Femto Forum Ltd . PO Box 23 . Dursley . GL11 . 5WA . UK
tel +44 (0)845 644 5823 **fax** +44 (0)845 644 5824 **email** info@femtoforum.org **www.femtoforum.org**

driving convergence worldwide

LTE eNB L1 API Definition v1.1

Femto Forum Technical Document

Document number:	FF_Tech_002_v1.11
Date issued:	12-10-2010
Document status:	*Document for public distribution*

Legal Notice

Terms and Acronyms

Term / Acronym	*Definition*
3GPP	3rd Generation Partnership Project
ACK	Acknowledge
API	Application Program Interface
BCH	Broadcast Channel
CC	Convolutional Code
CCE	Control Channel Element
CDD	Cyclic Delay Diversity
CFI	Control Format Indicator
CQI	Channel Quality Indicator
CRC	Cyclic Redundancy Check
CS	Cyclic Shift
CTC	Convolutional Turbo Codes
DCI	Downlink Control Information
DL	Downlink
DL-SCH	Downlink Shared Channel
DMRS	Demodulation Reference Symbol
DwPTS	Downlink Pilot Time Slot
EARFCN	Evolved Absolute Radio Frequency Channel Number
eNB	evolved Node B
EPC	Evolved Packet Core
EPRE	Energy Per Resource Element
EUTRA	Evolved Universal Terrestrial Radio Access
E-UTRAN	Evolved Universal Terrestrial Radio Access Network
FDD	Frequency Division Duplex
FEC	Forward Error Correction
FFT	Fast Fourier Transform
GP	Guard Period
HARQ	Hybrid Automatic Repeat Request
HCS	Header Check Sequence
HI	HARQ Indicator
HeMS	Home eNB Management System
IR	Incremental Redundancy
LTE	Long Term Evolution

MAC	Medium Access Control Layer
MBSFN	Multimedia Broadcast multicast services Single Frequency Network
MCH	Multicast Channel
MCS	Modulation and Coding Scheme
MIB	Master Information Block
MIMO	Multiple Input Multiple Output
NA	Not Applicable
NACK	Negative Acknowledge
OFDMA	Orthogonal Frequency Division Multiple Access
OS	Operating System
PCH	Paging Channel
PDCCH	Physical Downlink Control Channel
PDCP	Packet Data Convergence Protocol
PDSCH	Physical Downlink Shared Channel
PDU	Protocol Data Unit
PHICH	Physical Hybrid ARQ Indicator Channel
PHY	Physical Layer
PRACH	Physical Random Access Channel
PUCCH	Physical Uplink Control Channel
PUSCH	Physical Uplink Shared Channel
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase-Shift Keying
RA	Random Access
RACH	Random Access Channel
RB	Resource Block
RI	Rank Indicator
RRC	Radio Resource Control
RNTI	Radio Network Temporary Identifier
RSSI	Receive Signal Strength Indicator
RX	Receive
S1	The interface between the E-UTRAN and EPC
SAP	Service Access Point
SF	Subframe
SFN	System Frame Number
SISO	Single Input Single Output
SR	Scheduling Request
SRS	Sounding Reference Symbol
STC	Space Time Coding

STTD	Space Time Transmit Diversity
TB	Transport Block
TDD	Time Division Duplex
TLV	Tag Length Value
Tx	Transmit
UCI	Uplink Control Information
UE	User Equipment
UL	Uplink
UL-SCH	Uplink Shared Channel
UpPTS	Uplink Pilot Time Slot
WCDMA	Wideband Code Division Multiple Access
X2	Interface between two eNBs
ZT CC	Zero Tailed Convolutional Coding

Table of Contents

1 Introduction

This document describes an Application Programming Interface (API) which defines the interface between LTE L2/L3 software and L1 PHY. Specifically, this L1 API defines both P5 and P7 of the Femto Forum LTE FAPI.

The LTE standard [3] has been designed to support both TDD and FDD deployments. The LTE L1 API, described in this document, also supports TDD and FDD modes. Features which are specific to only TDD, or FDD, are clearly highlighted.

This document is divided into two sections. The first section provides a description of typical procedures which will occur between the L1 and L2/L3 software. The second section provides the definition of the L1 API messages.

1.1 LTE

LTE is standardized by 3GPP (http://www.3gpp.org) and designed as an evolution to the current WCDMA wireless network, which is in widespread use today. A critical requirement of LTE is the capability of supporting high data rates (300Mbps), and many aspects of the LTE network have been designed specifically to support high data rates and low latency.

Figure 1 shows the architecture of a LTE network. It consists of only two elements; the Evolved Pack Core (EPC) and the E-UTRAN Node B (eNB). The LTE L1 API resides within the eNB element. The two standardized interfaces in a LTE network are called S1 and X2. The L1 is not involved in either of these interfaces, and both are out of scope for this document.

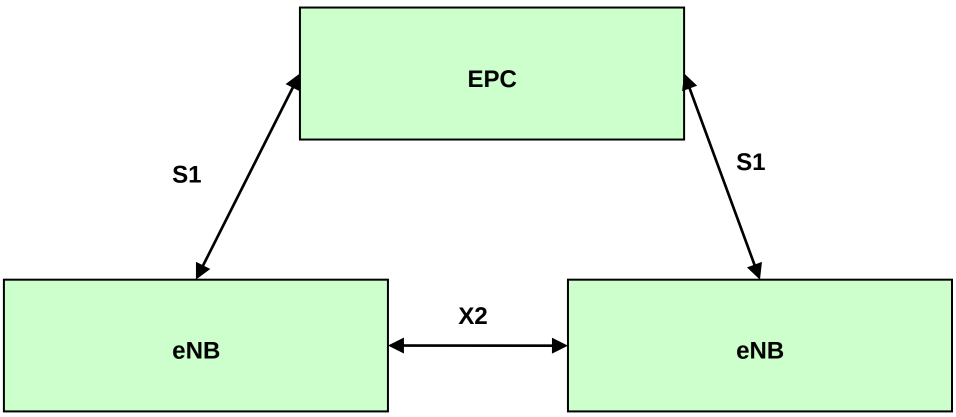

Figure 1: LTE Architecture

1.2 L1 API

The L1 API, defined in this document, resides within the eNB component. The functionality of an eNB is shown in Figure 2 and Figure 3. In both Figures the location of the L1 API is highlighted.

Figure 2 shows the protocol model for the eNB defined in the E-UTRAN architectural standard [4]. It highlights the separation of control- and data-plane information, which is maintained throughout the LTE network. Both control- and data-plane information is passed through the L1 API, however, each API message contains either control- or data-plane information, but never both.

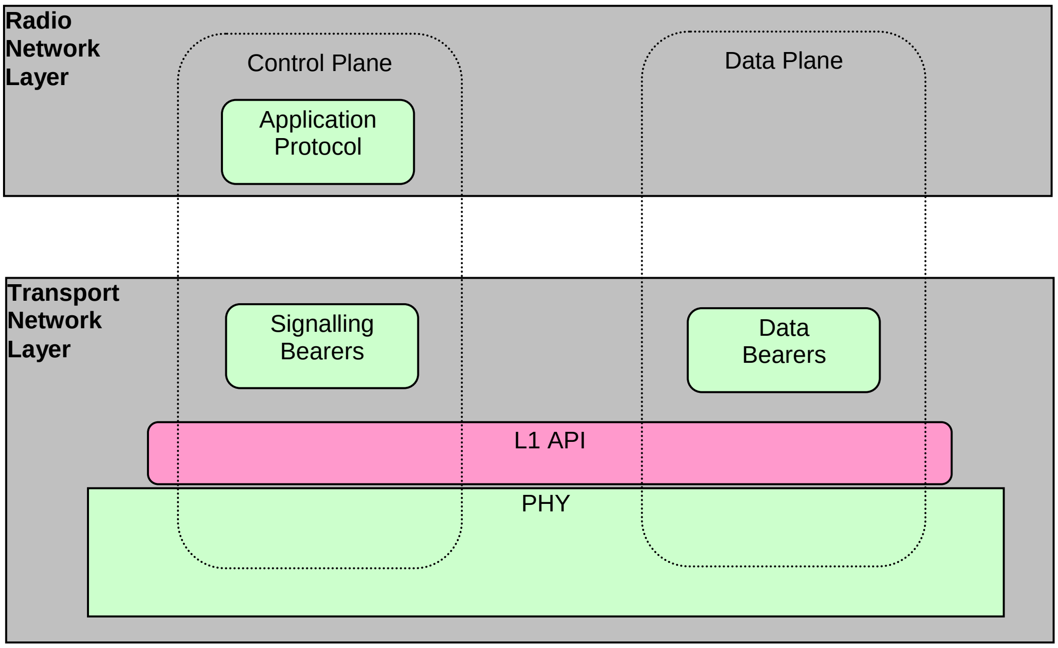

Figure 2: E-UTRAN protocol model

Figure 3 provides an example of how the different L2/L3 protocol layers will interact with the L1 API. In this example, a PHY control entity is responsible for configuration procedures (P5). The MAC layer is responsible for the exchange of data-plane messages with the PHY (P7). The PHY configuration sent over the P5 interface may be determined using SON techniques, information model parameters sent from the HeMS [11], or a combination of both methods.

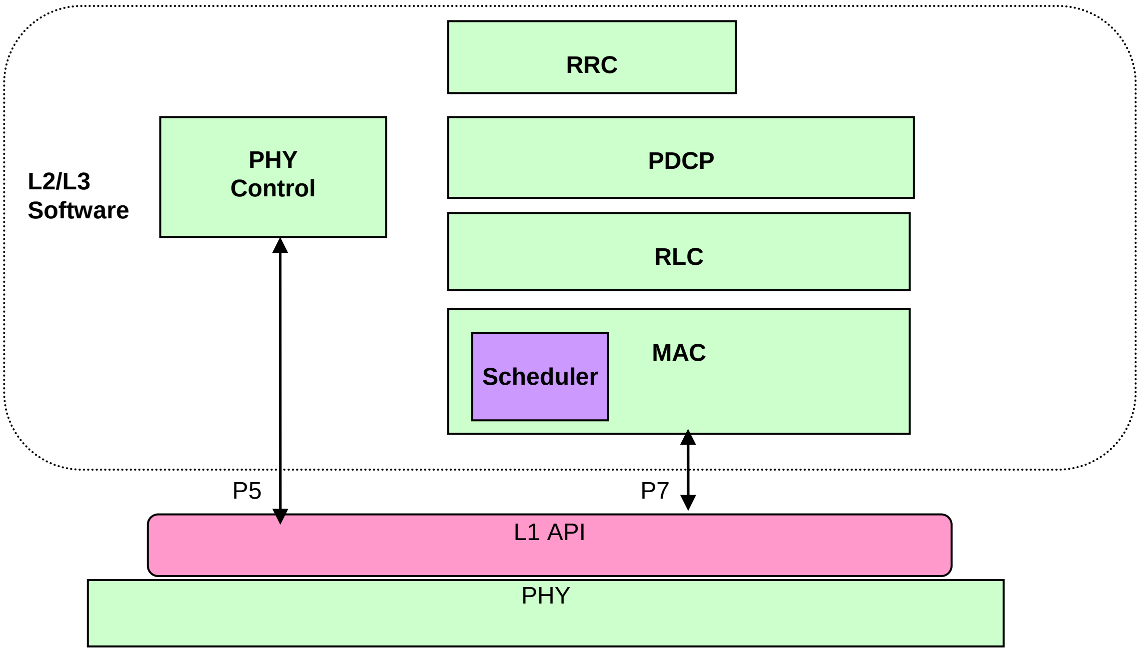

Figure 3: L1 API Interactions

2 L1 API Procedures

This section gives an overview of the procedures which use the L1 API. These procedures are split into two groups, namely, configuration procedures and subframe procedures. Configuration procedures handle the management of the PHY layer and are expected to occur infrequently. Subframe procedures determine the structure of each 1ms subframe and operate with a 1ms periodicity.

2.1 Configuration Procedures

The configuration procedures supported by the L1 API are:

- Initialization
- Termination
- Restart
- Reset
- Error notification

These procedures will move the PHY layer through the IDLE, CONFIGURED and RUNNING states, as shown in Figure 4. A list of the L1 API configuration messages which are valid in each state is given in Table 1.

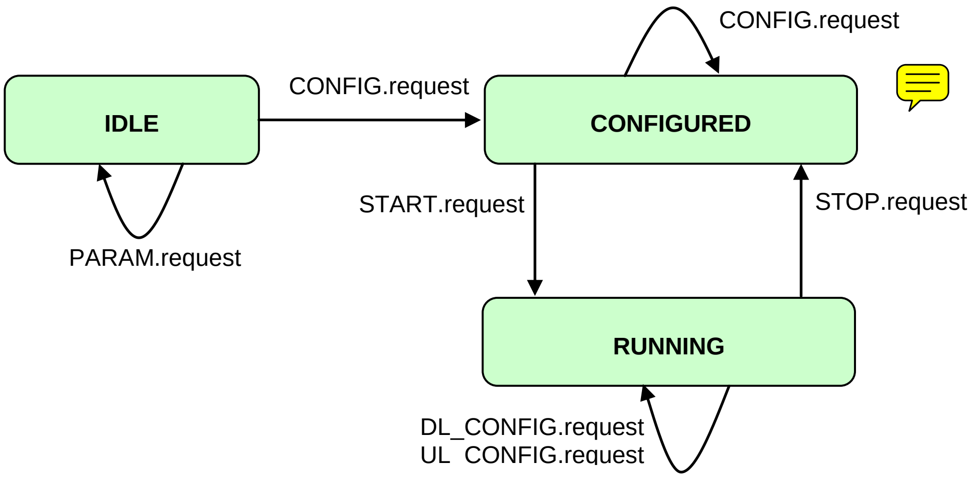

Figure 4: PHY layer state transactions on L1 API configuration messages

Idle State	*Configured State*	*Running State*
PARAM.request	PARAM.request	CONFIG.request
CONFIG.request	CONFIG.request	STOP.request
	START.request	

Table 1: L1 API configuration messages valid in each PHY state

2.1.1 Initialization

The initialization procedure moves the PHY from the IDLE state to the RUNNING state, via the CONFIGURED state. An overview of this procedure is given in Figure 5, the different stages are:

- The PARAM message exchange procedure
- The CONFIG message exchange procedure
- The START message exchange procedure

The initialization procedure is completed when the PHY sends the L2/L3 software a SUBFRAME.indication message.

The remainder of this section describes the PARAM, CONFIG and START message exchange procedures.

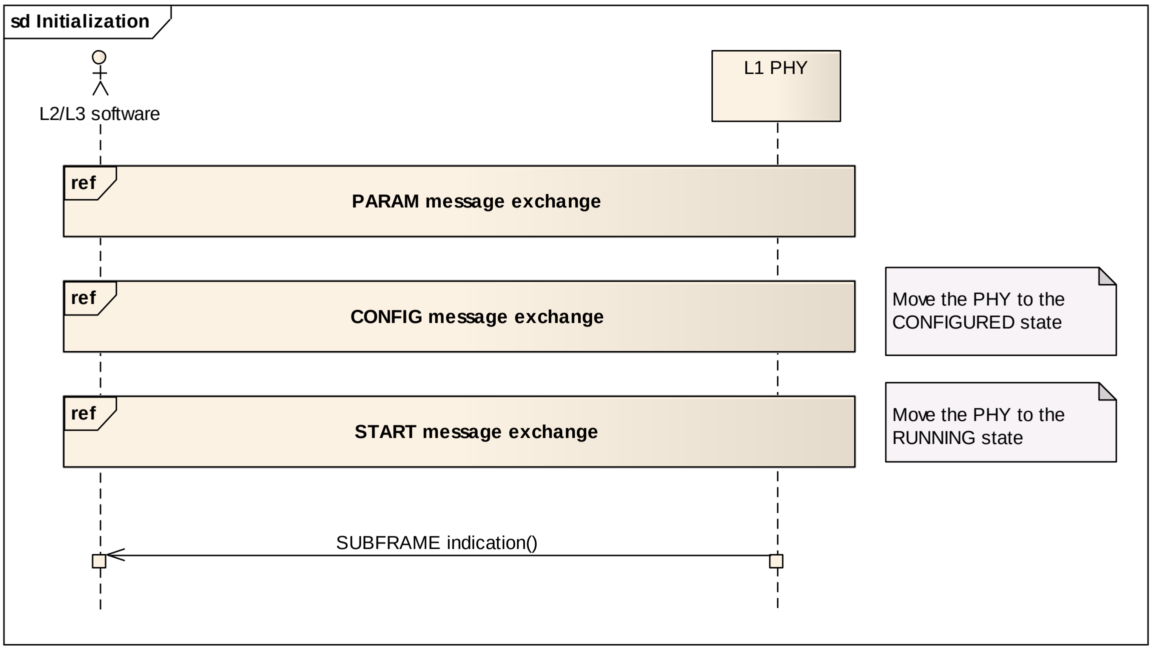

Figure 5: Initialization procedure

The PARAM message exchange procedure is shown in Figure 6. Its purpose is to allow the L2/L3 software to collect information about the PHY configuration and current state. The information returned by the PHY depends on its state, and is described in Table 2. The PARAM message exchange is optional.

PHY State	***Information Returned by PHY***
IDLE	The PHY indicates which capabilities it supports
CONFIGURED	The PHY returns its current configuration
RUNNING	The PHY returns invalid state

Table 2: Information returned by the PHY during a PARAM message exchange

From Figure 6 it can be seen that the PARAM message exchange procedure is initiated by the L2/L3 software sending a PARAM.request message to the PHY. It is recommended that the L2/L3 software starts a guard timer to wait for the response from the PHY. If the PHY is operating correctly it will return a PARAM.response message. In the IDLE and CONFIGURED states this message will include the current PHY state and a list of configuration information, as described in Table 2. In the RUNNING state this message will indicate an INVALID_STATE error, to determine the PHY capabilities it must be moved to the CONFIGURED state using the termination procedure. If the guard timer expires before the PHY responds this indicates the PHY is not operating correctly. This must be rectified before further L1 API commands are used; the rectification method is outside the scope of this document.

The CONFIG message exchange procedure is shown in Figure 7. Its purpose is to allow the L2/L3 software to configure the PHY. It can be used when the PHY is in any state. The procedure has slight differences depending on the PHY state, for clarity each case is described separately.

If the PHY is in the IDLE state the CONFIG.request message, sent by the L2/L3 software, must include all mandatory TLVs. The mandatory TLVs are highlighted later in Section 3.2.2.1. If all mandatory TLVs are included, and set to values supported by the PHY, L1 will return a CONFIG.response message indicating it is successfully configured and has moved to the CONFIGURED state. If the CONFIG.request message has missing mandatory

TLVs, invalid TLVs, or unsupported TLVs, the PHY will return a `CONFIG.response` message indicating an incorrect configuration. In this case, it will remain in the IDLE state and all received TLVs will be ignored.

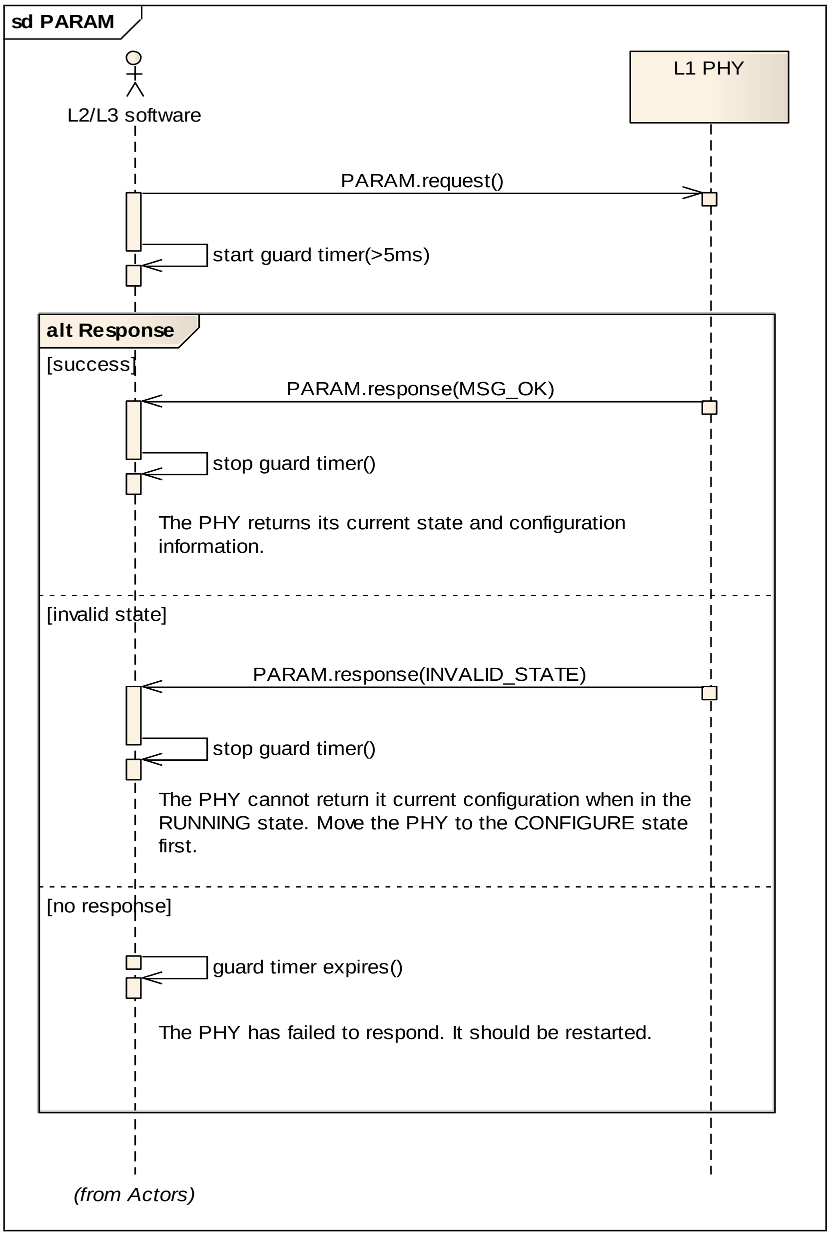

Figure 6: PARAM message exchange

If the PHY is in the CONFIGURED state the `CONFIG.request` message, sent by the L2/L3 software, may include only the TLVs that are required to change the PHY to a new configuration. If the PHY supports these new values, it will return a `CONFIG.response` message indicating it has been successfully configured. However, if the `CONFIG.request` message includes invalid TLVs, or unsupported TLVs, the PHY will return a `CONFIG.response` message indicating an incorrect configuration. In this case all received TLVs will be ignored and the PHY will continue with its previous configuration. In both cases, if the PHY receives a `CONFIG.request` while in the CONFIGURED state it will remain in the CONFIGURED state.

If the PHY is in the RUNNING state then a limited subset of CONFIG TLVs may be sent in a `CONFIG.request` message. The permitted TLVs are highlighted later in Section 3.2.2.1. If the `CONFIG.request` message has invalid TLVs, or TLVs which must not be reconfigured in the RUNNING state, the PHY will return a `CONFIG.response` message indicating an incorrect configuration. In this case, it will remain in the RUNNING state and all received TLVs will be ignored.

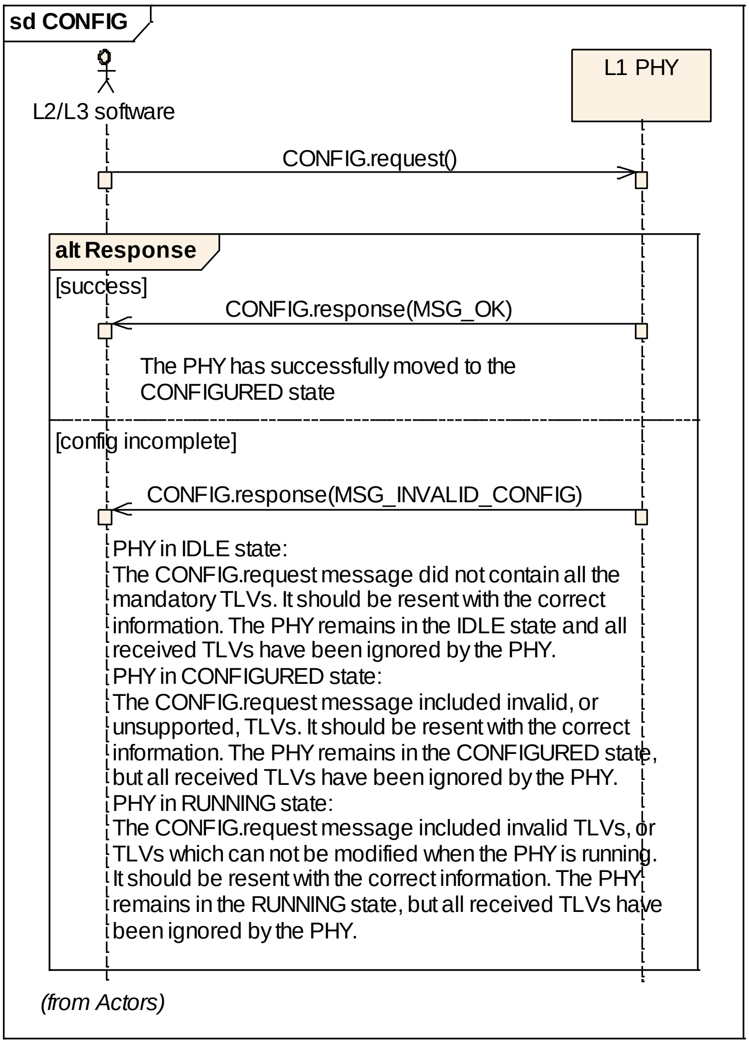

Figure 7: CONFIG message exchange

The START message exchange procedure is shown in Figure 8. Its purpose is to instruct a configured PHY to start transmitting as an eNB. The L2/L3 software initiates this procedure by sending a `START.request` message to the PHY. If the PHY is in the CONFIGURED state, it will issue a SUBFRAME indication. After the PHY has sent its first `SUBFRAME.indication` message it enters the RUNNING state.

If the PHY receives a `START.request` in either the IDLE or RUNNING state it will return an `ERROR.indication` including an INVALID_STATE error.

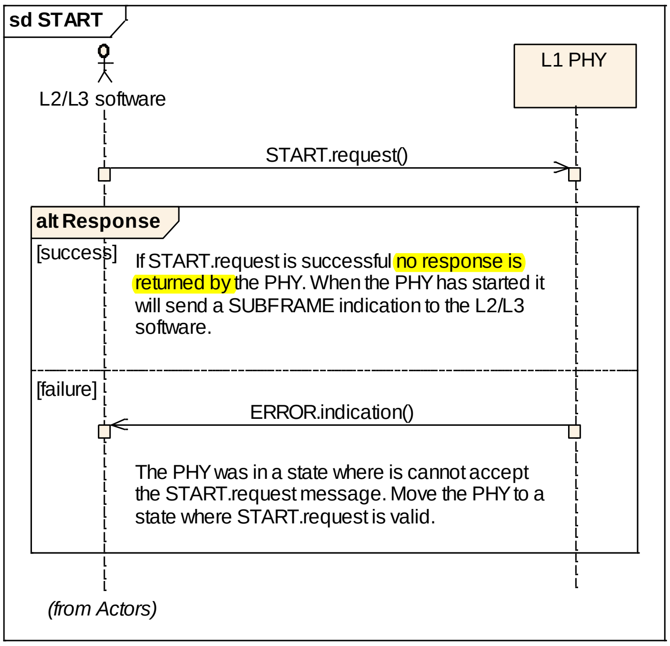

Figure 8: START message exchange

2.1.2 Termination

The termination procedure is used to move the PHY from the RUNNING state to the CONFIGURED state. This stops the PHY transmitting as an eNB. The termination procedure is shown in Figure 9 and initiated by the L2/L3 software sending a `STOP.request` message.

If the `STOP.request` message is received by the PHY while operating in the RUNNING state, it will stop all TX and RX operations and return to the CONFIGURED state. When the PHY has completed its stop procedure a `STOP.indication` message is sent to the L2/L3 software.

If the `STOP.request` message was received by the PHY while in the IDLE or CONFIGURED state, it will return an `ERROR.indication` message including an INVALID_STATE error. However, in this case the PHY was already stopped.

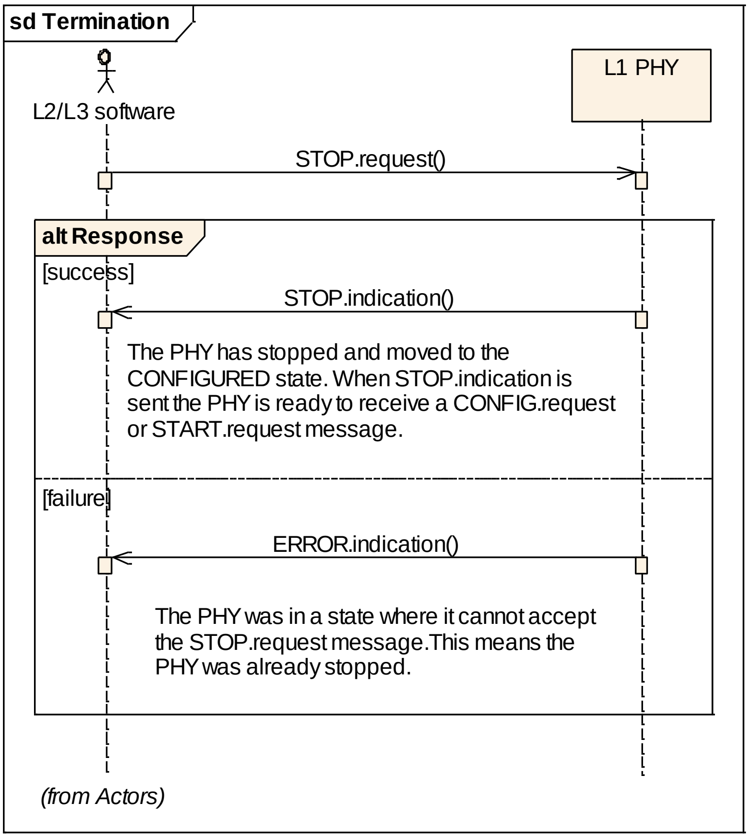

Figure 9: Termination procedure

2.1.3 Restart

The restart procedure is shown in Figure 10. It can be used by the L2/L3 software when it needs to stop transmitting, but later wants to restart transmission using the same configuration. To complete this procedure the L2/L3 software can follow the STOP message exchange shown in Figure 9. This moves the PHY to the CONFIGURED state. To restart transmission it should follow the START message exchange, shown in Figure 8, moving the PHY back to the RUNNING state.

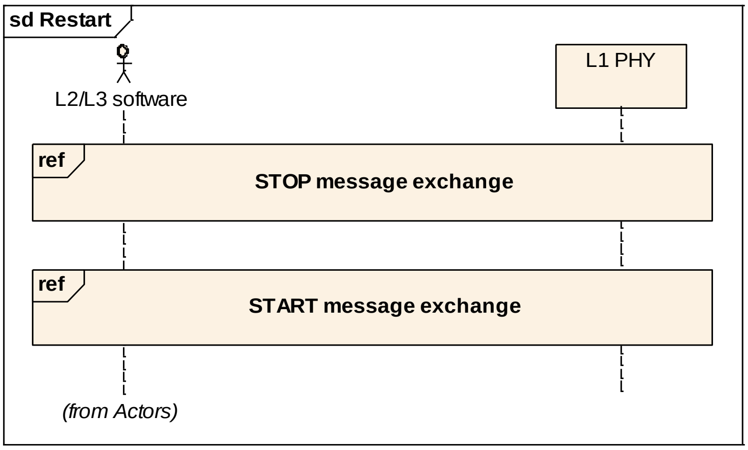

Figure 10: Restart procedure

2.1.4 Reset

The reset procedure is shown in Figure 11. This procedure is used when the L2/L3 software wants to return the PHY to the IDLE state. This can only be achieved by terminating the PHY (as shown in Figure 9) and then resetting the PHY. The method for resetting the PHY will be implementation specific and outside the scope of this document.

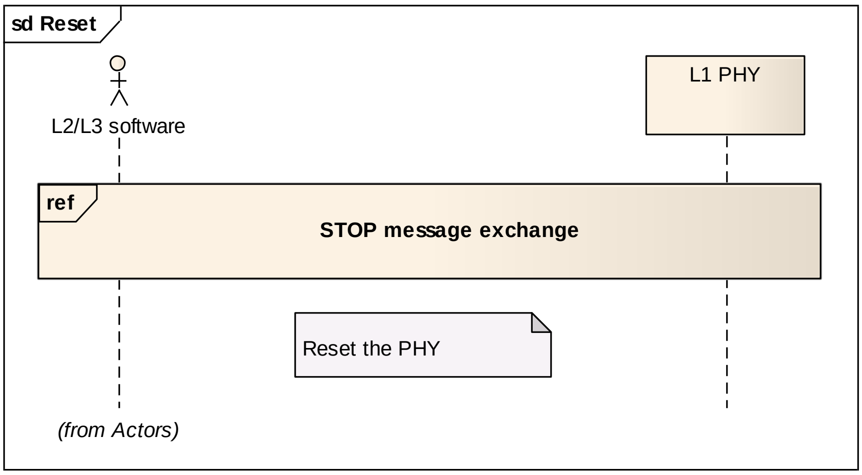

Figure 11: Reset procedure

2.1.5 Reconfigure

Two methods of reconfiguration are supported by the PHY. A major reconfiguration where the PHY is stopped, and a minor reconfiguration where the PHY continues running.

The major reconfigure procedure is shown in Figure 12. It is used when the L2/L3 software wants to make significant changes to the configuration of the PHY. The STOP message exchange, shown in Figure 9, is followed to halt the PHY and move it to the CONFIGURED state. The CONFIG message exchange, shown in Figure 7, is used to reconfigure the PHY. Finally, the START message exchange, shown in Figure 8, is followed to start the PHY and return it to the RUNNING state.

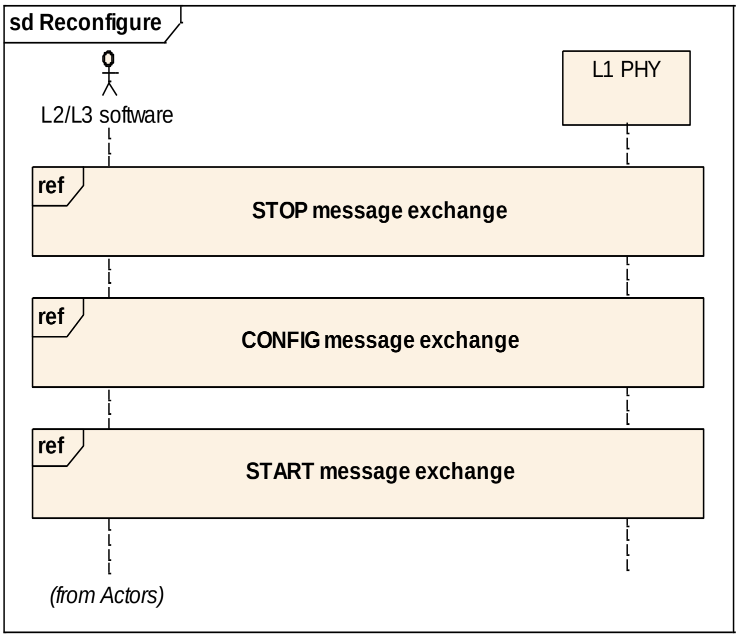

Figure 12: Major reconfiguration procedure

The minor reconfiguration procedure is shown in Figure 13. It is typically used in conjunction with a RRC system information update.

In the subframe where the L2/L3 software requires the configuration change it sends the `CONFIG.request` message to the PHY. Only a limited subset of CONFIG TLVs may be sent, these are highlighted later in Section 3.2.2.1. TLVs included in the `CONFIG.request` message for subframe N will be applied at the SFN/SF given in the `CONFIG.request` message. Reconfiguring the PHY while in the RUNNING state has a further restriction, the `CONFIG.request` message must be sent before the `DL_CONFIG.request` and `UL_CONFIG.request` message.

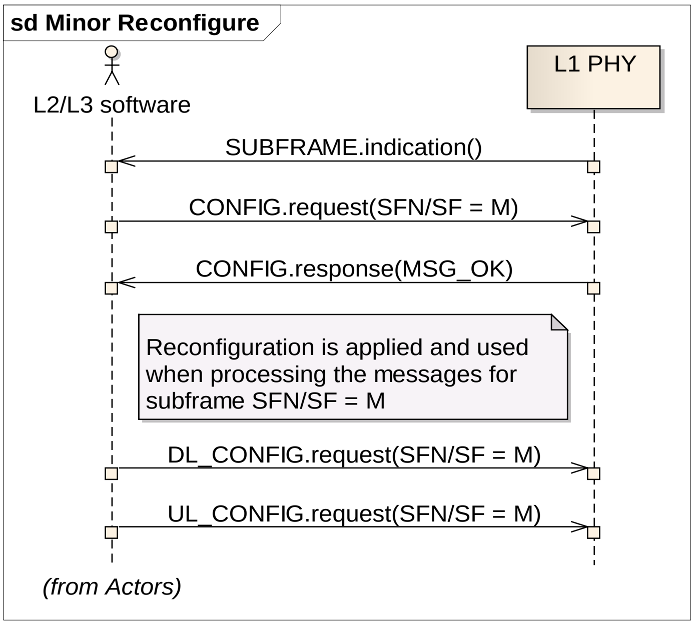

Figure 13: Minor reconfigure procedure

2.1.6 Query

The query procedure is shown in Figure 14. It is used by the L2/L3 software to determine the configuration and operational status of the PHY. The PARAM message exchange, shown in Figure 6, is used. This signalling sequence can be followed when the PHY is stopped, in the IDLE state and, optionally, the CONFIGURED state.

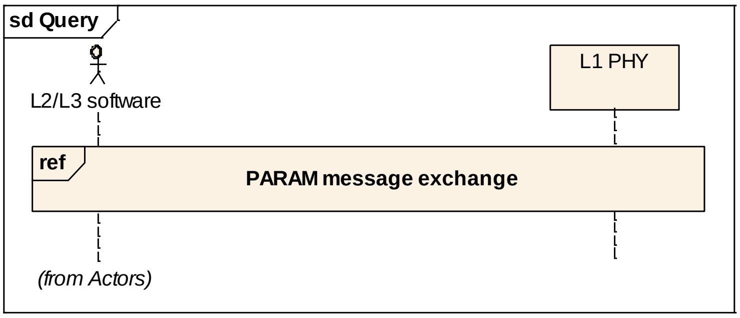

Figure 14: Query procedure

2.1.7 Notification

The notification procedure is shown in Figure 15. The PHY sends a notification message when it has an event of interest for the L2/L3 software. Currently, there is one notification message called `ERROR.indication`.

The `ERROR.indication` message has already been mentioned in multiple procedures. It is used by the PHY to indicate that the L2/L3 software has sent invalid information to the PHY.

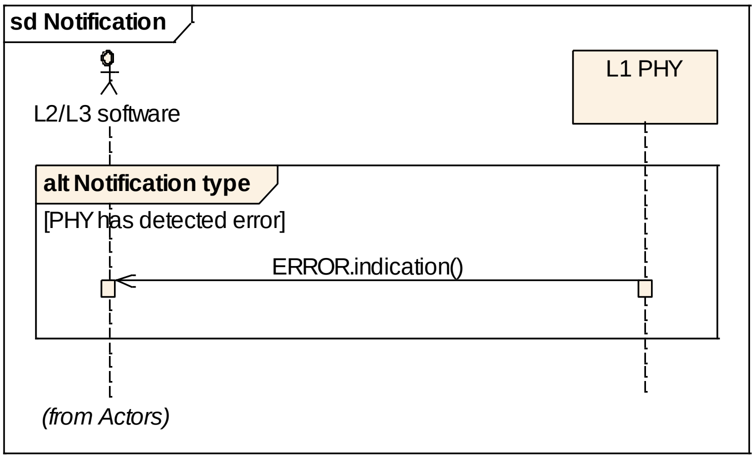

Figure 15: Notification procedures

2.2 Subframe Procedures

The subframe procedures have two purposes. Firstly, they are used to control the DL and UL frame structures. Secondly, they are used to transfer the subframe data between the L2/L3 software and PHY. The subframe procedures supported by the L1 API are:

- Transmission of a 1ms SUBFRAME message
- Synchronization of SFN/SF between the L2/L3 software and PHY
- Transmission of the BCH transport channel
- Transmission of the PCH transport channel
- Transmission of the DL-SCH transport channel and reception of ACK/NACK response
- Transmission of the MCH transport channel
- Reception of the RACH transport channel
- Reception of the UL-SCH transport channel and transmission of ACK/NACK response
- Reception of the sounding reference signal
- Reception of CQI and RI reporting
- Reception of scheduling request information

2.2.1 SUBFRAME Signal

A `SUBFRAME.indication` message is sent from the PHY, to the L2/L3 software, indicating the start of a 1ms subframe.

The periodicity of the `SUBFRAME.indication` message for TDD (frame structure 2) is shown in Figure 16 and Figure 17. In TDD two frame structures are possible, one with 5ms switch points and one with 10ms switch points [3]. The `SUBFRAME.indication` message is generated for every subframe (DL or UL).

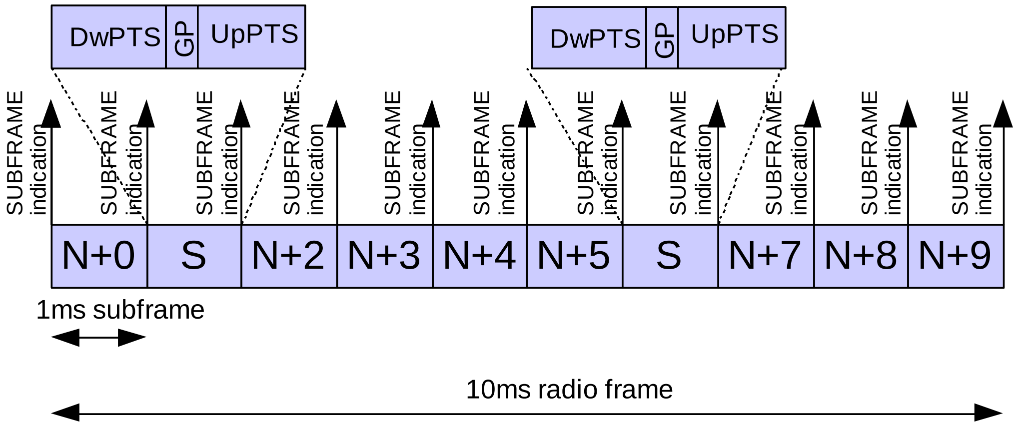

Figure 16: SUBFRAME signal for TDD using 5ms switch points

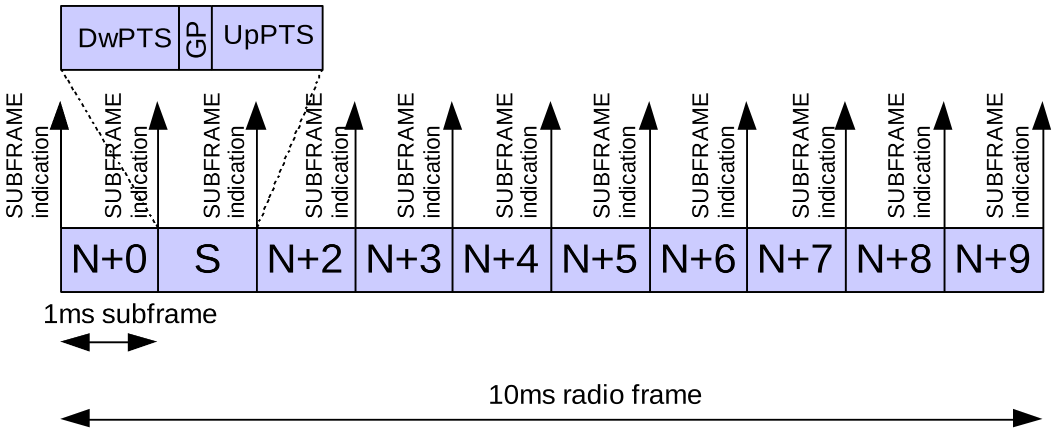

Figure 17: SUBFRAME signal for TDD using 10ms switch point

The periodicity of the `SUBFRAME.indication` message for FDD (frame structure 1) is shown in Figure 18. The subframe indication is generated for every DL subframe.

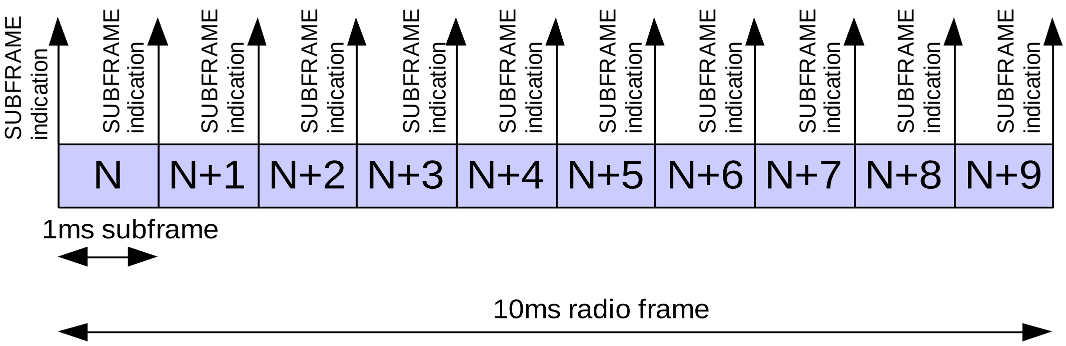

Figure 18: SUBFRAME signal for FDD

2.2.2 SFN/SF Synchronization

The SFN/SF synchronization procedure is used to maintain a consistent SFN/SF value between the L2/L3 software and PHY. Maintaining this synchronization is important since different subframes have different structures, and in TDD subframes are either downlink or uplink.

Two options are provided by the L1 API; the first option configures the PHY to use the SFN/SF value provided by the L2/L3 software. The second option configures the PHY to initialize the SFN/SF and ensure the L2/L3 software remains synchronous. The synchronization option is selected at compile time. For each option two procedures are described, the initial start-up synchronization and the maintenance of the synchronization.

2.2.2.1 L2/L3 Software is Master

The SFN/SF synchronization start-up procedure, where the L2/L3 software is master, is given in Figure 19. The start-up procedure followed is:

- After successful configuration the L2/L3 software sends a `START.request` message to move the PHY to the RUNNING state
- When the L2/L3 software is configured as master the initial PHY SFN/SF = M, where M could be any value. In the `SUBFRAME.indication` message, SFN/SF = M
- The L2/L3 software sends a `DL_CONFIG.request` message to the PHY containing the correct SFN/SF = N
- The PHY uses the SFN/SF received from the L2/L3 software. It changes its internal SFN/SF to match the value provided by the L2/L3 software

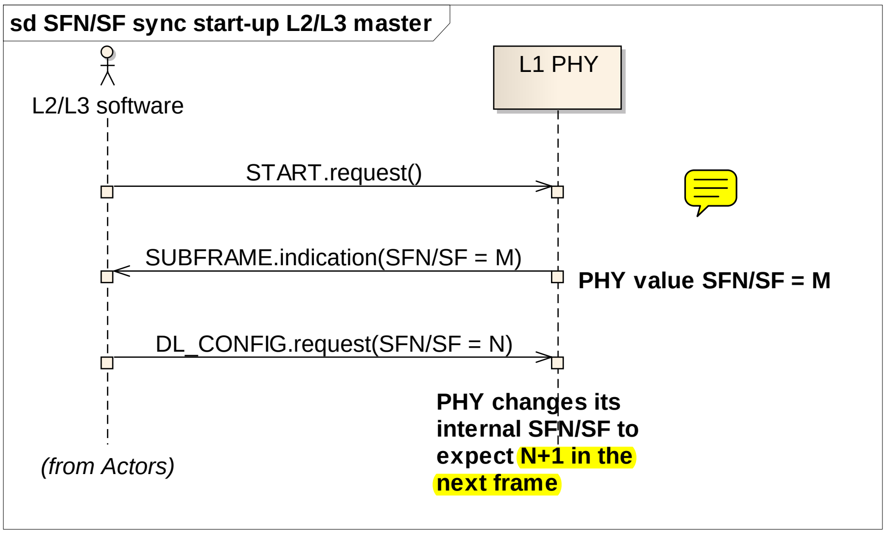

Figure 19: SFN/SF synchronization start-up with L2/L3 master

The SFN/SF synchronization maintenance procedure is shown in Figure 20. In this example, the L1 PHY is expecting the next DL_CONFIG.request to contain information regarding frame M. The procedure followed is:

- The PHY sends the SUBFRAME.indication message with SFN/SF = M.
- The L2/L3 software sends a DL_CONFIG.request message to the PHY containing SFN/SF = N
- If SFN/SF M = N
    - The PHY received the SFN/SF it was expecting. No SFN/SF synchronization is required
- If SFN/SF M ≠ N
    - The PHY received a different SFN/SF from the expected value. SFN/SF synchronization is required
    - The PHY uses the SFN/SF received from the L2/L3 software. It changes its internal SFN/SF to match the value provided by the L2/L3 software
    - The PHY returns an ERROR.indication message indicating the mismatch

This SFN/SF synchronization procedure assumes the L2/L3 software is always correct. However, it's possible the SFN/SF synchronization was unintended, and due to a L2/L3 software issue. The generation of an ERROR.indication message, with expected and received SFN/SF values, should allow the L2/L3 software to perform a correction with a further SFN/SF synchronization.

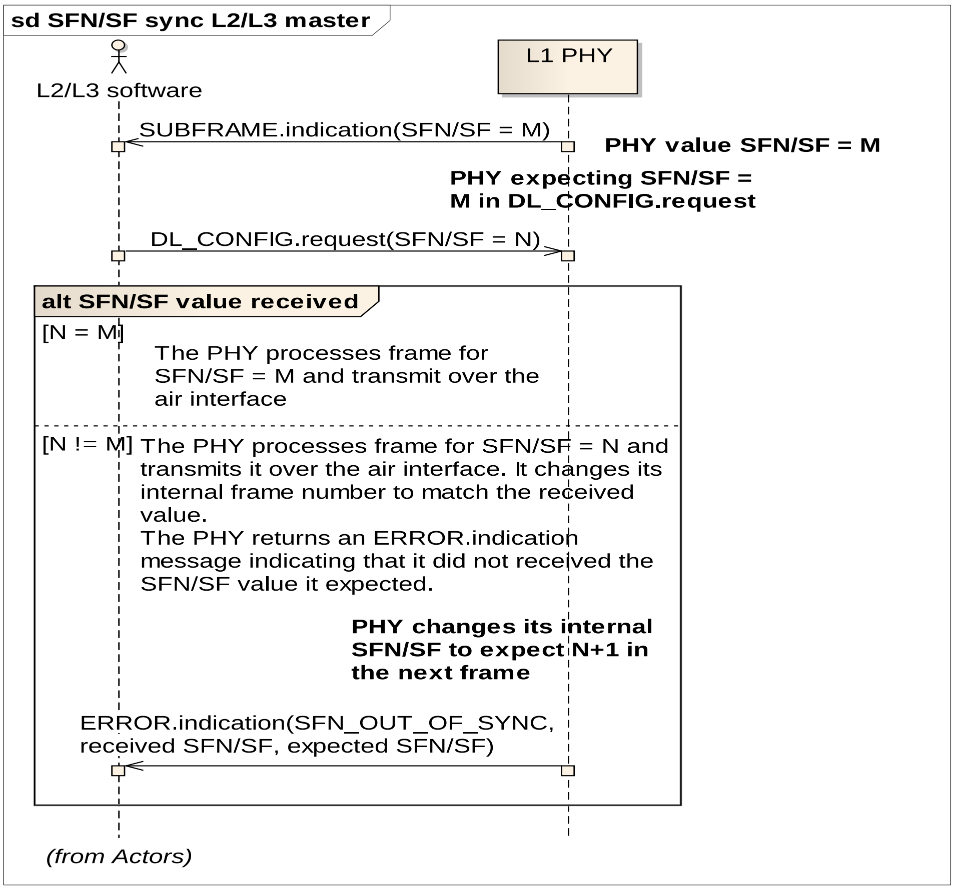

Figure 20: SFN/SF synchronization with L2/L3 master

2.2.2.2 L1 PHY is Master

The SFN/SF synchronization start-up procedure, where the L1 software is master, is given in Figure 21. The start-up procedure followed is:

- After successful configuration the L2/L3 software sends a `START.request` message to move the PHY to the RUNNING state
- If the L1 software is configured as master the initial PHY SFN/SF = M. The value of M is not deterministic, and could have been set by an external mechanism, such as GPS. The PHY sends a `SUBFRAME.indication` message to the L2/L3 software, with SFN/SF = M. The L2/L3 software uses the SFN/SF received from the PHY. It changes its internal SFN/SF to match the value provided by the PHY
- The L2/L3 software sends a `DL_CONFIG.request` message to the PHY containing SFN/SF = M

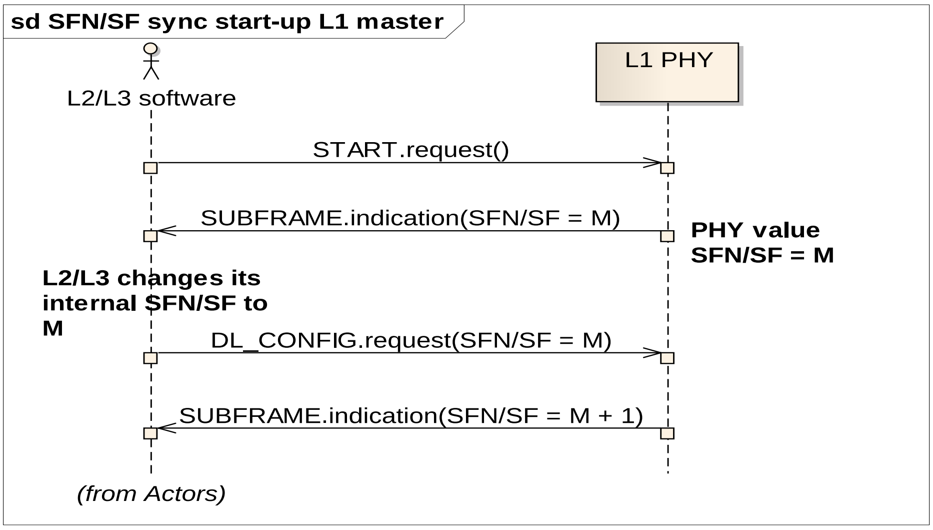

Figure 21: SFN/SF synchronization start-up with L1 master

The SFN/SF synchronization maintenance procedure is shown in Figure 22. In this example, the L1 PHY is expecting the next `DL_CONFIG.request` to contain information regarding frame M. The procedure followed is:

- The PHY sends a `SUBFRAME.indication` message to the L2/L3 software, with SFN/SF = M
- The L2/L3 software sends a `DL_CONFIG.request` message to the PHY containing SFN/SF = N
- If SFN/SF M = N
  - The PHY received the SFN/SF it was expecting. No SFN/SF synchronization is required
- If SFN/SF M ≠ N
  - The PHY received a different SFN/SF from the expected value. SFN/SF synchronization is required
  - The PHY discards the received `DL_CONFIG.request` message
  - The PHY returns an `ERROR.indication` message indicating the mismatch

This SFN/SF synchronization procedure will continue to discard `DL_CONFIG.request` messages and emit `ERROR.indication` messages until the L2/L3 software corrects its SFN/SF value.

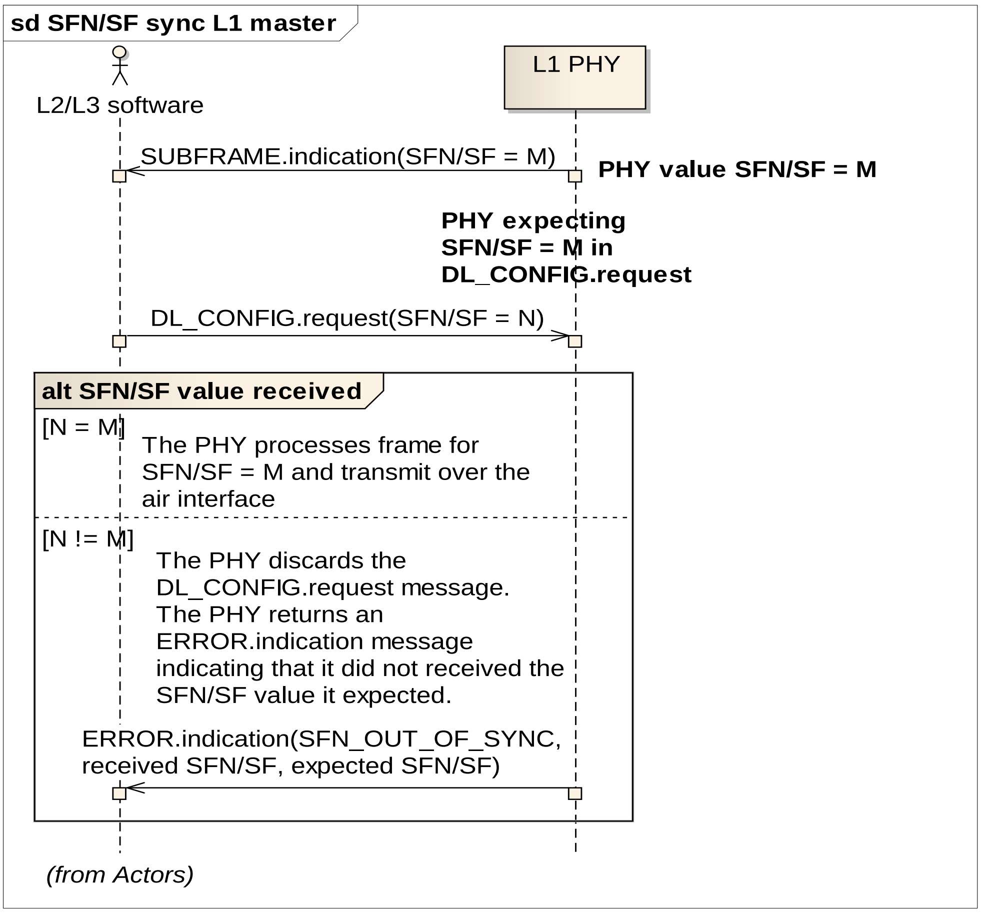

Figure 22: SFN/SF synchronization with L1 master

2.2.3 API Message Order

The L1 API has constraints of when certain subframe messages can be sent, or will be received, by the L2/L3 software.

The downlink API message constraints are shown in Figure 23:

- The SFN/SF included in the `SUBFRAME.indication` message is expected in the corresponding `DL_CONFIG.request`
- If the PHY is being reconfigured using the `CONFIG.request` message, this must be the first message for the subframe.
- If the PHY is being reconfigured using the `UE_CONFIG.request` message, this must be the next message for the subframe.
- The `DL_CONFIG.request` must be sent for every downlink subframe and must be the next message.
- The `UL_CONFIG.request` must be sent for every uplink subframe and must be the next message.
- The `TX.request` and `HI_DCI0.request` messages are optional. It is not a requirement that they are sent in every downlink subframe.
- There must be only 1 `DL_CONFIG.request`, 1 `UL_CONFIG.request`, 1 `HI_DCI0.request` and 1 `TX.request` for a subframe.

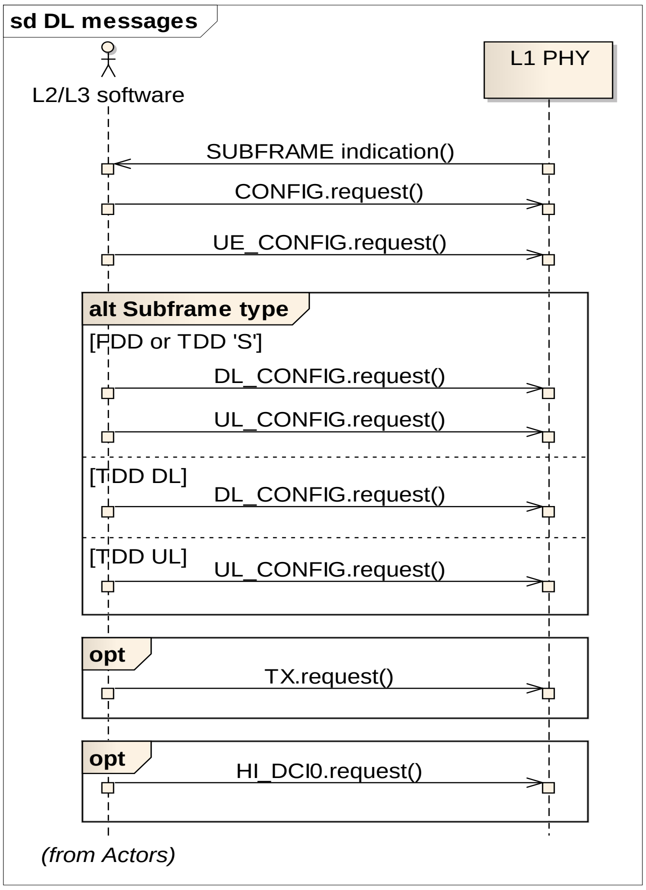

Figure 23: DL message order

The uplink API message constraints are shown in Figure 24:

- The UL API messages are optional. It is not a requirement that they are sent in every subframe.
- If present, the messages can be in any order
  - The `HARQ.indication` message is included if ACK/NACK responses were expected in the subframe.
  - The `CRC.indication` message is included if uplink data PDUs were expected in the subframe.
  - The `RX_ULSCH.indication` message is included if uplink data PDUs were expected in the subframe.
  - The `RX_SR.indication` message is included if SR PDUs were expected in the subframe.
  - The `RX_CQI.indication` message is included if CQI were expected in the subframe.
  - The `RACH.indication` message is included if any RACH preambles were detected in the subframe
  - The `SRS.indication` message is included if any sounding reference symbol information is expected in the subframe.
- There will be only 1 `HARQ.indication`, 1 `CRC.indication`, 1 `RX_ULSCH.indication`, 1 `RX_SR.indication`, 1 `RX_CQI.indication`, 1 `RACH.indication`, and 1 `SRS.indication` message per subframe

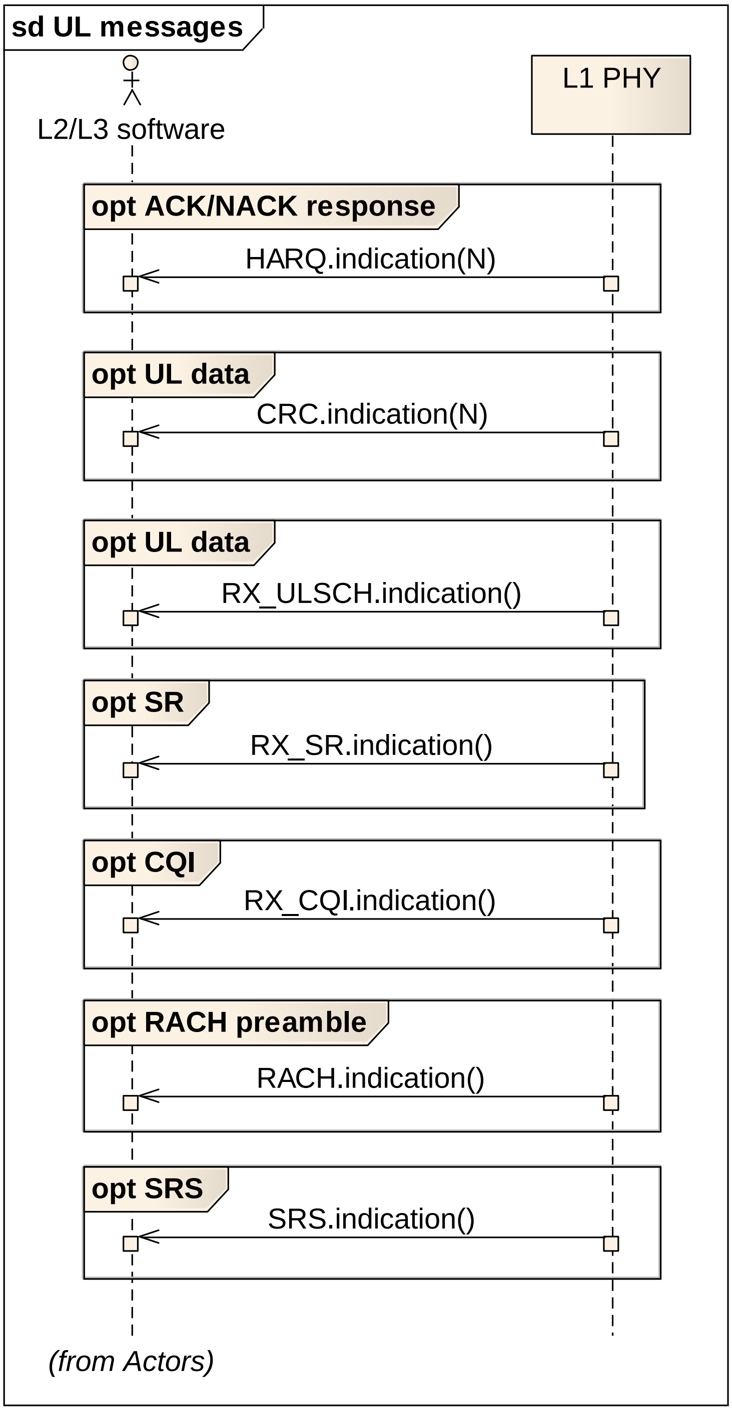

Figure 24: UL message order

2.2.4 Semi-Static Information

In LTE the majority of uplink and downlink data is determined by the scheduler on a subframe-by-subframe basis, however, there are several semi-static parameters which create periodic transmission patterns on the uplink. These semi-static parameters are either cell-specific or UE-specific. The cell-specific parameters are RACH and SRS regions which occur regularly and have a predefined pattern advertised on system information messages. The UE-specific parameters are CQI reporting, SR opportunities and SRS reporting, these are sent to the UE in RRC connection messages.

This L1 API supports the storage of the semi-static parameters in either the MAC or PHY. If stored in the MAC the L1 API is used to instruct the PHY when to allocate RACH and SRS regions, and when to expect the UE to transmit CQI, SR and SRS. If stored in the PHY semi-static parameters are passed from the MAC to PHY with the message exchange shown in Figure 25, a UE is release with the message exchange shown in Figure 26.

It is not expected that this parameter will be configurable; instead it will be a characteristic of the PHY.

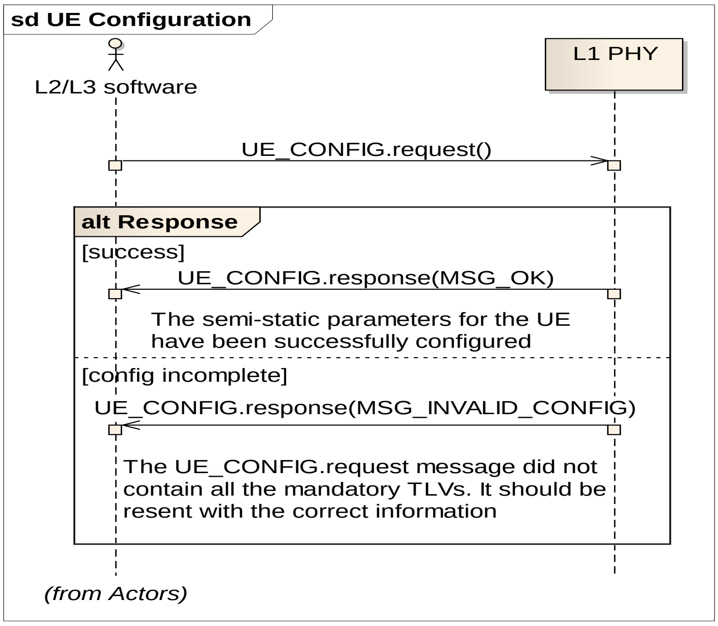

Figure 25: UE Configuration procedure

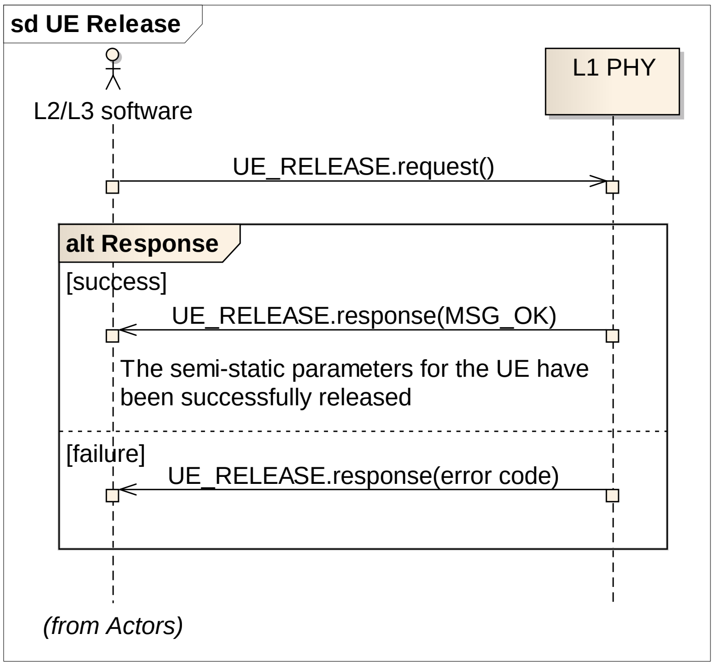

Figure 26: UE Release procedure

2.2.5 Uplink HARQ Signalling

Uplink HARQ signalling is used to acknowledge, or negatively acknowledge, downlink data transmissions sent to a UE. For FDD, the PUCCH location of this HARQ signalling is determined by the n_{CCE} value of the DCI which allocated the downlink grant. For TDD, the PUCCH location of this HARQ signalling is determined by a combination of the n_{CCE} value of the granting DCI and the DL-UL subframe configuration. Both the eNB and the UE need to calculate the HARQ location based on this information. In the eNB this calculation could be performed in either the MAC or PHY.

This L1 API supports the calculation of the uplink HARQ location in either the MAC or PHY. If the calculation is performed in the MAC the L1 API is used to instruct the PHY when and where to receive the HARQ If stored in the PHY HARQ parameters are passed from the MAC to PHY with the message exchange shown in Figure 25.

?

It is not expected that this parameter will be configurable; instead it will be a characteristic of the PHY.

2.2.6 Downlink

The procedures relating to downlink transmission are described in this Section.

2.2.6.1 BCH

The BCH transport channel is used to transmit the Master Information Block (MIB) information to the UE. The location of the MIB is defined in the LTE standards [1], and shown in Figure 27. It is transmitted in subframe 0 of each radio frame. When the radio frame (SFN mod 4) = 0 an updated MIB is transmitted in subframe 0. When the radio frame (SFN mod 4) ≠ 0 the MIB is repeated.

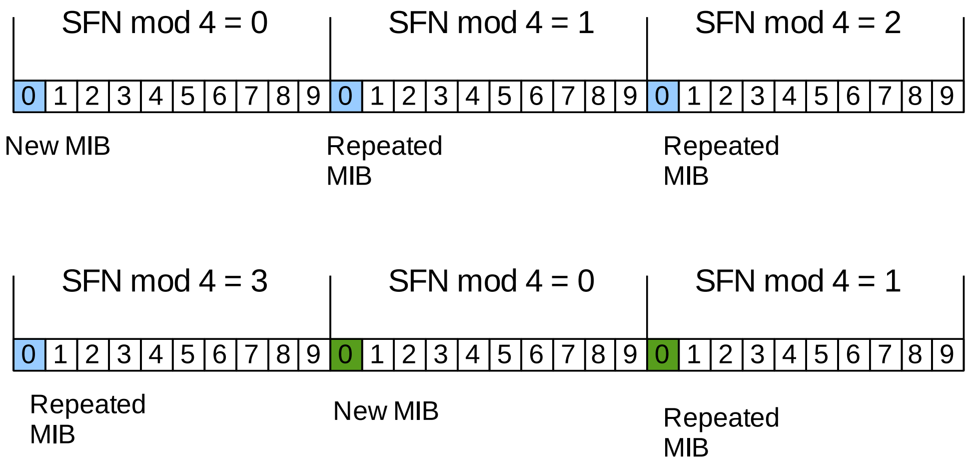

Figure 27: MIB scheduling on the BCH transport channel

The BCH procedure is shown in Figure 28. The L2/L3 software should provide a BCH PDU to the PHY in subframe SF=0, for each radio frame (SFN mod 4) = 0. This is once every 40ms. The L2/L3 software provides the following information:

- In `DL_CONFIG.request` a BCH PDU is included.
- In `TX.request` a MAC PDU containing the MIB is included.

If the PHY does not receive a BCH PDU in subframe SF=0, where radio frame (SFN mod 4) = 0, then no BCH will be transmitted.

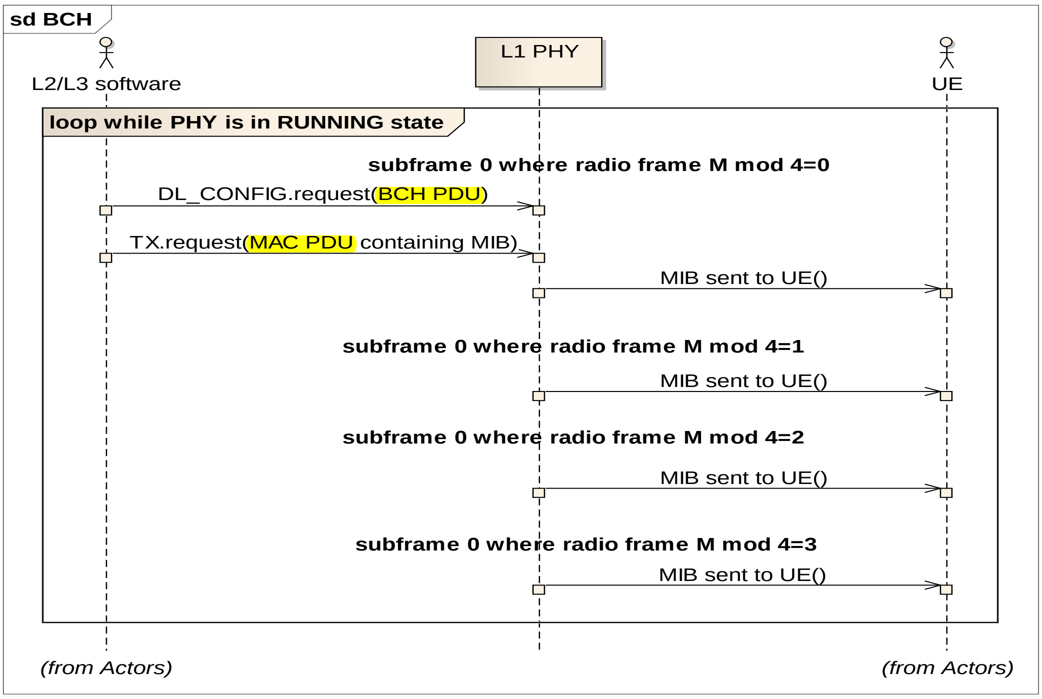

Figure 28: BCH procedure

2.2.6.2 PCH

The PCH transport channel is used to transmit paging messages to the UE. The UE has specific paging occasions where it listens for paging information [5]. The L2/L3 software is responsible for calculating the correct paging occasion for a UE. The PHY is only responsible for transmitting PCH PDUs when instructed by the `DL_CONFIG.request` message.

The PCH procedure is shown in Figure 29. To transmit a PCH PDU the L2/L3 software must provide the following information:

- In `DL_CONFIG.request` a PCH PDU and DCI PDU are included.
- In `TX.request` a MAC PDU containing the paging message is included.

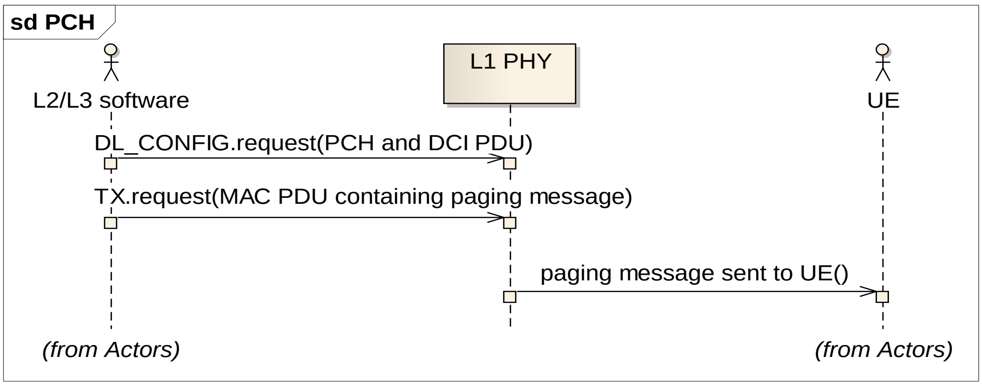

Figure 29: PCH procedure

2.2.6.3 DL-SCH

The DL-SCH transport channel is used to send data from the eNB to a single UE. This distinguishes the DL-SCH from the MCH, where data is sent to multiple UEs. HARQ is always applied on the DL-SCH transport channel. Therefore, together with scheduling downlink transmissions the L2/L3 software must schedule uplink bandwidth for the UE to return an ACK/NACK response.

The procedure for the DL-SCH transport channel is shown in Figure 30 To transmit a DL-SCH PDU the L2/L3 software must provide the following information:

干嘛用的？？？

- In `DL_CONFIG.request` a DLSCH PDU and DCI Format PDU are included. The DCI PDU contains control regarding the DL frame transmission
- In `TX.request` a MAC PDU containing the data is included
- If uplink HARQ signalling is calculated in the MAC a HARQ PDU is included in a later `UL_CONFIG.request`. The timing of this message is dependent on whether a TDD or FDD system is in operation. For TDD, it is also dependent on the DL/UL subframe configuration. There are multiple possible HARQ PDUs that can be used to indicate reception of the HARQ response on the uplink:
  - ULSCH_HARQ – is used if the UE is scheduled to transmit data and the ACK/NACK response
  - UCI_HARQ – is used if the UE is just scheduled to transmit the ACK/NACK response

  If the semi-static UE information is held in the MAC the following HARQ PDUs can also be used:
  - ULSCH_CQI_HARQ_RI – is used if the UE is scheduled to transmit data, a CQI report and the ACK/NACK response
  - UCI_SR_HARQ – is used if the UE has a SR opportunity and is scheduled to transmit the ACK/NACK response
  - UCI_CQI_HARQ – is used if the UE is scheduled to transmit a CQI report and the ACK/NACK response
  - UCI_CQI_SR_HARQ - is used if the UE is scheduled to transmit a CQI report, has a SR opportunity and is scheduled to transmit the ACK/NACK response
- If uplink HARQ signalling is calculated in the PHY no information is included in the L1 API regarding HARQ reception.
- The PHY will return the ACK/NACK response information in the `HARQ.indication` message

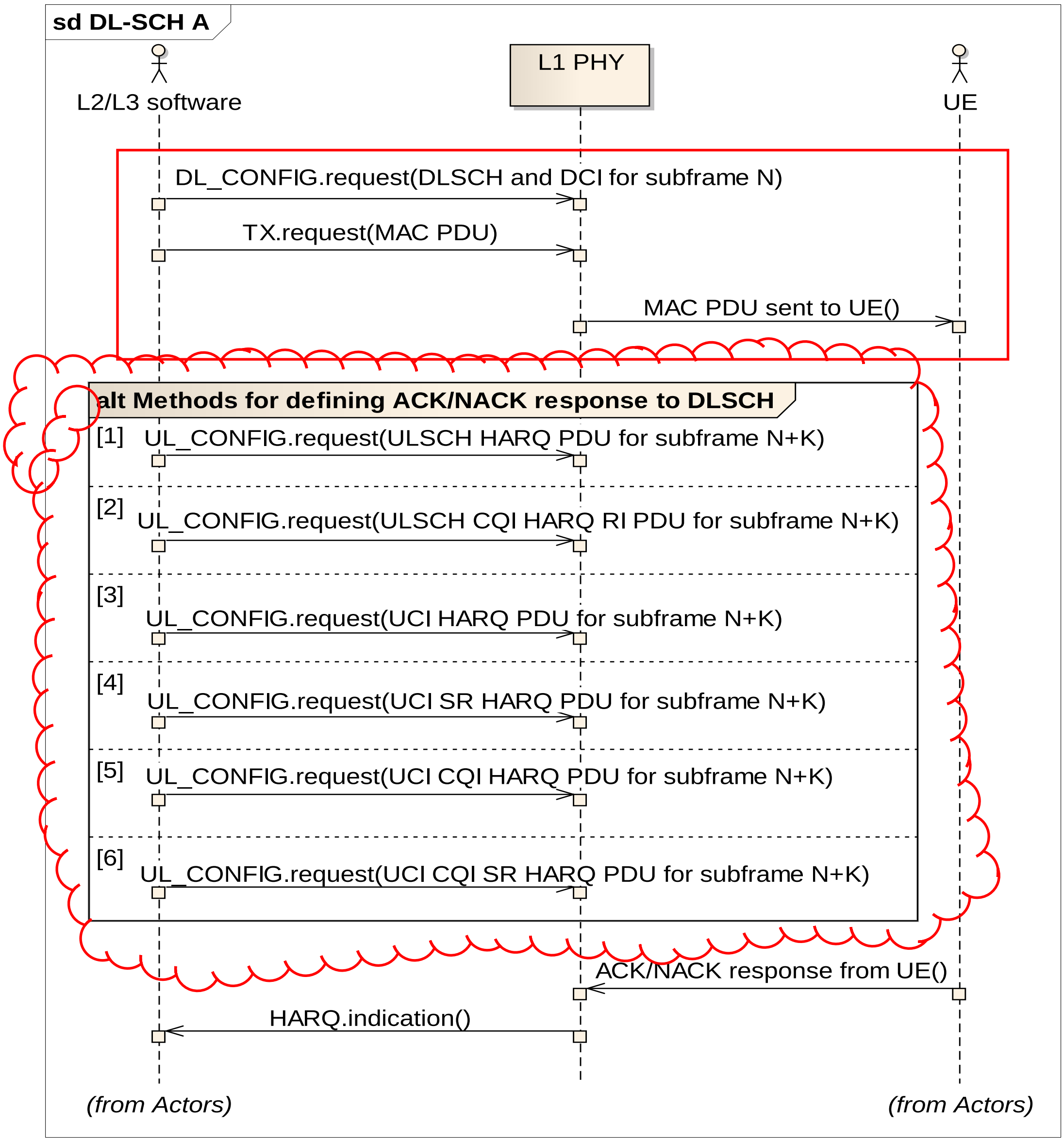

Figure 30: DLSCH procedure

With DCI Format 2 and 2A the DL SCH channel is combined to send two layer data transmission to a UE, this requires a single DCI PDU, but two DLSCH PDUs and two MAC PDUs. The procedure is shown in Figure 31. To initiate a two layer transmission the L2/L3 software must provide the following information:

- In `DL_CONFIG.request` a DCI Format 2 or 2A PDU is included. The DCI PDU contains control regarding the DL frame transmission. Two DLSCH PDUs are included one for each transport block specified in the DCI PDU.
- In `TX.request` two MAC PDUs containing the data are included.

(The remaining behaviour is identical to single-layer transmission)

- If uplink HARQ signalling is calculated in the MAC a HARQ PDU is included in a later `UL_CONFIG.request`. A single HARQ PDU is required to provide the ACK/NACK response for both transport blocks. The timing of this message is dependent on whether a TDD or FDD system is in operation. For TDD, it is also dependent on the DL/UL subframe configuration. There are multiple possible HARQ PDUs that can be used to indicate reception of the HARQ response on the uplink:
  - ULSCH_HARQ – is used if the UE is scheduled to transmit data and the ACK/NACK response

- UCI_HARQ – is used if the UE is just scheduled to transmit the ACK/NACK response

If the semi-static UE information is held in the MAC the following HARQ PDUs can also be used:

- ULSCH_CQI_HARQ_RI – is used if the UE is scheduled to transmit data, a CQI report and the ACK/NACK response
- UCI_SR_HARQ – is used if the UE is scheduled to transmit a scheduling request and the ACK/NACK response
- UCI_CQI_HARQ – is used if the UE is scheduled to transmit a CQI report and the ACK/NACK response
- UCI_CQI_SR_HARQ - is used if the UE is scheduled to transmit a CQI report, has a SR opportunity and is scheduled to transmit the ACK/NACK response

- If uplink HARQ signalling is calculated in the PHY no information is included in the L1 API regarding HARQ reception.
- The PHY will return the ACK/NACK response information in the `HARQ.indication` message

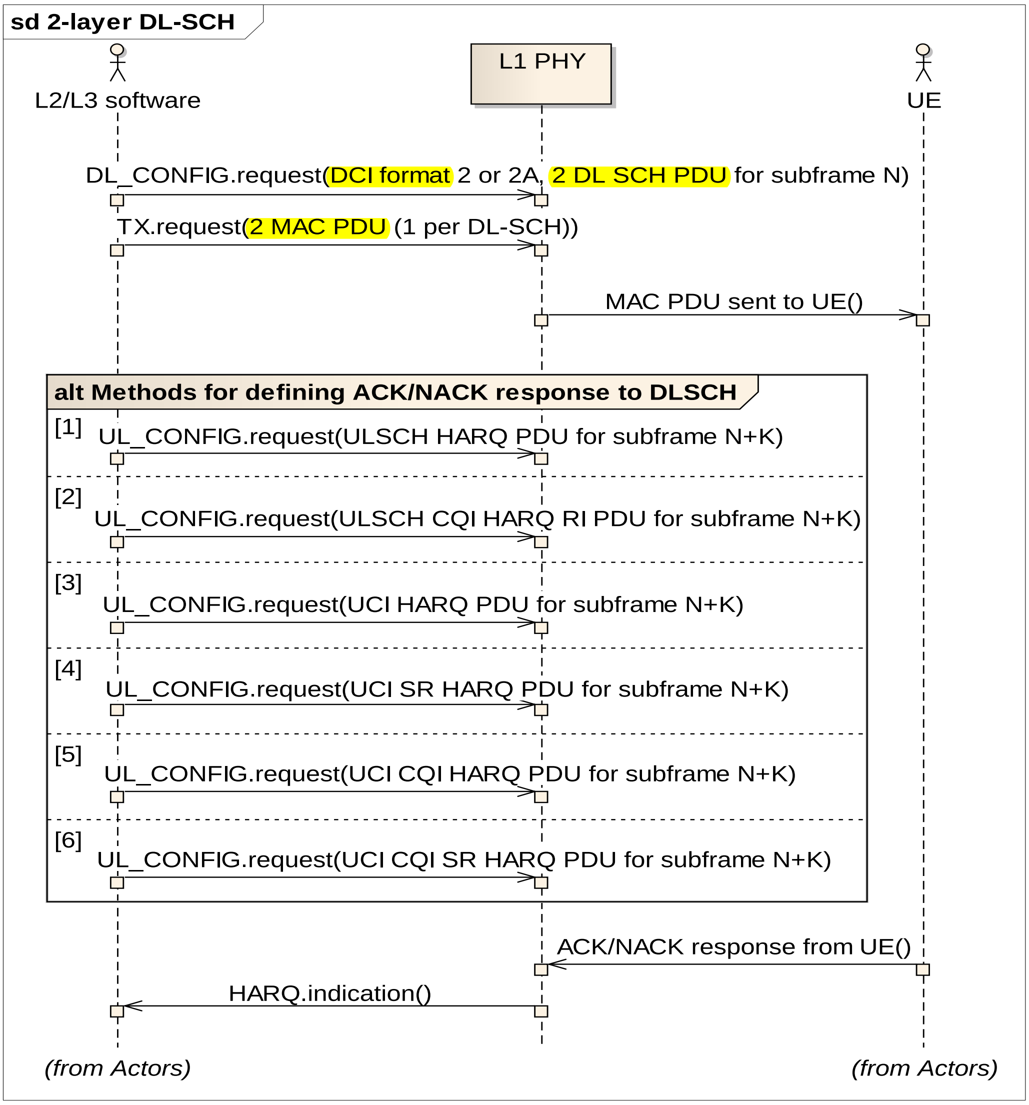

Figure 31: 2-layer DLSCH procedure

2.2.6.4 MCH

The MCH transport channel is used to send data, simultaneously, to multiple UEs. This means that HARQ is not used on this transport channel.

The MCH procedure is shown in Figure 32. To transmit a MCH PDU the L2/L3 software must provide the following information:

- In `DL_CONFIG.request` a MCH PDU is included
- In `TX.request` a MAC PDU containing the multicast data is included

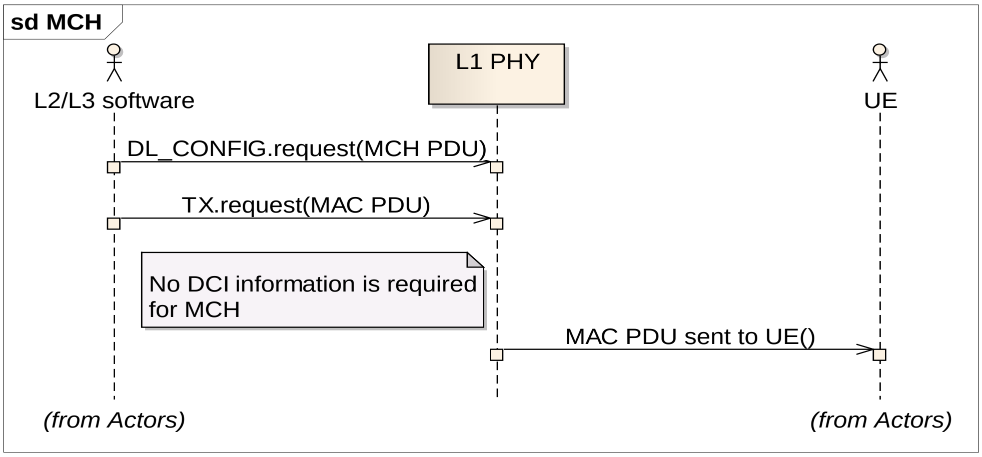

Figure 32: MCH procedure

比较好奇的一点，上行配置信息配置之后是传给了ue吗？ue需要获知这些信息吗？
师兄说通知了，每一次ul_config都会通知给ue。

2.2.7 Uplink

The procedures relating to uplink reception are described in this Section.

2.2.7.1 RACH

The RACH transport channel is used by the UE to send data to the eNB when it has no scheduled resources. Also, the L2/L3 software can indicate to the UE that it should initiate a RACH procedure. In LTE the occurrence of the RACH follows a pattern advertised on the System Information broadcast messages. The L1 API supports the storage of this information in either the MAC or PHY. If stored in the MAC the L1 API is used to instruct the PHY when it should allocate a PRACH, if stored in the PHY there is no need to include this information in the L1 API messages.

In the scope of the L1 API, the RACH procedure begins when the PHY receives a `UL_CONFIG.request` message indicating the presence of a RACH.

The RACH procedure is shown in Figure 33. To configure a RACH procedure the L2/L3 software must provide the following information:

- If the RACH pattern is stored in the MAC, in `UL_CONFIG.request` the RACH present field must be set. If the RACH pattern is stored in the PHY this step is not required.
- If a UE decides to RACH, and a preamble is detected by the PHY:
  - The PHY will include 1 RACH PDU in the `RACH.indication` message. This RACH PDU includes all detected preambles
- If no RACH preamble is detected by the PHY, then no `RACH.indication` message is sent

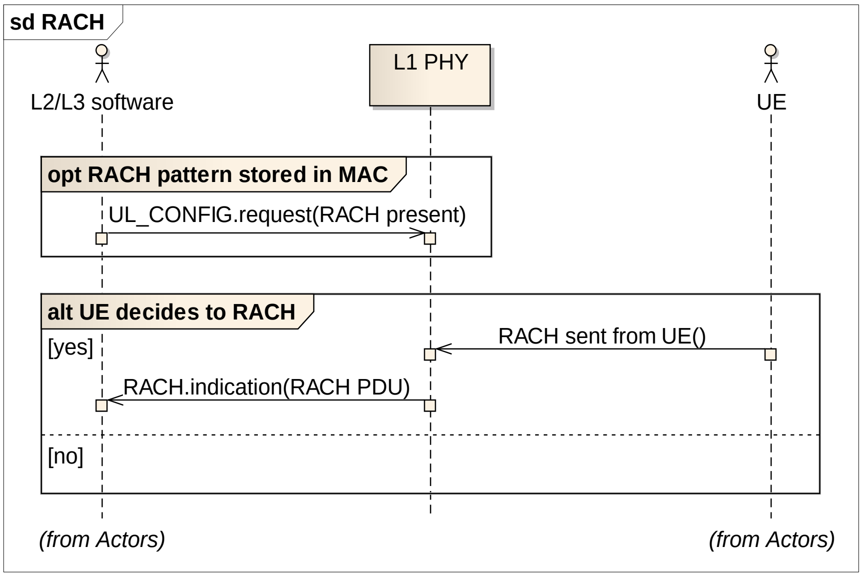

Figure 33: RACH procedure

2.2.7.2 UL-SCH

The UL-SCH transport channel is used to send data from the UE to the eNB. HARQ is always applied on the UL-SCH transport channel. Therefore, together with scheduling uplink transmissions the L2/L3 software must schedule downlink ACK/NACK responses.

The procedure for the UL-SCH transport channel is shown in Figure 34. To transmit an UL-SCH PDU the L2/L3 software must provide the following information:

- Within the `HI_DCI0.request` for subframe N a DCI PDU is included. The DCI Format 0 PDU contains control information regarding the UL frame transmission being scheduled.
- In `UL_CONFIG.request` for subframe N+K1 an ULSCH PDU is included. The timing of this message is dependent on whether a TDD or FDD system is in operation. For TDD, it is also dependent on the DL/UL subframe configuration. There are multiple possible ULSCH PDUs that can be used to schedule ULSCH data on the uplink:
  - ULSCH – is used if the UE is scheduled to only transmit data

  If the uplink HARQ signalling calculation is performed in the MAC the following ULSCH PDU can also be used:

  - ULSCH_HARQ – is used if the UE is scheduled to transmit data and an ACK/NACK response

  If the semi-static UE information is held in the MAC the following ULSCH PDUs can also be used:

  - ULSCH_CQI_RI – is used if the UE is scheduled to transmit data and a CQI report

  If both semi-static UE information and uplink HARQ signalling calculation is performed in the MAC the following ULSCH PDU can also be used:

  - ULSCH_CQI_HARQ_RI – is used if the UE is scheduled to transmit data, a CQI report and an ACK/NACK response

  ?
- If the Data Report Mode TLV = 0 in the `CONFIG.request` message, then:
  - The PHY will return CRC information for the received data in a the `CRC.indication` message
- The PHY will return the received uplink data in the `RX_ULSCH.indication` message. The `RX_ULSCH.indication` message repeats the CRC information given in the `CRC.indication` message.

- If a CQI was expected in the uplink subframe, the PHY will return the RX_CQI.indication message.
- The ACK/NACK response must be submitted to the PHY using a `HI_DCI0.request` message in subframe N+K1+K2. The timing of this message is dependent on whether a TDD or FDD system is in operation. For TDD, it is also dependent on the DL/UL subframe configuration.

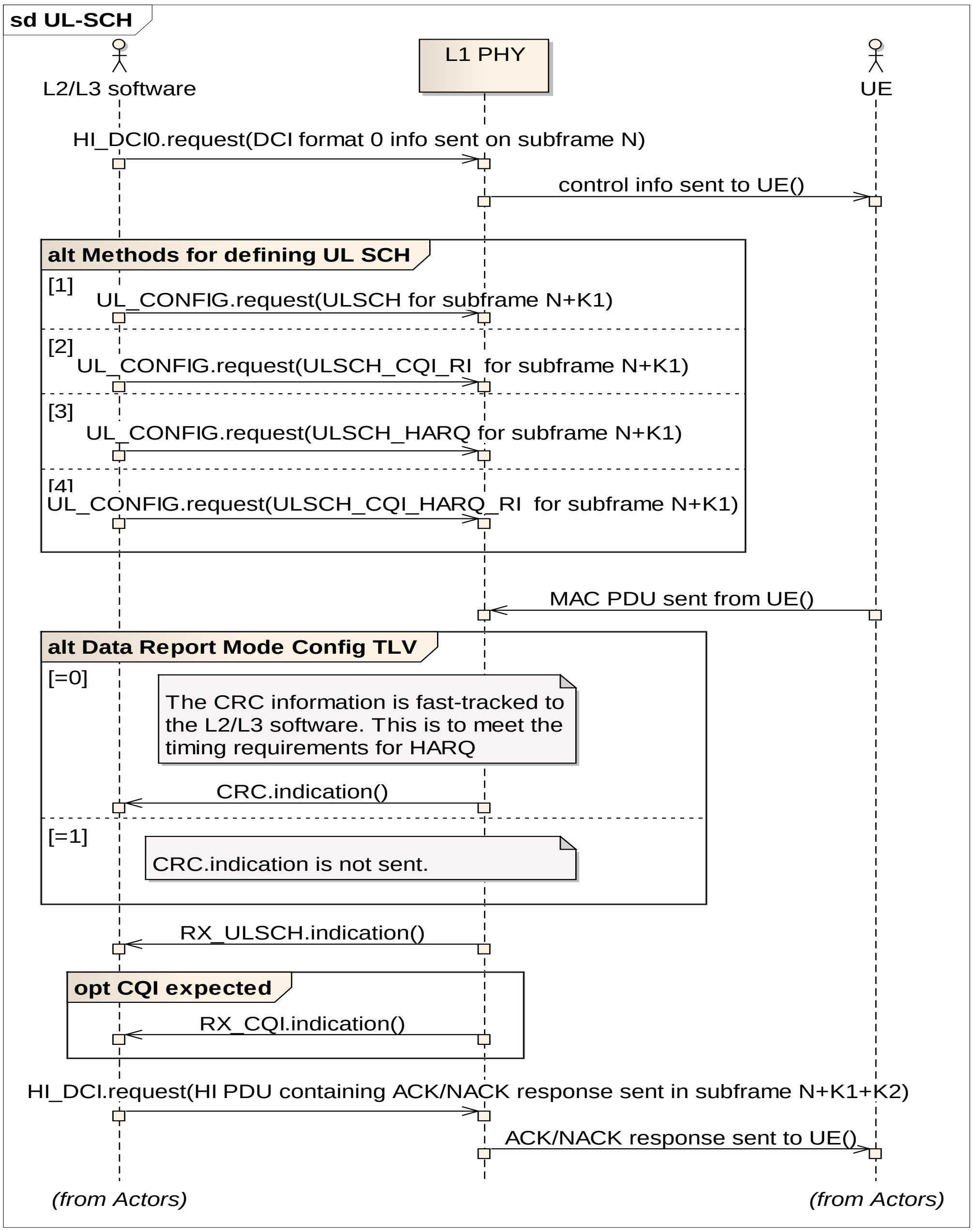

Figure 34: ULSCH procedure

2.2.7.3 SRS

The sounding reference signal (SRS) is used by L2/L3 software to determine the quality of the uplink channel. In LTE the occurrence of the SRS region follows a pattern advertised on the System Information broadcast messages. The transmission of the SRS by a UE is semi-static information. The L1 API supports the storage of this information in either the MAC or PHY. If stored in the MAC the L1 API is used to instruct the PHY when to allocate a SRS region and when to expect a SRS transmission from a UE, if stored in the PHY there is no need to include this information in the L1 API messages.

The SRS procedure is shown in Figure 35. To schedule a SRS the L2/L3 software must provide the following information:

- If the SRS information is stored in the MAC, in `UL_CONFIG.request` the SRS present field must be set and one SRS PDU per sounding UE is included. If the SRS information is stored in the PHY this step is not required.
- The PHY will return the SRS response to the L2/L3 software in the `SRS.indication` message

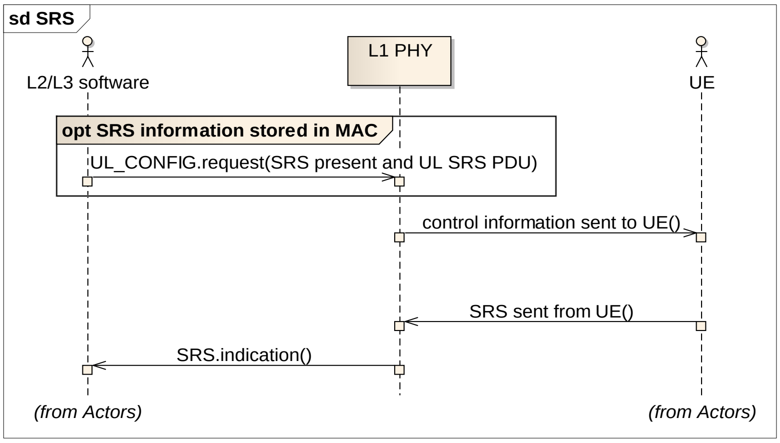

Figure 35: SRS procedure

2.2.7.4 CQI

The CQI reporting mechanism is used by the L2/L3 software to determine the quality of the downlink channel. CQI reporting is initiated through two methods. Firstly, during the RRC connection procedure the L2/L3 software will instruct the UE to transmit periodic CQI reports. Secondly, the L2/L3 software can use the PDCCH to instruct the UE to transmit an aperiodic CQI report.

The L1 API supports the storage of the periodic CQI information in either the MAC or PHY. If stored in the MAC the L1 API is used to instruct the PHY when to expect a CQI transmission from the UE, if stored in the PHY there is no need to include this information in the L1 API messages.

The CQI reporting procedure is shown in Figure 36. To schedule a CQI report the L2/L3 software must provide the following information:

- For an aperiodic report the DCI format 0 PDU is included in the `HI_DCI.request`. This instructs the UE to send a CQI report. For periodic CQI report no explicit DCI information is sent.
- If the CQI information is stored in the MAC:

- In the `UL_CONFIG.request`, where the L2/L3 software is expecting a CQI report, a CQI PDU is included. There are multiple possible CQI PDUs that can be used to indicate reception of the CQI report on the uplink:
  - ULSCH_CQI_RI – is used if the UE is scheduled to transmit data and a CQI report
  - UCI_CQI – is used if the UE is just scheduled to transmit a CQI report
  - UCI_CQI_SR– is used if the UE is just scheduled to transmit a CQI report and has a SR opportunity

  If the uplink HARQ signalling calculation is performed in the MAC the following CQI PDUs can also be used:
  - ULSCH_CQI_HARQ_RI – is used if the UE is scheduled to transmit data, a CQI report and the ACK/NACK response
  - UCI_CQI_HARQ - is used if the UE is scheduled to transmit a CQI report and the ACK/NACK response
  - UCI_CQI_SR_HARQ – is used if the UE is scheduled to transmit a CQI report, has a SR opportunity and is scheduled to transmit the ACK/NACK response
- If the CQI information is stored in the PHY, the L2/L3 software does not include any information relating to CQI in the `UL_CONFIG.request` message.
- The PHY will return the CQI report to the L2/L3 software in the `RX_CQI.indication` message

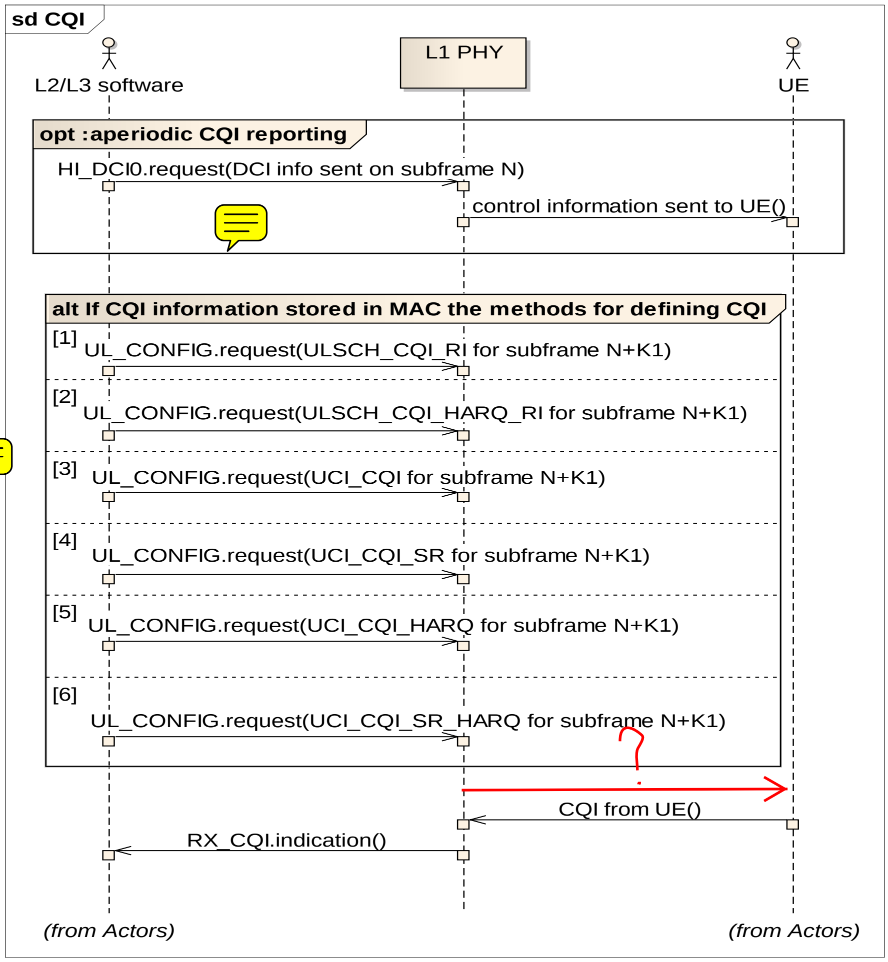

Figure 36: CQI procedure

2.2.7.5 SR

The scheduling request (SR) procedure is used by the UE to request additional uplink bandwidth. The L2/L3 software configures the SR procedure during the RRC connection procedure.

The L1 API supports the storage of the SR information in either the MAC or PHY. If stored in the MAC the L1 API is used to instruct the PHY when to expect a SR transmission from the UE, if stored in the PHY there is no need to include this information in the L1 API messages.

The SR procedure is shown in Figure 37. To schedule a SR the L2/L3 software must provide the following information:

- If the SR information is stored in the MAC:
  - In the `UL_CONFIG.request` a SR PDU is included. There are multiple possible SR PDUs that can be used to indicate reception of the SR on the uplink:
    - UCI_SR – is used if the UE has a SR opportunity
    - UCI_CQI_SR– is used if the UE is just scheduled to transmit a CQI report and has a SR opportunity

    If the uplink HARQ signalling calculation is performed in the MAC the following SR PDUs can also be used:

    - UCI_SR_HARQ – is used if the UE has a SR opportunity and is scheduled to transmit the ACK/NACK response
    - UCI_CQI_SR_HARQ – is used if the UE is scheduled to transmit a CQI report, has a SR opportunity and is scheduled to transmit the ACK/NACK response
- If the SR information is stored in the PHY, the L2/L3 software does not include any information relating to SR in the `UL_CONFIG.request` message.
- The PHY will return the SR to the L2/L3 software in the `RX_SR.indication` message

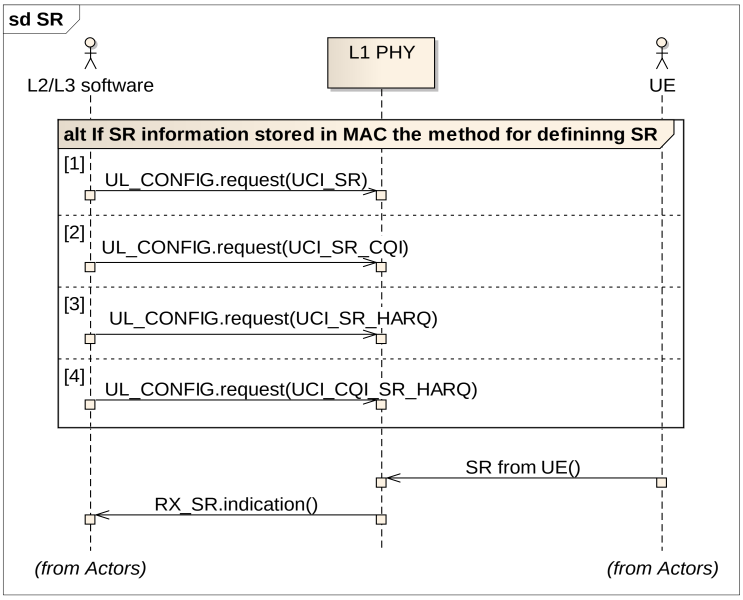

Figure 37: SR procedure

2.2.8 Error Sequences

The error sequences used for each subframe procedure are shown in Figure 38 to Figure 41. In all subframe procedures errors that are detected by the PHY are reported using the ERROR.indication message. In Section 3, the L1 API message definitions include a list of error codes applicable for each message.

The DL_CONFIG.request, UL_CONFIG.request, HI_DCI0.request and TX.request messages include information destined for multiple UEs. An error in information destined for one UE can affect a transmission destined for a different UE. For each message the ERROR.indication sent by the PHY will return the first error it encountered.

If the L2/L3 software receives an ERROR.indication message for DL_CONFIG.request, UL_CONFIG.request, HI_DCI0.request or TX.request, it should assume that the UE did not receive data and control sent in this subframe. This is similar to the UE experiencing interference on the air-interface and LTE mechanisms, such as, HARQ and ARQ, will enable the L2/L3 software to recover.

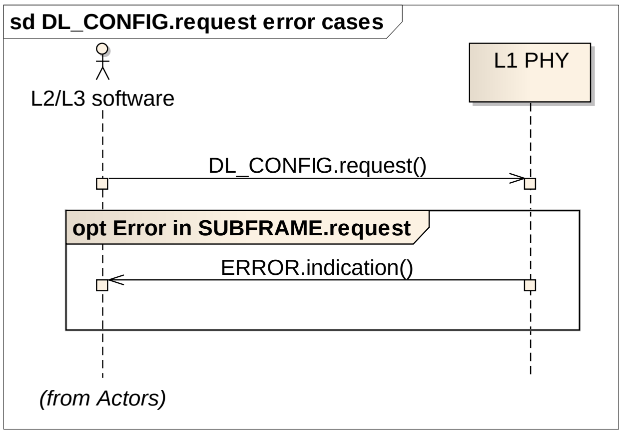

Figure 39: DL_CONFIG.request error sequence

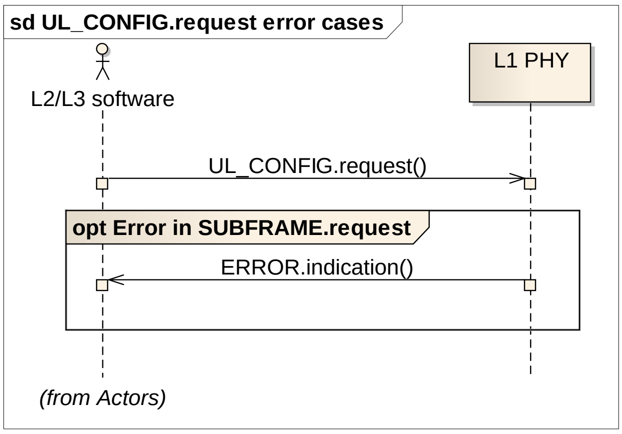

Figure 38: UL_CONFIG.request error sequence

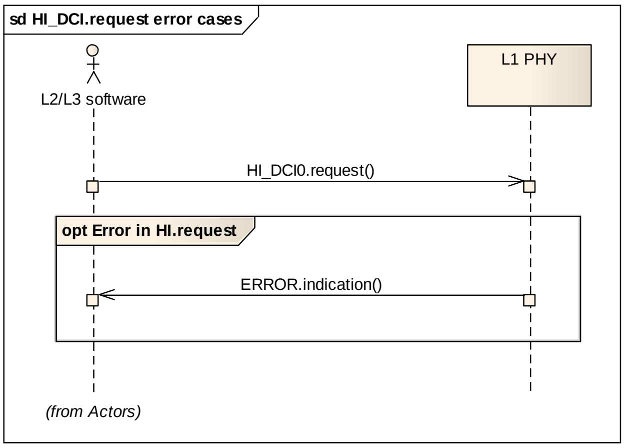

Figure 40: HI_DCI0.request error sequence

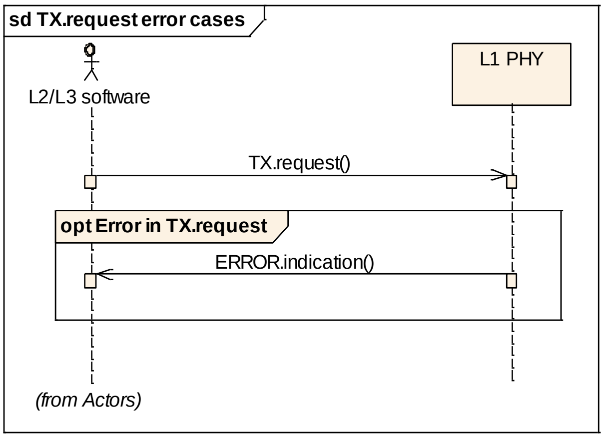

Figure 41: `TX.request` error sequence

3 L1 API Messages

This section provides a description of the L1 API message formats. It defines the L1 API message header, the message bodies and the error codes associated with the L1 API.

3.1 General Message Format

The general message format of the L1 API is shown in Table 3, where it can be seen that each L1 API message consists of a header followed by a message body.

The generic header consists of a message type ID, a message body length and a vendor-specific body length. The current list of message types is given in Table 4. The L1 API messages follow a standard naming convention where:

- All `.request` messages are sent from the L2/L3 software to the PHY.
- All `.response` messages are sent from the PHY to the L2/L3 software. These are sent in response to a .request.
- All `.indication` messages are sent from the PHY to the L2/L3 software. These are sent asynchronously.

The message body is different for each message type; however, each message body obeys the following rules:

- The first field in each response message is an error code. For each message it is indicated which error codes can be returned. A full list of error codes is given in Section 3.4.

A full description of each message body is given in the remainder of Section 3.

The API mechanism can use either little-endian byte order, or big-endian byte order. The selection of byte ordering is implementation specific. This document assumes that the API messages are transferred using a reliable in-order delivery mechanism.

Type	*Description*
uint8_t	Message type ID
uint8_t	Length of vendor-specific message body (bytes)
uint16_t	Length of message body (bytes)
Message body	
Vendor-specific message body	

Table 3: General L1 API message format

Message	*Value*	*Message Body Definition*
`PARAM.request`	0x00	See Section 3.2.1.1
`PARAM.response`	0x01	See Section 3.2.1.2
`CONFIG.request`	0x02	See Section 3.2.2.1
`CONFIG.response`	0x03	See Section 3.2.2.2
`START.request`	0x04	See Section 3.2.4.1
`STOP.request`	0x05	See Section 3.2.5.1
`STOP.indication`	0x06	See Section 3.2.5.2
`UE_CONFIG.request`	0x07	See Section 3.2.6.1
`UE_CONFIG.response`	0x08	See Section 3.2.6.2
`ERROR.indication`	0x09	See Section 3.2.8.1
`UE_RELEASE.request`	0x0a	See Section 3.2.7.1
`UE_RELEASE.response`	0x0b	See Section 3.2.7.2
RESERVED	0x0c-0x7f	
`DL_CONFIG.request`	0x80	See Section 3.3.1.2
`UL_CONFIG.request`	0x81	See Section 3.3.1.3
`SUBFRAME.indication`	0x82	See Section 3.3.1.1

Message	Value	Message Body Definition
HI_DCI0.request	0x83	See Section 3.3.1.3
TX.request	0x84	See Section 3.3.2.1
HARQ.indication	0x85	See Section 3.3.3.2
CRC.indication	0x86	See Section 3.3.3.3
RX_ULSCH.indication	0x87	See Section 3.3.3.1
RACH.indication	0x88	See Section 3.3.3.4
SRS.indication	0x89	See Section 3.3.3.7
RX_SR.indication	0x8a	See Section 3.3.3.4
RX_CQI.indication	0x8b	See Section 3.3.3.5
RESERVED	0x8c-0xff	

Table 4: L1 API Message Types

3.2 Configuration Messages

The configuration messages are used by the L2/L3 software to control and configure the PHY.

3.2.1 PARAM

The PARAM message exchange was described in Figure 6.

3.2.1.1 PARAM.request

This message can be sent by the L2/L3 when the PHY is in the IDLE state and, optionally, the CONFIGURED state. If it is sent when the PHY is in the RUNNING state, a MSG_INVALID_STATE error is returned in `PARAM.response`. No message body is defined for `PARAM.request`. The message length in the generic header = 0.

3.2.1.2 PARAM.response

The `PARAM.response` message is given in Table 5. From this table it can be seen that `PARAM.response` contains a list of TLVs providing information about the PHY. When the PHY is in the IDLE state this information relates to the PHY's overall capability. When the PHY is in the CONFIGURED state this information relates to the current configuration.

The full list of TLVs is given in Section 3.2.3. However, the set of TLVs which will be returned in the `PARAM.response` message depends on whether the PHY is TDD, or FDD, and on the current operational state of the PHY. Table 6 to Table 9 provide clarification on when a TLV will be included. Note: There is no requirement for the PHY to return the TLV's in the order specified in the Table.

Field	Type	Description
Error Code	uint8_t	See Table 86.
Number of TLVs	uint8_t	Number of TLVs contained in the message body.

Field	Type	Description
TLVs	Variable	See Table 6 to Table 9.

Table 5: `PARAM.response` message body

Description	Tag
PHY State	60
Downlink Bandwidth Support	40
Uplink Bandwidth Support	41
Downlink Modulation Support	42
Uplink Modulation Support	43
PHY Antenna Capability	44

Table 6: TLVs included in `PARAM.response` for TDD when PHY is in IDLE state

Description	Tag
PHY State	60
Downlink Bandwidth Support	40
Uplink Bandwidth Support	41
Downlink Modulation Support	42
Uplink Modulation Support	43
PHY Antenna Capability	44
Duplexing Mode	1
PCFICH Power Offset	2
P-B	3
DL Cyclic Prefix Type	4
UL Cyclic Prefix Type	5
RL Config	All TLVs in this grouping
PHICH Config	All TLVs in this grouping
SCH Config	All TLVs in this grouping
PRACH Config	All TLVs in this grouping

Description	Tag
PUSCH Config	All TLVs in this grouping
PUCCH Config	All TLVs in this grouping
SRS Config	All TLVs in this grouping
Uplink Reference Signal Config	All TLVs in this grouping
TDD Frame Structure Config	All TLVs in this grouping
Data Report Mode	50

Table 7: TLVs included in `PARAM.response` for TDD when PHY is in CONFIGURED state

Description	Tag
PHY State	60
Downlink Bandwidth Support	40
Uplink Bandwidth Support	41
Downlink Modulation Support	42
Uplink Modulation Support	43
PHY Antenna Capability	44

Table 8: TLVs included in `PARAM.response` for FDD when PHY is in IDLE state

Description	Tag
PHY State	60
Downlink Bandwidth Support	40
Uplink Bandwidth Support	41
Downlink Modulation Support	42
Uplink Modulation Support	43
PHY Antenna Capability	44
Duplexing Mode	1
PCFICH Power Offset	2
P-B	3
DL Cyclic Prefix Type	4

Description	*Tag*
UL Cyclic Prefix Type	5
RL Config	All TLVs in this grouping
PHICH Config	All TLVs in this grouping
SCH Config	All TLVs in this grouping
PRACH Config	All TLVs in this grouping
PUSCH Config	All TLVs in this grouping
PUCCH Config	All TLVs in this grouping
SRS Config	All TLVs in this grouping
Uplink Reference Signal Config	All TLVs in this grouping
Data Report Mode	50

Table 9: TLVs included in `PARAM.response` for FDD when PHY is in CONFIGURED state

3.2.1.3 PARAM Errors

The error codes which may be returned in `PARAM.response` are given in Table 10.

Error Code	*Description*
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `PARAM.request` was received when the PHY was in the RUNNING state.

Table 10: Error codes for `PARAM.response`

3.2.2 CONFIG

The CONFIG message exchange was described in Figure 7.

3.2.2.1 CONFIG.request

The `CONFIG.request` message is given in Table 11. From this table it can be seen that `CONFIG.request` contains a list of TLVs describing how the PHY should be configured. This message may be sent by the L2/L3 software when the PHY is in any state.

The full list of TLVs is given in Section 3.2.3. However, when the PHY is in the IDLE state there is a list of mandatory TLVs that must be included. For clarification Table 12 and Table 13 are provided. These indicate mandatory TLVs, which must be sent when the PHY is in IDLE state, and may be sent when the PHY is in the CONFIGURED state. The tables, also, indicate optional TLVs which may be sent when the PHY is in either the IDLE or CONFIGURED state. There is no requirement for the L2/L3 software to provide the TLVs in the order specified in the Tables.

When the PHY is in the RUNNING state a limited subset of the TLVs may be included. These TLVs are indicated in Table 14. Again, there is no requirement for the L2/L3 software to provide the TLVs in the order specified in the table.

Field	*Type*	*Description*
Number of TLVs	uint8_t	Number of TLVs contained in the message body.
TLVs	Variable	See Table 12 to Table 14.

Table 11: `CONFIG.request` message body

Description	*Tag*
Mandatory TLVs – These must be included when the PHY is in IDLE state, they may also be included when the PHY is in CONFIGURED state.	
Duplexing Mode	1
PCFICH Power Offset	2
P-B	3
DL Cyclic Prefix Type	4
UL Cyclic Prefix Type	5
RL Config	All TLVs in this grouping
PHICH Config	All TLVs in this grouping
SCH Config	All TLVs in this grouping
PRACH Config	All TLVs in this grouping
PUSCH Config	All TLVs in this grouping
PUCCH Config	All TLVs in this grouping
SRS Config	All TLVs in this grouping
Uplink Reference Signal Config	All TLVs in this grouping
TDD Frame Structure Config	All TLVs in this grouping
Data Report Mode	50
Optional TLVs – These may be included when the PHY is in either IDLE or CONFIGURED state.	
Currently there are no optional TLVs	

Table 12: TLVs included in `CONFIG.request` for TDD for IDLE and CONFIGURED states

Description	*Tag*
Mandatory TLVs – These must be included when the PHY is in IDLE state, they may also be included when the PHY is in CONFIGURED state.	
Duplexing Mode	1
PCFICH Power Offset	2
P-B	3
DL Cyclic Prefix Type	4
UL Cyclic Prefix Type	5
RL Config	All TLVs in this grouping
PHICH Config	All TLVs in this grouping
SCH Config	All TLVs in this grouping
PRACH Config	All TLVs in this grouping
PUSCH Config	All TLVs in this grouping
PUCCH Config	All TLVs in this grouping
SRS Config	All TLVs in this grouping
Uplink Reference Signal Config	All TLVs in this grouping
Data Report Mode	50
Optional TLVs – These may be included when the PHY is in either IDLE or CONFIGURED state.	
Currently there are no optional TLVs	

Table 13: TLVs included in `CONFIG.request` for FDD for IDLE and CONFIGURED states

Description	*Tag*
SFN/SF	51
Other TLVs are FFS	

Table 14: TLVs permitted in `CONFIG.request` in the RUNNING state

3.2.2.2 CONFIG.response

The `CONFIG.response` message is given in Table 15. If the configuration procedure was successful then the error code returned will be MSG_OK and no TLV tags will be included. If the configuration procedure was unsuccessful then MSG_INVALID_CONFIG will be returned, together with a list of TLVs identifying the problem.

Field	Type	Description
Error Code	uint8_t	See Table 86.
Number of Invalid or Unsupported TLVs	uint8_t	Number of invalid or unsupported TLVs contained in the message body.
Number of Missing TLVs	uint8_t	Number of missing TLVs contained in the message body. If the PHY is in the CONFIGURED state this will always be 0.
A list of invalid or unsupported TLVs – each TLV is presented in its entirety.		
TLV	Variable	Complete TLVs
A list of missing TLVs – each TLV is presented in its entirety		
TLV	Variable	Complete TLVs

Table 15: `CONFIG.response` message body

3.2.2.3 CONFIG Errors

The error codes that can be returned in `CONFIG.response` are given in Table 16.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_CONFIG	The configuration provided has missing mandatory TLVs, or TLVs that are invalid or unsupported in this state.

Table 16: Error codes for `CONFIG.response`

3.2.3 Configuration TLVs

The configuration TLVs that are used in the PARAM and CONFIG message exchanges follow the format given in Table 17. Each TLV consists of; a Tag parameter of 1 byte, a Length parameter of 1 byte and a Value parameter. The length of the Value parameter ensures the complete TLV is a multiple of 4-bytes (32-bits).

The individual TLVs are defined in Table 18.

Type	Description
uint8_t	Tag
uint8_t	Length (in bytes)
uint16_t	Value

Table 17: TLV format

Description	*Tag*	*Type*	*Value*
These TLVs are used by the L2/L3 software to configure a physical parameter in L1.			
Duplexing Mode	1	uint16_t	Type of duplexing mode Value : 0 : TDD, 1 : FDD, 2 : HD_FDD
PCFICH Power Offset	2	uint16_t	The power per antenna of the PCFICH with respect to the reference signal. Value: 0-> 10000, represents -6dB to 4dB in steps 0.001dB
P-B	3	uint16_t	Refers to downlink power allocation. See [6] section 5.2 Value is an index into the referenced table. Value: 0 → 3
DL Cyclic Prefix Type	4	uint16_t	Cyclic prefix type, used for DL See [8] section 5.2.1 0: CP_NORMAL, 1: CP_EXTENDED.
UL Cyclic Prefix Type	5	uint16_t	Cyclic prefix type, used for UL See [8] section 5.2.1 0: CP_NORMAL, 1: CP_EXTENDED.
RFConfig			
Downlink Channel Bandwidth	6	uint16_t	Downlink channel bandwidth in resource blocks. See [7] section 5.6. Value: 6,15, 25, 50, 75, 100
Uplink Channel Bandwidth	7	uint16_t	Uplink channel bandwidth in resource blocks. See [7] section 5.6 Value: 6,15, 25, 50, 75,100
Reference Signal Power	8	uint16_t	Normalized value levels (relative) to accommodate different absolute Tx Power used by eNb. Value: 0 → 255

Description	Tag	Type	Value
			Representing 0dB to -63.75dB in -0.25dB steps.
Tx Antenna Ports	9	uint16_t	The number of cell specific transmit antenna ports. See [8] section 6.2.1. Value:1,2,4
Rx Antenna Ports	10	uint16_t	The number of cell specific receive antenna ports. See [8] section 6.2.1. Value: 1, 2, 4
PHICH Config			
PHICH Resource	11	uint16_t	The number of resource element groups used for PHICH. See [8] section 6.9. 0: PHICH_R_ONE_SIXTH 1: PHICH_R_HALF 2: PHICH_R_ONE 3: PHICH_R_TWO
PHICH Duration	12	uint16_t	The PHICH duration for MBSFN and non-MBSFN sub-frames. See [8] section 6.9 0: PHICH_D_NORMAL 1: PHICH_D_EXTENDED
PHICH Power Offset	13	uint16_t	The power per antenna of the PHICH with respect to the reference signal. Value: 0-> 10000, represents -6dB to 4dB in steps 0.001dB
SCH Config			
Primary Synchronization signal EPRE/EPRERS	14	uint16_t	The power of synchronization signal with respect to the reference signal, Value: 0 → 9999 represents -6dB to 4dB in step 0.001dB
Secondary Synchronization signal EPRE/EPRERS	15	uint16_t	The power of synchronization signal with respect to the reference signal, Value: 0 → 9999 represents -6dB to 4dB in step 0.001dB
Physical Cell ID	16	uint16_t	The Cell ID sent with the synchronization signal. See [8] section 6.11.

Description	Tag	Type	Value
			Value: 0 → 503
PRACH Config			
Configuration Index	17	uint16_t	Provides information about the location and format of the PRACH. See [8] section 5.7. Table 5.7.1-2 for FDD, Table 5.7.1-3 for TDD The value is an index into the referenced tables. Value: 0 → 63
Root Sequence Index	18	uint16_t	PRACH Root sequence index. See [8] section 5.7.2. Value: 0 → 837
Zero Correlation Zone Configuration	19	uint16_t	Equivalent to N_{cs}, see [8] section 5.7.2. TDD: 0 → 6 FDD: 0 → 15
High Speed Flag	20	uint16_t	Indicates if unrestricted, or restricted, set of preambles is used. See [8] section 5.7.2. 0: HS_UNRESTRICTED_SET 1: HS_RESTRICTED_SET
Frequency Offset	21	uint16_t	The first physical resource block available for PRACH. see [8] section 5.7.1 Value: 0 → UL_channel_bandwidth – 6
PUSCH Config			
Hopping Mode	22	uint16_t	If hopping is enabled indicates the type of hopping used. See [8] section 5.3.4 0: HM_INTER_SF 1: HM_INTRA_INTER_SF
Hopping Offset	23	uint16_t	The offset used if hopping is enabled. See [8] section 5.3.4 Value: 0 → 98

Description	Tag	Type	Value
Number of Sub-bands	24	uint16_t	The number of sub-bands used for hopping. See [8] section 5.3.4. Value: 1 → 4
PUCCH Config			
Delta PUCCH Shift	25	uint16_t	The cyclic shift difference. See [8] section 5.4.1. Value: 1 → 3
N_CQI RB	26	uint16_t	The bandwidth, in units of resource blocks, that is available for use by PUCCH formats 2/2a/2b transmission in each slot. See Section 5.4 in [8]. Value: 0 → 98
N_AN CS	27	uint16_t	The number of cyclic shifts used for PUCCH formats 1/1a/1b in a resource block with a mix of formats 1/a/1/ab and 2/2a/2b. See Section 5.4 in [8]. Value: 0 → 7
N1Pucch-AN	28	uint16_t	$N_{\mathrm{PUCCH}}^{(1)}$, see [6] section 10.1 Value: 0 → 2047
SRS Config			
Bandwidth Configuration	29	uint16_t	The available SRS bandwidth of the cell. See [8] section 5.5.3 The value is an index into the referenced table. Value: 0 → 7
MaxUpPTS	30	uint16_t	Used for TDD only and indicates how SRS operates in UpPTS subframes. See [8] section 5.5.3.2 and [6] section 8.2 0: Disabled 1: Enabled
SRS Subframe Configuration	31	uint16	The subframe configuration. Needed if semi-static configuration is held in PHY.

Description	Tag	Type	Value
			Value: 0 → 15
SRS AckNack SRS Simultaneous Transmission	32	uint8	Indicates if SRS and ACK/NACK can be received in the same subframe. Needed if semi-static configuration is held in PHY. 0: no simultaneous transmission 1: simultaneous transmission
Uplink Reference Signal Config			
Uplink RS Hopping	33	uint16_t	Indicates the type of hopping to use. See [8] section 5.5.1. 0: RS_NO_HOPPING 1: RS_GROUP_HOPPING 2: RS_SEQUENCE_HOPPING
Group Assignment (Delta sequence-shift pattern)	34	uint16_t	The sequence shift pattern used if group hopping is enabled. See [8] section 5.5.1 Values: 0 → 29
Cyclic Shift 1 for DMRS	35	uint16_t	Specifies the cyclic shift for the reference signal used in the cell. See [8] section 5.5.1. The value is an index into the referenced table. Value: 0 → 7
TDD Frame Structure Config			
Subframe Assignment	36	uint16_t	For TDD mode only, indicates the DL/UL subframe structure. See [8] section 4.2. Value: 0 → 6
Special Subframe Patterns	37	uint16_t	For TDD mode only. Length of fields DwPTS, GP and UpPTS. See [8] section 4.2. Value: 0 → 8
These TLVs are used by L1 to report its physical capabilities to the L2/L3 software.			

Description	Tag	Type	Value
Downlink Bandwidth Support	40	uint16_t	The PHY downlink channel bandwidth capability (in resource blocks). See [7] section 5.6 Value: bitX :0 = no support, 1= support. Bit0: 6 Bit1: 15 Bit2: 25 Bit3: 50 Bit4: 75 Bit5: 100
Uplink Bandwidth Support	41	uint16_t	The PHY uplink channel bandwidth capability (in resource blocks). See [7] section 5.6 Value: bitX :0 = no support, 1= support. Bit0: 6 Bit1: 15 Bit2: 25 Bit3: 50 Bit4: 75 Bit5: 100
Downlink Modulation Support	42	uint16_t	The PHY downlink modulation capability. Value: bitX :0 = no support, 1= support. Bit0: QPSK Bit1: 16QAM Bit2: 64QAM
Uplink Modulation Support	43	uint16_t	The PHY uplink modulation capability. Value: bitX :0 = no support, 1= support. Bit0: QPSK Bit1: 16QAM Bit2: 64QAM
PHY Antenna Capability	44	uint16_t	Number of antennas supported. Value: 1, 2, 4

Description	Tag	Type	Value
These TLVs are used by the L2/L3 software to configure the interaction between L2/L3 and L1.			
Data Report Mode	50	uint16_t	The data report mode for the uplink data. 0: A `CRC.indication` message is sent in every subframe. If ULSCH data has been processed, the CRC.indication contains CRC results for the subframe. The CRC results are, also, given in the `RX.indication` message. 1: The `CRC.indication` message is not sent. The CRC results are given in the `RX.indication` message.
SFN/SF	51	uint16_t	The future SFN/SF subframe where the TLVs included in the message should be applied. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
These TLVs are used by L1 to report its current status.			
PHY State	60	uint16_t	Indicates the current operational state of the PHY. 0 = IDLE 1 = CONFIGURED 2 = RUNNING

Table 18: Configuration TLVs

3.2.4 START

The START message exchange was described in Figure 8.

3.2.4.1 START.request

This message can be sent by the L2/L3 when the PHY is in the CONFIGURED state. If it is sent when the PHY is in the IDLE, or RUNNING, state an `ERROR.indication` message will be sent by the PHY. No message body is defined for `START.request`. The message length in the generic header = 0.

3.2.4.2 START Errors

The error codes returned in an `ERROR.indication` generated by the `START.request` message are given in Table 19.

Error Code	Description
MSG_INVALID_STATE	The `START.request` was received when the PHY was in the IDLE or RUNNING state.

Table 19: Error codes for `ERROR.indication`

3.2.5 STOP

The STOP message exchange was described in Figure 9.

3.2.5.1 STOP.request

This message can be sent by the L2/L3 when the PHY is in the RUNNING state. If it is sent when the PHY is in the IDLE, or CONFIGURED, state an `ERROR.indication` message will be sent by the PHY. No message body is defined for `STOP.request`. The message length in the generic header = 0.

3.2.5.2 STOP.indication

This message is sent by the PHY to indicate that it has successfully stopped and returned to the CONFIGURED state. No message body is defined for `STOP.indication`. The message length in the generic header = 0.

3.2.5.3 STOP Errors

The error codes returned in an `ERROR.indication` generated by the `STOP.request` message are given in Table 20.

Error Code	***Description***
MSG_INVALID_STATE	The `STOP.request` was received when the PHY was in the IDLE or CONFIGURED state.

Table 20: Error codes for `ERROR.indication`

3.2.6 UE CONFIG

The UE CONFIG message exchange was described in Figure 25

3.2.6.1 UE_CONFIG.request

The `UE_CONFIG.request` message is given in Table 21. From this table it can be seen that `UE_CONFIG.request` contains a list of TLVs describing how the PHY should be configured with UE-specific parameters. This message may be sent by the L2/L3 software when the PHY is in the RUNNING state. The message is only valid if semi-static configuration is kept in PHY.

Field	***Type***	***Description***
Number of TLVs	uint8_t	Number of TLVs contained in the message body.
TLVs	Variable	See Table 22

Table 21: `UE CONFIG.request` message body

Description	***Tag***
Handle	100
RNTI	101

难道这些配置不需要传送给ue吗？

Description	Tag
CQI Config	All TLVs in this grouping
ACK/NACK Config	All TLVs in this grouping
SRS Config	All TLVs in this grouping
SR Config	All TLVs in this grouping
SPS Config	All TLVs in this grouping

Table 22: TLVs included in `UE_CONFIG.request`

3.2.6.2 UE_CONFIG.response

The `UE_CONFIG.response` message is given in Table 15. If the configuration procedure was successful then the error code returned will be MSG_OK and no TLV tags will be included. If the configuration procedure was unsuccessful then MSG_INVALID_CONFIG will be returned, together with a list of TLVs identifying the problem.

Only valid if semi-static configuration is stored in PHY.

Field	Type	Description
Error Code	uint8_t	See Table 86.
Number of Invalid or Unsupported TLVs	uint8_t	Number of invalid or unsupported TLVs contained in the message body.
Number of Missing TLVs	uint8_t	Number of missing TLVs contained in the message body. If the PHY is in the CONFIGURED state this will always be 0.
A list of invalid or unsupported TLVs – each TLV is presented in its entirety.		
TLV	Variable	Complete TLVs
A list of missing TLVs – each TLV is presented in its entirety		
TLV	Variable	Complete TLVs

Table 23: UE_`CONFIG.response` message body

3.2.6.3 UE_CONFIG Errors

The error codes that can be returned in `UE_CONFIG.response` are given in Table 16.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `UE_CONFIG.request` was received when the PHY was in the IDLE or CONFIGURED state.

Error Code	*Description*
MSG_INVALID_CONFIG	The configuration provided has missing mandatory TLVs, or TLVs that are invalid or unsupported in this state.

Table 24: Error codes for `CONFIG.response`

3.2.6.4 UE Configuration TLVs

The configuration TLVs that are used in the `UE_CONFIG` message exchanges follow the format given in Table 25. Each TLV consists of; a Tag parameter of 1 byte, a Length parameter of 1 byte and a Value parameter. The length of the Value parameter ensures the complete TLV is a multiple of 4-bytes (32-bits).

The individual TLVs are defined in Table 26.

Type	*Description*
uint8_t	Tag
uint8_t	Length (in bytes)
variable	Value

Table 25: TLV format

Description	Tag	Type	Value
Handle	100	uint32_t	An opaque handling to associate the received information in RX.indication
RNTI	101	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
CQI Config			
CQI PUCCH Resource Index	102	uint16_t	The PUCCH resource for periodc CQI reporting. Value: 0 → 1185.
CQI PMI Config Index	103	uint16_t	The periodic PMI reporting configuration. Value: 0 → 1023.
CQI RI Config Index	104	uint16_t	The periodic RI reporting configuration. Value: 0 → 1023.
CQI simultaneous ACK/NACK and CQI	105	uint8_t	Indicates if simultaneous transmission of CQI and ACK/NACK is allowed. Value: 0: no PUCCH Format 2a/2b 1: PUCCH Format 2a/2b can be used
ACK/NACK Config			
AN Repetition Factor	106	uint8_t	The ACK/NACK repetition factor Value: 2,4,6
AN n1PUCCH-ANRep	107	uint16_t	The ACK/NACK repetition PUCCH resource index Value: 0→ 2047
TDD Ack/Nack Feedback Mode	108	uint8_t	The TDD ACK/NACK Feedback Mode Value: 0: bundling 1: multiplexing
SRS Config			
SRS Bandwidth	109	uint8_t	SRS Bandwidth. This value is fixed for a UE and allocated in RRC connection setup.

Description	Tag	Type	Value
			See [8] section 5.5.3.2 Value: 0 → 3
SRS Hopping Bandwidth	110	uint8_t	Configures the frequency hopping on the SRS. This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.2. Value 0 → 3
Frequency Domain Position	111	uint8_t	Frequency-domain position, N_{RRC} This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.2 Value: 0 → 23
SRS Duration	112	uint8_t	The duration of the SRS configuration Value: 0: once 1: indefinite
I_{SRS} / SRS-ConfigIndex	113	uint16_t	Defines the periodicity and subframe location of the SRS. SRS Configuration Index. This value is fixed for a UE and allocated in RRC connection setup. See [6] section 8.2. Value: 0 → 1023.
Transmission Comb	114	uint8_t	Configures the frequency location of the SRS. This value is fixed for a UE and allocated in RRC connection setup. Value: 0 → 1
Sounding Reference Cyclic Shift	115	uint8_t	Configures the SRS sequence generation. This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.1. Value: 0 → 7
SR Config			
SR PUCCH Resource Index	116	uint16_t	The scheduling request PUCCH resource index. Value: 0 → 2047.

Description	Tag	Type	Value
SR Config Index	117	uint8_t	The scheduling request configuration index. Value: 0 → 155.
SPS Config			
SPS DL Config Scheduling Interval	118	uint16_t	SPS Configuration Interval Value: 10,20,32,40,64,80,128,160,320,640
SPS DL n1PUCCH AN Persistent	119	uint16_t	The SPS PUCCH AN Resource configuration. The TLV can be repeated four times. Value: 0 → 2047.

Table 26: Configuration TLVs for UE_CONFIG

3.2.7 UE RELEASE

The UE RELEASE message exchange was described in Figure 26.

3.2.7.1 UE_RELEASE.request

The `UE_RELEASE.request` message is given in Table 27. From this table it can be seen that `UE_RELEASE.request` contains a list of TLVs describing how the PHY should be configured with UE-specific parameters. This message may be sent by the L2/L3 software when the PHY is in the RUNNING state.

This message is used to release the semi-static information in PHY if kept in PHY.

Field	Type	Description
Number of TLVs	uint8_t	Number of TLVs contained in the message body.
TLVs	Variable	See Table 28

Table 27: `UE_RELEASE.request` message body

Description	Tag
Handle	100
RNTI	101

Table 28: TLVs included in `UE_RELEASE.request`

3.2.7.2 UE_RELEASE.response

The `UE_RELEASE.response` message is given in Table 29. If the configuration procedure was successful then the error code returned will be MSG_OK and no TLV tags will be included. If the configuration procedure was unsuccessful then MSG_INVALID_CONFIG will be returned, together with a list of TLVs identifying the problem.

Field	***Type***	***Description***
Error Code	uint8_t	See Table 30
Number of Invalid or Unsupported TLVs	uint8_t	Number of invalid or unsupported TLVs contained in the message body.
Number of Missing TLVs	uint8_t	Number of missing TLVs contained in the message body. If the PHY is in the CONFIGURED state this will always be 0.
A list of invalid or unsupported TLVs – each TLV is presented in its entirety.		
TLV	Variable	Complete TLVs
A list of missing TLVs – each TLV is presented in its entirety		
TLV	Variable	Complete TLVs

Table 29: UE_`RELEASE.response` message body

3.2.7.3 UE_RELEASE Errors

The error codes that can be returned in `UE_RELEASE.response` are given in Table 30.

Error Code	***Description***
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `UE_RELEASE.request` was received when the PHY was in the IDLE or CONFIGURED state.
MSG_INVALID_CONFIG	The configuration provided has missing mandatory TLVs, or TLVs that are invalid or unsupported in this state.

Table 30: Error codes for UE_`RELEASE.response`

3.2.8 PHY Notifications

The PHY notification messages are used by the PHY to inform the L2/L3 software of an event which occurred.

3.2.8.1 ERROR.indication

This message is used to report an error to the L2/L3 software. These errors all relate to API message exchanges. The format of `ERROR.indication` is given in Table 31.

Field	Type	Description
Message ID	uint8_t	Indicate which message received by the PHY has an error. Values taken from Table 4.
Error Code	uint8_t	The error code, see Section 3.4 for value. If the value is MSG_PDU_ERR then more detailed error information is included.
Error code dependent values	struct	The format of these bytes is dependent on the error code. See Table 32 to Table 36.

Table 31: ERROR`.indication` message body

Field	Type	Description
		Not used

Table 32: MSG_INVALID_STATE and MSG_BCH_MISSING

Field	Type	Description
Received SFN/SF	uint16_t	The SFN/SF value received in the message
Expected SFN/SF	uint16_t	The SFN/SF value the PHY was expecting to receive in the message

Table 33: SFN_OUT_OF_SYNC and MSG_INVALID_SFN

Field	Type	Description
Sub Error Code	uint8_t	The Sub Error Code for this message, see Section 3.4.1.
Direction	uint8_t	Indicates if this error was in a DL subframe configuration or an UL subframe configuration. 0 = DL, 1 = UL
RNTI	uint16_t	The RNTI in the received PDU. If the error occurred in a MCH, or BCH, PDU this value is set to 0
PDU Type	uint8_t	The PDU Type parameter specified in both DL subframe configuration and UL subframe configuration

Table 34: MSG_PDU_ERR

Field	Type	Description
Sub Error Code	uint8_t	The Sub Error Code for this message, see Section 3.4.1.

Field	Type	Description
PHICH Lowest UL RB Index	uint8_t	The PHICH RB Index parameters specified in each HI PDU

Table 35: MSG_HI_ERR

Field	Type	Description
Sub Error Code	uint8_t	The Sub Error Code for this message, see Section 3.4.1.
PDU Index	uint16_t	The PDU index parameter specified for each PDU

Table 36: MSG_TX_ERR

3.3 Subframe Messages

The subframe messages are used by the L2/L3 software to control the data transmitted, or received, every 1ms subframe.

3.3.1 SUBFRAME

3.3.1.1 SUBFRAME.indication

The `SUBFRAME.indication` message is given in Table 37. It is sent from the PHY every 1ms.

Field	Type	Description
SFN/SF	uint16_t	A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9

Table 37: `SUBFRAME.indication` message body

3.3.1.2 DL_CONFIG.request

The format of the `DL_CONFIG.request` message is shown in Table 38. A `DL_CONFIG.request` message indicates the SFN/SF subframe it contains information for. This control information is for a downlink subframe.

This message can be sent by the L2/L3 when the PHY is in the RUNNING state. If it is sent when the PHY is in the IDLE or CONFIGURED state an `ERROR.indication` message will be sent by the PHY.

The following combinations of PDUs are required:

- A BCH PDU does not have an associated DCI PDU
- A PCH PDU requires an associated DCI PDU
- A MCH PDU requires an associated DCI PDU
- A DLSCH allocated with Semi-Persistent Scheduling may not have an associated DCI PDU
- A DLSCH for a unique RNTI requires an associated DCI PDU. Therefore, 2 DLSCH for the same RNTI only require 1 DCI PDU

The PDUs included in this structure have no ordering requirements.

Field		Type	Description
SFN/SF		uint16_t	A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Length		uint16_t	The length of the downlink subframe configuration. Range 0 → 65535.
Number of PDCCH OFDM symbols		uint8_t	The number of OFDM symbols for the PDCCH. See [8] section 6.7. Value:0 → 4
Number of DCIs		uint8_t	The number of DCI PDUs included in this message. Range: 0 → 255
Number of PDUs		uint16_t	Number of PDUs that are included in this message. Range 0 → 514
Number of PDSCH RNTIs		uint8_t	Number of unique RNTIs sent on the PDSCH. - a PCH PDU will have an unique RNTI and should be included in this value - a DLSCH PDU can be one transport block sent to a UE with an unique RNTI. This RNTI should be included in this value - a DLSCH PDU can be one of two transport blocks sent to a UE. In this case the two DLSCH PDUs will share the same RNTI. Only one RNTI should be included in this value.
Transmission power for PCFICH		uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB steps.
For Number of PDUs			
	PDU Type	uint8_t	0: DCI DL PDU, see Section 3.3.1.2.1. 1: BCH PDU, see Section 3.3.1.2.2. 2: MCH PDU, see Section 3.3.1.2.3. 3: DLSCH PDU, see Section 3.3.1.2.4. 4: PCH PDU, see Section 3.3.1.2.5.
	PDU Size	uint8_t	Size of the PDU control information (in bytes). This length value includes the 2 bytes required for the PDU type and PDU size parameters.
	DL PDU Configuration	struct	See Sections 3.3.1.2.1 to 3.3.1.2.4.

Table 38: `DL_CONFIG.request` message body

3.3.1.2.1 DCI DL PDU

The format of a DCI DL PDU is shown in Table 39. The DCI DL PDU contains the information which the L2/L3 software must provide the PHY so it can create the DCI formats, related to the downlink, described in [9] section 5.3.3.1.

Field	*Type*	*Description*
DCI Format	uint8_t	Format of the DCI 0 = 1 1 = 1A 2 = 1B 3 = 1C 4 = 1D 5 = 2 6 = 2A
CCE Index	uint8_t	CCE index used to send the DCI. Value: 0 → 88
Aggregation Level	uint8_t	The aggregation level used Value: 1,2,4,8
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU Valid for all DCI formats Value: 1 → 65535.
Resource Allocation Type	uint8_t	Resource allocation type/header Valid for DCI formats: 1,2,2A 0=type 0 1=type 1
Virtual resource block assignment flag	uint8_t	Type of virtual resource block used Valid for DCI formats: 1A,1B,1D 0 = localized 1 = distributed
Resource Block Coding	uint32_t	The encoding for the resource blocks. It's coding is dependent on whether resource allocation type 0, 1, 2 is in use.

Field	*Type*	*Description*
		Resource allocation type 0 is explicitly signalled for DCI formats 1, 2, 2A Resource allocation type 1 is explicitly signalled for DCI formats 1, 2, 2A Resource allocation type 2 is implicit for DCI formats 1A, 1B, 1C, 1D See [6] section 7.1.6 for the encoding used for each format. Valid for DCI formats: 1,1A,1B,1C,1D ,2,2A See [6] section 7.1.6 for the encoding used for each format and a variable-length bitstring generated. Further information on the location of this bitstring within the resource block coding 32-bit parameter is implementation-specific
MCS_1	uint8_t	The modulation and coding scheme for 1st transport block Valid for DCI formats: 1,1A,1B,1C,1D ,2,2A Value: 0 → 31
Redundancy Version_1	uint8_t	The redundancy version for 1st transport block. Valid for DCI formats: 1,1A,1B,1C,1D ,2,2A Value: 0 → 3
New Data Indicator_1	uint8_t	The new data indicator for 1st transport block. Valid for DCI formats: 1,1A,1B,1C,1D ,2,2A
Transport block to codeword swap flag	uint8_t	Indicates the mapping of transport block to codewords Valid for DCI formats: 2,2A 0 = no swapping 1 = swapped
MCS_2	uint8_t	The modulation and coding scheme for 2nd transport block. Valid for DCI formats: 2,2A Value: 0 → 31
Redundancy Version_2	uint8_t	The redundancy version for 2nd transport block. Valid for DCI formats: 2,2A Value: 0 → 3

Field	*Type*	*Description*
New Data Indicator_2	uint8_t	The new data indicator for 2nd transport block. Valid for DCI formats: 2,2A
HARQ Process	uint8_t	HARQ process number Valid for DCI formats: 1,1A,1B,1D,2,2A Value: 0 →15
TPMI	uint8_t	The codebook index to be used for precoding Valid for DCI formats: 1B,1D 2 antenna_ports: 0 → 3 4 antenna_ports: 0 → 15
PMI	uint8_t	Confirmation for precoding Valid for DCI formats: 1B 0 = use precoding indicated in TPMI field 1 = use precoding indicated in last PMI report on PUSCH
Precoding Information	uint8_t	Precoding information Valid for DCI formats: 2,2A 2 antenna_ports: 0 → 7 4 antenna_ports: 0 → 63
TPC	uint8_t	Tx power control command for PUCCH. Valid for DCI formats: 1,1A,1B,1D,2,2A Value: 0,1,2,3 In case of DCI format 1A and RNTI=SI-RNTI,RA-RNTI or P-RNTI the TPC values are either 0,1 representing N_PRB=2 and N_PRB =3 respectively. In case of SPS-C-RNTI it represents the PUCCH resource index.
Downlink Assignment Index	uint8_t	The downlink assignment index. Only used in TDD mode, value ignored for FDD. Valid for DCI formats: 1,1A,1B,1D,2,2A Value: 1,2,3,4
N_{GAP}	uint8_t	Used in virtual resource block distribution Valid for DCI formats: 1A,1B,1C,1D

Field	***Type***	***Description***
		0= N_{GAP1} 1= N_{GAP2}
Transport block size index	uint8_t	The transport block size Valid for DCI formats: 1C Value: 0 → 31
Downlink power offset	uint8_t	Indicates the DL power offset type for multi-user MIMO transmission Valid for DCI formats: 1D Value: 0 → 1
Allocate PRACH flag	uint8_t	Indicates that PRACH procedure is initiated Valid for DCI formats: 1A 0 = false 1=true
Preamble Index	uint8_t	The preamble index to be used on the PRACH Valid for DCI formats: 1A Value: 0 → 63
PRACH Mask Index	uint8_t	The mask index to be used on the PRACH Valid for DCI formats: 1A Value: 0 → 15
RNTI type	uint8_t	RNTI type Valid for DCI format 1, 1A,2,2A 1 = C-RNTI 2 = RA-RNTI, P-RNTI, or SI-RNTI. 3 = SPS-CRNTI
Transmission power	uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB steps.

Table 39: DCI DL PDU

3.3.1.2.2 BCH PDU

The format of the BCH PDU is shown in Table 40.

Field	Type	Description
Length	uint16_t	The length (in bytes) of the associated MAC PDU, delivered in `TX.request`. This should be the actual length of the MAC PDU, which may not be a multiple of 32-bits.
PDU index	uint16_t	This is a count value which is incremented every time a BCH, MCH, PCH or DLSCH PDU is included in the `DL_CONFIG.request` message. This value is repeated in `TX.request` and associates the control information to the data. It is reset to 0 for every subframe Range 0 → 65535
Transmission power	uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB steps.

Table 40: BCH PDU

3.3.1.2.3 MCH PDU

The format of the MCH PDU is shown in Table 41. The contents of the MCH PDU are preliminary.

Field	Type	Description
Length	uint16_t	The length (in bytes) of the associated MAC PDU, delivered in `TX.request`. This should be the actual length of the MAC PDU, which may not be a multiple of 32-bits
PDU index	uint16_t	This is a count value which is incremented every time a BCH, MCH, PCH or DLSCH PDU is included in the `DL_CONFIG.request` message. This value is repeated in `TX.request` and associates the control information to the data. It is reset to 0 for every subframe Range 0 → 65535
RNTI	uint16_t	The RNTI associated with the MCH See [3] section 5.1.4 Value: 1 → 65535.

Field	Type	Description
Resource Allocation Type	uint8_t	Resource allocation type See [6] section 7.1.6 0 = type 0 1 = type 1 2 = type 2
Resource Block Coding	uint32_t	The encoding for the resource blocks. It's coding is dependent on whether resource allocation type 0, 1, 2 is in use. See [6] section 7.1.6 for the encoding used for each format.
Modulation	uint8_t	2: QPSK 4: 16QAM 6: 64QAM
Transmission power	uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB steps.

Table 41: MCH PDU

3.3.1.2.4 DLSCH PDU

The format of the DLSCH PDU is shown in Table 42.

Field	Type	Description
Length	uint16_t	The length (in bytes) of the associated MAC PDU, delivered in `TX.request`. This should be the actual length of the MAC PDU, which may not be a multiple of 32-bits.
PDU index	uint16_t	This is a count value which is incremented every time a BCH, MCH, PCH or DLSCH PDU is included in the `DL_CONFIG.request` message. This value is repeated in `TX.request` and associates the control information to the data. It is reset to 0 for every subframe Range 0 → 65535
RNTI	uint16_t	The RNTI associated with the UE See [3] section 5.1.4 Value: 1 → 65535.

Field	Type	Description
Resource Allocation Type	uint8_t	Resource allocation type See [3] section 7.1.6 0 = type 0 1 = type 1 2 = type 2
Virtual resource block assignment flag	uint8_t	Type of virtual resource block used. This should match the value sent in the DCI Format 1A, 1B, 1D PDU which allocated this grant. See [6] section 7.1.6.3 0 = localized 1 = distributed
Resource Block Coding	uint32_t	The encoding for the resource blocks. It's coding is dependent on whether resource allocation type 0, 1, 2 is in use. This should match the value sent in the DCI Format PDU which allocated this grant. See [6] section 7.1.6 for the encoding used for each format.
Modulation	uint8_t	2: QPSK 4: 16QAM 6: 64QAM
Redundancy Version	uint8_t	HARQ redundancy version. This should match the value sent in the DCI Format PDU which allocated this grant. Value: 0 → 3.
Transport Blocks	uint8_t	The transport block transmitted to this RNTI. A value of 2 indicates this is the second transport block for either DCI format 2 or 2A. For other DCI values this field will always be 1. Value: 1 → 2
Transport block to codeword swap flag	uint8_t	Indicates the mapping of transport block to codewords. This should match the value sent in the DCI Format 2, 2A PDU which allocated this grant. 0 = no swapping 1 = swapped
Transmission Scheme	uint8_t	The MIMO mode used in the PDU

Field	Type	Description
		See [6] section 7.1. 0: SINGLE_ANTENNA_PORT_0, 1: TX_DIVERSITY, 2: LARGE_DELAY_CDD, 3: CLOSED_LOOP_SPATIAL_MULTIPLEXING, 4: MULTI_USER_MIMO, 5: CLOSED_LOOP_RANK_1_PRECODING, 6: SINGLE_ANTENNA_PORT_5.
Number Of Layers	uint8_t	The number of layers used in transmission See [8] section 6.3.3 Value: 1 → 4
Number of subbands	uint8_t	Only valid when transmission scheme = 3, 4, 5. Defines the number of subbands and codebooks used for PMI. If value=1 then a single PMI value is supplied which should be used over all RB Value 0 -> 13
Number of subband entries {		
Codebook Index	uint8_t	Only valid when transmission scheme = 3, 4, 5. Defines the codebook used. When antenna port = 1: NA When antenna port = 2: 0..3 When antenna port = 4: 0..15
}		
UE Category Capacity	uint8_t	The UE capabilities category See [10] section 4.1. Value:1 → 5
P-A	uint8_t	The ratio of PDSCH EPRE to cell-specific RS EPRE among PDSCH REs in all the OFDM symbols not containing cell-specific RS in dB. See [6], section 5.2. 0: -6dB

Field	Type	Description
		1: -4.77dB 2: -3dB 3: -1.77dB 4: 0dB 5: 1dB 6: 2dB 7: 3dB
Delta power offset index	uint8_t	Delta power offset, value: 0..1, Refer to: Table 7.1.5-1 in [6] for Multi-user MIMO mode. Takes value zero for all other modes.
N_{GAP}	uint8_t	Used in virtual resource block distribution 0= N_{GAP1} 1= N_{GAP2}
N_{PRB}	uint8_t	Used with DCI format 1A and RNTI=SI-RNTI or RA-RNTI. This should match the value sent in the TPC field of the DCI 1A PDU which allocated this grant. 0: $N_{PRB} = 2$ 1: $N_{PRB} = 3$
numBfPRBperSubband	uint8_t	Number of PRBs that are treated as one subband
numBfVector	uint8_t	Number of beam forming vectors One beam forming vector is specified for each subband
bfVector	BfVectorType[numBfVector]	Beam forming vectors, see Table 43.

Table 42: DLSCH PDU

Field		Type	Description
subbandIndex		uint8_t	Index of subband for which the following beam forming vector is applied
numAntennas		uint8_t	Number of physical antennas
For each physical antenna			
	bfValue	uint16_t	Beam forming vector element for physical antenna #i real 8 bits followed by imaginery 8 bits

Table 43: BfVectorType Structure

3.3.1.2.5 PCH PDU

The format of the PCH PDU is shown in Table 44.

Field	***Type***	***Description***
Length	uint16_t	The length (in bytes) of the associated MAC PDU, delivered in `TX.request`. This should be the actual length of the MAC PDU, which may not be a multiple of 32-bits.
PDU index	uint16_t	This is a count value which is incremented every time a BCH, MCH, PCH or DLSCH PDU is included in the `DL_CONFIG.request` message. This value is repeated in `TX.request` and associates the control information to the data. It is reset to 0 for every subframe Range 0 → 65535
P-RNTI	uint16_t	The P-RNTI associated with the paging See [3] section 5.1.4 Value: 0xFFFE
Resource Allocation Type	uint8_t	Resource allocation type See [6] section 7.1.6 2 = type 2
Virtual resource block assignment flag	uint8_t	Type of virtual resource block used. This should match the value sent in the DCI Format 1A, 1B, 1D PDU which allocated this grant. See [6] section 7.1.6.3 0 = localized 1 = distributed
Resource Block Coding	uint32_t	The encoding for the resource blocks. It's coding is dependent on whether resource allocation type 0, 1, 2 is in use. This should match the value sent in the DCI Format PDU which allocated this grant. See [6] section 7.1.6 for the encoding used for each format.
MCS	uint8_t	For PCH PDU only QPSK modulation is allowed. 0: QPSK
Redundancy Version	uint8_t	For PCH PDU only redundancy version 0 is allowed

Field	Type	Description
		Value: 0
Number Of Transport Blocks	uint8_t	The number of transport blocks transmitted to this RNTI. Only 1 transport block is sent on the PCH per subframe. Value: 1
Transport block to codeword swap flag	uint8_t	Reserved. This parameter is not used on the PCH transport channel.
Transmission Scheme	uint8_t	The MIMO mode used in the PDU See [6] section 7.1. 0: SINGLE_ANTENNA_PORT_0, 1: TX_DIVERSITY, 6: SINGLE_ANTENNA_PORT_5.
Number Of Layers	uint8_t	The number of layers used in transmission See [8] section 6.3.3 Value: 1 → 4
Codebook Index	uint8_t	Reserved. This parameter is not used on the PCH transport channel.
UE Category Capacity	uint8_t	Reserved. This parameter is not used on the PCH transport channel.
P-A	uint8_t	The ratio of PDSCH EPRE to cell-specific RS EPRE among PDSCH REs in all the OFDM symbols not containing cell-specific RS in dB. See [6], section 5.2. 0: -6dB 1: -4.77dB 2: -3dB 3: -1.77dB 4: 0dB 5: 1dB 6: 2dB 7: 3dB
Transmission power	uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB

Field	Type	Description
		steps.
N_{PRB}	uint8_t	Used with DCI 1A format. This should match the value sent in the TPC field of the DCI 1A PDU which allocated this grant. 0: N_{PRB} = 2 1: N_{PRB} = 3

Table 44: PCH PDU

3.3.1.3 UL_CONFIG.request

The format of the `UL_CONFIG.request` message is shown in Table 45. An `UL_CONFIG.request` message indicates the SFN/SF subframe it contains information for. This control information is for an uplink subframe.

This message can be sent by the L2/L3 when the PHY is in the RUNNING state. If it is sent when the PHY is in the IDLE or CONFIGURED state an `ERROR.indication` message will be sent by the PHY.

The supported PDUs are dependent on whether the semi-static information and uplink HARQ signalling calculation is held in the MAC or PHY.

If the semi-static information and uplink HARQ signalling calculation is held in the MAC, the following combinations of PDUs are required:

- In order to support RACH in the subframe, the RACH present field must be true
- In order to support SRS in the subframe, the SRS present field must be true
- If the SRS present field is true, there can be 0 SRS PDU, or ≥1 SRS PDU present.
- The ULSCH PDU is present when a UE has been instructed to only send uplink data
- The ULSCH_CQI_RI, ULSCH_HARQ and ULSCH_CQI_HARQ_RI PDUs are present when a UE has been instructed to send uplink data and control
- The UCI_CQI, UCI_SR, UCI_SR_HARQ, UCI_CQI_HARQ, UCI_CQI_SR and UCI_CQI_SR_HARQ PDUs are present when a UE has been only been instructed to transmit control
- The following combinations can have the same RNTI values:
  - UCI_x + SRS
  - ULSCH_x + SRS

If the semi-static information and uplink HARQ signalling calculation is held in the PHY, the following combinations of PDUs are required:

- The ULSCH PDU is present when a UE has been instructed to send uplink data

If the semi-static information is held in the MAC, and the uplink HARQ signalling calculation is held in the PHY, the following combinations of PDUs are required:

- In order to support RACH in the subframe, the RACH present field must be true
- In order to support SRS in the subframe, the SRS present field must be true
- If the SRS present field is true, there can be 0 SRS PDU, or ≥1 SRS PDU present.
- The ULSCH PDU is present when a UE has been instructed to only send uplink data
- The ULSCH_CQI_RI is present when a UE has been instructed to send uplink data and control

- The UCI_CQI, UCI_SR and UCI_CQI_SR PDUs are present when a UE has been only been instructed to transmit control
- The following combinations can have the same RNTI values:
  - ULSCH_x + SRS

If the semi-static information is held in the PHY, and the uplink HARQ signalling calculation is held in the MAC, the following combinations of PDUs are required:

- The ULSCH PDU is present when a UE has been instructed to send uplink data
- The ULSCH HARQ_PDU is present when a UE has been instructed to send uplink data, and an ACK/NACK response is expected
- The UCI_HARQ PDU is present when an ACK/NACK response is expected from a UE not transmitting uplink data

Field	***Type***	***Description***
SFN/SF	uint16_t	A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Length	uint16_t	The length of the uplink subframe configuration. Range 0 → 65535.
Number of PDUs	uint8_t	The number of UL SCHs PDUs included in this message.
RACH/PRACH Frequency Resources	uint8_t	If semi-static information is held in the MAC For FDD: 0: No RACH in this subframe 1: RACH present in this subframe For TDD: Bits 0:5 indicate which RACH frequency resources are used in this subframe, see [8] section 5.7.1. 0: This RACH frequency index is not used 1: This RACH frequency index is used If semi-static information is held in the PHY this parameter is ignored.
SRS present	uint8_t	If semi-static information is held in the MAC 0: No SRS in this subframe 1: SRS present in this subframe If semi-static information is held in the PHY this parameter is ignored.
For Number of PDUs		
PDU Type	uint8_t	0: ULSCH, see Section 3.3.1.3.1.

Field	Type	Description
		1: ULSCH_CQI_RI, see Section 3.3.1.3.2. 2: ULSCH_HARQ, see Section 3.3.1.3.3. 3: ULSCH_CQI_HARQ_RI, see Section 3.3.1.3.4. 4: UCI_CQI, see Section 3.3.1.3.6. 5: UCI_SR, see Section 3.3.1.3.7. 6: UCI_HARQ, see Section 3.3.1.3.8. 7: UCI_SR_HARQ, see Section 3.3.1.3.9. 8: UCI_CQI_HARQ, see Section 3.3.1.3.10. 9 : UCI_CQI_SR, see Section 3.3.1.3.11 10 : UCI_CQI_SR_HARQ, see Section 3.3.1.3.11 11: SRS, see Section 3.3.1.3.16. 12 : HARQ_BUFFER , see Section 3.3.1.3.17
PDU Size	uint8_t	Size of the PDU control information (in bytes). This length value includes the 2 bytes required for the PDU type and PDU size parameters.
UL PDU Configuration	Struct	See Sections 3.3.1.3.1to 3.3.1.3.16.

Table 45: Uplink subframe configuration message body

3.3.1.3.1 ULSCH PDU

The format of the ULSCH PDU is given in Table 46.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the RX.indication
Size	uint16_t	The size of the ULSCH PDU in bytes as defined by the relevant UL grant. The size can be 0 if UCI over ULSCH without data is configured. The size of CQI/RI/HARQ are not added to this element.
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
Resource Block Start	uint8_t	The starting resource block for this ULSCH allocation. This should match the value sent in the DCI Format 0 PDU which allocated this grant. Value: 0 → 99

Field	Type	Description
Number of Resource Blocks	uint8_t	The number of resource blocks allocated to this ULSCH grant. This should match the value sent in the DCI Format 0 PDU which allocated this grant. Value: 1 → 100
Modulation Type	uint8_t	2: QPSK 4: 16QAM 6: 64QAM
Cyclic Shift 2 for DMRS	uint8_t	The 2^{nd} cyclic shift for DMRS assigned to the UE in the ULSCH grant. This should match the value sent in the DCI Format 0 PDU which allocated this grant. Value: 0 → 7
Frequency hopping enabled flag	uint8_t	Indicates if hopping is being used. This should match the value sent in the DCI Format 0 PDU which allocated this grant. See [6] Section 8.4. 0 = no hopping, 1= hopping enabled
Frequency hopping bits	uint8_t	The frequency hopping bits. This should match the value sent in the DCI Format 0 PDU which allocated this grant. See [6] Section 8.4 Value: 0 → 3
New Data Indication	uint8_t	Specify whether this received PUSCH is a new transmission from UE. This should match the value sent in the DCI Format 0 PDU which allocated this grant.
Redundancy Version	uint8_t	Redundancy version Value: 0 → 3
HARQ Process Number	uint8_t	HARQ Process number. TDD 0 → 15 FDD 0 → 7
UL Tx Mode	uint8_t	0 = SISO/SIMO 1 = MIMO
Current TX NB	uint8_t	The current HARQ transmission count of this transport block.

Field	Type	Description
		Valid if frequency hopping enabled.
N srs	uint8_t	Indicates if the resource blocks allocated for this grant overlap with the SRS configuration. 0 = no overlap 1 = overlap

Table 46: ULSCH PDU

3.3.1.3.2 ULSCH_CQI_RI PDU

The format of the ULSCH_CQI_RI PDU is given in Table 47. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
ULSCH PDU	struct	Description of contents given in Table 46
CQI_RI information	struct	Description of contents given in Table 48
Initial transmission parameters	struct	Description of contents given in Table 52

Table 47: ULSCH_CQI_RI PDU

The format of the CQI_RI information is given in Table 48.

Field	Type	Description
DL CQI/PMI Size Rank = 1	uint8_t	The size of the DL CQI/PMI in bits in case of rank 1 report. Value: 0 → 255
DL CQI/PMI Size Rank>1	uint8_t	The size of the DL CQI/PMI in bits in case of rank>1 report. Value: 0 → 255 In case of periodic report rank=1 and rank>1 size should be the same
RI Size	uint8_t	The size of RI in bits Value:1 → 2 0 in case of periodic report
Delta Offset CQI	uint8_t	Delta offset for CQI. This value is fixed for a UE and allocated in RRC connection setup. See [6] section 8.6.3

Field	Type	Description
		Value: 0 → 15
Delta Offset RI	uint8_t	Delta offset for RI. This value is fixed for a UE and allocated in RRC connection setup. See [6] section 8.6.3 Value: 0 → 15

Table 48: CQI_RI Information

3.3.1.3.3 ULSCH_HARQ PDU

The format of the ULSCH_HARQ PDU is given in Table 49. This PDU is only valid if the uplink HARQ signalling calculation is held in the MAC.

Field	Type	Description
ULSCH PDU	struct	Description of contents given in Table 46
HARQ information	struct	Description of contents given in Table 50
Initial transmission parameters	struct	Description of contents given in Table 52

Table 49: ULSCH_HARQ PDU

The format of the HARQ information is given in Table 50.

Field	Type	Description
HARQ Size	uint8_t	The size of the ACK/NACK in bits. Value: 1 → 2
Delta Offset HARQ	uint8_t	Delta offset for HARQ. This value is fixed for a UE and allocated in RRC connection setup. See [6] section 8.6.3 Value: 0 → 15
ACK_NACK mode	uint8_t	The format of the ACK/NACK response expected. For TDD only. 0 = BUNDLING

Field	Type	Description
		1 = MULTIPLEXING

Table 50: HARQ Information

3.3.1.3.4 ULSCH_CQI_HARQ_RI PDU

The format of the ULSCH_CQI_HARQ_RI PDU is given in Table 51. This PDU is only valid if both semi-static information and the uplink HARQ signalling calculation is held in the MAC.

Field	Type	Description
ULSCH PDU	struct	Description of contents given in Table 46
CQI_RI information	struct	Description of contents given in Table 48
HARQ information	struct	Description of contents given in Table 50
Initial transmission parameters	struct	Description of contents given in Table 52

Table 51: ULSCH_CQI_HARQ_RI PDU

3.3.1.3.5 Initial Transmission Parameters

The format of the Initial transmission parameters is given in Table 52.

Field	Type	Description
N srs initial	uint8_t	0 = last OFDM symbol is not punctured 1 = last OFDM symbol is punctured.
Initial number of resource blocks	uint8_t	The number of resource blocks used in the initial transmission of this transport block. Value: 1 → 100

Table 52: Initial Transmission Information

3.3.1.3.6 UCI_CQI PDU

The format of the UCI_CQI PDU is given in Table 53. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the RX_CQI.indication

Field	Type	Description
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
CQI Information	struct	Description of contents given in Table 60

Table 53: UCI_CQI PDU

3.3.1.3.7 UCI_SR PDU

The format of the UCI_SR PDU is given in Table 54. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the RX_SR.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
SR Information	struct	Description of contents given in Table 61

Table 54: UCI_SR PDU

3.3.1.3.8 UCI_HARQ PDU

The format of the UCI_HARQ PDU is given in Table 55. This PDU is only valid if the uplink HARQ signalling calculation is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the HARQ.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
HARQ Information	struct	Description of contents given in Table 62 for TDD and Table 63 for FDD.

Table 55: UCI_HARQ PDU

3.3.1.3.9 UCI_SR_HARQ PDU

The format of the UCI_SR_HARQ PDU is given in Table 56. This PDU is only valid if both semi-static information and the uplink HARQ signalling calculation is held in the MAC.

Field	*Type*	*Description*
Handle	uint32_t	An opaque handling returned in the RX_SR.indication and HARQ.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
SR Information	struct	Description of contents given in Table 61
HARQ Information	struct	Description of contents given in Table 62 for TDD and Table 63 for FDD.

Table 56: UCI_SR_HARQ PDU

3.3.1.3.10 UCI_CQI_HARQ PDU

The format of the UCI_CQI_HARQ PDU is given in Table 57. This PDU is only valid if both semi-static information and the uplink HARQ signalling calculation is held in the MAC.

For TDD when both HARQ and CQI, or HARQ and SR, are transmitted on PUCCH, multiple HARQ ACK/NACK responses are bundled according to table 7.3-1 of [6]. This is referred to in Table 76 as “Special Bundling” and implies a unique interpretation of the message fields.

Field	*Type*	*Description*
Handle	uint32_t	An opaque handling returned in the RX_CQI.indication and HARQ.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
CQI Information	struct	Description of contents given in Table 60
HARQ Information	struct	Description of contents given in Table 62 for TDD and Table 63 for FDD.

Table 57: UCI_CQI_HARQ PDU

3.3.1.3.11 UCI_CQI_SR_PDU

The format of the UCI_CQI_SR PDU is given in Table 58. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the RX_CQI.indication and RX_SR.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
CQI Information	struct	Description of contents given in Table 60
SR Information	struct	Description of contents given in Table 61

Table 58: UCI_CQI_SR PDU

3.3.1.3.12 UCI_CQI_SR_HARQ_PDU

The format of the UCI_CQI_SR HARQ_PDU is given in Table 59. This PDU is only valid if both semi-static information and the uplink HARQ signalling calculation is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the RX_CQI.indication, RX_SR.indication and HARQ.indication
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
CQI Information	struct	Description of contents given in Table 60
SR Information	struct	Description of contents given in Table 61
HARQ Information	struct	Description of contents given in Table 62 for TDD and Table 63 for FDD.

Table 59: UCI_CQI_SR_HARQ PDU

3.3.1.3.13 CQI Information

The format of the CQI Information used in UCI PDUs is given in Table 60.

Field	Type	Description
PUCCH Index	uint16_t	The PUCCH Index value $n_{\mathrm{PUCCH}}^{(2)}$ Value: 0 → 1185
DL CQI/PMI Size	uint8_t	The size of the DL CQI/PMI in bits. Value: 0 → 255

Table 60: CQI information

3.3.1.3.14 SR Information

The format of the SR Information used in UCI PDUs is given in Table 61.

Field	Type	Description
PUCCH Index	uint16_t	The PUCCH Index value $n_{\mathrm{PUCCH}}^{(1)}$ Value: 0 → 2047

Table 61: SR information

3.3.1.3.15 HARQ Information

The format of the HARQ Information used in UCI PDUs is dependent on whether TDD or FDD mode is used. For TDD it is given in Table 62. For TDD when both HARQ and CQI, or HARQ and SR, are transmitted on PUCCH, multiple HARQ ACK/NACK responses are bundled according to table 7.3-1 of [6]. This is referred to in Table 76 as "Special Bundling" and implies a unique interpretation of the message fields.

For FDD it is given in Table 63. For FDD only the PUCCH index value is required

Field	Type	Description
HARQ Size	uint8_t	For ACK_NACK Mode 0 (Bundling) and 1 (Multiplexing), this is the size of the ACK/NACK in bits. Value: 1 → 4 For Special Bundling this is the expected number of ACK/NACK responses (UDAI + NSPS) (see table 7.3-1 in [6]). Value: 0 → 9
ACK_NACK mode	uint8_t	The format of the ACK/NACK response expected. For TDD only. 0 = BUNDLING 1 = MULTIPLEXING

Field	Type	Description
Number of PUCCH Resources	uint8_t	The number of ACK/NACK responses received in this subframe. For TDD only. See [6] section 10.1 Value: 0 → 4 (Value 0 is only valid for Special Bundling.)
n_PUCCH_1_0	uint16_t	HARQ resource 0, value: 0 → 2047
n_PUCCH_1_1	uint16_t	HARQ resource 1, value: 0 → 2047
n_PUCCH_1_2	uint16_t	HARQ resource 2, value: 0 → 2047
n_PUCCH_1_3	uint16_t	HARQ resource 3, value: 0 → 2047

Table 62: HARQ information for TDD

Field	Type	Description
PUCCH Index	uint16_t	The PUCCH Index value for ACK/NACK Value: 0 → 2047
HARQ Size	uint8_t	The size of the ACK/NACK in bits. Value: 1 → 2

Table 63: HARQ information for FDD

3.3.1.3.16 SRS

The format of the SRS PDU is given in Table 64. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling returned in the SRS.indication
Size	uint16_t	The size of the PDU in bytes.
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU See [3] section 5.1.4 Value: 1 → 65535.
SRS Bandwidth	uint8_t	SRS Bandwidth. This value is fixed for a UE and allocated in RRC connection setup.

Field	Type	Description
		See [8] section 5.5.3.2 Value: 0 → 3
Frequency Domain Position	uint8_t	Frequency-domain position, N_{RRC} This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.2 Value: 0 → 23
SRS Hopping Bandwidth	uint8_t	Configures the frequency hopping on the SRS. This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.2. Value 0 → 3
Transmission Comb	uint8_t	Configures the frequency location of the SRS. This value is fixed for a UE and allocated in RRC connection setup. Value: 0 → 1
I_{SRS} / SRS-ConfigIndex	uint16_t	Defines the periodicity and subframe location of the SRS. SRS Configuration Index. This value is fixed for a UE and allocated in RRC connection setup. See [6] section 8.2. Value: 0 → 1023.
Sounding Reference Cyclic Shift	uint8_t	Configures the SRS sequence generation. This value is fixed for a UE and allocated in RRC connection setup. See [8] section 5.5.3.1. Value: 0 → 7

Table 64: SRS PDU

3.3.1.3.17 HARQ_BUFFER PDU

The HARQ Buffer PDU indicates for which UE the HARQ buffer should be released. The format of the HARQ_BUFFER PDU is given in Table 65: HARQ Buffer Release PDU. This PDU is only valid if semi-static information is held in the MAC.

Field	Type	Description
Handle	uint32_t	An opaque handling

Field	Type	Description
RNTI	uint16_t	The RNTI used for identifying the UE for which the HARQ buffer should be released. Value: 1 → 65535.

Table 65: HARQ Buffer Release PDU

3.3.1.4 HI_DCI0.request

The format of the `HI_DCI0.request` message is given in Table 66. This message contains two types of control information relating to the uplink. Firstly, it is used for the L2/L3 to send the ACK/NACK response for MAC PDUs received on the ULSCH, LTE has strict timing requirements for returning this information to the UE. Secondly, it is used for DCI control format information relating to the uplink which is broadcast on the PDCCH.

Section 2.2.3 contains a detailed description on when this message should be sent, and the correct value of SFN/SF.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF in this message should be the same as the corresponding `DL_CONFIG.request` message. A 2-byte value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of DCI	uint8_t	Number of DCI PDUs included in this message
Number of HI	uint8_t	Number of HI PDUs included in this message
For Number of DCI + HI PDUs		
PDU Type	uint8_t	0: HI PDU, see Section 3.3.1.4.1. 1: DCI UL PDU, see Section 3.3.1.4.2.
PDU Size	uint8_t	Size of the PDU control information (in bytes). This length value includes the 2 bytes required for the PDU type and PDU size parameters.
HI/DCI PDU Configuration	Struct	See Sections 3.3.1.4.1 to 3.3.1.4.2.

Table 66: `HI_DCI0.request` message body

3.3.1.4.1 HI PDU

The format of a HI PDU is shown in Table 67. The HI PDU contains the ACK/NACK response for a ULSCH transmission.

Field	Type	Description
Resource Block Start	uint8_t	This value is the starting resource block assigned to the ULSCH grant associated with this HI response. It should match

Field	Type	Description
		the value sent in the DCI format 0 which allocated the ULSCH grant See [6] section 9.1.2 Value: 0 → 100
Cyclic Shift 2 for DMRS	uint8_t	This value is the 2nd cyclic shift for DMRS assigned to the ULSCH grant associated with this HI response. It should match the value sent in the DCI format 0 which allocated the ULSCH grant See [6] section 9.1.2 Value: 0 → 7
HI Value	uint8_t	The PHICH value which is sent on the resource. 0: HI_NACK 1: HI_ACK
I_PHICH	uint8_t	Is used in the calculation of the PHICH location. For TDD only. See [6] section 9.1.2 1 = TDD subframe configuration 0 is used and the ULSCH grant associated with this HI was received in subframe 4 or 9 0 = in all other cases
Transmission power	uint16_t	Offset to the reference signal power. Value: 0 → 10000, representing -6 dB to 4 dB in 0.001 dB steps.

Table 67: HI PDU

3.3.1.4.2 DCI UL PDU

The format of a DCI UL PDU is shown in Table 68. The DCI UL PDU contains the information which the L2/L3 software must provide the PHY so it can create the DCI format 0 or format 3/3A described in [9] section 5.3.3.1.

Field	Type	Description
DCI Format	uint8_t	Format of the DCI 0 = 0 1 = 3 2 = 3A
CCE Index	uint8_t	CCE index used to send the DCI.

Field	Type	Description
		Value: 0 → 88
Aggregation Level	uint8_t	The aggregation level used Value: 1,2,4,8
RNTI	uint16_t	The RNTI used for identifying the UE when receiving the PDU Valid for all DCI formats Value: 1 → 65535.
Resource Block Start	uint8_t	The starting resource block for this ULSCH allocation. Valid for DCI format 0 Value: 0 → 100
Number of Resource Blocks	uint8_t	The number of resource blocks allocated to this ULSCH grant. Valid for DCI format 0 Value: 0 → 100
MCS	uint8_t	The modulation and redundancy version. See [6] section 8.6. Valid for DCI format 0 Value: 0 → 31
Cyclic Shift 2 for DMRS	uint8_t	The 2nd cyclic shift for DMRS assigned to the UE in the ULSCH grant. Valid for DCI format 0 Value: 0 → 7
Frequency enabled flag	uint8_t	Indicates if hopping is being used. See [6] Section 8.4. Valid for DCI format 0 0 = no hopping, 1= hopping enabled
Frequency hopping bits	uint8_t	The frequency hopping bits See [6] Section 8.4 Valid for DCI format 0

Field	*Type*	*Description*
		Value: 0 → 3
New Data Indication	uint8_t	The new data indicator for the transport block. Valid for DCI format 0
UE TX antenna selection	uint8_t	Indicates how the CRC is calculated on the PDCCH. See [9] section 5.3.2.2 Valid for DCI format 0 0 = Not configured; 1 = Configured and using UE port 0; 2 = Configured and using UE port 1.
TPC	uint8_t	Tx power control command for PUSCH. Valid for DCI format 0 Value: 0,1,2,3
CQI request	uint8_t	Aperiodic CQI request flag Valid for DCI format 0 0 = Aperiodic CQI not requested 1 = Aperiodic CQI requested
UL index	uint8_t	UL index. Valid for TDD mode only. Valid for DCI format 0 Value: 0,1,2,3
DL assignment index	uint8_t	DL assignment index. Valid for TDD mode only. Valid for DCI format 0 Value: 1,2,3,4
TPC bitmap	uint32_t	TPC commands for PUCCH and PUSCH Valid for DCI formats: 3,3A The encoding follows [9] section 5.3.3.1.6

Table 68: DCI UL PDU

3.3.1.5 SUBFRAME Errors

The error codes returned in an ERROR.indication generate by the DL_CONFIG.request message are given in Table 69.

Error Code	Description
MSG_INVALID_STATE	The DL_CONFIG.request was received when the PHY was in the IDLE or CONFIGURED state.
SFN_OUT_OF_SYNC	The DL_CONFIG.request was received with a different SFN/SF than the PHY expected. The PHY has followed the SFN/SF sync process, see Section 2.2.2.
MSG_BCH_MISSING	A BCH PDU was expected in the DL_CONFIG.request message for this subframe.
MSG_PDU_ERR	An error was received in DL_CONFIG.request. The sub-error code should be analyzed.

Table 69: Error codes for ERROR.indication generated by DL_CONFIG.request

The error codes returned in an ERROR.indication generate by the UL_CONFIG.request message are given in Table 69.

Error Code	Description
MSG_INVALID_STATE	The UL_CONFIG.request was received when the PHY was in the IDLE or CONFIGURED state.
MSG_PDU_ERR	An error was received in UL_CONFIG.request. The sub-error code should be analyzed.

Table 70: Error codes for ERROR.indication generated by UL_CONFIG.request

The error codes returned in an ERROR.indication generate by the HI_DCI0.request message are given in Table 71.

Error Code	Description
MSG_INVALID_STATE	The HI_DCI0.request was received when the PHY was in the IDLE or CONFIGURED state.
MSG_INVALID_SFN	The HI_DCI0.request message received in subframe N included a SFN/SF value which was not N-1. The message has been ignored.
MSG_HI_ERR	An error was received in HI_DCI0.request. The sub-error code should be analyzed

Table 71: Error codes for ERROR.indication generated by HI_DCI0.request

3.3.2 Downlink Data

3.3.2.1 TX.request

The format of the `TX.request` message is described in Table 72. This message contains the MAC PDU data for transmission over the air interface. The PDUs described in this message must follow the same order as `DL_CONFIG.request`.

This message can be sent by the L2/L3 when the PHY is in the RUNNING state. If it is sent when the PHY is in the IDLE, or CONFIGURED, state an `ERROR.indication` message will be sent by the PHY.

Field	*Type*	*Description*
SFN/SF	uint16_t	The SFN/SF in this message should be the same as the corresponding `DL_CONFIG.request` message. A 2-byte value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of PDUs	uint16_t	Number of PDUs included in this message.
For each PDU		
PDU Length	uint16_t	The total length (in bytes) of the PDU description and PDU data, without the padding bytes.
PDU index	uint16_t	This is a count value which starts from 0. It is incremented for each BCH, MCH, PCH or DLSCH PDU. This value was included in `TX.request` and associates the data to the control information. It is reset to 0 for every subframe Range 0 → 65535
numTLV	uint32_t	The number of TLVs describing the data of the transport block.
TLVs	TLV[numTLV]	Always a multiple of 32-bits. See Table 73

Table 72: `TX.request` message body

Field	*Type*	*Description*
tag	uint16_t	Range 0 -> 1 0: Payload is carried directly in the value field 1: Pointer to payload is in the value field
length	uint16_t	Length of the actual payload in bytes, without the padding bytes
value	Variable or uint32_t	Always a multiple of 32-bits.

Field	Type	Description
		Tag=0: Only the most significant bytes of the size indicated by "length" field are valid. Remaining bytes are zero padded to the nearest 32-bit boundary Tag=1: Pointer to the payload. Occupies 32-bits

Table 73: `TLV structure`

3.3.2.2 Downlink Data Errors

The error codes returned in an `ERROR.indication` generate by the `TX.request` message are given in Table 74.

Error Code	Description
MSG_INVALID_STATE	The `TX.request` was received when the PHY was in the IDLE or CONFIGURED state.
MSG_INVALID_SFN	The `TX.request` message received in subframe N included a SFN/SF value which was not N. The message has been ignored.
MSG_TX_ERR	An error was received in `TX.request`. The sub-error code should be analyzed.

Table 74: Error codes for `ERROR.indication`

3.3.3 Uplink Data

3.3.3.1 RX_ULSCH.indication

The format of the `RX_ULSCH.indication` message is shown in Table 75.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of PDUs	uint16_t	Number of PDUs included in this message.
For (Number of PDUs) {		
Handle	uint32_t	The handle received in the ULSCH PDU.

Field	Type	Description
RNTI	uint16_t	The RNTI passed to the PHY in a DL_CONFIG.request ULSCH PDU. See [3] section 5.1.4 Value: 1 → 65535.
Length	uint16_t	Length of PDU in bytes.
Data Offset	uint16_t	Gives the PDU#i data address offset from the beginning of the 'Number of PDUs' field. An offset of 0 indicates a CRC or decoding error.
UL_CQI	uint8_t	SNR Value: 0-255, representing -64dB to 63.5dB, with 0.5dB step size.
Timing Advance	uint16_t	The timing advance measured for this PDU and UE. Value: T_A from 0 to 1282 as defined in [6] section 4.2.3.
}		
PDU#1	Variable	Contents of PDU#1. This will be a MAC PDU.
PDU#2	Variable	Contents of PDU#2. This will be a MAC PDU
..	...	...
PDU#n	Variable	Contents of PDU#n. This will be a MAC PDU

Table 75: `RX.indication` message body

3.3.3.2 HARQ.indication

The format of the uplink HARQ control from the UE is dependent on whether a TDD or FDD PHY is used. To accommodate this difference two `HARQ.indication` messages are defined, one for TDD and one for FDD.

The `HARQ.indication` messages provide the following results for each ACK/NACK report.

- ACK – The PHY confidently detected an ACK
- NACK – The PHY confidently detected an NACK
- DTX – The PHY confidently detected that the UE did not transmit an ACK/NACK response
- ACK or NACK – The PHY is unsure whether it detected an ACK or NACK.

- ACK or DTX - The PHY is unsure whether it detected an ACK or DTX
- NACK or DTX - The PHY is unsure whether it detected an NACK or DTX
- ACK or NACK or DTX - The PHY is unsure whether it detected an ACK or NACK or DTX

3.3.3.2.1 TDD Format

The format of the `HARQ.indication` message for a TDD PHY is given in Table 76.

Field		*Type*	*Description*
SFN/SF		uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of HARQs		uint16_t	Number of HARQs included in this message.
For (Number of HARQs)			
	Handle	uint32_t	The handle received in the ULSCH PDU or UCI PDU.
	RNTI	uint16_t	The RNTI passed to the PHY in an uplink subframe configuration PDU. See [3] section 5.1.4 Value: 1 → 65535.
	Mode	uint8_t	The format of the ACK/NACK response expected. The bundling and multiplexing options are passed to the PHY in an uplink subframe configuration PDU. If the ACK/NACK is combined with either CQI or SR information then a special ACK/NACK encoding is used which reports the number of ACKs, rather than providing specific ACK/NACK values. This is identified separately and called SPECIAL_BUNDLING in this API. (see [6] section 7.3 and section 3.3.1.3.15 of this document for more information) 0 = BUNDLING 1 = MULTIPLEXING
	Number of ACK/NACK	uint8_t	The number of ACK/NACK results reported for this UE. See [6] section 10. Value: 1 → 4 For Special Bundling this is the expected number of ACK/NACK responses (UDAI + NSPS) (see table 7.3-1 in [6]). Value: 0 → 9
	HARQ Data	struct	The format of the data is dependent on the HARQ mode; BUNDLING, MULTIPLEXING, or SPECIAL BUNDLING. See Table

Field	Type	Description
		77 to Table 79.

Table 76: `HARQ.indication` message body for TDD

Field	Type	Description
Value 0	uint8_t	Indicates HARQ results. Range 1 → 7 1 = ACK 2 = NACK 3 = ACK or NACK 4 = DTX 5 = ACK or DTX 6 = NACK or DTX 7 = ACK or NACK or DTX
Value 1	uint8_t	Indicates HARQ results. Range 1 → 7, see above.

Table 77: TDD HARQ data format for mode = BUNDLING

Field	Type	Description
Value 0	uint8_t	Indicates HARQ results. Range 1 → 7 1 = ACK 2 = NACK 3 = ACK or NACK 4 = DTX 5 = ACK or DTX 6 = NACK or DTX 7 = ACK or NACK or DTX
Value 1	uint8_t	Indicates HARQ results. Range 1 → 7, see above.
Value 2	uint8_t	Indicates HARQ results. Range 1 → 7, see above.
Value 3	uint8_t	Indicates HARQ results. Range 1 → 7, see above.

Table 78: TDD HARQ data format for mode = MULTIPLEXING

Field	Type	Description
Value 0	uint8_t	Number of ACK among multiple ACK/NACK responses, see [6] table 7.3.-1 0 = 0 or None (UE detect at least one DL assignment is missed) 1 = 1 or 4 or 7 ACKs reported 2 = 2 or 5 or 8 ACKs reported 3 = 3 or 6 or 9 ACKs reported 4 = DTX (UE did not transmit anything)

Table 79: TDD HARQ data format for mode = SPECIAL BUNDLING

3.3.3.2.2 FDD Format

The format of the `HARQ.indication` message for a FDD PHY is given in Table 80.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of HARQs	uint16_t	Number of HARQs included in this message.
For (Number of HARQs)		
Handle	uint32_t	The handle received in the ULSCH PDU or UCI PDU.
RNTI	uint16_t	The RNTI passed to the PHY in an uplink subframe configuration PDU. See [3] section 5.1.4 Value: 1 → 65535.
HARQ TB1	uint8_t	HARQ feedback of 1st TB. Range 1 → 7 1 = ACK 2 = NACK 3 = ACK or NACK 4 = DTX 5 = ACK or DTX

Field	Type	Description
		6 = NACK or DTX 7 = ACK or NACK or DTX
HARQ TB2	uint8_t	HARQ feedback of 2nd TB. Range 1 → 7 1 = ACK 2 = NACK 3 = ACK or NACK 4 = DTX 5 = ACK or DTX 6 = NACK or DTX 7 = ACK or NACK or DTX

Table 80: `HARQ.indication` message body for FDD

3.3.3.3 CRC.indication

The format of the `CRC.indication` message is given in Table 81.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of CRCs	uint16_t	Number of CRCs included in this message.
For (Number of CRCs)		
Handle	uint32_t	The handle received in the ULSCH PDU.
RNTI	uint16_t	The RNTI passed to the PHY in an uplink subframe configuration PDU. See [3] section 5.1.4 Value: 1 → 65535.
CRC Flag	uint8_t	A flag indicating if a CRC error was detected. 0: CRC_CORRECT

Field	Type	Description
		1:CRC_ERROR

Table 81: `CRC.indication` message body

3.3.3.4 RX_SR.indication

The format of the RX_SR.`indication` message is given in Table 82.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 2-byte value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of SRs	uint16_t	Number of SRs included in this message.
For (Number of SRs)		
Handle	uint32_t	The handle received in the UCI PDU.
RNTI	uint16_t	The RNTI identifying the UE. For semi-static information held in the MAC this will be the value passed to the PHY in a `UL_CONFIG.request` SR PDU. See [3] section 5.1.4 Value: 0 → 65535.

Table 82: `RX_SR.indication` message body

3.3.3.5 RX_CQI.indication

The format of the `RX_CQI.indication` message is given in Table 83. The format of DL CQI feedback and reports varies depending upon the channel used for feedback (PUSCH or PUCCH) and the DL transmission mode. This is detailed in [9]. The formats differ in the fields reported and the resultant number of bits required for the report.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9

Field	Type	Description
Number of PDUs	uint16_t	Number of PDUs included in this message.
For (Number of PDUs) {		
Handle	uint32_t	The handle received in the ULSCH PDU or UCI PDU.
RNTI	uint16_t	The RNTI identifying the UE. For semi-static information held in the MAC this will be the value passed to the PHY in a `UL_CONFIG.request` CQI PDU. See [3] section 5.1.4 Value: 1 → 65535.
Length	uint16_t	Length of PDU in bytes.
Data Offset	uint16_t	Gives the PDU#i data address offset from the beginning of the 'Number of PDUs' field. An offset of 0 indicates a CRC or decoding error, or only RI received on PUSCH.
UL_CQI	uint8_t	SNR Value: 0-255, representing -64dB to 63.5dB, with 0.5dB step size.
RI	uint8_t	The rank indication reported by the UE on PUSCH Value: 0..4 0 = RI not received 1..4 = RI value
Timing Advance	uint16_t	The timing advance measured for this PDU and UE. Value: T_A from 0 to 1282 as defined in [6] section 4.2.3.
}		
PDU#1	Variable	Contents of PDU#1. Raw format CQI report as defined in [9]. The first bit of the CQI report is bit [0] of byte 0.
PDU#2	Variable	Contents of PDU#2. Raw format CQI report as defined in [9]. The first bit of the CQI report is bit [0] of byte 0.
..	...	...
PDU#n	Variable	Contents of PDU#n. Raw format CQI report as defined in [9]. The first bit of the CQI report is bit [0] of byte 0.

Table 83: `RX_CQI.indication` message body

3.3.3.6 RACH.indication

The format of the `RACH.indication` message is given in Table 84.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of Preambles	uint8_t	Number of RACH preambles
For (Number of Preambles) {		
RNTI	uint16_t	The RA-RNTI value See [3] section 5.1.4 Value: 1 → 65535.
Preamble	uint8_t	The detected preamble Value: 0 → 63
Timing Advance	uint16_t	The measured timing advance for the preamble. Value: 0 → 1282

Table 84: `RACH.indication` message body

3.3.3.7 SRS.indication

The format of the `SRS.indication` message is given in Table 85.

Field	Type	Description
SFN/SF	uint16_t	The SFN/SF of the SUBFRAME this information was received in. A 16-bit value where, [15:4] SFN, range 0 → 1023 [3:0] SF, range 0 → 9
Number of UEs	uint8_t	Number of UEs contributing to the uplink SRS
For (Number of UEs){		
Handle	uint32_t	The handle received in the SRS PDU.
RNTI	uint16_t	The RNTI passed to the PHY in an uplink subframe configuration PDU
Doppler estimation	uint16_t	FFS.

Field			Type	Description
				Values: 0 → 255,
	Timing Advance		uint16_t	The timing advance measured for the UE. Value: T_A from 0 to 1282 as defined in [6] section 4.2.3.
	Number of resource blocks		uint8_t	Number of resource blocks to be reported for this UE
	RB start		uint8_t	The starting point of the RBs to be reported.
	For (Number of RBs) {			
		SNR	uint8_t	Field size dependent on configured bandwidth SNR for RBs, each RBs report one SNR. Value: 0-255, representing -64dB to 63.5dB, with 0.5dB step size.”
	}			
}				

Table 85: `SRS.indication` message body

3.4 Error Codes

The list possible error codes returned in either `.response` messages or the `ERROR.indication` message is given in Table 86.

Value	Error Code	Description
0	MSG_OK	Message is OK.
1	MSG_INVALID_STATE	The received message is not valid in the PHY's current state.
2	MSG_INVALID_CONFIG	The configuration provided in the `CONFIG.request` message was invalid
3	SFN_OUT_OF_SYNC	The `DL_CONFIG.request` was received with a different SFN than the PHY expected.
4	MSG_SUBFRAME_ERR	An error was received in `DL_CONFIG.request` or `UL_CONFIG.request`. The sub-error code should be analyzed
5	MSG_BCH_MISSING	A BCH PDU was expected in the `DL_CONFIG.request` message for this subframe. However, it was not present.
6	MSG_INVALID_SFN	The received `HI_DCI0.request` or `TX.request` message included a SFN/SF value which was not expected. The message has been ignored.

Value	Error Code	Description
7	MSG_HI_ERR	An error was received in `HI_DCI.request`. The sub-error code should be analyzed
8	MSG_TX_ERR	An error was received in `TX.request`. The sub-error code should be analyzed

Table 86: L1 API error codes

3.4.1 Sub Error Codes

If the ERROR code is MSG_PDU_ERR then subframe error codes are provided. These sub error codes are given in Table 87.

Value	Error Code	Description
FFS		

Table 87: L1 API subframe error codes

4 References

Ref.	Title	Number	Version & Date
[1]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control	TS36.331	8.5.0
[2]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control(MAC) protocol specification	TS36.321	8.5.0
[3]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN) Overall Description; Stage 2	TS36.300	8.8.0
[4]	3GPP Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Architecture description	TS36.401	8.5.0
[5]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) procedures in idle mode	TS36.304	8.5.0
[6]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer procedures	TS36.213	8.6.0
[7]	3GPP Base Station(BS) Radio Transmission and Reception	TS36.104	8.5.0
[8]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channel and Modulation	TS36.211	8.6.0

[9]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding	TS36.212	8.6.0
[10]	3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment(UE) radio access capabilities	TS36.306	8.3.0
[11]	3GPP Home eNodeB (HeNB) Operations, Administration, Maintenance and Provisioning (OAM&P); Information model for Type 1 interface HeNB to HeNB Management System (HeMS)	TS32.592	0.4.0

5 Revision History

Version	***Description***
1.11	Prepared for public release