**3GPP TSG-RAN WG2 Meeting #131 R2-250xxx**

**Bengaluru, India, 25th – 29th Aug. 2025**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.3* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | **38.322** | **CR** | **0065** | **rev** | **1** | **Current version:** | **18.2.0** |  |
|  | | | | | | | | |
| *For* [***HELP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network | **X** | Core Network |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | Introduction of R19 XR enhancements for RLC spec. | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | vivo | | | | | | | | | |
| ***Source to TSG:*** | R2 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | NR\_XR\_Ph3-Core | | | | |  | ***Date:*** | | | 2025-09-01 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | **B** |  | | | | | ***Release:*** | | | Rel-19 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change 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](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)  Rel-20 (Release 20)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | New mechanisms have been agreed to enhance support for XR services in Rel-19. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | 1. Enhancements to support multiple entry DSR in MAC, data volume is calculated for each i'th DSR-ReportingThreshold in RLC. 2. Enhancements to support RLC timely retransmission, including timer based retransmission and timer based polling. 3. Enhancements to suppport avoiding unnecessary RLC retransmission. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Enhancements related to XR services cannot be supported in Rel-19. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 3.1, 5.2.3.1.1, 5.2.3.2.1, 5.2.3.2.3, 5.2.3.2.x, 5.3.2, 5.3.3.2, 5.3.3.4, 5.3.4, 5.5, 6.2.3.10, 7.1, 7.3, 7.4 | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | | **X** |  | Other core specifications | | | | TS 38.331 CR 5395  TS 38.331 CR 5403  TS 38.323 CR 0149  TS 38.321 CR 2102  TS 38.306 CR 1321  TS 38.300 CR 1007 | | |
| ***affected:*** | |  | **X** | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  | **X** | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | |  | | | | | | | | |

Start of change

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [1].

**Data field element:** An RLC SDU or an RLC SDU segment that is mapped to the Data field.

**Delay-critical RLC SDU:** RLC SDU corresponding to a PDCP PDU indicated as delay-critical by PDCP (see TS 38.323 [4]).

**NR sidelink communication**: AS functionality enabling at least V2X Communication as defined in TS 23.287 [6] and ProSe communication (including ProSe non-Relay, UE-to-Network Relay and UE-to-UE Relay communication (including ProSe UE-to-UE Relay communication with integrated discovery)) as defined in TS 23.304 [8], between two or more nearby UEs, using NR technology but not traversing any network node.

**NR sidelink discovery**: AS functionality enabling ProSe non-Relay Discovery, ProSe UE-to-Network Relay discovery and ProSe UE-to-UE Relay discovery for Proximity based Services as defined in TS 23.304 [8] between two or more nearby UEs, using NR technology but not traversing any network node.

**RLC data volume:** The amount of data available for transmission in an RLC entity.

**RLC SDU segment:** A segment of an RLC SDU.

Next change

5.2.3.1.1 General

The transmitting side of an AM RLC entity shall prioritize transmission of RLC control PDUs over AMD PDUs. The transmitting side of an AM RLC entity shall prioritize transmission of AMD PDUs containing previously transmitted RLC SDUs or RLC SDU segments over transmission of AMD PDUs containing not previously transmitted RLC SDUs or RLC SDU segments.

The transmitting side of an AM RLC entity shall maintain a transmitting window according to the state variable TX\_Next\_Ack as follows:

- a SN falls within the transmitting window if TX\_Next\_Ack <= SN < TX\_Next\_Ack + AM\_Window\_Size;

- a SN falls outside of the transmitting window otherwise.

The transmitting side of an AM RLC entity shall not submit to lower layer any AMD PDU whose SN falls outside of the transmitting window.

For each RLC SDU received from the upper layer, the AM RLC entity shall:

- associate a SN with the RLC SDU equal to TX\_Next and construct an AMD PDU by setting the SN of the AMD PDU to TX\_Next;

- increment TX\_Next by one.

When submitting an AMD PDU that contains a segment of an RLC SDU, to lower layer, the transmitting side of an AM RLC entity shall:

- set the SN of the AMD PDU to the SN of the corresponding RLC SDU.

The transmitting side of an AM RLC entity can receive a positive acknowledgement (confirmation of successful reception by its peer AM RLC entity) for an RLC SDU by the following:

- STATUS PDU from its peer AM RLC entity.

When receiving a positive acknowledgement for an RLC SDU with SN = x, the transmitting side of an AM RLC entity shall:

- send an indication to the upper layers of successful delivery of the RLC SDU;

- set TX\_Next\_Ack equal to the SN of the RLC SDU with the smallest SN, whose SN falls within the range TX\_Next\_Ack <= SN <= TX\_Next and for which a positive acknowledgment has not been received yet.

When receiving a positive acknowledgement for an RLC SDU with SN = x, the transmitting side of an AM RLC entity should cancel any pending remaining time based retransmissions for the corresponding RLC SDU or RLC SDU segment(s).

If *stopReTxDiscardedSDU* is configured, when indicated from upper layer to discard a particular RLC SDU (see TS 38.323 [4]), the transmitting side of an AM RLC entity shall not consider the corresponding RLC SDU or RLC SDU segment(s) for transmission or retransmission if the RLC SDU or the RLC SDU segment(s) thereof has been submitted to lower layers.

NOTE: The corresponding RLC SDU or RLC SDU segment(s) above includes RLC SDU or RLC SDU segment(s) which have been already considered or pending for transmission or retransmission.

Next change

#### 5.2.3.2 Receive operations

5.2.3.2.1 General

The receiving side of an AM RLC entity shall maintain a receiving window according to the state variable RX\_Next as follows:

- a SN falls within the receiving window if RX\_Next <= SN < RX\_Next + AM\_Window\_Size;

- a SN falls outside of the receiving window otherwise.

When receiving an AMD PDU from lower layer, the receiving side of an AM RLC entity shall:

- either discard the received AMD PDU or place it in the reception buffer (see clause 5.2.3.2.2);

- if the received AMD PDU was placed in the reception buffer:

- update state variables, reassemble and deliver RLC SDUs to upper layer and start/stop *t-Reassembly* and *t-RxDiscard* as needed (see clause 5.2.3.2.3).

When *t-Reassembly* expires, the receiving side of an AM RLC entity shall:

- update state variables and start *t-Reassembly* as needed (see clause 5.2.3.2.4).

When *t-RxDiscard* expires, the receiving side of an AM RLC entity shall:

- update state variables and start *t-RxDiscard* as needed (see clause 5.2.3.2.x).

##### 5.2.3.2.2 Actions when an AMD PDU is received from lower layer

When an AMD PDU is received from lower layer, where the AMD PDU contains byte segment numbers y to z of an RLC SDU with SN = x, the receiving side of an AM RLC entity shall:

- if x falls outside of the receiving window; or

- if byte segment numbers y to z of the RLC SDU with SN = x have been received before:

- discard the received AMD PDU.

- else:

- place the received AMD PDU in the reception buffer;

- if some byte segments of the RLC SDU contained in the AMD PDU have been received before:

- discard the duplicate byte segments.

5.2.3.2.3 Actions when an AMD PDU is placed in the reception buffer

When an AMD PDU with SN = x is placed in the reception buffer, the receiving side of an AM RLC entity shall:

- if x >= RX\_Next\_Highest:

- update RX\_Next\_Highest to x+ 1.

- if all bytes of the RLC SDU with SN = x are received:

- reassemble the RLC SDU from AMD PDU(s) with SN = x, remove RLC headers when doing so and deliver the reassembled RLC SDU to upper layer;

- if x = RX\_Highest\_Status:

- update RX\_Highest\_Status to the SN of the first RLC SDU with SN > current RX\_Highest\_Status for which not all bytes have been received.

- if x = RX\_Next:

- update RX\_Next to the SN of the first RLC SDU with SN > current RX\_Next for which not all bytes have been received.

- if *t-Reassembly* is running:

- if RX\_Next\_Status\_Trigger = RX\_Next; or

- if RX\_Next\_Status\_Trigger = RX\_Next + 1 and there is no missing byte segment of the SDU associated with SN = RX\_Next before the last byte of all received segments of this SDU; or

- if RX\_Next\_Status\_Trigger falls outside of the receiving window and RX\_Next\_Status\_Trigger is not equal to RX\_Next + AM\_Window\_Size:

- stop and reset *t-Reassembly*.

- if *t-Reassembly* is not running (includes the case *t-Reassembly* is stopped due to actions above):

- if RX\_Next\_Highest> RX\_Next +1; or

- if RX\_Next\_Highest = RX\_Next + 1 and there is at least one missing byte segment of the SDU associated with SN = RX\_Next before the last byte of all received segments of this SDU:

- start *t-Reassembly*;

- set RX\_Next\_Status\_Trigger to RX\_Next\_Highest.

- if *t-RxDiscard* is configured and running:

- if RX\_Next\_Discard\_Trigger = RX\_Next; or

- if RX\_Next\_ Discard\_Trigger = RX\_Next + 1 and there is no missing byte segment of the SDU associated with SN = RX\_Next before the last byte of all received segments of this SDU; or

- if RX\_Next\_Discard\_Trigger falls outside of the receiving window and RX\_Next\_Discard\_Trigger is not equal to RX\_Next + AM\_Window\_Size:

- stop and reset *t-RxDiscard*.

- if *t-RxDiscard* is configured and not running (includes the case *t-RxDiscard* is stopped due to actions above):

- if RX\_Next\_Highest> RX\_Next +1; or

- if RX\_Next\_Highest = RX\_Next + 1 and there is at least one missing byte segment of the SDU associated with SN = RX\_Next before the last byte of all received segments of this SDU:

- start *t-RxDiscard*;

- set RX\_Next\_Discard\_Trigger to RX\_Next\_Highest.

##### 5.2.3.2.4 Actions when *t-Reassembly* expires

When *t-Reassembly* expires, the receiving side of an AM RLC entity shall:

- update RX\_Highest\_Status to the SN of the first RLC SDU with SN >= RX\_Next\_Status\_Trigger for which not all bytes have been received;

- if RX\_Next\_Highest> RX\_Highest\_Status +1: or

- if RX\_Next\_Highest = RX\_Highest\_Status + 1 and there is at least one missing byte segment of the SDU associated with SN = RX\_Highest\_Status before the last byte of all received segments of this SDU:

- start *t-Reassembly*;

- set RX\_Next\_Status\_Trigger to RX\_Next\_Highest.

##### 5.2.3.2.x Actions when *t-RxDiscard* expires

When *t-RxDiscard* expires, the receiving side of an AM RLC entity shall:

- discard the AMD PDU(s) in the reception buffer with SN < RX\_Next\_Discard\_Trigger, if any;

- update RX\_Next to the SN of the first RLC SDU with SN >= RX\_Next\_Discard\_Trigger for which not all bytes have been received;

- if RX\_Next\_Highest > RX\_Next +1; or

- if RX\_Next\_Highest = RX\_Next + 1 and there is at least one missing byte segment of the SDU associated with SN = RX\_Next before the last byte of all received segments of this SDU:

- start *t-RxDiscard*;

- set RX\_Next\_Discard\_Trigger to RX\_Next\_Highest.

Next change

5.3.2 Retransmission

The transmitting side of an AM RLC entity can receive a negative acknowledgement (notification of reception failure by its peer AM RLC entity) for an RLC SDU or an RLC SDU segment by the following:

- STATUS PDU from its peer AM RLC entity.

When receiving a negative acknowledgement for an RLC SDU or an RLC SDU segment by a STATUS PDU from its peer AM RLC entity, the transmitting side of the AM RLC entity shall:

- if the SN of the corresponding RLC SDU falls within the range TX\_Next\_Ack <= SN < = the highest SN of the AMD PDU among the AMD PDUs submitted to lower layer; and

- if *stopReTxDiscardedSDU* is configured and no discard indication for the RLC SDU has been received from upper layers, or if *stopReTxDiscardedSDU* is not configured:

- consider the RLC SDU or the RLC SDU segment for which a negative acknowledgement was received for retransmission.

When receiving an indication from upper layer (e.g., PDCP) that the condition for remaining-time-based RLC retransmission has been met for an RLC SDU, the transmitting side of the AM RLC entity shall:

- if the RLC SDU or the RLC SDU segment(s) thereof has been submitted to lower layers; and

- if the RLC SDU or the RLC SDU segment(s) thereof has not been positively acknowledged:

- consider the RLC SDU or the RLC SDU segment(s) for retransmission.

When an RLC SDU or an RLC SDU segment is considered for retransmission, the transmitting side of the AM RLC entity shall:

- if the RLC SDU or RLC SDU segment is considered for retransmission for the first time:

- set the RETX\_COUNT associated with the RLC SDU to zero.

- else, if it (the RLC SDU or the RLC SDU segment that is considered for retransmission) is not pending for retransmission already and the RETX\_COUNT associated with the RLC SDU has not been incremented due to another negative acknowledgment in the same STATUS PDU:

- increment the RETX\_COUNT.

- if RETX\_COUNT = *maxRetxThreshold*:

- indicate to upper layers that max retransmission has been reached.

When retransmitting an RLC SDU or an RLC SDU segment, the transmitting side of an AM RLC entity shall:

- if needed, segment the RLC SDU or the RLC SDU segment;

- form a new AMD PDU which will fit within the total size of AMD PDU(s) indicated by lower layer at the particular transmission opportunity;

- submit the new AMD PDU to lower layer.

When forming a new AMD PDU, the transmitting side of an AM RLC entity shall:

- only map the original RLC SDU or RLC SDU segment to the Data field of the new AMD PDU;

- modify the header of the new AMD PDU in accordance with the description in clause 6.2.2.4;

- set the P field according to clause 5.3.3.

Next change

5.3.3.2 Transmission of a AMD PDU

Upon notification of a transmission opportunity by lower layer, for each AMD PDU submitted for transmission such that the AMD PDU contains either a not previously transmitted RLC SDU or an RLC SDU segment containing not previously transmitted byte segment, the transmitting side of an AM RLC entity shall:

- increment PDU\_WITHOUT\_POLL by one;

- increment BYTE\_WITHOUT\_POLL by every new byte of Data field element that it maps to the Data field of the AMD PDU;

- if PDU\_WITHOUT\_POLL >= pollPDU; or

- if BYTE\_WITHOUT\_POLL >= pollByte:

- include a poll in the AMD PDU as described below.

Upon notification of a transmission opportunity by lower layer, for each AMD PDU submitted for transmission, the transmitting side of an AM RLC entity shall:

- if both the transmission buffer and the retransmission buffer becomes empty (excluding transmitted RLC SDUs or RLC SDU segments awaiting acknowledgements and excluding RLC SDUs or RLC SDU segments for which the transmission and retransmission are stopped as specified in clause 5.2.3.1.1) after the transmission of the AMD PDU; or

- if no new RLC SDU can be transmitted after the transmission of the AMD PDU (e.g. due to window stalling); or

- if an indication is received from upper layer (e.g., PDCP) that the condition for remaining-time-based RLC polling has been met for an RLC SDU:

- include a poll in the AMD PDU as described below.

NOTE: Empty RLC buffer (excluding transmitted RLC SDUs or RLC SDU segments awaiting acknowledgements and excluding RLC SDUs or RLC SDU segments for which the transmission and retransmission are stopped as specified in clause 5.2.3.1.1) should not lead to unnecessary polling when data awaits in the upper layer. Details are left up to UE implementation.

To include a poll in an AMD PDU, the transmitting side of an AM RLC entity shall:

- set the P field of the AMD PDU to "1";

- set PDU\_WITHOUT\_POLL to 0;

- set BYTE\_WITHOUT\_POLL to 0.

Upon submission of an AMD PDU including a poll to lower layer, the transmitting side of an AM RLC entity shall:

- set POLL\_SN to the highest SN of the AMD PDU among the AMD PDUs submitted to lower layer;

- if *t-PollRetransmit* is not running:

- start *t-PollRetransmit*.

- else:

- restart *t-PollRetransmit*.

Next change

#### 5.3.3.3 Reception of a STATUS report

Upon reception of a STATUS report from the receiving RLC AM entity the transmitting side of an AM RLC entity shall:

- if the STATUS report comprises a positive or negative acknowledgement for the RLC SDU with sequence number equal to POLL\_SN:

- if *t-PollRetransmit* is running:

- stop and reset *t-PollRetransmit*.

NOTE: When all RLC SDUs with SNs up to POLL\_SN are already positively or negatively acknowledged or indicated as discarded from upper layer (e.g., PDCP), the transmitting side of an AM RLC entity may stop and reset the running *t-PollRetransmit*.

Next change

#### 5.3.3.4 Expiry of *t-PollRetransmit*

Upon expiry of *t-PollRetransmit*, the transmitting side of an AM RLC entity shall:

- if both the transmission buffer and the retransmission buffer are empty (excluding transmitted RLC SDU or RLC SDU segment awaiting acknowledgements and excluding RLC SDUs or RLC SDU segments for which the transmission and retransmission are stopped as specified in clause 5.2.3.1.1); or

- if no new RLC SDU or RLC SDU segment can be transmitted (e.g. due to window stalling):

- consider the RLC SDU with the highest SN among the RLC SDUs submitted to lower layer for retransmissio n (excluding RLC SDUs or RLC SDU segments for which the transmission and retransmission are stopped as specified in clause 5.2.3.1.1); or

- consider any RLC SDU which has not been positively acknowledged for retransmission (excluding RLC SDUs or RLC SDU segments for which the transmission and retransmission are stopped as specified in clause 5.2.3.1.1).

- include a poll in an AMD PDU, if any, as described in clause 5.3.3.2.

Next change

5.3.4 Status reporting

An AM RLC entity sends STATUS PDUs to its peer AM RLC entity in order to provide positive and/or negative acknowledgements of RLC SDUs (or portions of them).

Triggers to initiate STATUS reporting include:

- Polling from its peer AM RLC entity:

- When an AMD PDU with SN = x and the P field set to "1" is received from lower layer, the receiving side of an AM RLC entity shall:

- if the AMD PDU is to be discarded as specified in clause 5.2.3.2.2; or

- if x < RX\_Highest\_Status or x >= RX\_Next + AM\_Window\_Size:

- trigger a STATUS report.

- else:

- delay triggering the STATUS report until x < RX\_Highest\_Status or x >= RX\_Next + AM\_Window\_Size.

NOTE 1: This ensures that the RLC Status report is transmitted after HARQ reordering.

- Detection of reception failure of an AMD PDU

- The receiving side of an AM RLC entity shall trigger a STATUS report when *t-Reassembly* expires.

- Detection of discard of an AMD PDU:

- The receiving side of an AM RLC entity shall trigger a STATUS report when*t-RxDiscard* expires.

NOTE 2: The expiry of *t-Reassembly* triggers both RX\_Highest\_Status to be updated and a STATUS report to be triggered, but the STATUS report shall be triggered after RX\_Highest\_Status is updated.

NOTE X: The expiry of *t-RxDiscard* triggers both RX\_Next to be updated and a STATUS report to be triggered, but the STATUS report shall be triggered after RX\_Next is updated.

When STATUS reporting has been triggered, the receiving side of an AM RLC entity shall:

- if *t-StatusProhibit* is not running:

- at the first transmission opportunity indicated by lower layer, construct a STATUS PDU and submit it to lower layer.

- else:

- at the first transmission opportunity indicated by lower layer after *t-StatusProhibit* expires, construct a single STATUS PDU even if status reporting was triggered several times while *t-StatusProhibit* was running and submit it to lower layer.

When a STATUS PDU has been submitted to lower layer, the receiving side of an AM RLC entity shall:

- start *t-StatusProhibit*.

When constructing a STATUS PDU, the AM RLC entity shall:

- for the RLC SDUs with SN such that RX\_Next <= SN < RX\_Highest\_Status that has not been completely received yet, in increasing SN order of RLC SDUs and increasing byte segment order within RLC SDUs, starting with SN = RX\_Next up to the point where the resulting STATUS PDU still fits to the total size of RLC PDU(s) indicated by lower layer:

- for an RLC SDU for which no byte segments have been received yet:

- include in the STATUS PDU a NACK\_SN which is set to the SN of the RLC SDU.

- for a continuous sequence of byte segments of a partly received RLC SDU that have not been received yet:

- include in the STATUS PDU a set of NACK\_SN, SOstart and SOend.

- for a continuous sequence of RLC SDUs that have not been received yet:

- include in the STATUS PDU a set of NACK\_SN and NACK range;

- include in the STATUS PDU, if required, a pair of SOstart and SOend.

- set the ACK\_SN to the SN of the next not received RLC SDU which is not indicated as missing and is not discarded due to the expiry of *t-RxDiscard* in the resulting STATUS PDU.

Next change

5.5 Data volume calculation

For the purpose of MAC buffer status reporting, the UE shall consider the following as RLC data volume:

- RLC SDUs and RLC SDU segments that have not yet been included in an RLC data PDU;

- RLC data PDUs that are pending for initial transmission;

- RLC data PDUs that are pending for retransmission (RLC AM).

For the purpose of MAC single entry delay status reporting, the UE shall consider the following as delay-critical RLC data volume:

- delay-critical RLC SDUs and delay-critical RLC SDU segments that have not yet been included in an RLC data PDU;

- RLC data PDUs pending for initial transmission, and containing a delay-critical RLC SDU or a delay-critical RLC SDU segment;

- RLC data PDUs that are pending for retransmission (RLC AM).

For the purpose of MAC multiple entry delay status reporting, the UE shall evaluate the delay-reporting RLC data volume in ascending order of *DSR-ReportingThreshold* based on the DSR data indication from upper layer (e.g., PDCP), and consider the following as delay-reporting RLC data volume associated with the i:th *DSR-ReportingThreshold* in *dsr-ReportingThresList*, where i starts from 1:- for an RLC SDU that the DSR data indication associated with the i:th *DSR-ReportingThreshold* is received from upper layer (e.g., PDCP):

- the RLC SDU or RLC SDU segment(s) thereof that has not yet been included in an RLC data PDU, and are not considered as delay-reporting RLC data volume associated with any of the k:th *DSR-ReportingThreshold* where k < i;

- the corresponding RLC data PDUs pending for initial transmission, and containing the RLC SDU or RLC SDU segment(s) thereof, and that are not considered as delay-reporting RLC data volume associated with any of the k:th *DSR-ReportingThreshold* where k < i;

- if i=1, RLC data PDUs that are pending for retransmission (RLC AM).

In addition, if a STATUS PDU has been triggered and *t-StatusProhibit* is not running or has expired, the UE shall estimate the size of the STATUS PDU that will be transmitted in the next transmission opportunity, and consider this as part of RLC data volume for MAC buffer status reporting, as part of delay-critical RLC data volume for MAC single entry delay status reporting, and as part of the delay-reporting RLC data volume associated with the first (i.e. i=1) *DSR-ReportingThreshold* for MAC multiple entry delay status reporting.

Next change

#### 6.2.3.10 Acknowledgement SN (ACK\_SN) field

Length: 12 bits or 18 bits (configurable).

The ACK\_SN field indicates the SN of the next not received RLC SDU which is not reported as missing and is not discarded due to the expiry of *t-RxDiscard* in the STATUS PDU. When the transmitting side of an AM RLC entity receives a STATUS PDU, it interprets that all RLC SDUs up to but not including the RLC SDU with SN = ACK\_SN have been received by its peer AM RLC entity, excluding those RLC SDUs indicated in the STATUS PDU with NACK\_SN, portions of RLC SDUs indicated in the STATUS PDU with NACK\_SN, SOstart and SOend, RLC SDUs indicated in the STATUS PDU with NACK\_SN and NACK\_range, and portions of RLC SDUs indicated in the STATUS PDU with NACK\_SN, NACK range, SOstart and SOend.

Next change

7.1 State variables

This clause describes the state variables used in AM and UM entities in order to specify the RLC protocol. The state variables defined in this clause are normative.

All state variables and all counters are non-negative integers.

All state variables related to AM data transfer can take values from 0 to 4095 for 12 bit SN or from 0 to 262143 for 18 bit SN. All arithmetic operations contained in the present document on state variables related to AM data transfer are affected by the AM modulus (i.e. final value = [value from arithmetic operation] modulo 4096 for 12 bit SN and 262144 for 18 bit SN).

All state variables related to UM data transfer can take values from 0 to 63 for 6 bit SN or from 0 to 4095 for 12 bit SN. All arithmetic operations contained in the present document on state variables related to UM data transfer are affected by the UM modulus (i.e. final value = [value from arithmetic operation] modulo 64 for 6 bit SN and 4096 for 12 bit SN).

When performing arithmetic comparisons of state variables or SN values, a modulus base shall be used.

TX\_Next\_Ack and RX\_Next shall be assumed as the modulus base at the transmitting side and receiving side of an AM RLC entity, respectively. This modulus base is subtracted from all the values involved, and then an absolute comparison is performed (e.g. RX\_Next <= SN < RX\_Next + AM\_Window\_Size is evaluated as [RX\_Next – RX\_Next] modulo 2[*sn-FieldLength*] <= [SN – RX\_Next] modulo 2[*sn-FieldLength*] < [RX\_Next + AM\_Window\_Size – RX\_Next] modulo 2[*sn-FieldLength*]), where *sn-FieldLength* is 12 or 18 for 12 bit SN and 18 bit SN, respectively.

RX\_Next\_Highest– UM\_Window\_Size shall be assumed as the modulus base at the receiving UM RLC entity. This modulus base is subtracted from all the values involved, and then an absolute comparison is performed (e.g. (RX\_Next\_Highest– UM\_Window\_Size) <= SN < RX\_Next\_Highest is evaluated as [(RX\_Next\_Highest– UM\_Window\_Size) – (RX\_Next\_Highest– UM\_Window\_Size)] modulo 2[*sn-FieldLength*] <= [SN – (RX\_Next\_Highest– UM\_Window\_Size)] modulo 2[*sn-FieldLength*] < [RX\_Next\_Highest– (RX\_Next\_Highest– UM\_Window\_Size)] modulo 2[*sn-FieldLength*]), where *sn-FieldLength* is 6 or 12 for 6 bit SN and 12 bit SN, respectively.

The transmitting side of each AM RLC entity shall maintain the following state variables:

a) TX\_Next\_Ack – Acknowledgement state variable

This state variable holds the value of the SN of the next RLC SDU for which a positive acknowledgment is to be received in-sequence, and it serves as the lower edge of the transmitting window. It is initially set to 0, and is updated whenever the AM RLC entity receives a positive acknowledgment for an RLC SDU with SN = TX\_Next\_Ack.

b) TX\_Next – Send state variable

This state variable holds the value of the SN to be assigned for the next newly generated AMD PDU. It is initially set to 0, and is updated whenever the AM RLC entity constructs an AMD PDU with SN = TX\_Next and contains an RLC SDU or the last segment of a RLC SDU.

c) POLL\_SN – Poll send state variable

This state variable holds the value of the highest SN of the AMD PDU among the AMD PDUs submitted to lower layer when POLL\_SN is set according to clause 5.3.3.2. It is initially set to 0.

The transmitting side of each AM RLC entity shall maintain the following counters:

a) PDU\_WITHOUT\_POLL – Counter

This counter is initially set to 0. It counts the number of AMD PDUs sent since the most recent poll bit was transmitted.

b) BYTE\_WITHOUT\_POLL – Counter

This counter is initially set to 0. It counts the number of data bytes sent since the most recent poll bit was transmitted.

c) RETX\_COUNT – Counter

This counter counts the number of retransmissions of an RLC SDU or RLC SDU segment (see clause 5.3.2). There is one RETX\_COUNT counter maintained per RLC SDU. This counter is reset to zero for each RLC SDU when indicated by upper layer.

The receiving side of each AM RLC entity shall maintain the following state variables:

a) RX\_Next – Receive state variable

This state variable holds the value of the SN following the last in-sequence completely received RLC SDU, and it serves as the lower edge of the receiving window. It is initially set to 0, and is updated whenever the AM RLC entity receives an RLC SDU with SN = RX\_Next.

b) RX\_Next\_Status\_Trigger – *t-Reassembly* state variable

This state variable holds the value of the SN following the SN of the RLC SDU which triggered *t-Reassembly*.

c) RX\_Highest\_Status – Maximum STATUS transmit state variable

This state variable holds the highest possible value of the SN which can be indicated by "ACK\_SN" when a STATUS PDU needs to be constructed. It is initially set to 0.

d) RX\_Next\_Highest – Highest received state variable

This state variable holds the value of the SN following the SN of the RLC SDU with the highest SN among received RLC SDUs. It is initially set to 0.

x) RX\_Next\_Discard\_Trigger – *t-RxDiscard* state variable

This state variable holds the value of the SN following the SN of the RLC SDU which triggered *t-RxDiscard*. This state variable shall be maintained only when the AM RLC entity is configured with *t-RxDiscard*.

Each transmitting UM RLC entity shall maintain the following state variables:

a) TX\_Next – UM send state variable

This state variable holds the value of the SN to be assigned for the next newly generated UMD PDU with segment. It is initially set to 0, and is updated after the UM RLC entity submits a UMD PDU including the last segment of an RLC SDU to lower layers.

Each receiving UM RLC entity shall maintain the following state variables:

a) RX\_Next\_Reassembly – UM receive state variable

This state variable holds the value of the earliest SN that is still considered for reassembly. It is initially set to 0. For groupcast and broadcast of NR sidelink communication or for SL-SRB4 of NR sidelink discovery, it is initially set to the SN of the first received UMD PDU containing an SN. For the receiving UM RLC entity configured for MCCH or MTCH, it is up to UE implementation to set the initial value of RX\_Next\_Reassembly to a value before RX\_Next\_Highest.

b) RX\_Timer\_Trigger – UM *t-Reassembly* state variable

This state variable holds the value of the SN following the SN which triggered *t-Reassembly*.

c) RX\_Next\_Highest– UM receive state variable

This state variable holds the value of the SN following the SN of the UMD PDU with the highest SN among received UMD PDUs. It serves as the higher edge of the reassembly window. It is initially set to 0. For groupcast and broadcast of NR sidelink communication or for SL-SRB4 of NR sidelink discovery, it is initially set to the SN of the first received UMD PDU containing an SN. For the receiving UM RLC entity configured for MCCH or MTCH, it is initially set to the SN of the first received UMD PDU containing an SN.

Next change

## 7.3 Timers

The following timers are configured by TS 38.331 [5]:

a) *t-PollRetransmit*

This timer is used by the transmitting side of an AM RLC entity in order to retransmit a poll (see clause 5.3.3).

b) *t-Reassembly*

This timer is used by the receiving side of an AM RLC entity and receiving UM RLC entity in order to detect loss of RLC PDUs at lower layer (see clauses 5.2.2.2 and 5.2.3.2). If *t-Reassembly* is running, *t-Reassembly* shall not be started additionally, i.e. only one *t-Reassembly* per RLC entity is running at a given time.

c) *t-StatusProhibit*

This timer is used by the receiving side of an AM RLC entity in order to prohibit transmission of a STATUS PDU (see clause 5.3.4).

x) *t-RxDiscard*

This timer is used by the receiving side of an AM RLC entity in order to detect discard of AMD PDU(s) (see clause 5.2.3.2.x). If *t-RxDiscard* is running, *t-RxDiscard* shall not be started additionally, i.e. only one *t-RxDiscard* per RLC entity is running at a given time.

Next change

## 7.4 Configurable parameters

The following parameters are configured by TS 38.331 [5]:

a) *maxRetxThreshold*

This parameter is used by the transmitting side of each AM RLC entity to limit the number of retransmissions corresponding to an RLC SDU, including its segments (see clause 5.3.2).

b) *pollPDU*

This parameter is used by the transmitting side of each AM RLC entity to trigger a poll for every *pollPDU* PDUs (see clause 5.3.3).

c) *pollByte*

This parameter is used by the transmitting side of each AM RLC entity to trigger a poll for every *pollByte* bytes (see clause 5.3.3).

x) *stopReTxDiscardedSDU*

This parameter is used by the transmitting side of each AM RLC entity to determine whether to stop RLC transmission and retransmission of discarded SDUs (see clause 5.2.3).

End of change