependent, but when a STATUS PDU includes information about which ~~PU~~PDUs have been received and which are detected as missing, information shall not be included about ~~PU~~PDUs with

SN≥VR(H) i.e. ~~PU~~PDUs that have not yet reached the receiver. Information about ~~PU~~PDUs with SN<VR(R) shall not be given except when this is necessary in order to use the BITMAP SUFI, see 9.2.2.11.5.

Length: variable number of bits.

The SUFI (Super-Field) includes three sub-fields: type information (type of super-field, e.g. list, bitmap, acknowledgement, etc), length information (providing the length of a variable length field within the following value field) and a value.

Figure 9.7 shows the structure of the super-field. The size of the type sub-field is non-zero but the size of the other sub-fields may be zero.

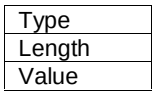


Figure 9.7: The Structure of a Super-Field

The length of the type field is 4 bits and it may have any of following values.

Bit	Description
0000	No More Data (**NO_MORE**)
0001	Window Size (**WINDOW**)
0010	Acknowledgement (**ACK**)
0011	List (**LIST**)
0100	Bitmap (**BITMAP**)
0101	Relative list (**RIist)**
0110	Move Receiving Window **(MRW)**
0111	Move Receiving Window Acknowledgement **(MRW_ACK)**
1000-1111	Reserved (PDUs with this encoding are invalid for this version of the protocol)

The length sub-field gives the length of the variable size part of the following value sub-field and the length of it depends on the super-field type. The value sub-field includes the value of the super-field, e.g. the bitmap in case of a BITMAP super-field, and the length is given by the length of the type sub-field.

9.2.2.11.1 The No More Data super-field

The 'No More Data' super-field indicates the end of the data part of a STATUS PDU and is shown in Figure 9.8 below. It shall always be placed as the last SUFI if it is included in a STATUS PDU. All data after this SUFI shall be regarded as padding and shall be neglected.

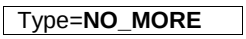


Figure 9.8: NO_MORE field in a STATUS PDU

9.2.2.11.2 The Acknowledgement super-field

The 'Acknowledgement' super-field consists of a type identifier field (ACK) and a sequence number (LSN) as shown in Figure 9.9 below. The acknowledgement super-field is also indicating the end of the data part of a STATUS PDU. Thus, no 'NO_MORE' super-field is needed in the STATUS PDU when the 'ACK' super-field is present. The ACK SUFI shall always be placed as the last SUFI if it is included in a STATUS PDU. All data after this SUFI shall be regarded as padding and shall be neglected.

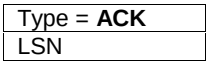


Figure 9.9: The ACK fields in a STATUS PDU

LSN

Length: 12 bits

Acknowledges the reception of all ~~PU~~PDUs with sequence numbers < LSN (Last Sequence Number) that are *not* indicated to be erroneous in earlier parts of the STATUS PDU. This means that if the LSN is set to a different value than VR(R) all erroneous ~~PU~~PDUs must be included in the same STATUS PDU and if the LSN is set to VR(R), the erroneous ~~PU~~PDUs can be split into several STATUS PDUs. At the transmitter, if the value of the LSN =< the value of the first error indicated in the STATUS PDU, VT(A) will be updated according to the LSN, otherwise VT(A) will be updated according to the first error indicated in the STATUS PDU. VT(A) is only updated based on STATUS PDUs where ACK SUFI (or MRW_ACK SUFI) is included. The LSN should not be set to a value > VR(H).

9.2.2.11.3 The Window Size super-field

The 'Window Size' super-field consists of a type identifier (WINDOW) and a window size number (WSN) as shown in Figure 9.10 below. The receiver is always allowed to change the Tx window size of the peer entity during a connection, but the minimum and the maximum allowed value is given by RRC configuration. The Rx window of the receiver is not changed.

Type = **WINDOW**
WSN

Figure 9.10: The WINDOW fields in a STATUS PDU

WSN

Length: 12 bits

The value of VT(WS) to be used by the transmitter. The range of the WSN is [0, 2^{12}-1]. The minimum value of VT(WS) is 1, if WSN is zero the SUFI shall be discarded by this version of the protocol. The variable VT(WS) is set equal to WSN upon reception of this SUFI. If WSN is greater than Configured_Tx_Window_Size, VT(WS) shall be set equal to Configured_Tx_Window_Size.

9.2.2.11.4 The List super-field

The List Super-Field consists of a type identifier field (LIST), a list length field (LENGTH) and a list of LENGTH number of pairs as shown in Figure 9.11 below:

Type = **LIST**
LENGTH
SN_1
L_1
SN_2
L_2
…
SN_{LENGTH}
L_{LENGTH}

Figure 9.11: The List fields in a STATUS PDU for a list

LENGTH

Length: 4 bits

The number of (SN_i , L_i)-pairs in the super-field of type LIST. The value "0000" is invalid and the list is discarded.

SN_i

Length: 12 bits

Sequence number of ~~PU~~PDU, which was not correctly received.

L_i

Length: 4 bits

Number of consecutive ~~PU~~PDUs not correctly received following ~~PU~~PDU with sequence number SN_i.

9.2.2.11.5 The Bitmap super-field

The Bitmap Super-Field consists of a type identifier field (BITMAP), a bitmap length field (LENGTH), a first sequence number (FSN) and a bitmap as shown in Figure 9.12 below:

Type = **BITMAP**
LENGTH
FSN
Bitmap

Figure 9.12: The Bitmap fields in a STATUS PDU

LENGTH

Length: 4 bits

The size of the bitmap in octets equals LENGTH+1, i.e. LENGTH="0000" means that the size of the bitmap is one octet and LENGTH="1111" gives the maximum bitmap size of 16 octets.

FSN

Length: 12 bits

The sequence number for the first bit in the bitmap. FSN shall not be set to a value lower than VR(R)-7 when the Rx window size is less then half the maximum RLC AM sequence number. If the Rx window size is larger, FSN shall not be set to a value lower than VR(R).

Bitmap

Length: Variable number of octets given by the LENGTH field.

Status of the SNs in the interval [FSN, FSN + (LENGTH+1)*8 - 1] indicated in the bitmap where each position (from left to right) can have two different values (0 and 1) with the following meaning (bit_position$\in$ [0,(LENGTH+1)*8 - 1]):

1: SN = (FSN + bit_position) has been correctly received.

0: SN = (FSN + bit_position) has not been correctly received.

9.2.2.11.6 The Relative List super-field

The Relative List super-field consists of a type identifier field (RLIST), a list length field (LENGTH), the first sequence number (FSN) and a list of LENGTH number of codewords (CW) as shown in Figure 9.134 below.

Type = **RLIST**
LENGTH
FSN
CW_1
CW_2
…
CW_{LENGTH}

Figure 9.13: The RList fields in a STATUS PDU

LENGTH

Length: 4 bits

The number of codewords (CW) in the super-field of type RLIST.

FSN

Length: 12 bits

The sequence number for the first erroneous ~~PU~~PDU in the RLIST, i.e. LENGTH="0000" means that only FSN is present in the SUFI.

CW

Length: 4 bits

The CW consists of 4 bits where the three first bits are part of a number and the last bit is a status indicator and it shall be interpreted as follows:

Code Word	Description
$X_1X_2X_3$ 0	Next 3 bits of the number are $X_1X_2X_3$ and the number continues in the next CW. The most significant bit within this CW is X_1.
$X_1X_2X_3$ 1	Next 3 bits of the number are $X_1X_2X_3$ and the number is terminated. The most significant bit within this CW is X_1. This is the most significant CW within the number.

By default, the number given by the CWs represents a distance between the previous indicated erroneous ~~PU~~PDU up to and including the next erroneous ~~PU~~PDU.

One special value of CW is defined:

`000 1` 'Error burst indicator'.

The error burst indicator means that the next CWs will represent the number of subsequent erroneous ~~PU~~PDUs (not counting the already indicated error position). After the number of errors in a burst is terminated with XXX 1, the next codeword will again by default be the least significant bits (LSB) of the distance to the next error.

9.2.2.11.7 The Move Receiving Window Acknowledgement super-field

The 'Move Receiving Window Acknowledgement' super-field acknowledges the reception of a MRW SUFI. The format is given in the figure below.

Type = **MRW_ACK**
N
SN_ACK

Figure 9.14: The MRW-ACK fields in a STATUS PDU

N

Length: 4 bits

The N field shall be set equal to the N_{LENGTH} field in the received MRW SUFI if the SN_ACK field is equal to the SN_MRW_{LENGTH} field. Otherwise N shall be set to 0.

With the aid of this field in combination with the SN_ACK field, it can be determined if the MRW_ACK corresponds to a previously transmitted MRW SUFI.

SN_ACK

Length: 12 bits

The SN_ACK field indicates the updated value of VR(R) after the reception of the MRW SUFI. With the aid of this field in combination with the N field, it can be determined if the MRW_ACK corresponds to a previously transmitted MRW SUFI.

9.2.2.11.8 The Move Receiving Window (MRW) super-field

The 'Move Receiving Window' super-field is used to request the RLC receiver to move its receiving window and to indicate the amount of discarded SDUs, as a result of a SDU discard in the RLC transmitter. The format is given in the figure below.

Type = **MRW**
LENGTH
SN_MRW_1
SN_MRW_2
...
SN_MRW_{LENGTH}
N_{LENGTH}

Figure 9.15: The MRW fields in a STATUS PDU

LENGTH

Length: 4 bits

The number of SN_MRW_i fields in the super-field of type MRW**.** The values "0001" through "1111" indicate 1 through 15 SN_MRW_i respectively. The value "0000" indicates that one SN_MRW_i field is present and that the discarded SDU extends above the configured Tx window in the transmitter.

SN_MRW_i

Length: 12 bits

SN_MRW_i is used to indicate the end of each discarded SDU. SN_MRW_i is the sequence number of the ~~PU~~PDU that contains the LI of the i:th discarded SDU (except when $N_{LENGTH} = 0$, see definition of N_{LENGTH}).

Additionally SN_MRW_{LENGTH} requests the RLC receiver to discard all ~~PU~~PDUs with sequence number < SN_MRW_{LENGTH}, and to move the receiving window accordingly. In addition, the receiver has to discard the first N_{LENGTH} LIs and the corresponding data bytes in the ~~PU~~PDU with sequence number SN_MRW_{LENGTH}.

N_{LENGTH}

Length: 4 bits

N_{LENGTH} is used together with SN_MRW_{LENGTH} to indicate the end of the last discarded SDU.

N_{LENGTH} indicates which LI in the ~~PU~~PDU with sequence number SN_MRW_{LENGTH} corresponds to the last discarded SDU. $N_{LENGTH} = 0$ indicates that the last SDU ended in the ~~PU~~PDU with sequence number $SN\_MRW_{LENGTH} - 1$ and that the first data byte in the ~~PU~~PDU with sequence number SN_MRW_{LENGTH} is the first data byte to be reassembled next.

9.2.2.12 Reserved (R)

Length: 1 bit

This bit in the Piggybacked STATUS PDU is used to achieve octet alignment and for this purpose it is coded as 0. Otherwise the PDU is treated as invalid and hence shall be discarded by this version of the protocol.

9.2.2.13 Reset Sequence Number (RSN)

Length: 1 bit

This field is used to indicate the sequence number of the transmitted RESET PDU. If this RESET PDU is a retransmission of the original RESET PDU then the retransmitted RESET PDU would have the same sequence number value as the original RESET PDU. Otherwise it will have the next reset sequence number. The initial value of this field is zero. The value of this field shall be reinitialized when the RLC is re-established. It shall not be reinitialized when the RLC is reset.

9.2.2.14 Hyper Frame Number Indicator (HFNI)

Length: 20 bit

This field is used to indicate the hyper frame number (HFN) to the peer entity. With the aid of this field the HFN in UE and UTRAN can be synchronised.

9.3 Protocol states

9.3.1 State model for transparent mode entities

Figure 9.16 illustrates the state model for transparent mode RLC entities (both transmitting and receiving). A transparent mode entity can be in one of following states.

9.3.1.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is created and transparent data transfer ready state is entered.

9.3.1.2 Transparent Data Transfer Ready State

In the transparent data transfer ready, transparent mode data can be exchanged between the entities. Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

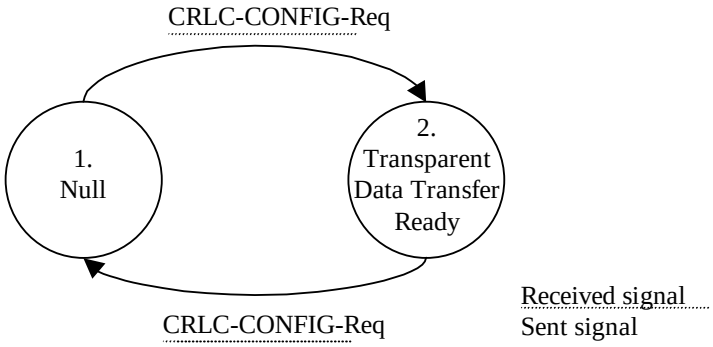


Figure 9.16: The state model for transparent mode entities

9.3.2 State model for unacknowledged mode entities

Figure 9.17 illustrates the state model for unacknowledged mode RLC entities (both transmitting and receiving). An unacknowledged mode entity can be in one of following states.

9.3.2.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is created and unacknowledged data transfer ready state is entered.

9.3.2.2 Unacknowledged Data Transfer Ready State

In the unacknowledged data transfer ready, unacknowledged mode data can be exchanged between the entities. Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

9.3.2.3 Local Suspend State

Upon reception of CRLC-SUSPEND-Req from higher layer (RRC) the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send RLC-PDUs with SN>=VT(US)+N. Upon reception of CRLC-RESUME-Req from higher layer (RRC) the RLC entity is resumed and the Data Transfer Ready state is entered.

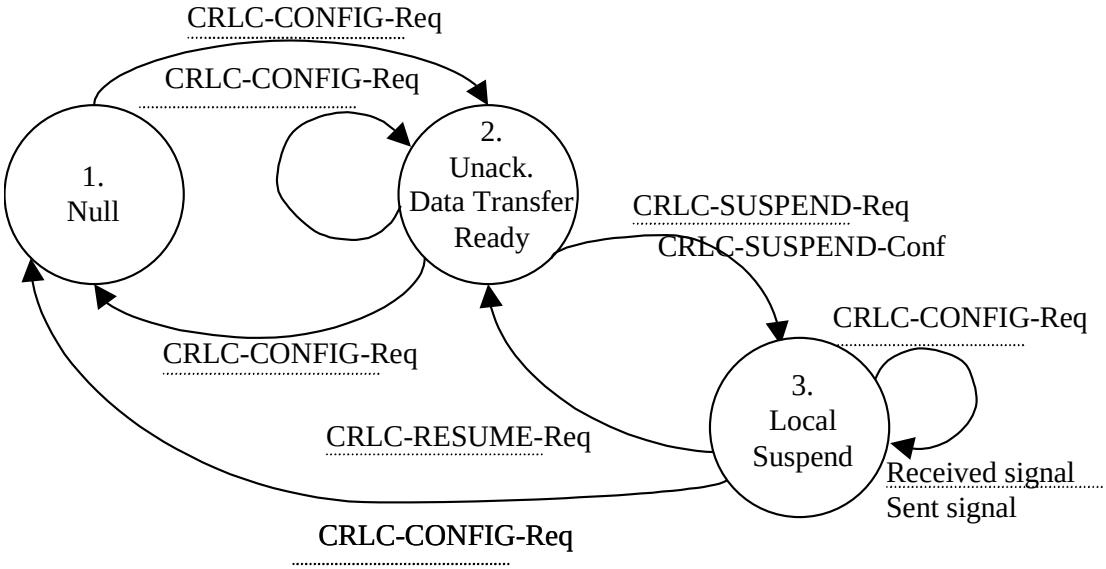


Figure 9.17: The state model for unacknowledged mode entities

9.3.3 State model for acknowledged mode entities

Figure 9.18 illustrates the state model for the acknowledged mode RLC entity (both transmitting and receiving). An acknowledged mode entity can be in one of following states.

9.3.3.1 Null State

In the null state the RLC entity does not exist and therefore it is not possible to transfer any data through it.

Upon reception of an CRLC-CONFIG-Req from higher layer the RLC entity is created and acknowledged data transfer ready state is entered.

9.3.3.2 Acknowledged Data Transfer Ready State

In the acknowledged data transfer ready state, acknowledged mode data can be exchanged between the entities. Upon reception of a CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

Upon errors in the protocol, the RLC entity sends a RESET PDU to its peer and enters the reset pending state.

Upon reception of a RESET PDU, the RLC entity resets the protocol (see subclause 11.4.3), sets the hyper frame number HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) equal to the HFNI field in the RESET PDU and responds to the peer entity with a RESET ACK PDU.

Upon reception of a RESET ACK PDU, the RLC takes no action.

9.3.3.3 *Reset* Pending State

In the reset pending state the entity waits for a response from its peer entity and no data can be exchanged between the entities. Upon reception of CRLC-CONFIG-Req from higher layer the RLC entity is terminated and the null state is entered.

Upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU, the RLC entity resets the protocol (see subclause 11.4.4),sets the hyper frame number HFN (DL HFN when the RESET ACK is received in UE or UL HFN when the RESET ACK is received in UTRAN) equal to the HFNI field in the RESET ACK and one of the following state transitions take place.

The RLC entity enters the acknowledged data transfer ready state if Reset Pending State was entered from Acknowledged Data Transfer Ready State or if Reset Pending State was entered from Local Suspend State and a CRLC-RESUME–Req was received in Reset Pending State.

The RLC entity enters into Local Suspend State if Reset Pending State was entered from Local Suspend State or if Reset Pending State was entered from Acknowledged Data Transfer Ready State and a CRLC-SUSPEND-Req was received in Reset Pending State.

Upon reception of a RESET ACK PDU with a different RSN value as in the corresponding RESET PDU the RESET ACK PDU is discarded.

Upon reception of a RESET PDU, the RLC entity resets the protocol (see subclause 11.4.3), sets the hyper frame number HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) equal to the HFNI field in the RESET PDU , sends a RESET ACK PDU and stays in the reset pending state.

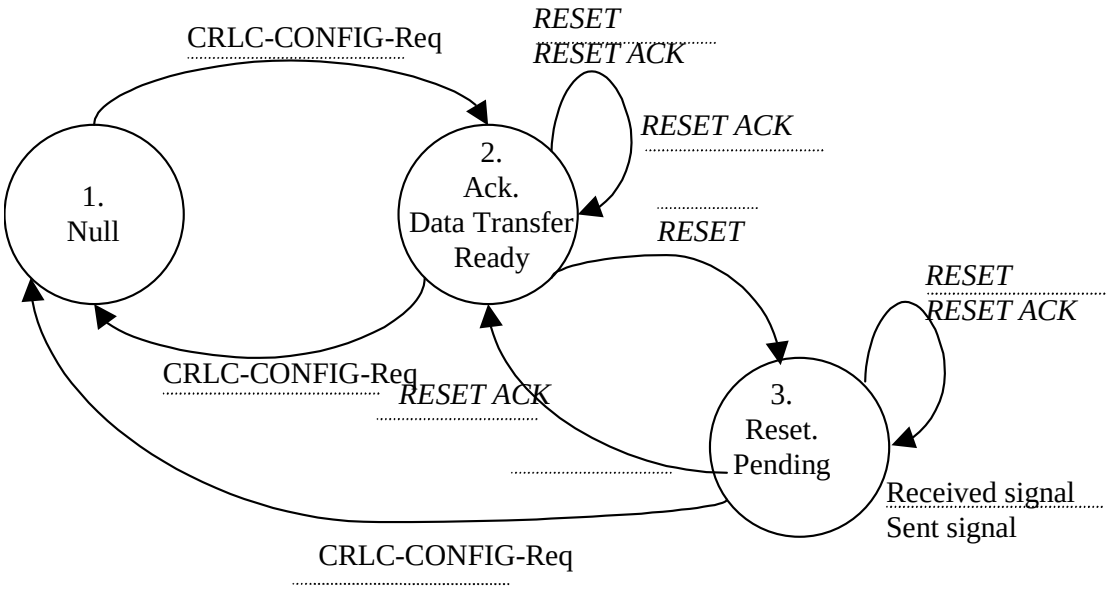


Figure 9.18: The state model for the acknowledged mode entities when reset is performed

9.3.3.4 Local Suspend State

Upon reception of CRLC-SUSPEND-Req from higher layer (RRC) in Acknowledge Data Transfer Ready State the RLC entity is suspended and the Local Suspend state is entered. In the Local Suspend state RLC shall not send a RLC-PDUs with a SN>=VT(S)+N. Upon reception of CRLC-RESUME-Req from higher layer (RRC) in this state, the RLC entity is resumed and the Data Transfer Ready state is entered.

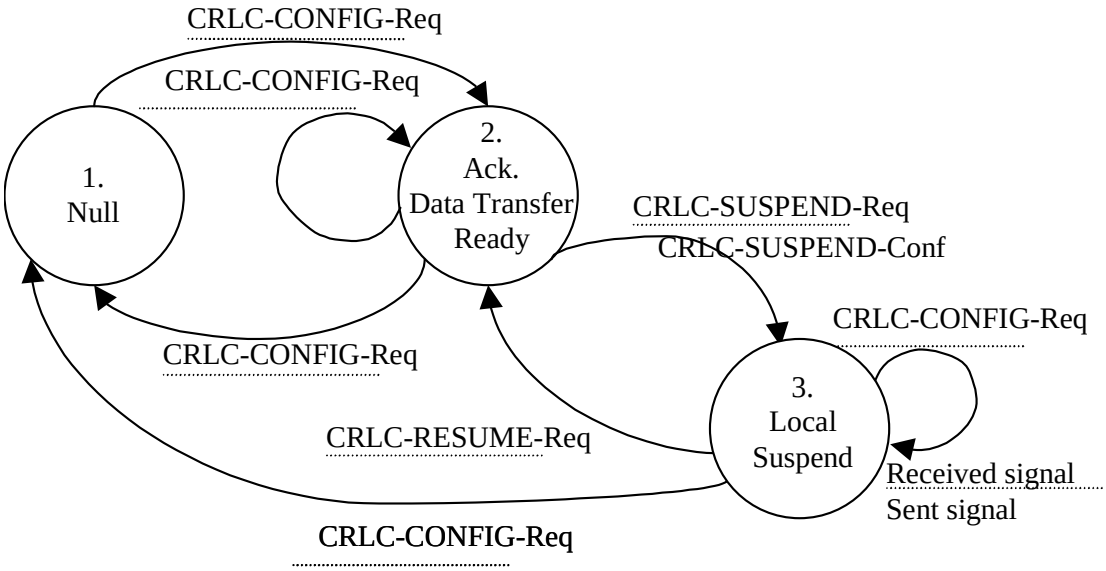


Figure 9.19: The state model for the acknowledged mode entities when local suspend is performed

9.4 State variables

This sub-clause describes the state variables used in the specification of the peer-to-peer protocol. All state variables are non-negative integers. ~~PU~~PDUs are sequentially and independently numbered and may have the value 0 through n minus 1 (where n is the modulus of the sequence numbers). The modulus equals 2^{12} for AM and 2^{7} for UM; the sequence numbers cycle through the entire range: 0 through $2^{12} - 1$ for AM and 0 through $2^{7} - 1$ for UM. All arithmetic operations on the following state variables and sequence numbers contained in this specification are affected by the modulus: VT(S), VT(A), VT(MS), VR(R), VR(H), VR(MR), VT(US) and VR(US). When performing arithmetic comparisons of transmitter variables, VT(A) is assumed to be the base. When performing arithmetic comparisons of receiver variables, VR(R) is assumed to be the base.

The RLC maintains the following state variables at the transmitter.

a) VT(S) - Send state variable.

   The sequence number of the next ~~PU~~PDU to be transmitted for the first time (i.e. excluding retransmission). It is updated after transmission of a PDU, which includes not earlier transmitted ~~PU~~PDUs and after transmission of a MRW SUFI which includes $SN\_MRW_{LENGTH} \geq VT(S)$. The initial value of this variable is 0.

b) VT(A) - Acknowledge state variable.

   The sequence number of the next in-sequence ~~PU~~PDU expected to be acknowledged, which forms the lower edge of the window of acceptable acknowledgements. VT(A) is updated based on receipt of a STATUS PDU including an ACK and/or MRW_ACK super-field. The initial value of this variable is 0.

c) VT(DAT).

   This state variable counts the number of times a ~~PU~~PDU has been transmitted. There is one VT(DAT) for each ~~PU~~PDU and it is incremented each time the ~~PU~~PDU is transmitted. The initial value of this variable is 0.

d) VT(MS) - Maximum Send state variable.

   The sequence number of the first ~~PU~~PDU not allowed by the peer receiver [i.e. the receiver will allow up to VT(MS) – 1], VT(MS) = VT(A) + VT(WS). This value represents the upper edge of the transmit window. The transmitter

shall not transmit a ~~PU~~PDU with SN ≥ VT(MS). VT(MS) is updated when either VT(A) or VT(WS) is updated. The ~~PU~~PDU with SN VT(S)-1 can be transmitted also when VT(S) ≥ VT(MS).

e) VT(US) – UM data state variable.

This state variable gives the sequence number of the next UMD PDU to be transmitted. It is updated each time a UMD PDU is transmitted. The initial value of this variable is 0.

f) VT(~~PU~~PDU).

This state variable is used when the poll every Poll_~~PU~~PDU ~~PU~~PDU function is used. It is incremented with 1 for each ~~PU~~PDU that is transmitted. It should be incremented for both new and retransmitted ~~PU~~PDUs. When it reaches Poll_~~PU~~PDU a new poll is transmitted and the state variable is set to zero. The initial value of this variable is 0.

g) VT(SDU).

This state variable is used when the poll every Poll_SDU SDU function is used. It is incremented with 1 for each SDU that is transmitted. When it reaches Poll_SDU a new poll is transmitted and the state variable is set to zero. The poll bit should be set in the ~~PU~~PDU that contains the last segment of the SDU. The initial value of this variable is 0.

h) VT(RST) - Reset state variable.

It is used to count the number of times a RESET PDU is transmitted. VT(RST) is incremented with 1 each time a RESET PDU is transmitted. VT(RST) is reset only upon the reception of a RESET ACK PDU, i.e. VT(RST) is not reset when a RLC reset occurs which was initiated from the peer RLC entity. The initial value of this variable is 0.

i) VT(MRW) – MRW command send state variable.

It is used to count the number of times a MRW command is transmitted. VT(MRW) is incremented with 1 each time an MRW command is transmitted. VT(MRW) is reset when the discard procedure is terminated. The initial value of this variable is 0.

- VT(WS) – Transmitter window size state variable.

The size that shall be used for the transmitter window. VT(WS) is set equal to the WSN field when the transmitter receives a STATUS PDU including a Window Size super-field. The initial value of this variable is Configured_Tx_Window_size.

The RLC maintains the following state variables at the receiver:

a) VR(R) - Receive state variable.

The sequence number of the next in-sequence ~~PU~~PDU expected to be received. It is set equal to SNmax+1 upon receipt of the next in-sequence ~~PU~~PDU, where SNmax is the sequence number of the highest received in-sequence ~~PU~~PDU. The initial value of this variable is 0.

b) VR(H) - Highest expected state variable.

The sequence number of the highest expected ~~PU~~PDU. This state variable is set equal to SN+1 only when a new ~~PU~~PDU is received with VR(MR)>SN≥VR(H). The initial value of this variable is 0.

c) VR(MR) - Maximum acceptable Receive state variable.

The sequence number of the first ~~PU~~PDU not allowed by the receiver [i.e. the receiver will allow up to VR(MR) – 1], VR(MR) = VR(R) + Configured_Rx_Window_Size. The receiver shall discard ~~PU~~PDUs with SN ≥ VR(MR).

d) VR(US) - Receiver Send Sequence state variable.

The sequence number of the next PDU to be received. It shall set equal to SN + 1 upon reception of a PDU. The initial value of this variable is 0.

e) VR(EP) - Estimated PDU Counter state variable.

The number of ~~PU~~PDUs that should be received yet as a consequence of the transmission of the latest status report. In acknowledged mode, this state variable is updated at the end of each transmission time interval. It is decremented by the number of ~~PU~~PDUs that should have been received during the transmission time interval. If VR(EP) is equal to zero, then check if all ~~PU~~PDUs requested for retransmission in the latest status report have been received.

9.5 Timers

a) Timer_Poll.

This timer is only used when the poll timer trigger is used. It is started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The timer is stopped when receiving a STATUS PDU that contains an acknowledgement of all AMD PDUs with SN up to and including VT(S)-1 at the time the poll was submitted to lower layer,or when a negative acknowledgement of the same ~~PU~~PDU is received. The value of the timer is signalled by RRC.

If the timer expires and no STATUS PDU fulfilling the criteria above has been received, the receiver is polled once more (either by the transmission of a PDU which was not yet sent, or by a retransmission) and the timer is restarted at the time specified above, with a new value of VT(S)-1.

If a new poll is sent when the timer is running the timer is restarted at the time specified above, with a new value of VT(S)-1.

b) Timer_Poll_Prohibit.

This timer is only used when the poll prohibit function is used. It is used to prohibit transmission of polls within a certain period. The timer shall be started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time a PDU containing a poll is submitted to lower layer until the timer has expired. A poll shall be delayed until the prohibit time expires if a poll is triggered during the prohibit time. Only one poll shall be transmitted when the prohibit time expires even if several polls were triggered during the prohibit time. This timer will not be stopped by a received STATUS PDU. The value of the timer is signalled by RRC.

c) Timer_EPC.

This timer is only used when the EPC function is used and it accounts for the roundtrip delay, i.e. the time when the first retransmitted ~~PU~~PDU should be received after a status report has been sent. The timer is started when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN) and when it expires VR(EP) can start its counting-down process (see subclause 9.7.4). The value of the timer is signalled by RRC.

d) Timer_Discard.

This timer is used for the SDU discard function. In the transmitter, the timer is activated upon reception of a SDU from higher layer. One timer is used for each SDU that is received from higher layer. If the SDU has not been acknowledged and/or transmitted when the timer expires, the SDU is discarded. Following which, if the SDU discard function uses explicit signalling, a Move Receiving Window request is sent to the receiver. The value of the timer is signalled by RRC.

e) Timer_Poll_Periodic.

This timer is only used when the timer based polling is used. The timer is started when the RLC entity is created. Each time the timer expires, the timer is restarted and a poll is triggered (either by the transmission of a PDU which was not yet sent, or by a retransmission). If there is no ~~PU~~PDU to be transmitted and all ~~PU~~PDUs have already been acknowledged, a poll shall not be triggered and the timer shall only be restarted. The value of the timer is signalled by RRC.

f) Timer_Status_Prohibit.

This timer is only used when the STATUS prohibit function is used. It prohibits the receiving side from sending status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. The timer is started when the successful or unsuccessful transmission of the last STATUS PDU in a status report is indicated by lower layer (in UE) or the last STATUS PDU in a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired and no new status report containing the mentioned SUFIs can be transmitted during the prohibit time. The timer does not prohibit transmission of the SUFIs MRW, MRW_ACK, WINDOW or NO_MORE. The value of the timer is signalled by RRC.

g) Timer_Status_Periodic.

This timer is only used when timer based status report sending is used. The timer is started when the RLC entity is created. Each time the timer expires the transmission of a status report is triggered and the timer is restarted. The value of the timer is signalled by RRC.

h) Timer_RST.

This timer is used to detect the loss of RESET ACK PDU from the peer RLC entity. This timer is started when the successful or unsuccessful transmission of a RESET PDU is indicated by lower layer (in UE) or a RESET PDU is submitted to lower layer (in UTRAN). It will only be stopped upon reception of RESET ACK PDU, i.e. this timer is not stopped when an RLC reset occurs which was initiated from the peer RLC entity. If it expires, RESET PDU will be retransmitted. The value of the timer is signalled by RRC.

i) Timer_MRW.

This timer is used as part of the Move Receiving Window protocol. It is used to trigger the retransmission of a status report containing an MRW SUFI field. The timer is started when the successful or unsuccessful transmission of a STATUS PDU containing the MRW SUFI is indicated by lower layer (in UE) or a STATUS PDU containing the MRW SUFI is submitted to lower layer (in UTRAN). Each time the timer expires the MRW SUFI is retransmitted and the timer is restarted (at the time specified above). It shall be stopped when one of the termination criteria for the SDU discard is fulfilled. The value of the timer is signalled by RRC.

9.6 Protocol Parameters

The values of the protocol parameters in this section are signalled by RRC.

a) MaxDAT.

It is the maximum value for the number of retransmissions of a ~~PU~~PDU. This parameter is an upper limit of counter VT(DAT). When the value of VT(DAT) comes to MaxDAT, error recovery procedure will be performed.

b) Poll_~~PU~~PDU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_~~PU~~PDU ~~PU~~PDU. This is an upper limit for the VT(~~PU~~PDU) state variable, when VT(~~PU~~PDU) reaches Poll_~~PU~~PDU a poll is transmitted to the peer entity.

c) Poll_SDU.

This parameter indicates how often the transmitter should poll the receiver in case of polling every Poll_SDU SDU. This is an upper limit for the VT(SDU) state variable, when VT(SDU) reaches Poll_SDU a poll is transmitted to the peer entity.

d) Poll_Window.

This parameter indicates when the transmitter should poll the receiver in case of performing window-based polling. The range of values of this parameter shall be 0 ≤ Poll_Window ≤ 100. A poll is triggered for each ~~PU~~PDU when :J ≥ Poll_Window, where J is the window transmission percentage defined by

$$J = \frac{(4096+VT(S) - VT(A)) \textbf{ mod } 4096}{VT(WS)} * 100 ,$$

where the constant 4096 is the modulus for AM described in Subclause 9.4.

.

e) MaxRST.

It is the maximum value for the number of retransmission of RESET PDU. This parameter is an upper limit of counter VT(RST). When the value of VT(RST) comes to MaxRST, the higher layer (RRC) is notified.

f) Configured_Tx_Window_Size.

The maximum allowed transmitter window size.

g) Configured_Rx_Window_Size.

The allowed receiver window size.

h) MaxMRW.

It is the maximum value for the number of retransmissions of a MRW command. This parameter is an upper limit of counter VT(MRW). When the value of VT(MRW) comes to MaxMRW, error recovery procedure will be performed.

9.7 Specific functions

9.7.1 Polling function for acknowledged mode

The transmitter of AMD PDUs may poll the receiver for a status report (consisting of one or several STATUS PDUs). The Polling bit in the AMD PDU indicates the poll request. If there is no ~~PU~~PDU to be transmitted and all ~~PU~~PDUs have already been acknowledged, the receiver shall not be polled. There are several triggers for setting the polling bit. The network (RRC) controls, which triggers should be used for each RLC entity. Following triggers are possible:

1) Last ~~PU~~PDU in buffer.

The sender triggers a poll when the last ~~PU~~PDU available for transmission is transmitted.

2) Last ~~PU~~PDU in retransmission buffer.

The sender triggers a poll when the last ~~PU~~PDU to be retransmitted is transmitted.

3) Poll timer.

The timer Timer_Poll is started when the successful or unsuccessful transmission of a PDU containing a poll is indicated by lower layer (in UE) or a PDU containing a poll is submitted to lower layer (in UTRAN) and if the criterion for stopping the timer has not occurred before the timer Timer_Poll expires a new poll is triggered.

4) Every Poll_~~PU~~PDU ~~PU~~PDU.

The sender triggers a poll every Poll_~~PU~~PDU ~~PU~~PDU. Both retransmitted and new ~~PU~~PDUs shall be counted.

5) Every Poll_SDU SDU.

   The sender triggers a poll every Poll_SDU SDU.

6) Window based.

   The sender triggers a poll when it has reached Poll_Window% of the transmission window.

7) Timer based.

   The sender triggers a poll periodically.

Either the trigger "Last ~~PU~~PDU in buffer" and "Last ~~PU~~PDU in retransmission buffer" or "Timer based" can be chosen to avoid deadlock for every RLC entity. The network also controls if the poll prohibit function shall be used. The poll bit shall be set to 0 if the poll prohibit function is used and the timer Timer_Poll_Prohibit is active. If a poll was triggered during the prohibit time defined in subclause 9.5 b) (Timer_Poll_Prohibit), the poll shall be delayed until the timer expires. Only one poll shall be transmitted when the timer expires even if several polls were triggered during the prohibit time.This function has higher priority than any of the above mentioned triggers.

9.7.2 STATUS transmission for acknowledged mode

The receiver of AMD PDUs transmits status reports (each status report consists of one or several STATUS PDUs) to the sender in order to inform about which ~~PU~~PDUs that have been received and not received. There are several triggers for sending a status report. The network (RRC) controls which triggers should be used for each RLC entity, except for one, which is always present. The receiver shall always send a status report when receiving a poll request. Except for that trigger following triggers are configurable:

1) Detection of missing ~~PU~~PDU(s).

   If the receiver detects one or several missing ~~PU~~PDUs it shall trigger the transmission of a status report to the sender.

2) Timer based STATUS transfer.

   The receiver triggers the transmission of a status report periodically to the sender. The timer Timer_Status_Periodic controls the time period.

3) The EPC mechanism.

   The timer Timer_EPC is started and the state variable VR(EP) is set when the successful or unsuccessful transmission of the first STATUS PDU of a status report is indicated by lower layer (in UE) or the first STATUS PDU of a status report is submitted to lower layer (in UTRAN). If not all ~~PU~~PDUs requested for retransmission have been received before the variable VR(EP) has reached zero, a new status report is transmitted to the peer entity. A more detailed description of the EPC mechanism is given in subclause 9.7.4.

There are two functions that can prohibit the receiver from sending a status report. The network (RRC) controls which functions should be used for each RLC entity. If any of the following functions is used the sending of the status report shall be delayed, even if any of the triggering conditions above are fulfilled:

1) STATUS prohibit.

   The Timer_Status_Prohibit is started when the successful or unsuccessful transmission of the last STATUS PDU of a status report is indicated by lower layer (in UE) or the last STATUS PDU of a status report is submitted to lower layer (in UTRAN). The prohibit time is calculated from the time the last STATUS PDU of a status report is submitted to lower layer until the timer has expired. The receiving side is not allowed to transmit a status report during the prohibit time. If a status report was triggered during the prohibit time, the status report is transmitted after the prohibit time has expired. The receiver shall only send one status report, even if there are several triggers during the prohibit time. This timer only prohibits the transmission of status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. Status reports containing other SUFIs are not prohibited.

2) The EPC mechanism.

   If the EPC mechanism is active and the sending of a status report is triggered it shall be delayed until the EPC mechanism has ended. The receiver shall only send one status report, even if there are several triggers when the timer is active or the counter is counting down. This mechanism only prohibits the transmission of status reports containing any of the SUFIs LIST, BITMAP, RLIST or ACK. Status reports containing other SUFIs are not prohibited.

9.7.3 SDU discard function for acknowledged, unacknowledged, and transparent mode

The SDU discard function allows to discharge RLC PDU from the buffer on the transmitter side, when the transmission of the RLC PDU does not success for a long time. The SDU discard function allows to avoid buffer overflow. There will be several alternative operation modes of the RLC SDU discard function, and which discard function to use will be given by the QoS requirements of the Radio Access Bearer.

The following is a list of operation modes for the RLC SDU discard function.

Table 9.2: List of criteria's that control when to perform SDU discard

Operation mode	Presence
Timer based discard, with explicit signalling	Network controlled
Timer based discard, without explicit signalling	Network controlled
SDU discard after MaxDAT number of retransmissions	Network controlled

9.7.3.1 Timer based discard, with explicit signalling

This alternative uses a timer based triggering of SDU discard (Timer_Discard). This makes the SDU discard function insensitive to variations in the channel rate and provides means for exact definition of maximum delay. However, the SDU loss rate of the connection is increased as SDUs are discarded.

For every SDU received from a higher layer, timer monitoring of the transmission time of the SDU is started. If the transmission time exceeds a predefined value for a SDU in acknowledged mode RLC, this SDU is discarded in the transmitter and a Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded.

The MRW command is defined as a super-field in the RLC STATUS PDU (see subclause 9.2), and piggybacked to status information of transmissions in the opposite direction. If the MRW command has not been acknowledged by receiver, it will be retransmitted. Therefore, SDU discard variants requiring peer-to-peer signalling are only possible for full duplex connections.

9.7.3.2 Timer based discard, without explicit signalling

This alternative uses the same timer based trigger for SDU discard (Timer_Discard) as the one described in the subclause 9.7.3.1. The difference is that this discard method does not use any peer-to-peer signalling. This function is applied only for unacknowledged and transparent mode RLC and peer-to-peer signalling is never needed. The SDUs are simply discarded in the transmitter, once the transmission time is exceeded.

9.7.3.3 SDU discard after MaxDAT number of retransmissions

This alternative uses the number of retransmissions as a trigger for SDU discard, and is therefore only applicable for acknowledged mode RLC. This makes the SDU discard function dependent of the channel rate. Also, this variant of the SDU discard function strives to keep the SDU loss rate constant for the connection, on the cost of a variable delay. SDU discard

is triggered at the transmitter, and a MRW command is necessary to convey the discard information to the receiver, like in the timer based discard with explicit signalling.

9.7.4 The Estimated PDU Counter for acknowledged mode

The Estimated PDU Counter is a mechanism used for scheduling the retransmission of status reports in the receiver side. With this mechanism, the receiver will send a new status report in which it requests for ~~PU~~PDUs not yet received. The time between two subsequent status report retransmissions is not fixed, but it is controlled by both the timer Timer_EPC and the state variable VR(EP), which adapt this time to the round trip delay and the current bit rate, indicated in the TFI, in order to minimise the delay of the status report retransmission.

When a STATUS report is triggered by some mechanisms and it is submitted to lower layer (in UTRAN) or the successful or unsuccessful transmission of it is indicated by lower layer (in UE) to request for retransmitting one or more missing ~~PU~~PDUs, the variabe VR(EP) is set equal to the number of requested ~~PU~~PDUs. At least one requested ~~PU~~PDU is needed to activate the EPC mechanism. The variable VR(EP) is a counter, which is decremented every transmission time interval with the estimated number of ~~PU~~PDUs that should have been transmitted during that transmission time interval.

A special timer, called Timer_EPC, controls the maximum time that the variable VR(EP) needs to wait before it will start counting down. This timer starts immediately after a transmission of a retransmission request from the receiver (when the first STATUS PDU of the status report is submitted to lower layer (in UTRAN) or the successful or unsuccessful transmission of it is indicated by lower layer(in UE)). The timer Timer_EPC typically depends on the roundtrip delay, which consists of the propagation delay, processing time in the transmitter and receiver and the frame structure. This timer can also be implemented as a counter, which counts the number of 10 ms radio frames that could be expected to elapse before the first requested AMD PDU is received.

If not all of these requested ~~PU~~PDUs have been received correctly when VR(EP) is equal to zero, a new status report will be transmitted and the EPC mechanism will be reset accordingly. The timer Timer_EPC will be started once more when the first STATUS PDU of the status report is submitted to lower layer (in UTRAN) or the successful or unsuccessful transmission of it is indicated by lower layer (in UE). If all of the requested ~~PU~~PDUs have been received correctly, the EPC mechanism ends.

~~9.7.5 Multiple payload units in an RLC PDU for acknowledged mode~~

~~The possibility to include multiple payload units (PU) into one RLC AMD PDU is part of the service capabilities of a UE in acknowledged mode. For Release 99, there shall be only one PU per AMD PDU.~~

~~A payload unit is the smallest unit that can be separately addressed for retransmission and is of fixed size, containing data and optionally, length indicators and/or padding. The padding space of a PU can be used to piggyback STATUS PDUs.~~

~~The size of the PU is set by the RRC~~.

9.7.5~~6~~ Local Suspend function for acknowledged and unacknowledged mode

The higher layer (RRC) may suspend the RLC entity. The CRLC-SUSPEND-Req indicates this request. The RLC entity shall, when receiving this request, not send RLC PDUs with SN>=VT(S)+N for AM and SN>=VT(US)+N for UM ,where N is given by the CRLC_SUSPEND-Req primitive. The RLC entity shall acknowledge the CRLC-SUSPEND-Req ordering a suspend with a CRLC-SUSPEND-Conf with the current value of VT(S) for AM and VT(US) for UM. The suspend state is left when a CRLC-RESUME-Req primitive indicating resume is received.

9.7.6~~7~~ RLC stop, RLC Continue function

The higher layer may stop the RLC entity. The stop parameter in the CRLC-CONFIG-Req primitive indicates this request. The RLC entity shall, when receiving this request, not submit any RLC PDUs to lower layer or receive any RLC PDUs. The data transmission and reception is continued when the continue parameter in the CRLC-CONFIG-Req primitive is received. If the continue parameter is received when the RLC entity is not stopped, no action shall be taken.

When the RLC entity is stopped, the RLC timers are not affected. triggered polls and status transmissions are delayed until the RLC entity is continued.

10 Handling of unknown, unforeseen and erroneous protocol data

The list of error cases is reported below:

a) Inconsistent state variables.

If the RLC entity receives a PDU including "erroneous Sequence Number", state variables between peer entities may be inconsistent. Following shows "erroneous Sequence Number" examples:

- Each Sequence Number of missing ~~PU~~PDU informed by SUFI LIST, BITMAP or RLIST is not within the value between "Acknowledge state variable(VT(A))" and "Send state variable(VT(S)) – 1", and

-- LSN of SUFI ACK is not within the value between "Acknowledge state variable(VT(A))" and "Send state variable(VT(S))".

In case of error situations the following actions are foreseen:

1) RLC entity should use RESET procedure in case of an unrecoverable error.

2) RLC entity should discard invalid PDU.

3) RLC entity should notify upper layer of unrecoverable error occurrence in case of failed retransmission.

b) Inconsistent status indication of a ~~PU~~PDU

If a received STATUS PDU indicates different status for the same ~~PU~~PDU, then the transmitter shall discard the STATUS PDU.

11 Elementary procedures

11.1 Transparent mode data transfer procedure

11.1.1 Purpose

The transparent mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in transparent mode. Figure 11.1 below illustrates the elementary procedure for transparent mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

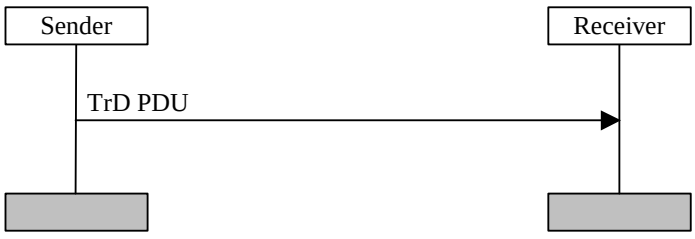


Figure 11.1: Transparent mode data transfer procedure

11.1.2 Initiation

The sender initiates this procedure upon a request of transparent mode data transfer from higher layer. When the sender is in data transfer ready state it shall put the data received from the higher layer into TrD PDUs. If required RLC shall perform segmentation.

Channels that can be used are DTCH, CCCH (uplink only), SHCCH (uplink only), BCCH and PCCH. The type of logical channel depends on if the RLC entity is located in the user plane (DTCH) or in the control plane (CCCH/BCCH/SHCCH/PCCH). One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can not be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

11.1.2.1 TrD PDU contents to set

The TrD PDU includes a complete SDU or a segment of an SDU. How to perform the segmentation is decided upon when the service is established. No overhead or header is added, instead segmentation is done based on which of the transport formats of the transport channel that will be used. A particular transport format informs the receiver how the segmentation was performed.

11.1.3 Reception of TrD PDU

Upon reception of a TrD PDU, the receiving entity reassembles (if segmentation was performed) the PDUs into RLC SDUs. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

11.1.4 Abnormal cases

11.1.4.1 Undefined SDU size at receiver

If the TrD PDUs are reassembled to a SDU which have a size that is not allowed the SDU shall be discarded.

11.1.4.2 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard all PDUs that contain segments of the associated SDU.

11.2 Unacknowledged mode data transfer procedure

11.2.1 Purpose

The unacknowledged mode data transfer procedure is used for transferring data between two RLC peer entities, which are operating in unacknowledged mode. Figure 11.2 below illustrates the elementary procedure for unacknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.2: Unacknowledged mode data transfer procedure

11.2.2 Initiation

The sender initiates this procedure upon a request of unacknowledged mode data transfer from higher layer.

When the sender is in data transfer ready state it shall segment the data received from the higher layer into PDUs.

Channels that can be used are DTCH, DCCH, CCCH (downlink only), CTCH, SHCCH (downlink only). The type of logical channel depends on if the RLC entity is located in the user plane (DTCH, CTCH) or in the control plane (DCCH/CCCH(downlink only)/SHCCH(downlink only)). One or several PDUs may be transmitted in each transmission time interval (TTI). For each TTI, MAC decides which PDU size shall be used and how many PDUs shall be transmitted.

The VT(US) state variable shall be updated for each UMD PDU that is transmitted.

11.2.2.1 UMD PDU contents to set

The Sequence Number field shall be set equal to VT(US).

The Extension bit shall be set to 1 if the next field is a length indicator field, otherwise it shall be set to zero.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the header fields and the end of the segment. If padding is needed, another LI field set to only 1's shall be added unless the padding size is one octet for PDUs with 15-bit LIs. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

11.2.3 Reception of UMD PDU

Upon reception of a UMD PDU, the receiver shall update VR(US) state variable according to the received PDU(s).

The PDUs are reassembled into RLC SDUs. If a PDU with sequence number < VR(US) is missing then all SDUs that have segments in this PDU shall be discarded. RLC delivers the RLC SDUs to the higher layer through the UM-SAP.

11.2.4 Abnormal cases

11.2.4.1 Length Indicator value 1111110

Upon reception of an UMD PDU that contains Length Indicator value 1111110 or 111111111111110 ("piggybacked STATUS PDU", in case 7bit or 15 bit Length Indicator field is used, respectively) the receiver shall discard that UMD PDU. This Length Indicator value is not used in unacknowledged mode data transfer.

11.2.4.2 Invalid length indicator value

If the length indicator of a PDU has a value that is larger than the PDU size – the number of octets containing LIs in the PDU – 1 and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PDU shall be discarded and treated as a missing PDU.

11.2.4.3 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard the associated SDU.The next UMD PDU shall carry the first segment of the oldest SDU not discarded. The state variable VT(US) shall be updated so that the receiver can detect at least one missing PDUs. To avoid that the receiver should discard one extra SDU, a LI field shall be added in the first PDU transmitted after a Discard Operation. The value of the LI field shall be either the value indicating that the previous SDU filled exactly the previous RLC PDU or the value indicating that the first data octet in this RLC PDU is the first octet of a RLC SDU.

11.3 Acknowledged mode data transfer procedure

11.3.1 Purpose

The acknowledged mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in acknowledged mode. Figure 11.3 below illustrates the elementary procedure for acknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.3: Acknowledged mode data transfer procedure

11.3.2 Initiation

The sender initiates this procedure upon a request of acknowledged mode data transfer from higher layer or upon retransmission of ~~PU~~PDUs. Retransmitted ~~PU~~PDUs have higher priority than ~~PU~~PDUs transmitted for the first time.

The sender is only allowed to retransmit ~~PU~~PDUs that have been indicated missing by the receiver. An exception is the ~~PU~~PDU with SN VT(S)-1 which can be retransmitted. In addition, a ~~PU~~PDU that has not yet been acknowledged, may be retransmitted if Configured_Tx_Window_Size is less than 2048.

RLC shall segment the data received from the higher layer into ~~PU~~AMD PDUs. ~~When the sender is in data transfer ready state one or several PUs are included in one AMD PDU, which is sent to the receiver.~~ The PDUs shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane. One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can ~~not~~ be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

The VT(DAT) state variables shall be updated for each AMD PDU that is transmitted. The PDU shall not ~~include any PU with~~have a Sequence Number ≥ VT(MS), except ~~the PU with~~for the sequence number VT(S)-1; a PDU with this sequence number ~~which~~ may be ~~included~~ sent also when VT(S) ≥ VT(MS).

If the poll bit is set in any of the AMD PDUs and the timer Timer_Poll shall be used, the sender shall start the timer Timer_Poll when the successful or unsuccessful transmission of a PDU with the set poll bit is indicated by lower layer (in UE) or submitted to lower layer (in UTRAN).

If timer based SDU discard is used, the timer Timer_Discard shall be started when the RLC entity receives an SDU from higher layer. One timer is used for each SDU that is received from higher layer.

If the trigger for polling, "Every Poll_~~PU~~PDU ~~PU~~PDU", is used, the VT(~~PU~~PDU) shall be increased by 1 for each ~~PU~~PDU that is transmitted.

If the trigger for polling, "Every Poll_SDU SDU", is used, the VT(SDU) shall be increased by 1 for each SDU that is transmitted.

11.3.2.1 AMD PDU contents to set

If the PDU is transmitted for the first time, the Sequence Number field shall be set equal to VT(S) and VT(S) shall be updated.

The setting of the Polling bit is specified in subclause 11.3.2.1.1.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the header fields and the end of the segment. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

How to perform the segmentation of a SDU is specified in subclause 11.3.2.1.2.

11.3.2.1.1 Setting of the Polling bit

The Polling bit shall be set to 1 if any of following conditions are fulfilled except when the poll prohibit function is used and the timer Timer_Poll_Prohibit is active (the different triggers are described in 9.7.1):

1) Last ~~PU~~PDU in buffer is used and the last ~~PU~~PDU available for transmission is transmitted.
2) Last ~~PU~~PDU in retransmission buffer is used and the last ~~PU~~PDU to be retransmitted is transmitted.
3) Poll timer is used and timer Timer_Poll has expired.
4) Every Poll_~~PU~~PDU ~~PU~~PDU is used and when VT(~~PU~~PDU)=Poll_~~PU~~PDU.
5) Every Poll_SDU is used and VT(SDU)=Poll_SDU and the PDU contains the last segment of that SDU.
6) Window based polling is used, , and J≥Poll_Window, where J is defined in subclause 9.6.

.

7) Timer based polling is used and Timer_Poll_Periodic has expired.
8) Poll prohibit shall be used, the timer Timer_Poll_Prohibit has expired and one or several polls were prohibited during the time Timer_Poll_Prohibit was active.

11.3.2.1.2 Segmentation of a SDU

Upon reception of a SDU, RLC shall segment the SDU to fit into the fixed size of a ~~PU~~PDU. The segments are inserted in the data field of a ~~PU~~PDU. A length indicator shall be added to each ~~PU~~PDU that includes a border of an SDU, i.e. if a ~~PU~~PDU does not contain an LI, the SDU continues in the next ~~PU~~PDU. The length indicator indicates where the border occurs in the ~~PU~~PDU. The data after the indicated border can be either a new SDU, padding or piggybacked information. If padding or piggybacking is added another LI shall be added unless the padding size is one octet for PDUs with 15-bit LIs, see subclauses 9.2.2.8 and 9.2.2.9.

11.3.3 Reception of AMD PDU by the receiver

Upon reception of a AMD PDU, the receiver shall update VR(R), VR(H) and VR(MR) state variables according to the received ~~PU~~PDU~~(s)~~.

If ~~any of the~~a received ~~PU~~PDU~~s~~ includes a Polling bit set to 1, the STATUS PDU transfer procedure shall be initiated.

If the detection of missing ~~PU~~PDU(s) shall be used and the receiver detects that a ~~PU~~PDU is missing, the receiver shall initiate the STATUS PDU transfer procedure.

11.3.4 Abnormal cases

11.3.4.1 Timer_Poll timeout

Upon expiry of the Timer_Poll, the sender shall retransmit the poll. The poll can be retransmitted in either a new PDU or a retransmitted PDU.

11.3.4.2 Receiving a ~~PU~~PDU outside the receiving window

Upon reception of a ~~PU~~PDU with SN<VR(R) or SN≥VR(MR), the receiver shall discard the ~~PU~~PDU. The poll bit shall be considered even if a complete PDU is discarded.

11.3.4.3 Timer_Discard timeout

11.3.4.3.1 SDU discard with explicit signalling

Upon expiry of Timer_Discard, the sender shall initiate the SDU discard with explicit signalling procedure.

11.3.4.4 VT(DAT) ≥ MaxDAT

If SDU discard after MaxDAT number of retransmission is used and VT(DAT) ≥ MaxDAT for any ~~PU~~PDU, the sender shall initiate the SDU discard with explicit signalling procedure for the SDUs to which the ~~PU~~PDU with VT(DAT) ≥MaxDAT belongs.

If the SDU discard is not used, the sender shall initiate the RLC reset procedure when VT(DAT) ≥ MaxDAT.

11.3.4.5 Invalid length indicator value

If the length indicator of a ~~PU~~PDU has a value that is larger than the ~~PU~~PDU size – the number of octets containing LIs in the ~~PU~~PDU and is not one of the predefined values listed in the table of subclause 9.2.2.8, the ~~PU~~PDU shall be discarded and treated as a missing ~~PU~~PDU.

11.4 RLC reset procedure

11.4.1 Purpose

The RLC reset procedure is used to reset two RLC peer entities, which are operating in acknowledged mode. Figure 11.4 below illustrates the elementary procedure for a RLC reset. The sender can be either the UE or the network and the receiver is either the network or the UE. During the reset procedure the hyper frame numbers (HFN) in UTRAN and UE are synchronised. Two HFNs used for ciphering needs to be synchronised, DL HFN in downlink and UL HFN in uplink. In the reset procedure, the highest UL HFN and DL HFN used by the RLC entity are exchanged between UE and UTRAN. After the reset procedure is terminated, the UL HFN and DL HFN shall be increased with one in both UE and UTRAN, and the updated HFN values shall be used after the reset procedure.

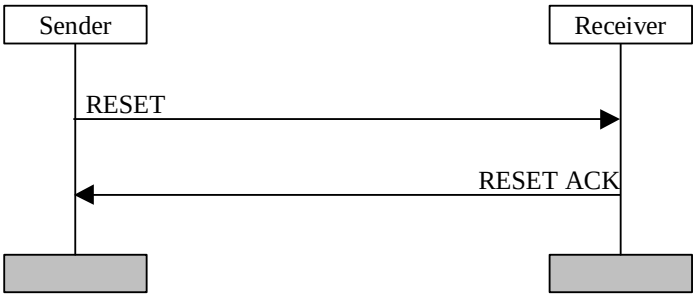


Figure 11.4: RLC reset procedure

11.4.2 Initiation

The procedure shall be initiated when a protocol error occurs.

The sender sends the RESET PDU when it is in data transfer ready state and enters reset pending state. The sender shall start the timer Timer_RST and increase VT(RST) with 1. The RESET PDU shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

The RESET PDU has higher priority than data PDUs.

When a reset procedure has been initiated it can only be ended upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU, i.e., a reset procedure is not interrupted by the reception of a RESET PDU from the peer entity.

11.4.2.1 RESET PDU contents to set

The size of the RESET PDU shall be equal to one of the allowed PDU sizes. The hyper frame number indicator field (HFNI) shall be set equal to the currently used HFN (DL HFN when the RESET is sent by UTRAN or UL HFN when the RESET is sent by the UE). The RSN field shall indicate the sequence number of the RESET PDU. This sequence number is incremented every time a new RESET PDU is transmitted, but not when a RESET PDU is retransmitted.

11.4.3 Reception of the RESET PDU by the receiver

Upon reception of a RESET PDU the receiver shall respond with a RESET ACK PDU. The receiver resets the state variables in 9.4 to their initial value and resets configurable parameters to their configured value. Both the transmitter and receiver side of the AM RLC entity are reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded.

When a RESET PDU is received, the receiver shall set the HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) equal to the HFNI field in the received RESET PDU.

The RESET ACK PDU shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

The RESET ACK PDU has higher priority than data PDUs.

11.4.3.1 RESET ACK PDU contents to set

The size of the RESET ACK PDU shall be equal to one of the allowed PDU sizes. The RSN field shall always be set to the same value as in the corresponding RESET PDU. The hyper frame number indicator field (HFNI) shall be set equal to the currently used HFN (DL HFN when the RESET ACK is sent by UTRAN or UL HFN when the RESET ACK is sent by the UE).

11.4.4 Reception of the RESET ACK PDU by the sender

When the sender is in reset pending state and receives a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU the Timer_RST shall be stopped and the value of the HFN (DL HFN when the RESET ACK is received in UE or UL HFN when the RESET ACK is received in UTRAN) shall be set equal to the HFNI field in the received RESET ACK PDU. The sender resets the state variables in 9.4 to their initial value and resets configurable parameters to their configured value. Both the transmitter and receiver side of the AM RLC entity is reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded.

The sender shall enter data transfer ready state.

Upon reception of a RESET ACK PDU with a different RSN value as in the corresponding RESET PDU the RESET ACK PDU is discarded.

Upon reception of a RESET ACK PDU in data transfer ready state the RESET ACK PDU is discarded.

11.4.5 Abnormal cases

11.4.5.1 Timer_RST timeout

Upon expiry of Timer_RST the sender shall retransmit the RESET PDU and increase VT(RST) with 1. In the retransmitted RESET PDU the value of the RSN field shall not be incremented.

11.4.5.2 VT(RST) ≥ MaxRST

If VT(RST) becomes larger or equal to MaxRST the RRC layer shall be informed.

11.4.5.3 Reception of the RESET PDU by the sender

Upon reception of a RESET PDU in reset pending state, the sender shall respond with a RESET ACK PDU. The sender resets the state variables in 9.4 to their initial value, resets configurable parameters to their configured value. However, VT(RST) and Timer_RST are not reset. Both the transmitter and receiver side of the AM RLC entity are reset. All RLC PDUs in the AM RLC receiver shall be discarded. The RLC SDUs in the AM RLC transmitter that were transmitted before the reset shall be discarded.The hyper frame number, HFN (DL HFN when the RESET is received in UE or UL HFN when the RESET is received in UTRAN) is set equal to the HFNI field in the received RESET PDU. The sender shall stay in the reset pending state. The sender shall enter data transfer ready state only upon reception of a RESET ACK PDU with the same RSN value as in the corresponding RESET PDU.

11.5 STATUS report transfer procedure

11.5.1 Purpose

The status report transfer procedure is used for transferring of status information between two RLC peer entities, which are operating in acknowledged mode. Figure 11.5 below illustrates the elementary procedure for status report transfer. A status report consists of one or several STATUS PDUs. The receiver is the receiver of AMD PDUs and it is either the UE or the network and the sender is the sender of AMD PDUs and it is either the network or the UE.

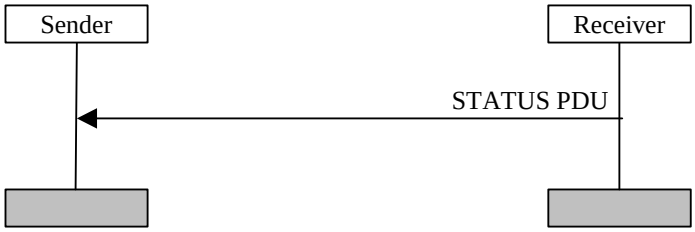


Figure 11.5: Status report transfer procedure

11.5.2 Initiation

The receiver in any of the following cases initiates this procedure:

1) The poll bit in a received AMD PDU is set to 1.

2) Detection of missing ~~PU~~PDUs is used and a missing ~~PU~~PDU is detected.

3) The timer based STATUS transfer is used and the timer Timer_Status_Periodic has expired.

The receiver shall transmit a status report on the DCCH logical channel if the receiver is located in the control plane and on the DTCH if it is located in the user plane. Separate logical channels can be assigned for AMD PDU transfer and for Control PDU transfer.

The STATUS PDUs have higher priority than data PDUs.

There are two functions that can prohibit the receiver from sending a status report. If any of following conditions are fulfilled the sending of the status report shall be delayed, even if any of the conditions above are fulfilled:

1) STATUS prohibit is used and the timer Timer_Status_Prohibit is active.

   The status report shall be transmitted after the Timer_Status_Prohibit has expired. The receiver shall send only one status report, even if there are several triggers when the timer is active. The rules for when the timer Timer_status_Prohibit is active are defined in subclause 9.5.

2) The EPC mechanism is used and the timer Timer_EPC is active or VR(EP) is counting down.

   The status report shall be transmitted after the VR(EP) has reached 0. The receiver send only one status report, even if there are several triggers when the timer is active or the counter is counting down. The rules for when the timer Timer_EPC is active are defined in subclause 9.5.

If the timer based STATUS transfer shall be used and the Timer_Status_Periodic has expired it shall be restarted.

If the EPC mechanism shall be used, the timer Timer_EPC shall be started and the VR(EP) shall be set equal to the number ~~PU~~PDUs requested to be retransmitted.

11.5.2.1 Piggybacked STATUS PDU

It is possible to piggyback a STATUS PDU on an AMD PDU. If a PDU includes padding, a piggybacked STATUS PDU can be inserted instead of the padding. The sending of a piggybacked STATUS PDU follows the same rules as the sending of an ordinary STATUS PDU.

11.5.2.2 STATUS PDU contents to set

The size of the STATUS PDU shall be equal to one of the allowed PDU sizes. The information that needs to be transmitted in a status report can be split into several STATUS PDUs if one STATUS PDU does not accommodate all the information. A SUFI can not be split into several STATUS PDUs. Indication of the same ~~PU~~ PDU shall not be given in more than one STATUS PDU of a STATUS report, but the ACK SUFI can be present in more than one STATUS PDU of a status report.

Which SUFI fields to use is implementation dependent, but the status report shall include information about ~~PU~~PDUs that have been received and information about all ~~PU~~PDUs detected as missing. No information shall be given for ~~PU~~PDUs with SN≥VR(H), i.e. ~~PU~~PDUs that have not yet reached the receiver.

Padding shall be inserted if the SUFI fields do not fill an entire STATUS PDU. If the PDU contains padding the last SUFI field shall be either an ACK SUFI or a NO_MORE SUFI. If there is no padding in the STATUS PDU, NO_MORE SUFI or ACK SUFI does not need to be included in the STATUS PDU.

11.5.3 Reception of the STATUS PDU by the sender

The sender shall upon reception of the STATUS PDU/piggybacked STATUS PDU update the state variables VT(A) and VT(MS) according to the received STATUS PDU/piggybacked STATUS PDU.

If the STATUS PDU includes negatively acknowledged ~~PU~~PDUs, the acknowledged data transfer procedure shall be initiated and the ~~PU~~PDUs shall be retransmitted. If a ~~PU~~PDU is indicated as missing more thaen once in a STATUS PDU, the ~~PU~~PDU shall be retransmitted only once. Retransmitted ~~PU~~PDUs have higher priority than new ~~PU~~PDUs.

11.5.4 Abnormal cases

11.5.4.1 VR(EP) reaches zero and the requested ~~PU~~PDUs have not been received

If the EPC mechanism is used and VR(EP) has reached zero and not all ~~PU~~PDUs requested for retransmission have been received, the receiver shall:

- Retransmit the status report. The retransmitted status report may contain new or different SUFI fields in order to indicate that some ~~PU~~PDUs have been received and that some new have been lost.

11.6 SDU discard with explicit signalling procedure

11.6.1 Purpose

An SDU can be discarded with explicit signalling when MaxDAT number of retransmissions is reached or the transmission time exceeds a predefined value (Timer_Discard) for a SDU in acknowledged mode RLC. Move Receiving Window (MRW) command is sent to the receiver so that AMD PDUs carrying that SDU are discarded in the receiver and the receiver window is updated accordingly. Note that when the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded.

The MRW command is defined as a super-field in the RLC STATUS PDU, and can be piggybacked to status information of transmissions in the opposite direction.

Figure 11.6 below illustrates the elementary procedure for SDU discard with explicit signalling. The sender is the sender of AMD PDUs and it is either the UE or the network and the receiver is the receiver of AMD PDUs and it is either the network or the UE.

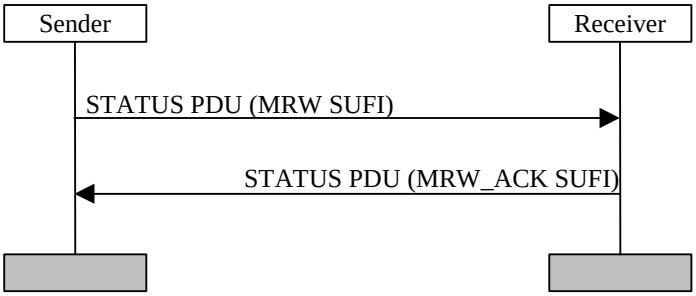


Figure 11.6: SDU discard with explicit signalling

11.6.2 Initiation

This procedure is initiated by the sender when the following conditions are fulfilled:

1) Timer based SDU discard with explicit signalling is used, and Timer_Discard expires for an SDU.

2) SDU discard after MaxDAT number of retransmissions is used, and MaxDAT number of retransmissions is reached for an SDU.

The sender shall discard all ~~PU~~PDUs that contain segments of the associated SDUs. If the concatenation function is active, PDUs carrying segments of other SDUs that have not timed out shall not be discarded. VT(A) shall be updated when the procedure is terminated, and VT(S) shall be updated when a new MRW SUFI which includes $SN\_MRW_{LENGTH} \geq VT(S)$ is transmitted.

The sender shall transmit a status report on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane.

This status report is sent even if the 'STATUS prohibit' is used and the timer 'Timer_Status_Prohibit' or 'Timer_EPC' is active.

The STATUS PDUs have higher priority than data PDUs.

The sender shall start timer Timer_MRW. If a new SDU discard procedure is triggered when Timer_MRW is running, no new MRW SUFIs shall be sent before the current SDU discard procedure is terminated by one of the termination criteria.

11.6.2.1 Piggybacked STATUS PDU

It is possible to piggyback a STATUS PDU on an AMD PDU. If a PDU includes padding a piggybacked STATUS PDU can be inserted instead of the padding.

11.6.2.2 STATUS PDU contents to set

The size of the STATUS PDU shall be equal to one of the allowed PDU sizes. The discard information shall not be split into several MRW SUFIs.

The status report shall include the MRW SUFI, other SUFI fields can be used additionally. MRW SUFI shall convey information about the discarded SDU(s) to the receiver.

In order to discard a single SDU that ends in a PDU with SN≥ VT(A)+Configured_Tx_Window_Size, the LENGTH field in the MRW SUFI shall be set to "0000". If more then one SDU are discarded with the same MRW SUFI, at least the first discarded SDUs must end (i.e. the LI must be located) in a PDU with SN in the interval VT(A)≤ SN <VT(A)+Configured_Tx_Window_Size.

Padding shall be inserted if the SUFI fields do not fill the entire STATUS PDU. If the STATUS PDU contains padding the last SUFI field shall be either an ACK SUFI or a NO_MORE SUFI. If there is no padding in the STATUS PDU, NO_MORE SUFI or ACK SUFI does not need to be included in the STATUS PDU.

11.6.3 Reception of the STATUS PDU by the receiver

The receiver shall upon reception of the STATUS PDU/piggybacked STATUS PDU discard PUs and update the state variables VR(R), VR(H) and VR(MR) according to the received STATUS PDU/piggybacked STATUS PDU. Additionally the receiver should ~~indicate~~ inform the higher layers ~~of~~ about all of the discarded SDUs.

The receiver shall initiate the transmission of a status report containing an MRW_ACK SUFI.

In the MRW_ACK SUFI, SN_ACK shall be set to the new value of VR(R), updated after reception of the MRW SUFI. The N field in the MRW_ACK SUFI shall be set to N_{LENGTH} field in the received MRW SUFI if the SN_ACK field is equal to SN_MRW_{LENGTH}. Otherwise N shall be set to 0.

The last discarded data byte is the byte indicated by the N_{LENGTH}:th LI field of the ~~PU~~PDU with sequence number SN_MRW_{LENGTH} and the succeeding data byte is the first data byte to be reassembled after the discard. When $N_{LENGTH} = 0$, the first data byte of the ~~PU~~PDU with sequence number SN_MRW_{LENGTH} is the first data byte to be reassembled after the discard.

If the MRW SUFI indicates an SN_MRW_i outside the interval $VR(R) \leq SN\_MRW_i < VR(MR)$, the Rx shall consider the sequence number to be below VR(R), unless LENGTH="0000" or at least the first indicated SN_MRW_i in the MRW SUFI is within the interval $VR(R) \leq SN\_MRW_i < VR(MR)$, in which case the sequence number shall be considered to be above or equal to VR(MR).

11.6.4 Termination

The procedure is terminated in the sender in the following cases:

1. On the reception of a STATUS PDU which contains an MRW_ACK SUFI with $SN\_ACK > SN\_MRW_{LENGTH}$ and the N field is equal to zero.
2. On the reception of a STATUS PDU which contains an ACK SUFI indicating $VR(R) > SN\_MRW_{LENGTH}$
3. On reception of a STATUS PDU which contains an MRW_ACK with $SN\_ACK = SN\_MRW_{LENGTH}$ and N is equal to the N_{LENGTH} indicated in the transmitted MRW SUFI.

If one of the termination criteria above is fulfilled, Timer_MRW is stopped and the discard procedure is terminated.

When VT(MRW) reaches MaxMRW, the procedure is terminated and an RLC reset is performed.

11.6.5 Expiration of timer Timer_MRW

If Timer_MRW expires before the discard procedure is terminated, the MRW SUFI shall be retransmitted, VT(MRW) is incremented by one and Timer_MRW restarted. MRW SUFI shall be exactly the same as previously transmitted even though some new SDUs would have been discarded during the running of the Timer_MRW. If the retransmitted STATUS PDU contains other SUFIs than the MRW SUFI, the status information indicated by these SUFIs shall be updated.

11.6.6 Abnormal cases

11.6.6.1 Obsolete/corrupted MRW command

If the MRW command contains outdated information about the receiver window (receiver window already moved further than MRW command is indicating), the MRW command shall be discarded and a status report containing SUFI MRW_ACK shall be transmitted indicating the value of VR(R) and the N field shall be set to zero.

11.6.6.2 VT(MRW) equals MaxMRW

If the number of retransmission of a MRW command (i.e. VT(MRW)) reaches MaxMRW, an error indication shall be passed to RRC and RESET procedure shall be performed.

11.6.6.3 Reception of obsolete MRW_ACK

The received MRW_ACK shall be discarded in the following cases.

1. If timer Timer_MRW is not active.
2. If the SN_ACK field in the received $MRW\_ACK < SN\_MRW_{LENGTH}$ in the transmitted MRW SUFI.
3. If the SN_ACK field in the received MRW_ACK is equal to the SN_MRW_{LENGTH} in the transmitted MRW SUFI and the N field in the received MRW_ACK is not equal to the N_{LENGTH} field in the transmitted MRW SUFI

4. If the SN_ACK field in the received MRW_ACK > SN_MRW_{LENGTH} in the transmitted MRW SUFI and the N field in the received MRW_ACK is not equal to zero.

11.7 Ciphering

The ciphering function is performed in RLC, according to the following rules if a radio bearer is using a non-transparent RLC mode (AM or UM). The data unit that is ciphered, depends on the transmission mode as described below.

- For RLC UM mode, the ciphering unit is the UMD PDU excluding the first octet, i.e. excluding the RLC UM PDU header. This is shown below in Figure 11.7.1.

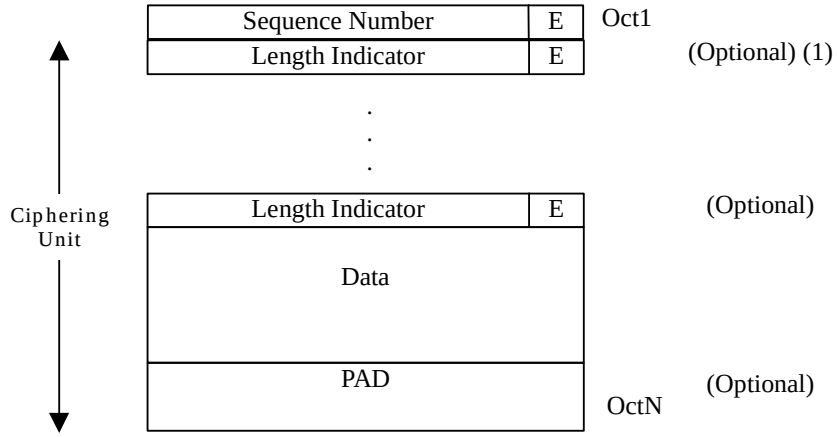


Figure 11.7.1: Ciphering unit for a UMD PDU

- For RLC AM mode, the ciphering unit is the AMD PDU excluding the two first octets, i.e. excluding the RLC AM PDU header. This is shown below in Figure 11.7.2.

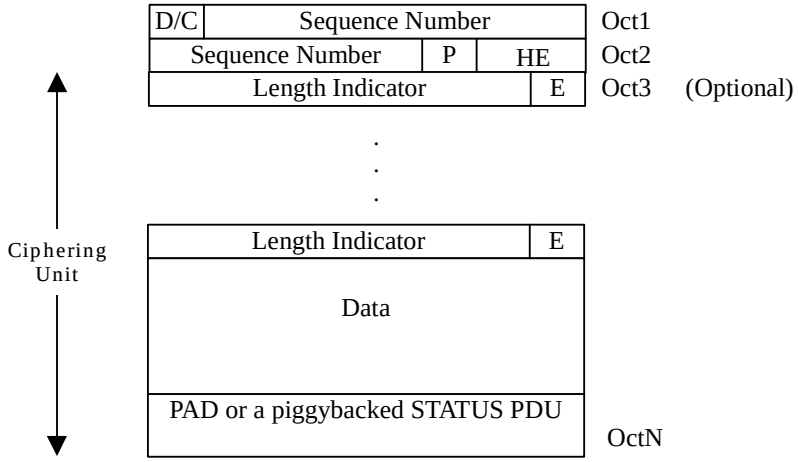


Figure 11.7.1: Ciphering unit for a AMD PDU

The ciphering algorithm and key to be used are configured by upper layers [8] and the ciphering method shall be applied as specified in [10].

The parameters that are required by RLC for ciphering are defined in [10] and are input to the ciphering algorithm. The parameters required by RLC which are provided by upper layers [8] are listed below:

- RLC AM HFN (Hyper frame number for radio bearers that are mapped onto RLC AM)

- RLC UM HFN (Hyper frame number for radio bearers that are mapped onto RLC UM)
- BEARER (Radio Bearer ID)
- CK (Ciphering Key)

[...]

~~Annex B (informative):~~
~~Pseudo code describing AMD PDU header Compression~~

~~The following Pseudo-Code is an example of algorithm to describe the exact Header Compression Operation that takes place when several PUs are packed into one RLC PDU.~~

```
/* Prior to calling this procedure it must be checked that <pus_in_pdu> consecutive PU:s
 are to be transmitted (or there is padding in the end)*/

Compress_PDU (pus_in_pdu, pu_size) {

 li_addition = 0; // reset the variable that counts data in full pu:s

 Loop through pus_in_pdu {

 d_e_flag = E-flag for this PU;

 If (d_e_flag == FALSE) {

 Append PU data to PDU data; // complete PU is SDU-data
 li_addition += pu_size; // to be added to the next LI

 } else { // E-flag is TRUE, so LI-field(s) exist

 Previous E-flag in PDU = TRUE; // Either in PDU header or pdu_li_vector;

 j = 0; // reset LI-counter for this PU
 pu_data_size = 0; // reset data size counter for this PU

 Loop until (d_e_flag == FALSE) {

 d_li = next LI; // in octet j of PU;
 d_e_flag = next E_FLAG; // in octet j of PU;

 if (d_li is not PADDING) {

 pu_data_size += d_li; // to keep track of data segment size in this PU);
 d_li += li_addition; // to add data from previous PU:s to LI-value);
 li_addition = 0; // reset li_addition;

 }

 Append (d_li + d_e_flag) to pdu_li_vector;
```

```
 j++; // go to next li_octet, if d_e_flag is TRUE);
 } /* end-of-loop (exit when d_e_flag is TRUE) */
 Append pu_data_size segments starting from j to RLC-PDU data;
 } /* end-of e-flag == TRUE */
 } /* end-of loop through PU:s in PDU */
} /* end-of Compress_PDU */
```

3GPP TSG-RAN WG2 Meeting #18
Sophia-Antepolis, France, 19 - 23 Feb. 2001

R2-010721

CR-Form-v3

CHANGE REQUEST

⌘ **25.322** **CR** **118** ⌘ rev **r2** ⌘ Current version: **3.5.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network **X** Core Network ☐

Title:	⌘	Padding Blocks and TFC Selection pre-empting
Source:	⌘	TSG-RAN WG2
Work item code:	⌘	***Date:*** ⌘ 10 Jan. 2001
Category:	⌘	**F** ***Release:*** ⌘ R99

Use one of the following categories:
F *(essential correction)*
A *(corresponds to a correction in an earlier release)*
B *(Addition of feature),*
C *(Functional modification of feature)*
D *(Editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	It was decided at the RLC ad hoc meeting that the creation of RLC padding blocks would not be allowed at all when the TFCS provided by UTRAN followed a set of guidelines described in 25.331. In order to allow some level of functionality even when the guidelines are not followed this restriction is introduced in the TFC Selection in MAC. Now it is necessary to introduce a definition of what RLC padding blocks are. Also, in order to guaranty that the selection of a TFC by MAC does not result in the introduction of padding because of some race condition between TFC selection and the SDU discard function, it was clarified that SDU discard will not pre-empt TFC selection.
Summary of change:	⌘	1. Padding PDUs are defined in UM 2. Padding PDUs are defined in AM 3. It is now explicitly stated that it is allowed to wait after the TFC selection procedure has been performed before discarding SDUs in all three modes.
Consequences if not approved:	⌘	This is linked to CR66 to MAC. Unless the padding PDUs are defined in RLC, that CR cannot stand by itself.

Clauses affected:	⌘	11.1.4.2, 11.2.2, 11.2.4.3, 11.3.2, 11.3.4.3.1
Other specs Affected:	⌘	☐ Other core specifications ⌘ ☐ Test specifications ☐ O&M Specifications
Other comments:	⌘	

How to create CRs using this form:
Comprehensive information and tips about how to create CRs can be found at: http://www.3gpp.org/3G_Specs/CRs.htm.
Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://www.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2000-09 contains the specifications resulting from the September 2000 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

Transparent mode data transfer procedure

11.1.1 Purpose

The transparent mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in transparent mode. Figure 11.1 below illustrates the elementary procedure for transparent mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.1: Transparent mode data transfer procedure

11.1.2 Initiation

The sender initiates this procedure upon a request of transparent mode data transfer from higher layer. When the sender is in data transfer ready state it shall put the data received from the higher layer into TrD PDUs. If required RLC shall perform segmentation.

Channels that can be used are DTCH, CCCH (uplink only), SHCCH (uplink only), BCCH and PCCH. The type of logical channel depends on if the RLC entity is located in the user plane (DTCH) or in the control plane (CCCH/BCCH/SHCCH/PCCH). One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can not be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

11.1.2.1 TrD PDU contents to set

The TrD PDU includes a complete SDU or a segment of an SDU. How to perform the segmentation is decided upon when the service is established. No overhead or header is added, instead segmentation is done based on which of the transport formats of the transport channel that will be used. A particular transport format informs the receiver how the segmentation was performed.

11.1.3 Reception of TrD PDU

Upon reception of a TrD PDU, the receiving entity reassembles (if segmentation was performed) the PDUs into RLC SDUs. RLC delivers the RLC SDUs to the higher layer through the Tr-SAP.

11.1.4 Abnormal cases

11.1.4.1 Undefined SDU size at receiver

If the TrD PDUs are reassembled to a SDU which have a size that is not allowed the SDU shall be discarded.

11.1.4.2 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard ~~all PDUs that contain segments of~~ the associated SDU. In the case where the TFC selection exchange has been initiated by sending the RLC Entity Info parameter to MAC, the UE may wait until after it provides MAC with the requested set of PDUs before discarding the afore-mentioned SDU.

11.2 Unacknowledged mode data transfer procedure

11.2.1 Purpose

The unacknowledged mode data transfer procedure is used for transferring data between two RLC peer entities, which are operating in unacknowledged mode. Figure 11.2 below illustrates the elementary procedure for unacknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.2: Unacknowledged mode data transfer procedure

11.2.2 Initiation

The sender initiates this procedure upon a request of unacknowledged mode data transfer from higher layer.

When the sender is in data transfer ready state it shall segment the data received from the higher layer into PDUs.

Channels that can be used are DTCH, DCCH, CCCH (downlink only), CTCH, SHCCH (downlink only). The type of logical channel depends on if the RLC entity is located in the user plane (DTCH, CTCH) or in the control plane (DCCH/CCCH(downlink only)/SHCCH(downlink only)). One or several PDUs may be transmitted in each transmission time interval (TTI). For each TTI, MAC decides which PDU size shall be used and how many PDUs shall be transmitted.

The VT(US) state variable shall be updated for each UMD PDU that is transmitted.

A UMD PDU will be considered to be a padding PDU if it consists only of an RLC Header with one length indicator (indicating that the rest of the PDU is padding) and padding.

11.2.2.1 UMD PDU contents to set

The Sequence Number field shall be set equal to VT(US).

The Extension bit shall be set to 1 if the next field is a length indicator field, otherwise it shall be set to zero.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the header fields and the end of the segment. If padding is needed, another LI field set to only 1's shall be added unless the padding size is one octet for PDUs with 15-bit LIs. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

11.2.3 Reception of UMD PDU

Upon reception of a UMD PDU, the receiver shall update VR(US) state variable according to the received PDU(s).

The PDUs are reassembled into RLC SDUs. If a PDU with sequence number < VR(US) is missing then all SDUs that have segments in this PDU shall be discarded. RLC delivers the RLC SDUs to the higher layer through the UM-SAP.

11.2.4 Abnormal cases

11.2.4.1 Length Indicator value 1111110

Upon reception of an UMD PDU that contains Length Indicator value 1111110 or 111111111111110 ("piggybacked STATUS PDU", in case 7bit or 15 bit Length Indicator field is used, respectively) the receiver shall discard that UMD PDU. This Length Indicator value is not used in unacknowledged mode data transfer.

11.2.4.2 Invalid length indicator value

If the length indicator of a PDU has a value that is larger than the PDU size – the number of octets containing LIs in the PDU – 1 and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PDU shall be discarded and treated as a missing PDU.

11.2.4.3 SDU discard without explicit signalling

Upon expiry of the Timer_Discard on the sender side the sender shall discard the associated SDU. In the case where the TFC selection exchange has been initiated by sending the RLC Entity Info parameter to MAC, the UE may wait until after it provides MAC with the requested set of PDUs before discarding the afore-mentioned SDU.

The next UMD PDU shall carry the first segment of the oldest SDU not discarded. The state variable VT(US) shall be updated so that the receiver can detect at least one missing PDUs. To avoid that the receiver should discard one extra SDU, a LI field shall be added in the first PDU transmitted after a Discard Operation. The value of the LI field shall be either the value indicating that the previous SDU filled exactly the previous RLC PDU or the value indicating that the first data octet in this RLC PDU is the first octet of a RLC SDU.

11.3 Acknowledged mode data transfer procedure

11.3.1 Purpose

The acknowledged mode data transfer procedure is used for transferring of data between two RLC peer entities, which are operating in acknowledged mode. Figure 11.3 below illustrates the elementary procedure for acknowledged mode data transfer. The sender can be either the UE or the network and the receiver is either the network or the UE.

Figure 11.3: Acknowledged mode data transfer procedure

11.3.2 Initiation

The sender initiates this procedure upon a request of acknowledged mode data transfer from higher layer or upon retransmission of PUs. Retransmitted PUs have higher priority than PUs transmitted for the first time.

The sender is only allowed to retransmit PUs that have been indicated missing by the receiver. An exception is the PU with SN VT(S)-1 which can be retransmitted. In addition, a PU that has not yet been acknowledged, may be retransmitted if Configured_Tx_Window_Size is less than 2048.

RLC shall segment the data received from the higher layer into PUs. When the sender is in data transfer ready state one or several PUs are included in one AMD PDU, which is sent to the receiver. The PDUs shall be transmitted on the DCCH logical channel if the sender is located in the control plane and on the DTCH if it is located in the user plane. One or several PDUs may be transmitted in each transmission time interval (TTI) and MAC decides how many PDUs shall be transmitted in each TTI. In the UE, the PDUs that can not be transmitted in a TTI (i.e. MAC has indicated that some of the available PDUs can not be transmitted) shall be buffered according to the discard configuration set by RRC.

The VT(DAT) state variables shall be updated for each AMD PDU that is transmitted. The PDU shall not include any PU with Sequence Number ≥ VT(MS), except the PU with sequence number VT(S)-1 which may be included also when VT(S) ≥ VT(MS).

If the poll bit is set in any of the AMD PDUs and the timer Timer_Poll shall be used, the sender shall start the timer Timer_Poll when the successful or unsuccessful transmission of a PDU with the set poll bit is indicated by lower layer (in UE) or submitted to lower layer (in UTRAN).

If timer based SDU discard is used, the timer Timer_Discard shall be started when the RLC entity receives an SDU from higher layer. One timer is used for each SDU that is received from higher layer.

If the trigger for polling, "Every Poll_PU PU", is used, the VT(PU) shall be increased by 1 for each PU that is transmitted.

If the trigger for polling, "Every Poll_SDU SDU", is used, the VT(SDU) shall be increased by 1 for each SDU that is transmitted.

In AM, a PDU will be considered to be a padding PDU if it is:

- An AMD PDU consisting only of an RLC Header with one length indicator (indicating that the rest of the PDU is padding) and padding.
- A Status PDU consisting only of a NO_MORE SUFI.

11.3.2.1 AMD PDU contents to set

If the PDU is transmitted for the first time, the Sequence Number field shall be set equal to VT(S) and VT(S) shall be updated.

The setting of the Polling bit is specified in subclause 11.3.2.1.1.

One length indicator field shall be included for each end of a SDU that the PDU includes. The LI shall be set equal to the number of octets between the end of the header fields and the end of the segment. If the PDU is exactly filled with the last segment of a SDU and there is no room for an LI field, an LI field set to only 0's shall be included as the first length indicator in the following PDU. If the PDU with 15-bit LIs has only one octet left after filling with the last segment of a SDU and there is no room for a 15-bit LI field, an LI field set to the predefined value 11111111 1111011 shall be included in the next PDU.

How to perform the segmentation of a SDU is specified in subclause 11.3.2.1.2.

11.3.2.1.1 Setting of the Polling bit

The Polling bit shall be set to 1 if any of following conditions are fulfilled except when the poll prohibit function is used and the timer Timer_Poll_Prohibit is active (the different triggers are described in 9.7.1):

1) Last PU in buffer is used and the last PU available for transmission is transmitted.
2) Last PU in retransmission buffer is used and the last PU to be retransmitted is transmitted.
3) Poll timer is used and timer Timer_Poll has expired.
4) Every Poll_PU PU is used and when VT(PU)=Poll_PU.
5) Every Poll_SDU is used and VT(SDU)=Poll_SDU and the PDU contains the last segment of that SDU.
6) Window based polling is used, , and J≥Poll_Window, where J is defined in subclause 9.6.

.

7) Timer based polling is used and Timer_Poll_Periodic has expired.
8) Poll prohibit shall be used, the timer Timer_Poll_Prohibit has expired and one or several polls were prohibited during the time Timer_Poll_Prohibit was active.

11.3.2.1.2 Segmentation of a SDU

Upon reception of a SDU, RLC shall segment the SDU to fit into the fixed size of a PU. The segments are inserted in the data field of a PU. A length indicator shall be added to each PU that includes a border of an SDU, i.e. if a PU does not contain an LI, the SDU continues in the next PU. The length indicator indicates where the border occurs in the PU. The data after the indicated border can be either a new SDU, padding or piggybacked information. If padding or piggybacking is added another LI shall be added unless the padding size is one octet for PDUs with 15-bit LIs, see subclauses 9.2.2.8 and 9.2.2.9.

11.3.3 Reception of AMD PDU by the receiver

Upon reception of a AMD PDU, the receiver shall update VR(R), VR(H) and VR(MR) state variables according to the received PU(s).

If any of the PUs includes a Polling bit set to 1, the STATUS PDU transfer procedure shall be initiated.

If the detection of missing PU(s) shall be used and the receiver detects that a PU is missing, the receiver shall initiate the STATUS PDU transfer procedure.

11.3.4 Abnormal cases

11.3.4.1 Timer_Poll timeout

Upon expiry of the Timer_Poll, the sender shall retransmit the poll. The poll can be retransmitted in either a new PDU or a retransmitted PDU.

11.3.4.2 Receiving a PU outside the receiving window

Upon reception of a PU with SN<VR(R) or SN≥VR(MR), the receiver shall discard the PU. The poll bit shall be considered even if a complete PDU is discarded.

11.3.4.3 Timer_Discard timeout

11.3.4.3.1 SDU discard with explicit signalling

Upon expiry of Timer_Discard, the sender shall initiate the SDU discard with explicit signalling procedure. In the case where the TFC selection exchange has been initiated by sending the RLC Entity Info parameter to MAC, the UE may wait until after it provides MAC with the requested set of PDUs before discarding the afore-mentioned SDU.

11.3.4.4 VT(DAT) ≥ MaxDAT

If SDU discard after MaxDAT number of retransmission is used and VT(DAT) ≥ MaxDAT for any PU, the sender shall initiate the SDU discard with explicit signalling procedure for the SDUs to which the PU with VT(DAT) ≥MaxDAT belongs.

If the SDU discard is not used, the sender shall initiate the RLC reset procedure when VT(DAT) ≥ MaxDAT.

11.3.4.5 Invalid length indicator value

If the length indicator of a PU has a value that is larger than the PU size – the number of octets containing LIs in the PU and is not one of the predefined values listed in the table of subclause 9.2.2.8, the PU shall be discarded and treated as a missing PU.