FlexRAN Reference Solution L2-L1

API Specification Software v20.2

February 2020

Intel Confidential

Document Number: 571742-1.8

Contents

Figures

Tables

Revision History

Date	Revision	Description
February 2020	1.8	Updates in support of Release 20.02. New reserved rows in Table 41 and Table 91
June 2018	1.7	Updates in support of Release 1.6.0. pmchstart added in Table 69 DLSubframeCommonControlStructure
April 2018	1.6	Updates in support of Release 1.5.0. Added PHY_SHUTDOWN confirmation and request APIs, and added noiseDemodCellWide status and report.
December 2017	1.5	Updates in support of Release 1.4.0.
September 2017	1.4	Updates in support of Release 1.3.0.
June 2017	1.3	Updates in support of Release 1.2.0.
June 2017	1.2	Supports Release 1.1.2.
April 2017	1.1	Updated in support of Release 1.1.0.
December 2016	1.0	First Issue

1.0 Preface

1.1 About this Document

This document describes the FlexRAN reference physical layer (PHY) for a Long Term Evolution (LTE) wireless base station using Intel® Xeon® processors and Intel FerryBridge field-programmable gate array (FPGA). This Application Program Interface (API) sits between the L2+ and the physical layer of a LTE base station.

This document describes the different procedures supported by this API, the command sequences for each procedure, the command timing requirements, configuration parameters and any additional information that a user of this API requires to know prior to integrating the L2+ stack with the PHY.

1.2 Audience

Developers, applications engineers, and other personnel who are looking to implement, support, or otherwise become familiar with the FlexRAN reference design, can log on and register at the Intel® Business Link to access the full range of support documentation. For detailed information on specific devices and available features, always consult the Intel® Business Link. Table 1 gives a summary of the Intel publications related to this document.

1.3 Related Documentation

A comprehensive library of FlexRAN reference and application documentation is available from Intel.

Table 1. Related Documentation

Title	Document Number
FlexRAN Reference Solution L1 User Guide	570228
FlexRAN Reference Solution RefPHYDocument	572318
FlexRAN Reference Solution L1 XML Configuration User Guide	571741
FlexRAN Reference Solution v1.5.0 Release Notes	575822

Table 2. 3GPP 36 Series Standard Document References

Reference	Title	Number	Version
[36.331]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control*	TS36.331	10.10.0 († 11.4.0)

Reference	Title	Number	Version
[36.321]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification*	TS36.321	10.9.0 († 11.3.0)
[36.300]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN) Overall Description; Stage 2*	TS36.300	10.10.0
[36.401]	*3GPP Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Architecture description*	TS36.401	10.4.0
[36.304]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) procedures in idle mode*	TS36.304	10.6.0
[36.213]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer procedures*	TS36.213	10.10.0 († 11.4.0)
[36.104]	*3GPP Base Station (BS) Radio Transmission and Reception*	TS36.104	10.9.0
[36.211]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channel and Modulation*	TS36.211	10.7.0 († 11.3.0)
[36.212]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding*	TS36.212	10.8.0 († 11.3.0)
[36.306]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio access capabilities*	TS36.306	10.10.0
[36.214]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer; Measurements*	TS36.214	10.10.0
[36.141]	*3GPP Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) conformance testing*	TS36.141	11.11.0
	SCAPI Small Cell Forum "LTE eNB L1 API Definition" v2.4 November 18, 2013		

Note: † EPDDCH

1.4 Terminology

Table 3. Terminology

Term / Acronym	Definition
3GPP	3rd Generation Partnership Project
ACK	Acknowledge
API	Application Program Interface
BCH	Broadcast Channel
CA	Carrier Aggregation
CC	Convolutional Code
CCE	Control Channel Element
CDD	Cyclic Delay Diversity
CL	Convergence Layer
CFI	Control Format Indicator
CNM	Continuous Network Monitor
CQI	Channel Quality Indicator
CRC	Cyclic Redundancy Check

Term / Acronym	Definition
CS	Cyclic Shift
CSI-RS	Channel State Indication Reference Signals
CTC	Convolutional Turbo Codes
DCI	Downlink Control Information
DL	Downlink
DL-SCH	Downlink Shared Channel
DFS	Dynamic Frequency Selection
DMA	Direct Memory Access
DMRS	Demodulation Reference Symbol
DwPTS	Downlink Pilot Time Slot
EARFCN	E-UTRA Absolute Radio Frequency Channel Number
ECCE	Enhanced Control Channel Element
eNB	evolved Node B
EPC	Evolved Packet Core
EPDCCH	Enhanced Physical Downlink Control Channel
EPRE	Energy Per Resource Element
EREG	Enhanced Resource Element Group
E-UTRA	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
FIFO	First-In First-Out
FPGA	Field-Programmable Gate Array
GP	Guard Period
HARQ	Hybrid Automatic Repeat Request
HCS	Header Check Sequence
HeMS	Home eNodeB Management System
HI	HARQ Indicator
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

Term / Acronym	Definition
MIB	Master Information Block
MIMO	Multiple Input Multiple Output
MU-MIMO	Multi User MIMO
NA	Not Applicable
NACK	Negative Acknowledge
NMM	Network Monitor Mode
OFDMA	Orthogonal Frequency Division Multiple Access
OS	Operating System
OTA	Over the Air
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
PRS	Positioning Reference Signals
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

Term / Acronym	Definition
SON	Self-Organizing Network
SR	Scheduling Request
SRS	Sounding Reference Symbol
STC	Space Time Coding
STTD	Space Time Transmit Diversity
SU-MIMO	Single User MIMO
TB	Transport Block
TDD	Time Division Duplex
TLV	Tag Length Value
TTI	Transmission Time Interval
Tx	Transmit
UCI	Uplink Control Information
UE	User Equipment
UE-SRS	UE Specific Reference Signals
UL	Uplink
ULCCH	Uplink Control Channel
ULSCH	Uplink Shared Channel
UpPTS	Uplink Pilot Time Slot
WLS	Wireless Subsystem Interface
WCDMA	Wideband Code Division Multiple Access
X2	Interface between two eNBs
ZBC	Zero Buffer Copy
ZT CC	Zero Tailed Convolutional Coding

2.0 Introduction

This document describes the FlexRAN Reference Solution L2-L1 API, which provides the interface to the LTE PHY running on an Intel® Xeon® processor. The L2/L3 protocols expected to use this API are the Radio Resource Control (RRC) layer [36.331] (refer to Table 2), Medium Access Control (MAC) layer [36.321] (refer to Table 2), and the scheduler. It is assumed that these run also on an Intel® Xeon® processor on a different core.

The LTE standard [36.300] has been designed to support both Time Division Duplex (TDD) and Frequency Division Duplex (FDD) deployments. The L1 API described in this document currently supports both FDD and TDD modes; the differences between these modes are highlighted in the document.

This document is divided into three main sections:

- *L1 API Procedures* provides a description of typical procedures that occur between the L1 and L2/L3 software.
- *L1 API Messages* provides the definition and encoding of the L1 API messages.
- *Physical Implementation* provides a brief overview of transport mechanisms between the L1 and L2/L3 software.

The API described in this document facilitates a broad range of LTE functionality. The user should consult the relevant L1 release note to determine the actual functionality supported by the product.

2.1 Long Term Evolution

LTE is standardized by the 3GPP* Mobile Broadband Standard (refer to Table 2),and designed as an evolution to the current Wideband Code Division Multiple Access (WCDMA) wireless network in widespread use today. A critical requirement of LTE is the capability of supporting high data rates (Up to 1 Gbs). Many aspects of the LTE network have been designed specifically to support high data rates and low latency.

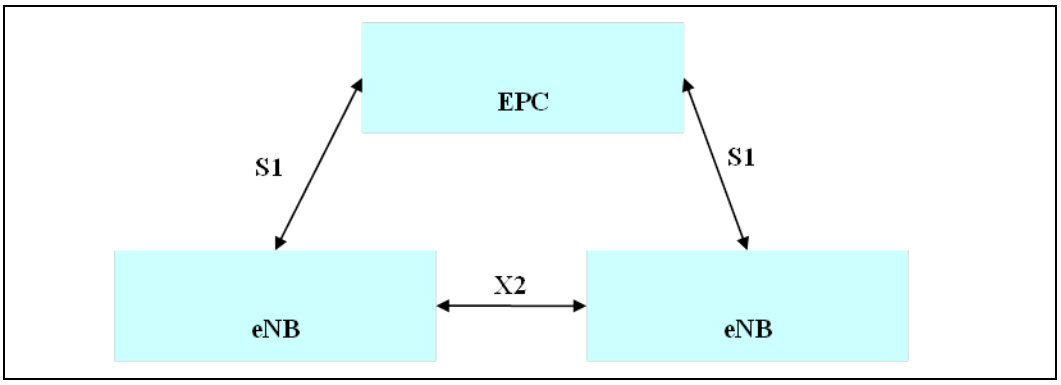
Figure 1 shows the architecture of an LTE network. It consists of only two elements:

- the Evolved Pack Core (EPC)
- the E-UTRAN Node B (eNB).

The LTE L1 API resides within the eNB element. The two standardized interfaces in an LTE network are called S1 and X2.

Note: The L1 is not involved in either of these interfaces, and both are out of scope for this document.

Figure 1. LTE Architecture



2.2 L1 API

The L1 API, defined in this document, resides within the eNB component. Figure 2 and Figure 3 shows the functionality of an eNB and highlight the location of the L1 API in that model.

Figure 2 shows the protocol model for the eNB defined in the E-UTRAN architectural standard [36.401]. It highlights the separation of control- and data-plane information, which is maintained throughout the LTE network. Each API message contains either control or data-plane information, but never both, even though both control-plane and data-plane information is passed through the L1 API.

Figure 2. E-UTRAN Protocol Model

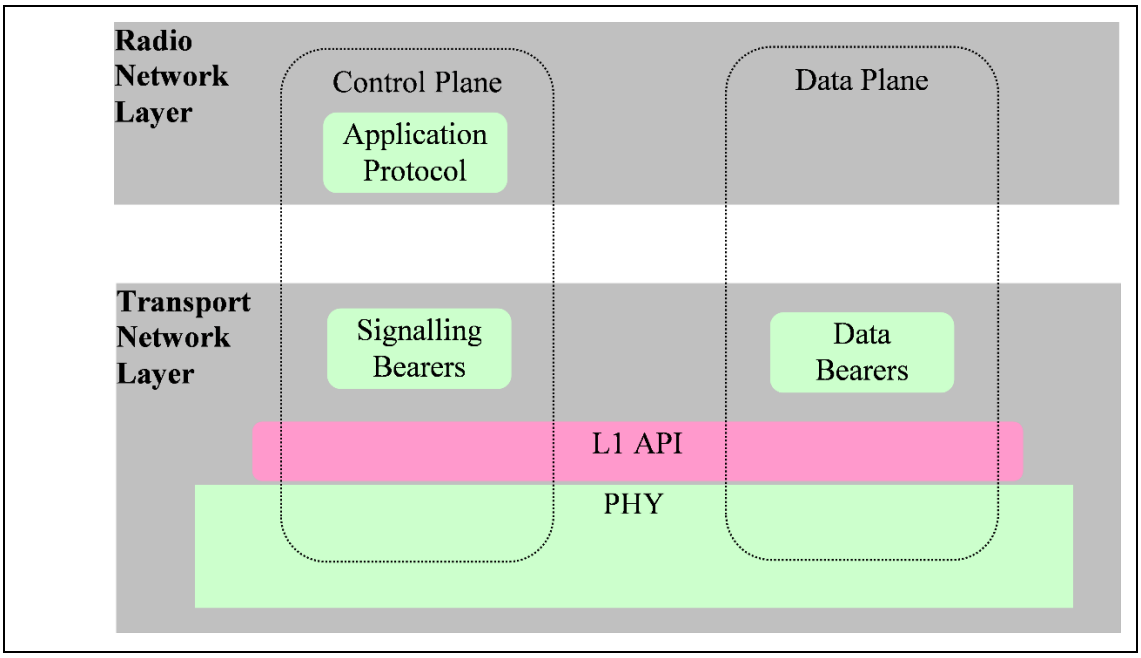


Figure 3 provides an example of how the different L2/L3 protocol layers interact with the L1 API. In this example, a PHY control entity is responsible for configuration procedures.

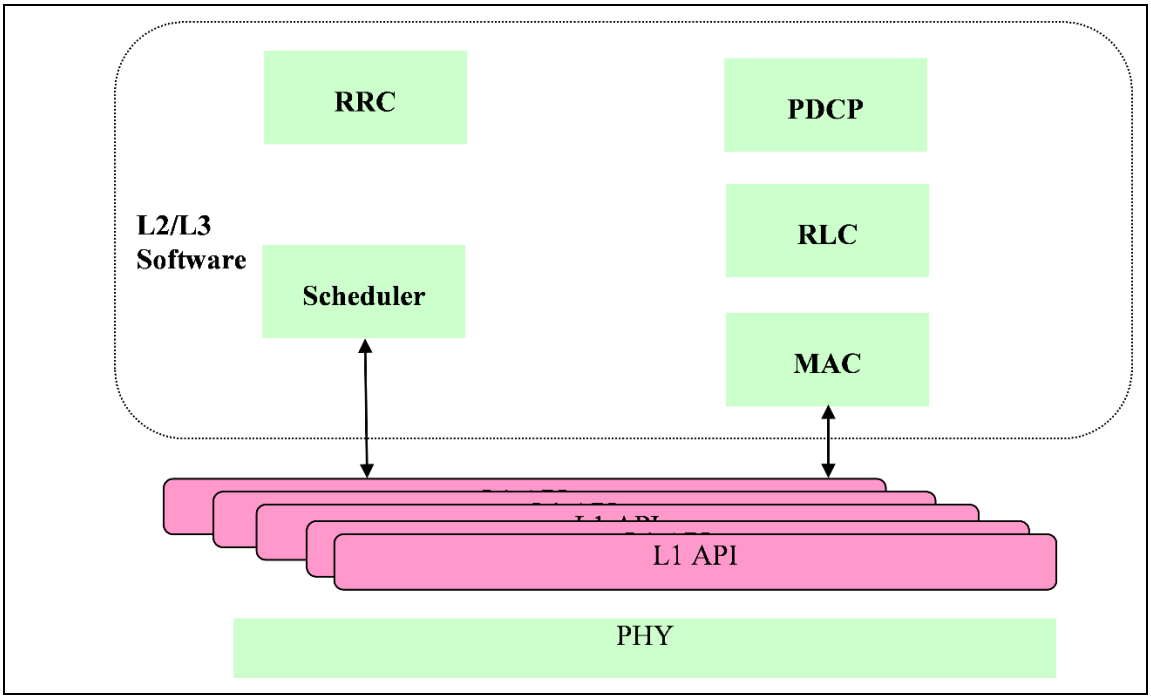
The MAC layer is responsible for the exchange of data-plane messages with the PHY. The PHY configuration sent over the L1 interface may be determined using Self-Organizing Network (SON) techniques, information model parameters sent from the Home eNodeB Management System (HeMS), or a combination of both methods. The MAC layer is responsible for the exchange of data-plane messages with the PHY.

Finally, the scheduler is responsible for deciding the subframe structure, and receiving measurement information from the PHY.

Note: There is no requirement for the eNB protocol to follow this example.

The L1 API is a collection of messages that can be routed anywhere by the eNB software. For carrier aggregation support there is one instance of the API per each component carrier used.

Figure 3. L1 API Interactions



3.0 L1 API Procedures

This section provides an overview of the procedures that use the L1 API:

- Configuration procedures handle the management of the PHY layer, and are expected to occur infrequently.
- Subframe procedures determine the structure of each 1 ms subframe and operate with a 1 ms periodicity.

3.1 Configuration Procedures

The configuration procedures supported by the L1 API are:

- Initialization
- Termination
- Restart
- Reset
- Reconfigure
- Query
- Notification

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

Figure 4. PHY Layer State Transactions on L1 API Configuration Messages

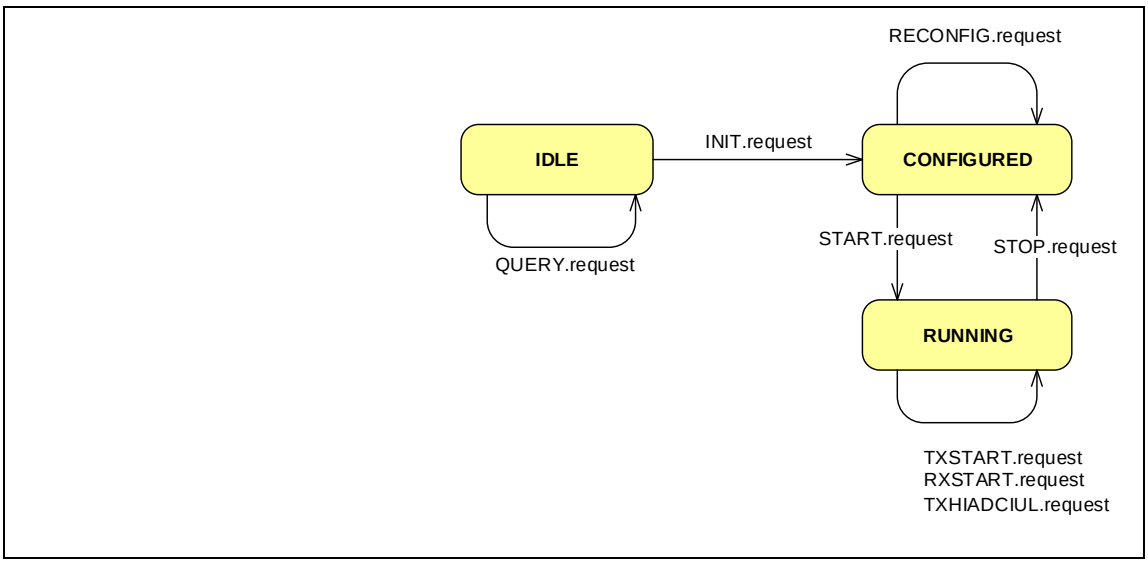


Table 4. L1 API Request Message Valid in each PHY State

Idle State	Configured State	Running State
`INIT.request`	`RECONFIG.request`	`STOP. request`
	`START.`	`TXSTART.request`

Idle State	Configured State	Running State
		RXSTART.request
		TXHIADCIUL.request
QUERY.request		

3.1.1 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:

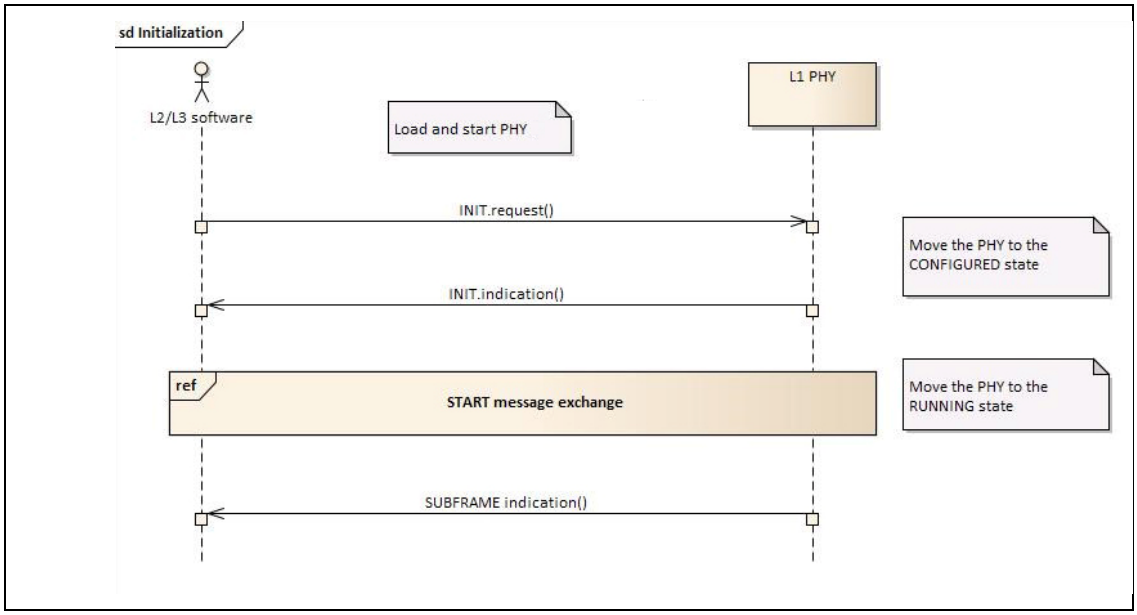
- The PHY application must be executed in the target processor.
- L2/L3 software must go through the INIT message exchange procedure.
- The START message exchange procedure must be executed.

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

The remainder of this section describes the INIT and START message exchange procedures.

3.1.1.1 INIT Message Exchange Procedure

Figure 5. Initialization Procedure



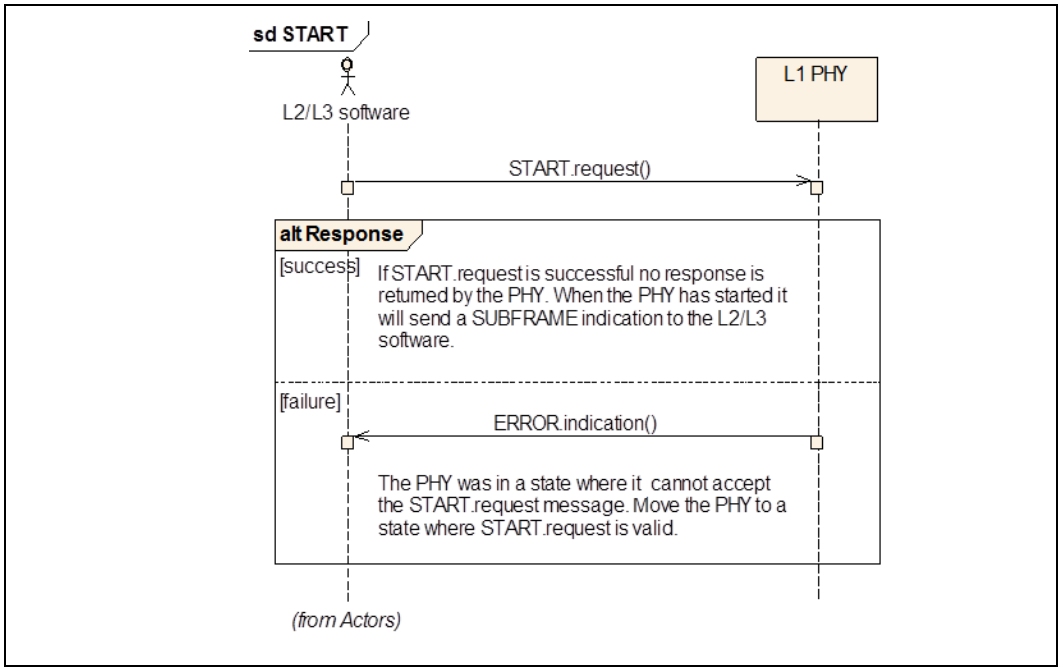
The INIT message exchange procedure is shown in Figure 5. Its purpose is to allow the L2/L3 software to configure the PHY. It is used only after a system reset.

The PHY_INIT message must contain all the entries defined in Section 4.2.2, PHY_INIT Indication, otherwise the command will return an error message indicating an incorrect configuration in the PHY_INIT.indication message and the PHY remains in the IDLE state.

3.1.1.2 START Message Exchange Procedure

The START message exchange procedure is shown in Figure 6. Its purpose is to instruct a configured PHY to start transmitting as an eNB. The L2/L3 software initiates this procedure by sending a PHY_START.request message to the PHY. If the PHY is in the CONFIGURED state, it starts an internal 1ms timer and waits for an internal command to issue a SUBFRAME indication in the form of a PHY_TXSTART.indication. An absence of a PHY_TXSTART.indication past the expected time for the indication to be issued constitutes an error that the L2/L3 software must interpret as the command being issued in the wrong state.

Figure 6. Start Message Exchange



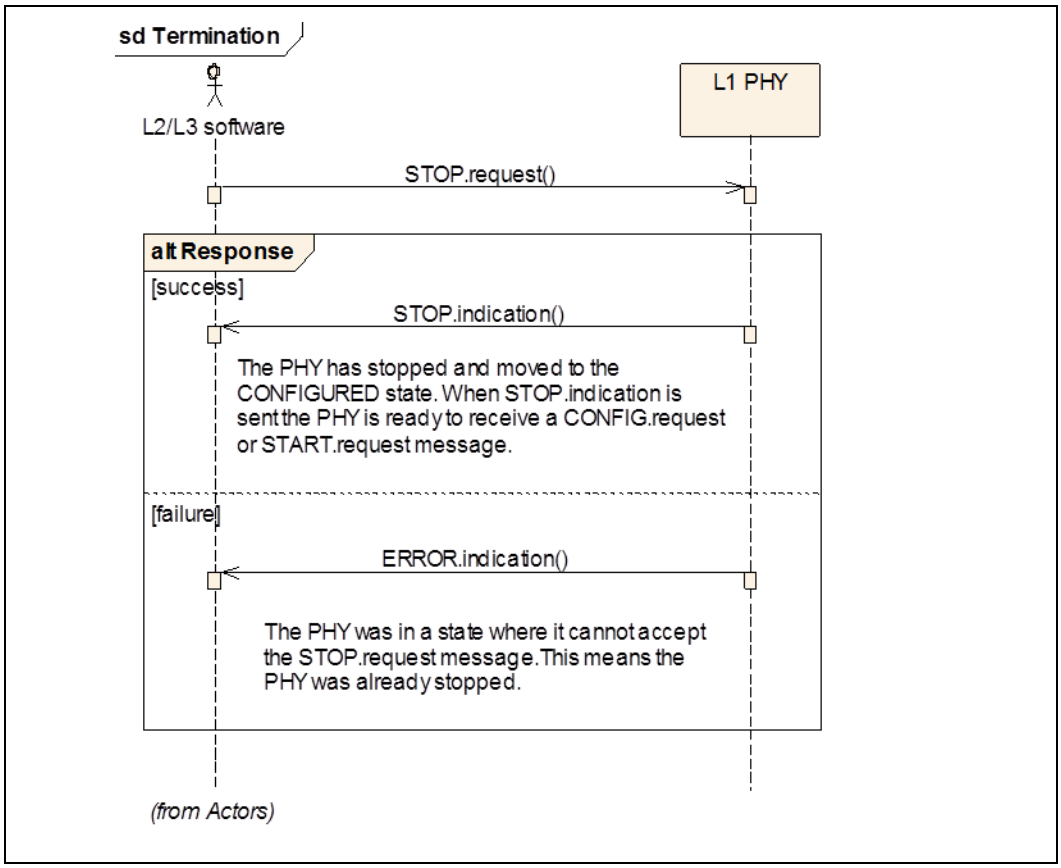
3.1.2 Termination

The termination procedure moves 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 7 and initiated by the L2/L3 software sending a STOP.request message.

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

If the PHY_STOP.request message was received by the PHY while in the IDLE or CONFIGURED state, it will return a PHY_STOP.indication message with a status field reporting the error. However, in this case, the PHY was already stopped.

Figure 7. Termination Procedure

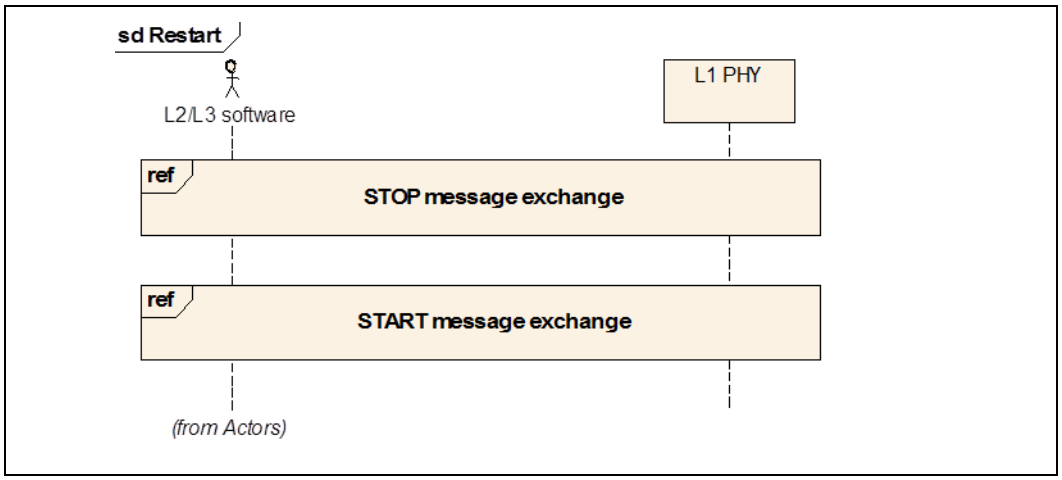


3.1.3 Restart

The restart procedure is shown in Figure 8. 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 7. This moves the PHY to the `CONFIGURED` state. To restart transmission it should follow the `START` message exchange, shown in Figure 6, moving the PHY back to the `RUNNING` state.

Note: Currently this procedure is supported in timer mode only.

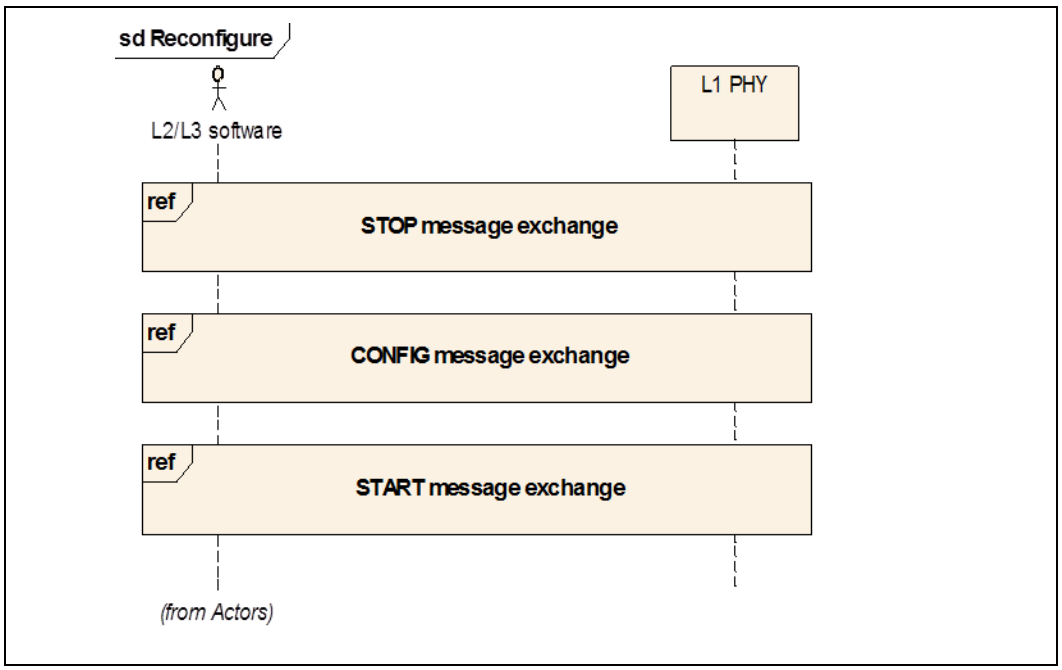
Figure 8. Restart Procedure



3.1.4 Reconfigure

The reconfigure procedure is shown in Figure 9. It is used when the L2/L3 software makes significant changes to the configuration of the PHY. The `STOP` message exchange, shown in Figure 7, is followed to halt the PHY and move it to the `CONFIGURED` state. The `RECONFIG` message exchange is used to reconfigure the PHY. Finally, the `START` message exchange, shown in Figure 6, is followed to restart the PHY and return it to the `RUNNING` state.

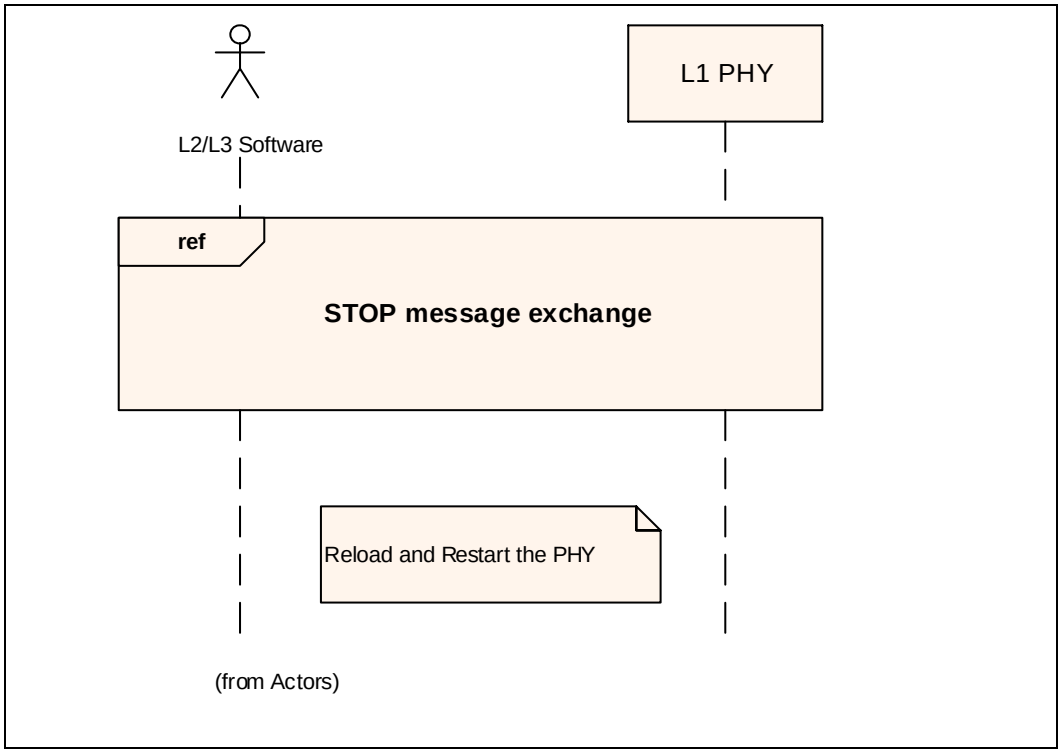
Figure 9. Reconfigure Procedure



3.1.5 Reset

The reset procedure is shown in Figure 10. 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 7) and then restarting the PHY on the target device.

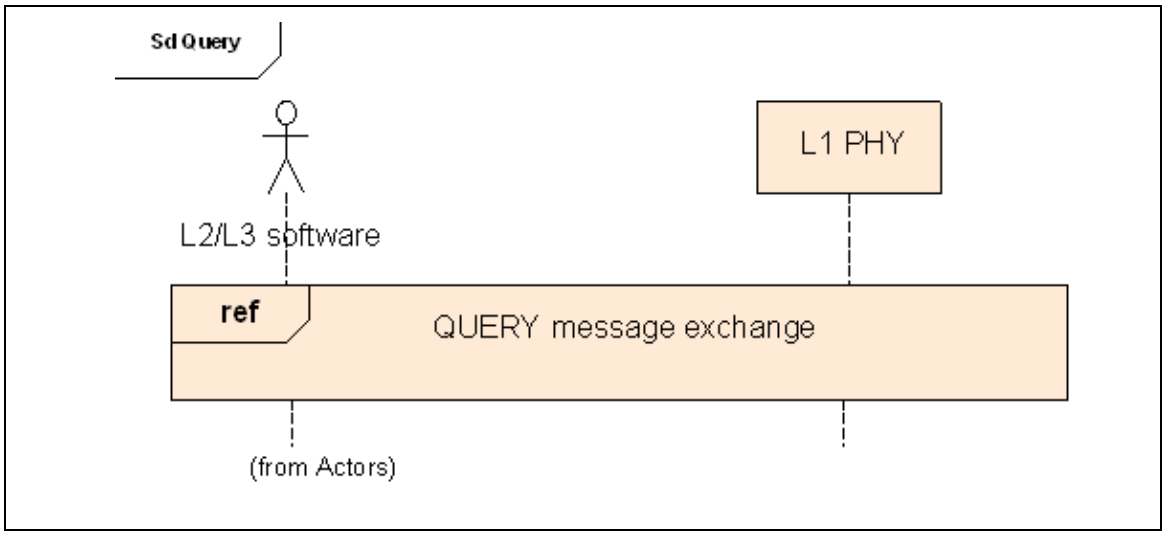
Figure 10. Reset Procedure



3.1.6 Query

The query procedure is shown in Figure 11. This procedure allows the L2/L3 code to find out what version of the PHY API is being used to allow easy identification of the API feature set supported by the PHY.

Figure 11. Query Procedure

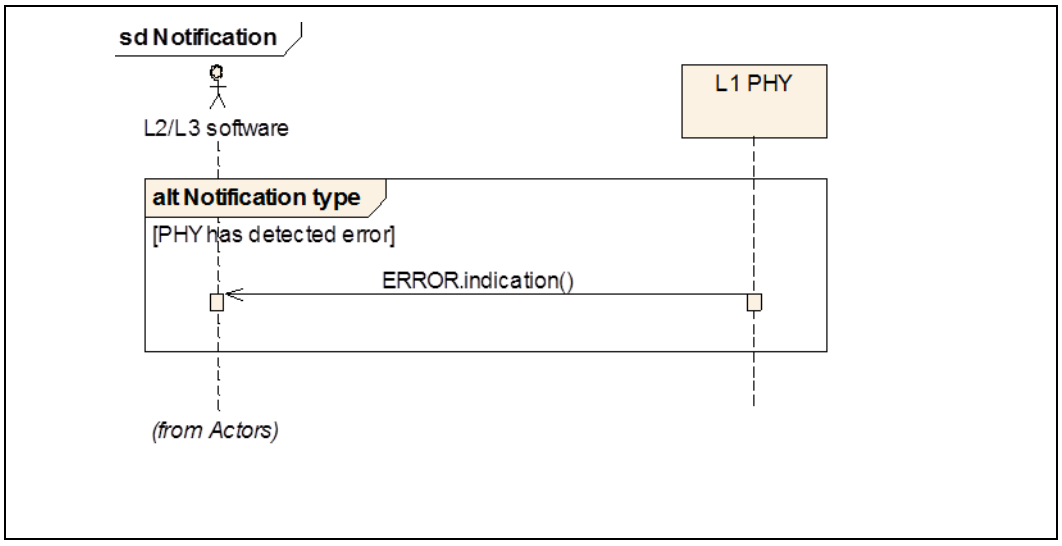


3.1.7 Notification

The notification procedure is shown in Figure 12. 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 is used by the PHY to indicate that the L2/L3 software has sent invalid information to the PHY or that the information was not delivered within the valid time window.

Figure 12. Notification Procedures



3.2 Subframe Procedures

The subframe procedures have two purposes:

- They are used to control the DL and UL frame structures.

- 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 1 ms SUBFRAME signal
- Synchronization of System Frame Number/Subframe (SFN/SF) between the L2/L3 software and PHY
- Transmission of the Broadcast Channel (BCH) transport channel
- Transmission of the Paging Channel (PCH) transport channel
- Transmission of the Downlink Shared Channel (DL-SCH) transport channel and reception of Acknowledge/Negative Acknowledge (ACK/NACK) response
- Transmission of the MCH transport channel
- Transmission of Positioning Reference Signals
- Transmission of Channel State Indication Reference Signals
- Transmission of User Equipment (UE) Specific Reference Signals
- Reception of the (RACH) transport channel
- Reception of the Uplink Shared Channel (UL-SCH) transport channel and transmission of ACK/NACK response
- Reception of the sounding reference signal
- Reception of Channel Quality Indicator (CQI) and Rank Indicator (RI) reporting
- Reception of scheduling request information
- Rssi measurements
- Transmit Power measurements
- Thermal Noise Power Measurements
- Received Interference Power Measurements

3.2.1 SUBFRAME Indication

A SUBFRAME indication is sent from the PHY to the L2/L3 software, indicating the start of a 1 ms frame. The actual subframe indication message used is the PHY_TXSTART.indication message. In the case of TDD where there could be UL or DL only subframes a PHY_RXSTART.indication message is issued for the UL only subframes.

The SUBFRAME indications issued in TDD mode (frame structure 2) are done at 1 ms intervals and there are two possible TDD frame structures. The first one uses a 5 ms switch point periodicity and the second one uses 10 ms switch point periodicity. Figure 13 shows the 5 ms switch point periodicity case and Figure 14 illustrates the 10 ms switch point periodicity case.

Subframes 0 and 5 and Downlink Pilot Time Slot (DwPTS) are always reserved for downlink transmission. Uplink Pilot Time Slot (UpPTS) and the subframe immediately following the special subframe are always reserved for uplink transmission. GP is the guard period.

Figure 13. SUBFRAME Signal for TDD with 5 ms Switch Point Periodicity

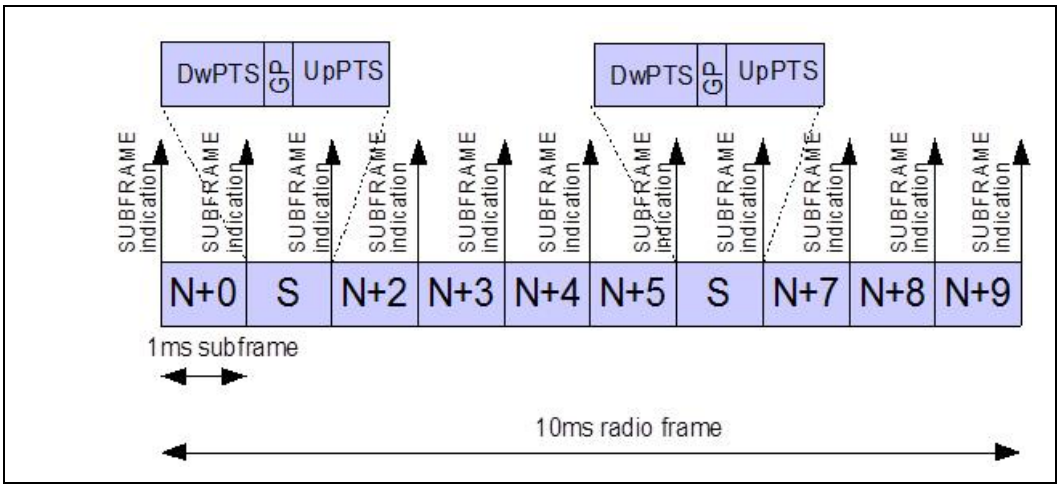
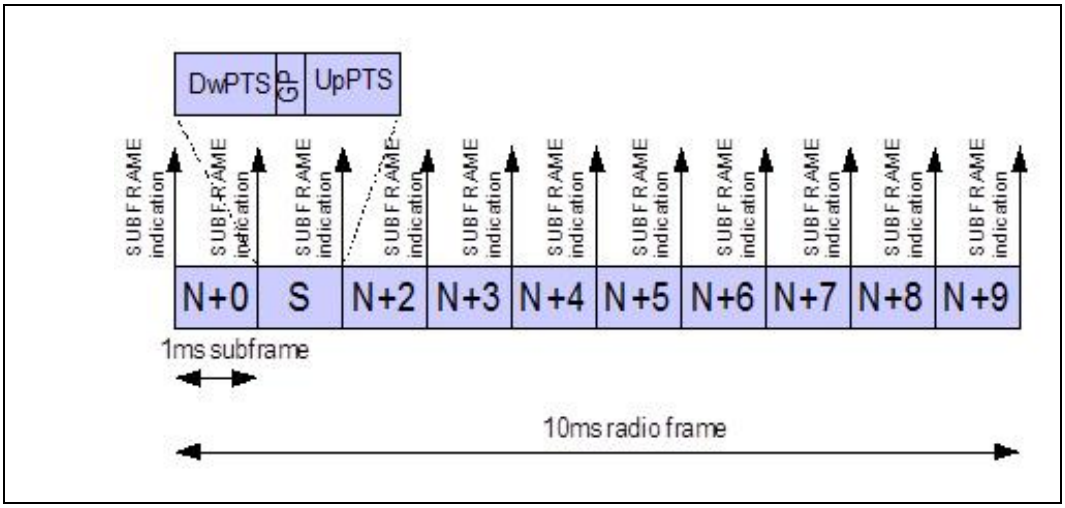
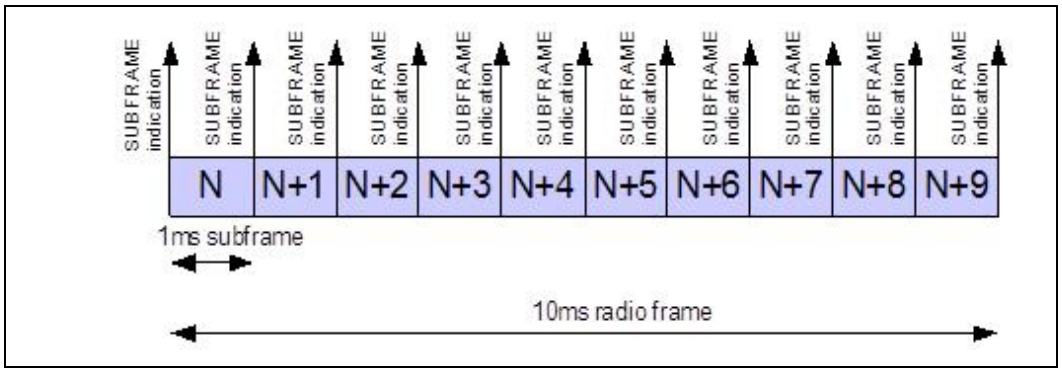


Figure 14. SUBFRAME Signal for TDD with 10 ms Switch Point Periodicity



The location of the SUBFRAME indication for FDD (frame structure 1) is shown in Figure 15. The subframe indication is generated at the start of every DL subframe.

Figure 15. SUBFRAME Signal for FDD



3.2.2 SFN/SF Synchronization

The SFN/SF synchronization procedure is used to maintain a consistent SFN/SF value between the L2/L3 software and PHY.

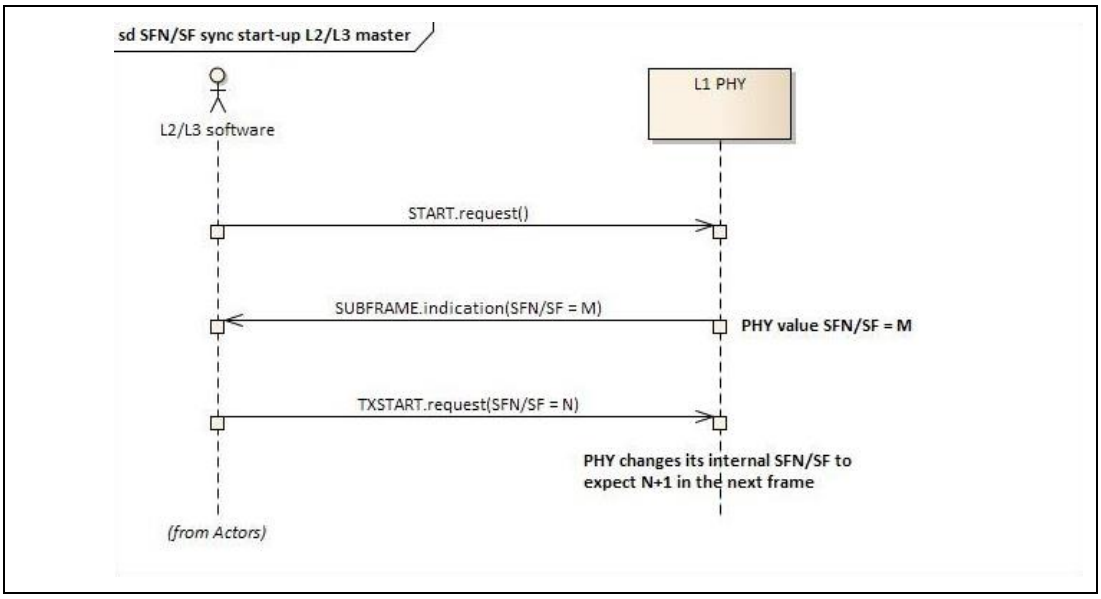
Currently only one option is provided by the L1 API. So the PHY uses the SFN/SF value provided by the L2/L3 software.

The SFN/SF synchronization start-up procedure, where the L2/L3 software is master, is given in Figure 16.

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 is the value passed by the L2/L3 during PHY_INIT. The PHY sends a `SUBFRAME.indication` message to the L2/L3 software with `SFN/SF = M`
- The L2/L3 software sends a `SUBFRAME.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 16. SFN/SF Synchronization start-up with L2/L3 Master



The SFN/SF synchronization maintenance procedure is shown in Figure 17. In this example, the L1 PHY is expecting the next `SUBFRAME.request` to contain information regarding frame M. The procedure followed is:

- The PHY sends a `SUBFRAME.indication` message, `SFN/SF = M`.
- The L2/L3 software sends a `SUBFRAME.request` message to the PHY containing `SFN/SF = N`
- If `SFN/SF M = N-1`

The PHY received the SFN/SF it was expecting. No SFN/SF synchronization is required.

- If `SFN/SF M ≠ N-1`

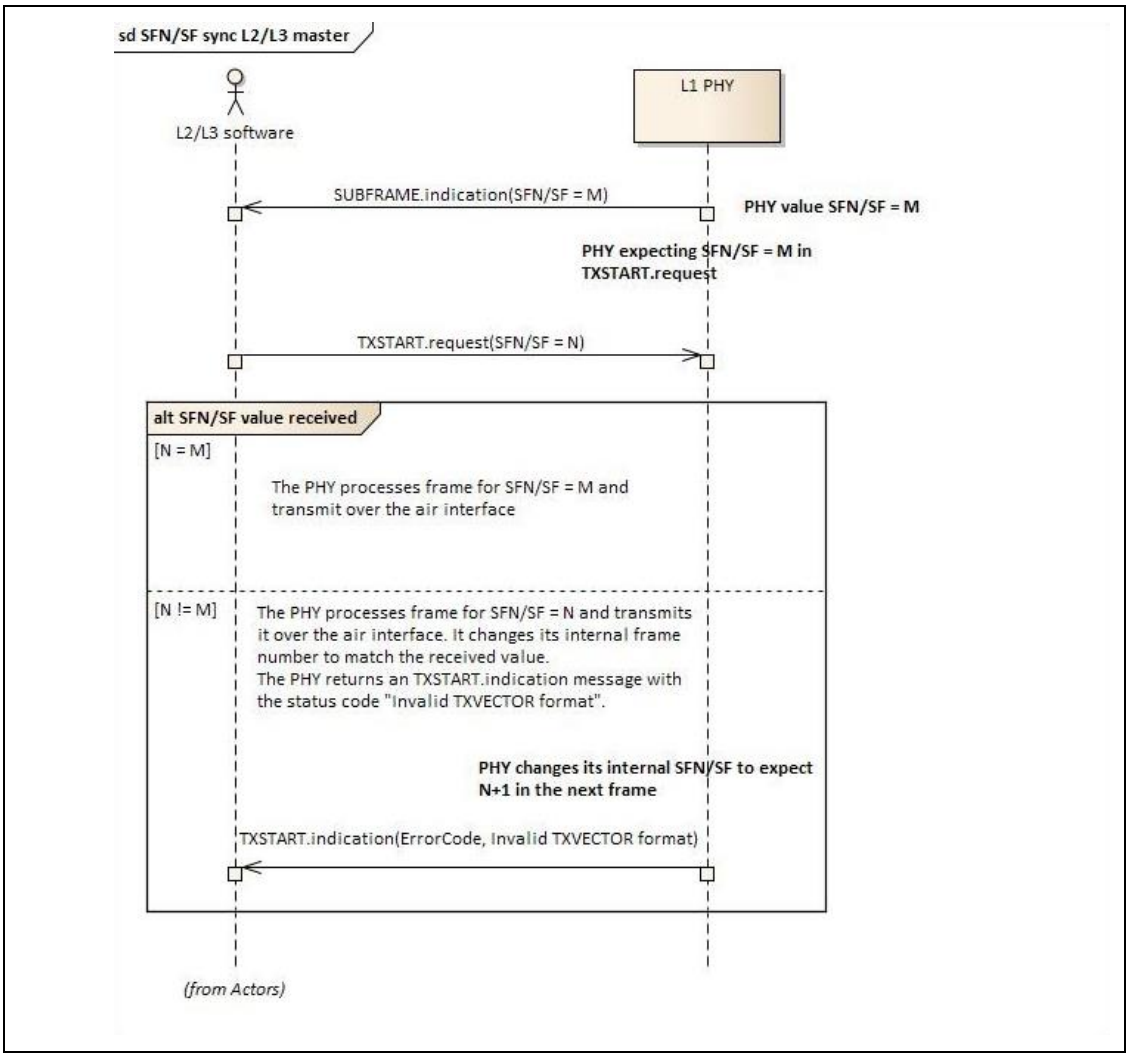
– The PHY received a different SFN/SF from the expected value. SFN/SF synchronization is required.

– 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 with the status field containing invalid TX VECTOR format.

This SFN/SF synchronization procedure assumes the L2/L3 software is always correct. However, it is possible that the SFN/SF synchronization was unintended, due to a L2/L3 software issue. The recommended method is for the PHY to have robust vector generation support enabled, for details see PHY_INIT parameters in Section 4.2.3.

Figure 17. SFN/SF Synchronization with L2/L3 Master



When robust mode is enabled the PHY at the beginning of the Transmission Time Interval (TTI) checks the status of the First-In First-Out (FIFO) buffer from the host to see if the API messages are present or not. If the FIFO is empty, then the PHY inserts its own dummy TXVECTOR and RXVECTOR with default parameters (i.e DL No channels, only pilots, synchronization signals and PBCH if required and UL No channels).

After finishing the subframe processing, the PHY checks the FIFO one more time to see if any late API messages have arrived. If there are messages, the PHY pops the messages from the FIFO and cleans/free(s) all the memory associated with the messages. It then checks the RxVector to see if the L2/L3 is expecting any RxSDU indications or RXSTATUS indications for PUSCH, PUCCH. If messages were expected, then these are generated with TIMEOUT_RXSTART_REQ in the status field so they would be discarded by the L2/L3. All of these actions happen within the same TTI.

The L2/L3 code needs to be able to deal with messages that appear out of order in relation to the SFN/SF that it maintains whenever a TIMEOUT_RXSTART_REQ message or an ERROR_IND stating TIMEOUT_TXSTART_REQ have been received from the PHY since the PHY will increment the SFN/SF internally in the event of the L2/L3 not being able to keep up with the TTI deadline.

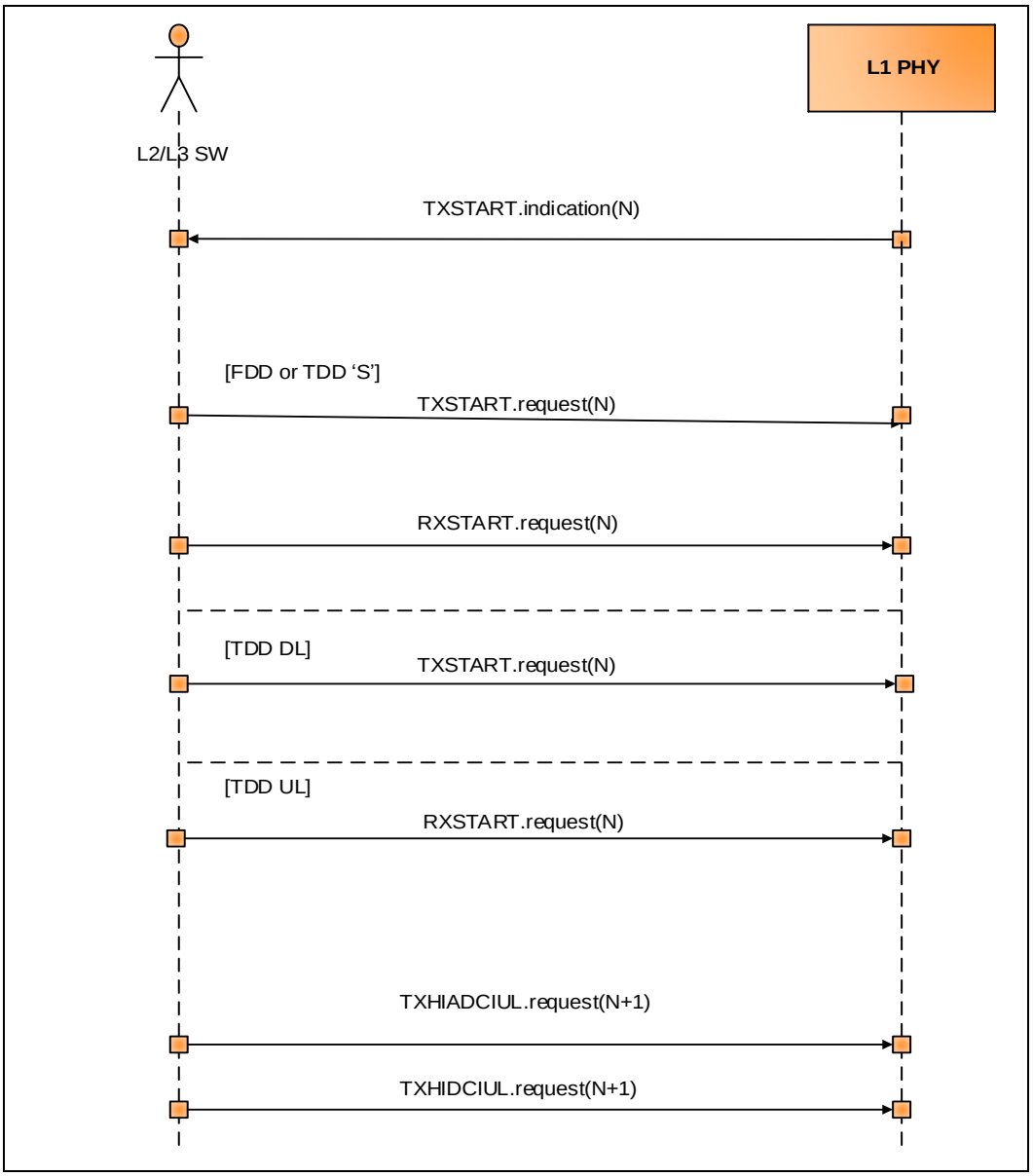
3.2.3 API Message Order

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 18:

- The SFN/SF issued by the PHY in the SUBFRAME.indication message is expected back in the corresponding SUBFRAME.request from the L2/L3.
- The SUBFRAME.request must be sent for every subframe and must be the next message. The TXSTART.request and RXSTART.request are mandatory in FDD mode and the TXHIADCIUL.request is optional.
- In TDD mode TXSTART.request is mandatory in all DL subframes and RXSTART.request is mandatory in all UL subframes.
- If present, the TXHIADCIUL.request must be the next message for the subframe.
- There must be only one TXSTART.request, one RXSTART.request and one TXHIADCIUL.request for a subframe.

Figure 18. DL Message Order

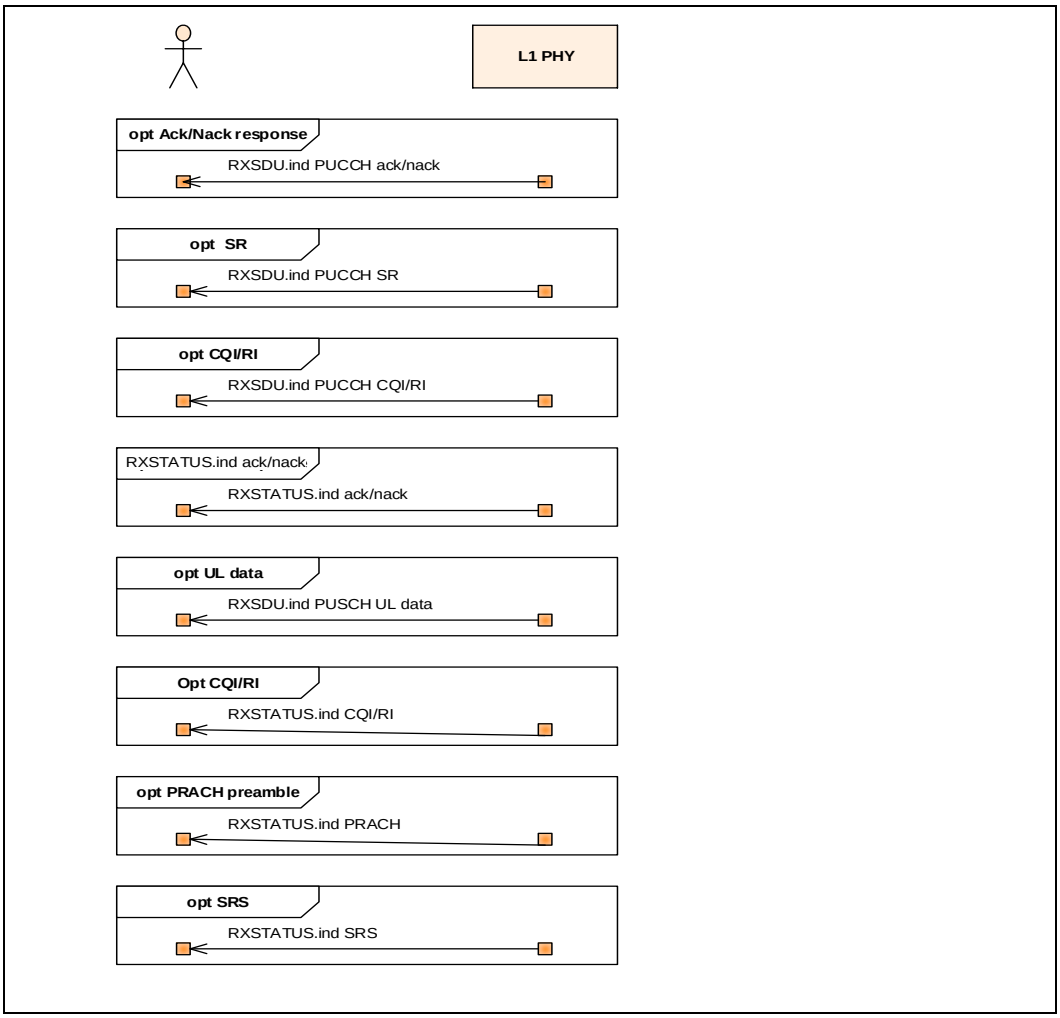


The uplink API message constraints are shown in Figure 19:

- The first messages delivered if present are `PUCCH HARQ ACK/NACK's`, Schedule request(s), CQI and RI using the `RXSDU`.indication with `channelType` being PUCCH.
- If present the Multiplexed PUSCH Control messages for `HARQ ACK/NACK's` are delivered using `RXSTATUS.indication` messages.
- If present the PUSCH messages are delivered next using `RXSDU.indication` messages with `channelType` being PUSCH.
- If present Multiplexed PUSCH Control CQI and RI messages are delivered using `RXSTATUS.indication` messages
- If present the SRS message is delivered using `RXSTATUS.indication`.
- If present the PRACH message is delivered using `RXSTATUS.indication`.

There are as many HARQ indications, RXSDU indications, CQI indications and RI indications as specified by the L2/L3 in the `RXSTART.request` message for a subframe. There are only one SRS and one PRACH indication messages in the form of `RXSTATUS.indication` per subframe in FDD mode. In TDD mode there could be multiple PRACH indication messages depending on the contents of `prachTddFreqResEnabled` variable and the signals received from the UEs.

Figure 19. UL Message Order



3.2.4 API Message Timing

Understanding the L1 API message timing in LTE is complex. The DL-SCH and UL-SCH transport channels both use Hybrid Automatic Repeat Request (HARQ). This means that a DL-SCH has an uplink component to return the `ACK/NACK` response, while the UL-SCH requires a downlink component. In addition, the scheduling of UL-SCH is performed by transmitting control information on the downlink PDCCH physical channel.

The `ACK/NACK` response is required in subframe N+K, and the UE applies the uplink PDCCH control in subframe N+KPUSCH. For FDD K=4 and KPUSCH=4. For TDD K>=4.

LTE FDD defines a maximum of eight HARQ processes for the DL-SCH and 8 HARQ processes for the UL-SCH. To achieve a maximum throughput, a HARQ process must be scheduled every subframe. Therefore, the transmission of data and reception of ACK/NACK response must be completed in eight DL, or UL, subframes.

A sample L1 API message timing is shown in Figure 20 for FDD and K=4. The L1 API timing is:

- Following subframe indication N:

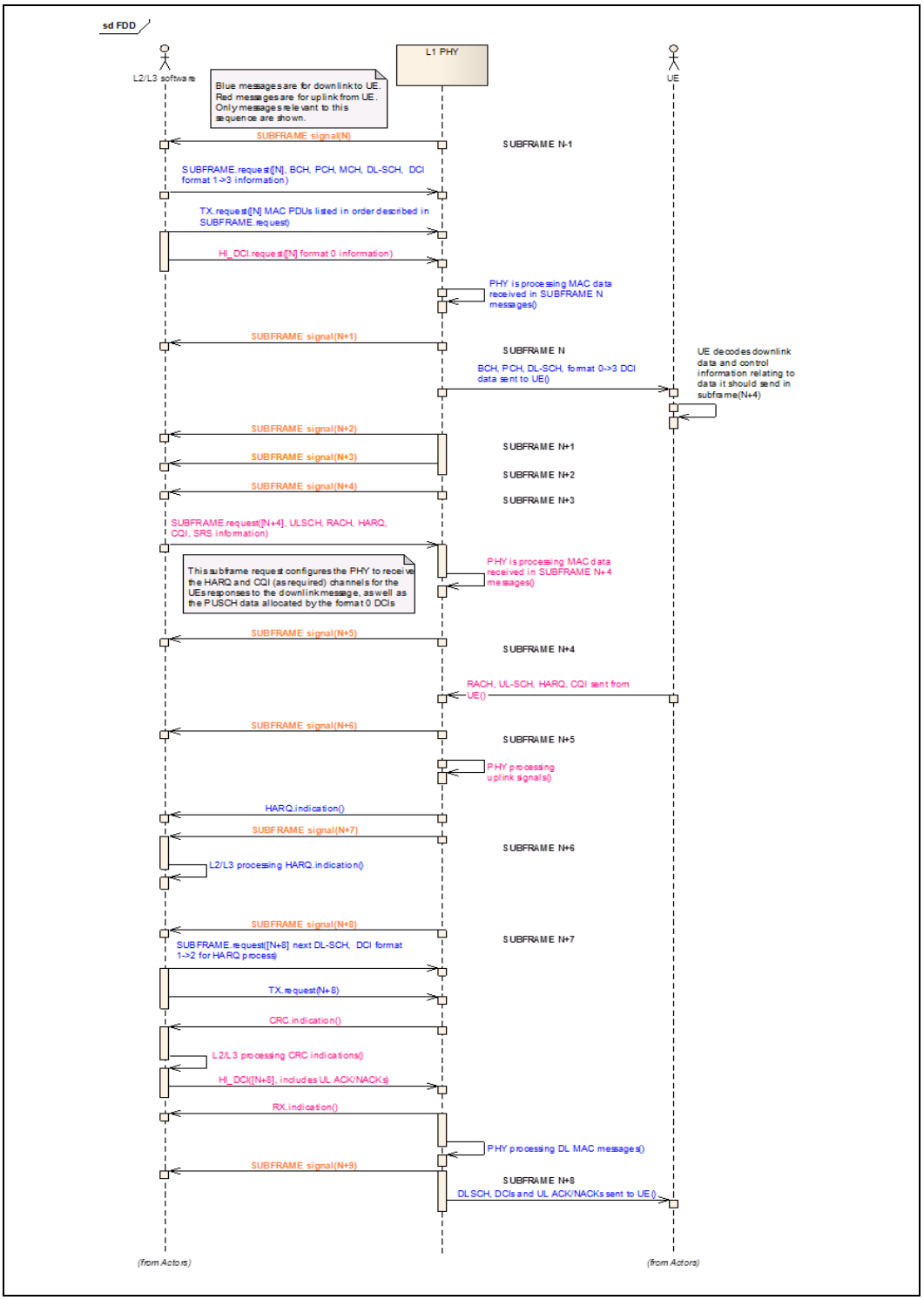
– A TXSTART.request is sent to the PHY. This message includes downlink transport channels and DCI format 1 → 2 information. The DCI information relates to the downlink.
– TXSDU.request(s) is (are) sent to the PHY. The message(s) include the data for the downlink transport channels.
– A TXHIADCIUL.request is sent to the PHY. This message includes DCI format 0 and 3 information, and relates to the uplink.
– TXDCIULSDU.request(s) is (are) sent to the PHY. The message(s) include the data for the downlink control channel(s).
– The eNodeB transmits during air subframe N.

- Following subframe indication (N+4):

– A RXSTART.request is sent to the PHY. This message includes uplink transport channels as well as SR and SRS information used to maintain the uplink connection. Also, it includes CQI and HARQ information which relate to the downlink connection.
– The UE transmits during air subframe (N+4).

- The RXSDU.harq.indication or RXSTATUS.harq.indication, with the ACK/NACK for downlink transmission in subframe N, is sent to the L2/L3 software after subframe indication for (N+6).
- The RXSDU.indication(s) for the uplink transmission in air subframe (N+4) are sent to the L2/L3 software after subframe indication (N+8). This uplink data corresponds to the DCI format 0 information sent in subframe N, and uplink PDUs sent in subframe N+4.
- To maintain maximum data throughput, a TXSTART.request is sent to the PHY in subframe N+8 for the same HARQ process as subframe N.
- A TXHIADCIUL.request is sent to the PHY after subframe indication N+8, containing the ACK/NACK information for the uplink data received in subframe N+4.

Figure 20. API Command Timing Example



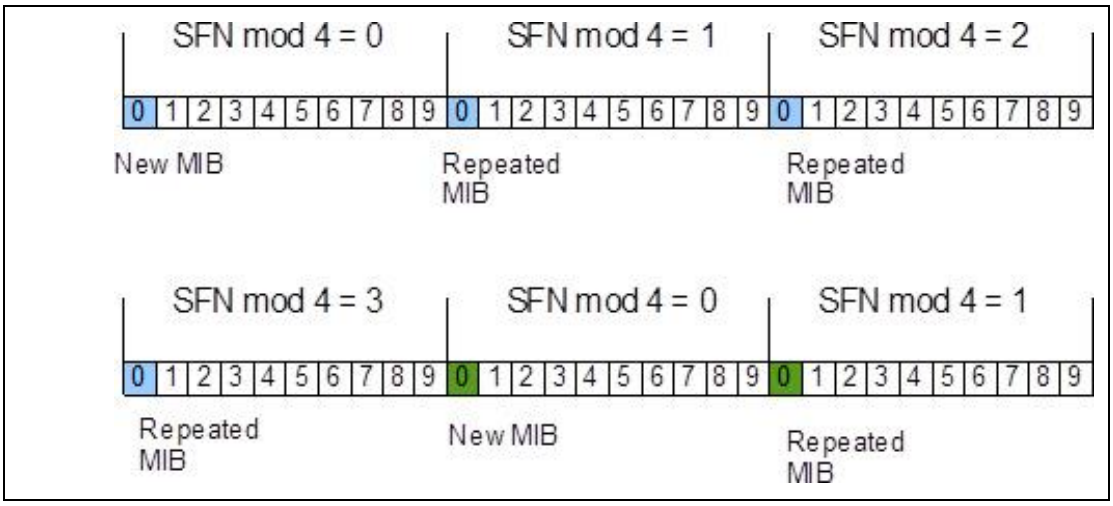
3.2.5 Downlink

This section describes the procedures relating to downlink transmission.

3.2.5.1 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 [36.331], (refer to Table 2) and shown in Figure 21. 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 with a different scrambler preset to delineate the SFN boundary.

Figure 21. MIB Scheduling on the BCH Transport Channel



The BCH procedure is shown in Figure 22. 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 40 ms.

If the PHY has been configured for autonomous BCH support (refer to the `phyCfg` information under the PHY initialization in Section 4.2.3) then the PHY takes care of generating the correct MIB information based on the PHY_INIT parameters set by the L2/L3. If there is a need to change one of the parameters in the MIB a Reconfiguration procedure must be initiated by the L2/L3.

If the automatic PHY processing of PBCH was not set by the L2/L3, then it must provide the following information:

- In `TXSTART.request` the `TXVECTOR` must have a `DLChannelDescriptor` with a `channelType` of PBCH and the `numberOfChannels` entry must consider a PBCH channel being present.
- A `TXSDU.request` with a MAC PDU containing the MIB and with a `channelType` of PBCH in the TXSDU header.

If the PHY does not receive a BCH PDU in subframe SF=0, where radio frame (SFN mod 4) = 0 (refer to Figure 22), it will generate a message based on the information that is available from the PHY_INIT for the MIB as a way of avoiding a loss of BCH Sync at the UEs connected to the eNb. Figure 23 provides an illustration of the procedure when Automatic PBCH generation is enabled.

Figure 22. BCH Procedure with Automatic PBCH Generation Disabled

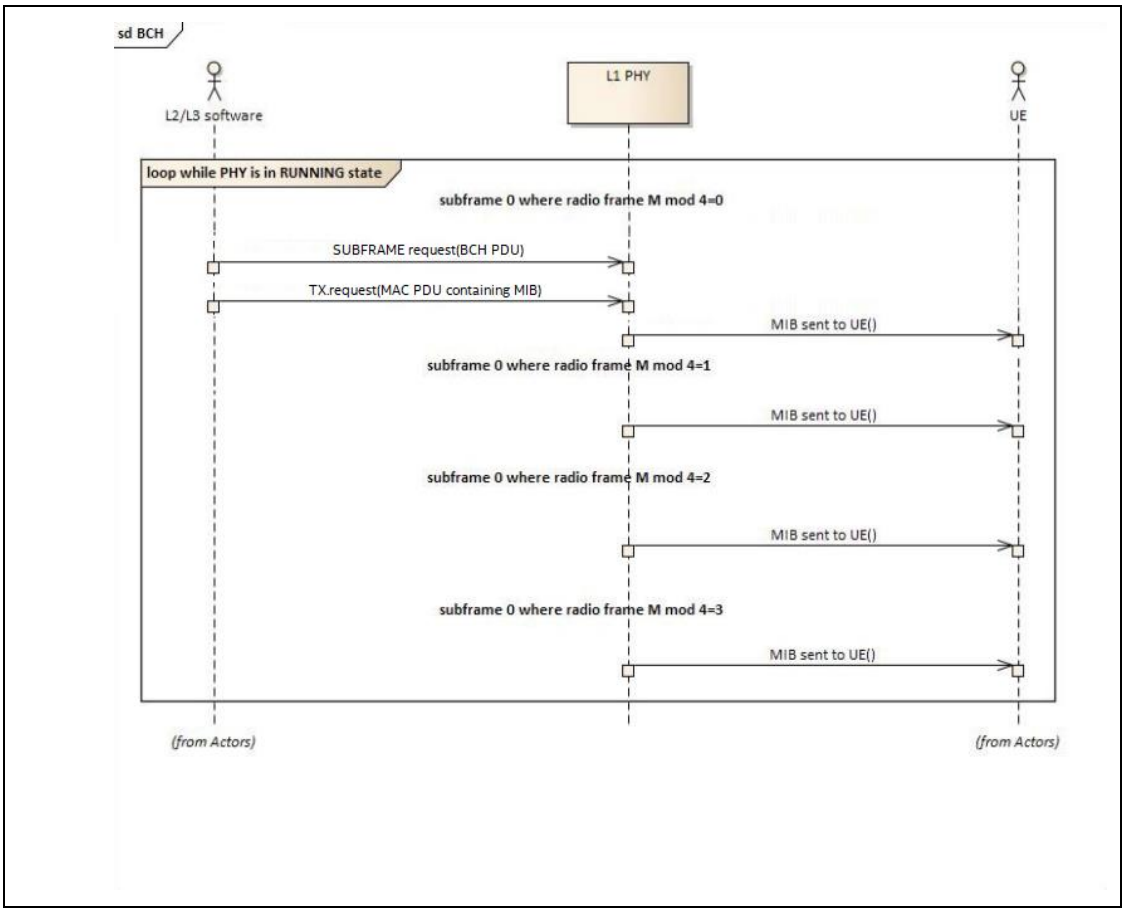
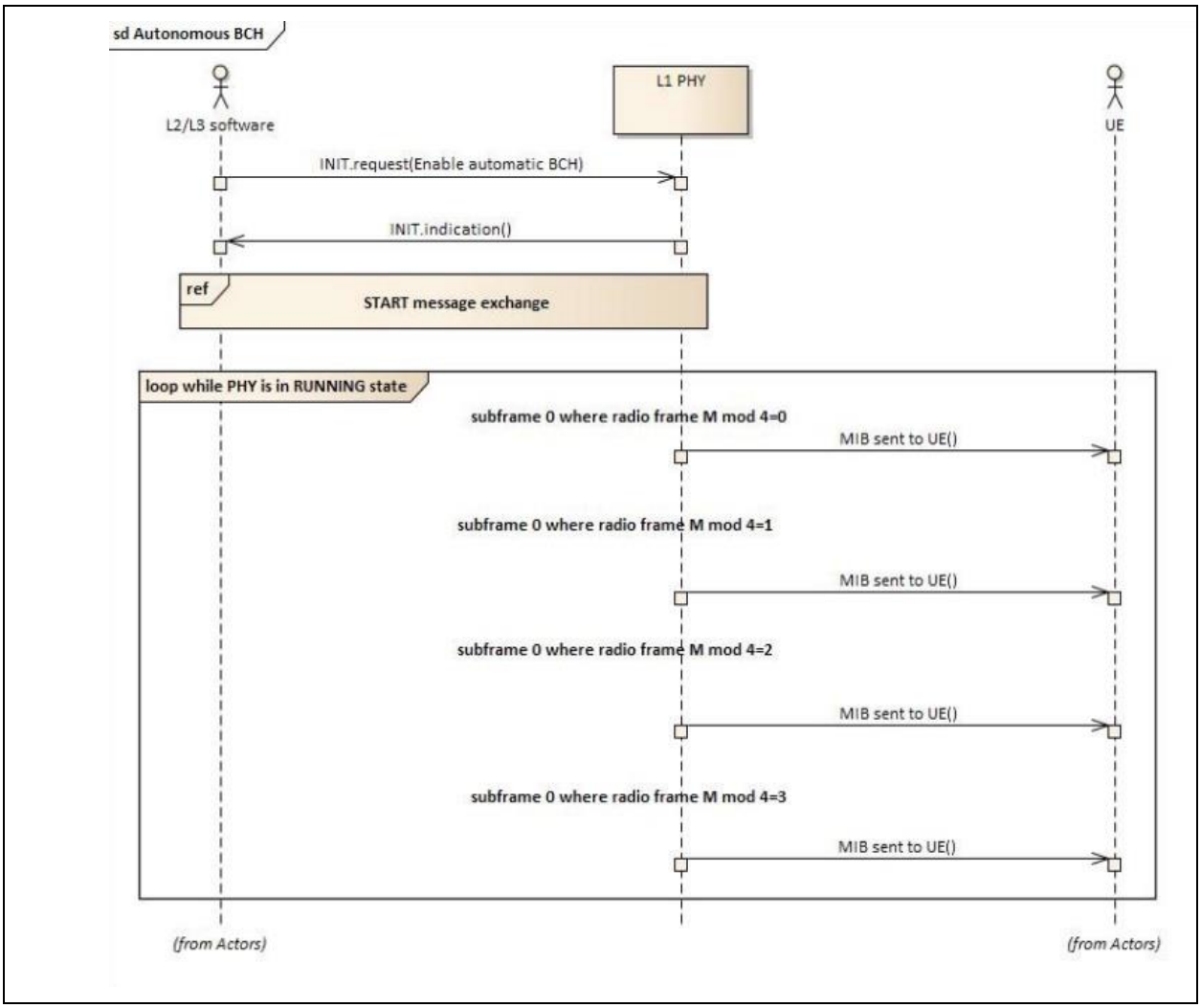


Figure 23. BCH Procedure with Automatic PBCH Enabled



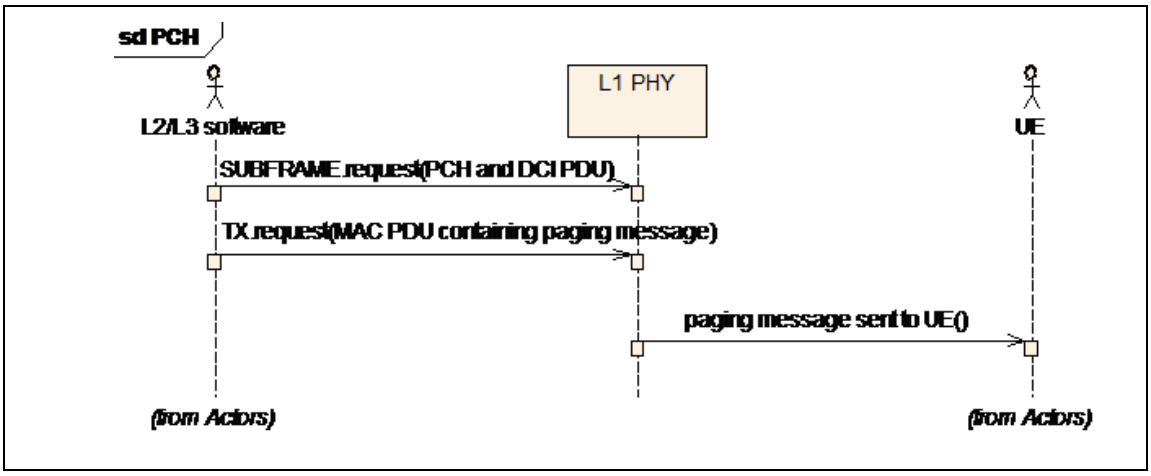
3.2.5.2 PCH

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 [36.304]. The L2/L3 software is responsible for calculating the correct paging occasion for a UE. The PHY is only responsible for transmitting PCH PDUs as payload to a PDSCH when instructed by the `PHY_TXSTART.request` message.

The PCH procedure is shown in Figure 24. To transmit a PCH PDU the L2/L3 software must provide the following information:

- In `PHY_TXSTART.request` a DLSCH descriptor and a PDCCH descriptor are included.
- `PHY_TXSDU.request` for both PDSCH and PDCCH are issued by the L2/L3 including the MAC PDU containing the paging message and the DCI message with the PDSCH assignments associated with the PCH.

Figure 24. PCH Procedure



3.2.5.3 DL-SCH

The DL-SCH transport channel is used to send data from the eNB to a single UE. 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 25. To transmit a DL-SCH PDU the L2/L3 software must provide the following information:

- In PHY_TXSTART.request a PDSCH and DCI descriptors are included.
- Then a PHY_TXSDU.request must be provided for the data associated with the PDSCH, with one PHY_TXSDU.request per code word being used in the PDSCH. The TXSDU header reflects the code word identifier associated with the PDU and in the case that two code words are used the header section cwId contains the appropriate code word identifier.
- The DCI descriptors contain control information regarding the DL frame transmission.
- Later a PHY_RXSTART.request is included describing the configuration for the HARQ information to be received from the UE. The timing of this subframe was explained in Section 0
-
- API Message Timing. There are multiple configuration scenarios that can be used to indicate reception of the HARQ response on the uplink:

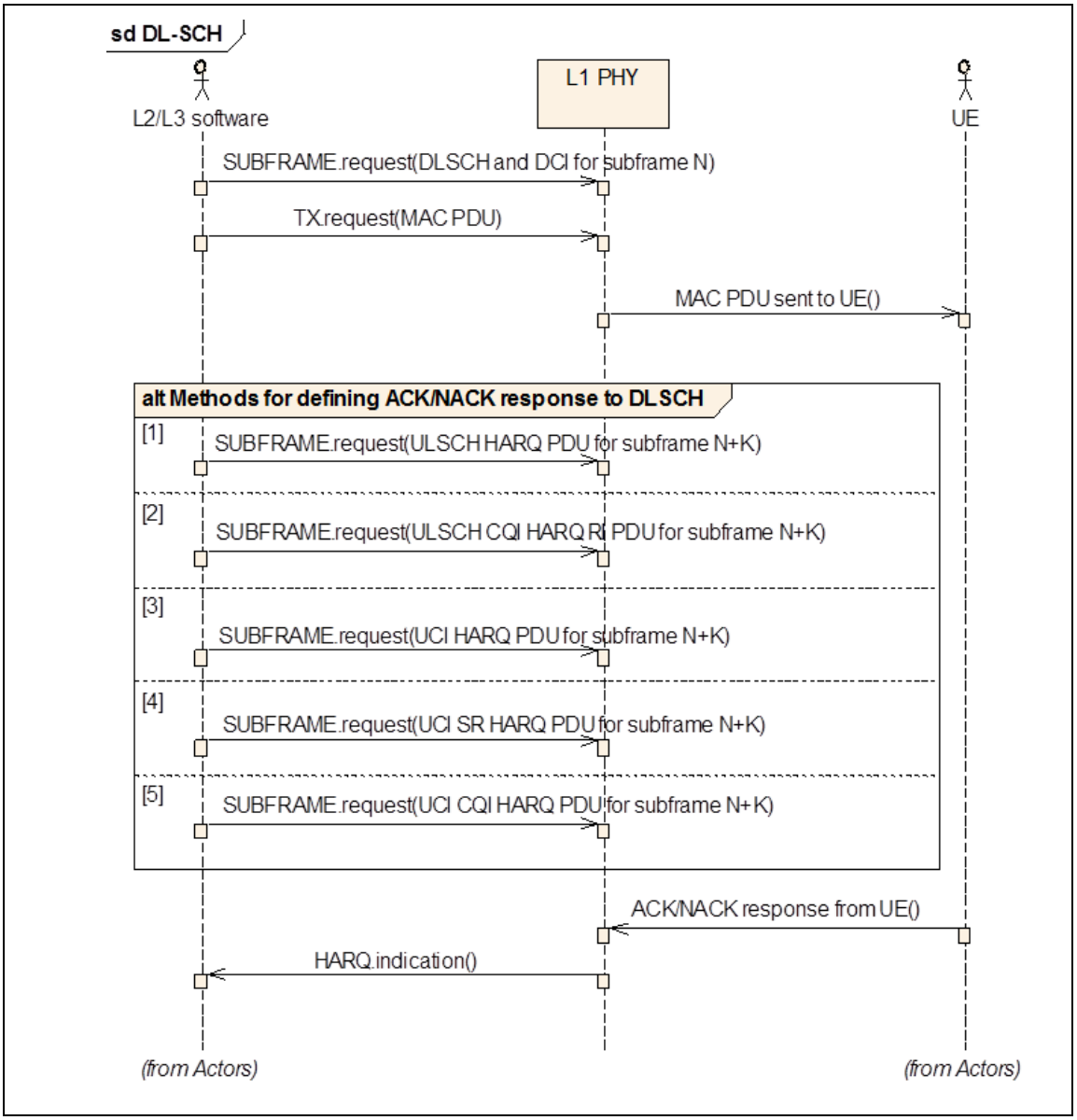
– ULSCH_HARQ – when the UE is scheduled to transmit data and the ACK/NACK response
– ULSCH_CQI_HARQ_RI – when the UE is scheduled to transmit data, a CQI report and the ACK/NACK response
– UCI_HARQ – when the UE is just scheduled to transmit the ACK/NACK response
– ULSCH_UCI_HARQ - when the UE is scheduled to transmit data and ACK/NACK response using simultaneous transmission of PUCCH and PUSCH. This is added for 3GPP release 10.
– UCI_SR_HARQ – when the UE is scheduled to transmit a scheduling request and the ACK/NACK response
– UCI_CQI_HARQ – when the UE is scheduled to transmit a CQI report and the ACK/NACK response
– ULSCH_ CSI_UCI_HARQ – is used if the UE is scheduled to transmit data, a CSI report and ACK/NACK response using simultaneous transmission of PUCCH and PUSCH. This is added for 3GPP release 10.
– The L2/L3 must properly configure the RXVECTOR structures to match the expected format for the HARQ response.
  - The PHY returns the ACK/NACK response information in two different forms depending on whether the information came multiplexed within the PUSCH (in which case a RXSTATUS.indication

message is used) or whether it came via the PUCCH (in which case a `RXSDU.indication` message is issued and the HARQ information is contained in the payload).

Section 0,

API Message Timing explained the strict timing requirements in LTE. For FDD to maintain maximum downlink throughput the L2/L3 software must analyze the `HARQ.indication` message within 1 ms. This allows the L2/L3 software to generate the next `TXSTART.request` message, using the same HARQ process, in subframe N+8.

Figure 25. DL SCH Procedure



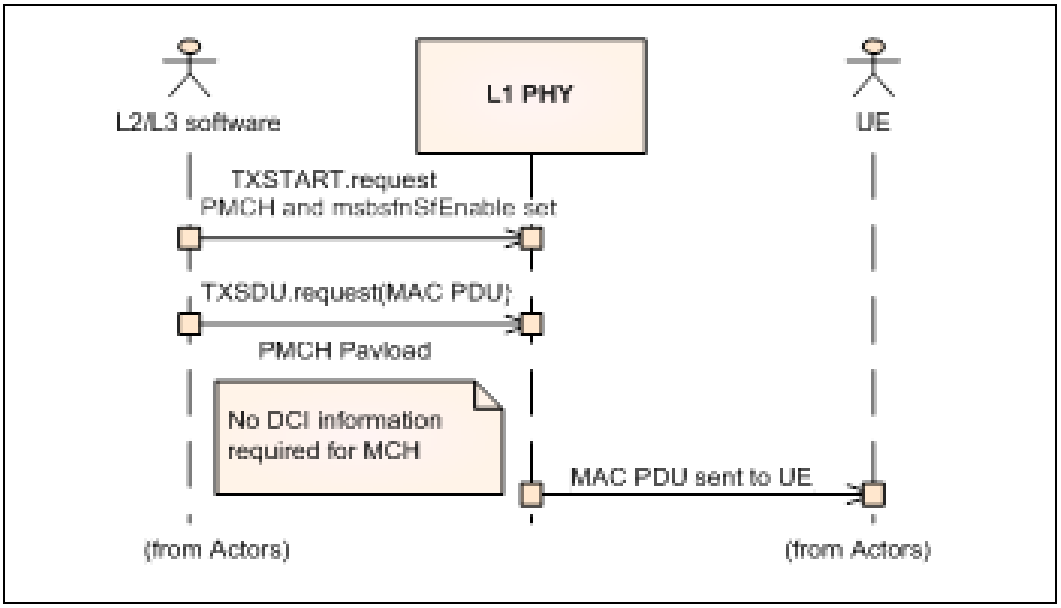
3.2.5.4 MCH

The MCH transport channel is used to send data to multiple UEs simultaneously. The MCH transport channel is transmitted in a subframe configured with a Multimedia Broadcast Single Frequency Network (MBSFN) region. HARQ is not used.

The MCH procedure is shown in Figure 26. To transmit a MCH TXSDU the L2/L3 software must provide the following information:

- In `PHY_TXSTART.request` the `mbsfnSfEnable` field in the `DLSubframeParam` Structure should be set. An entry in the `DlChannelDescriptor` Structure for a PMCH must be provided with the descriptor that specifies the parameters for the channel.
- A `PHY_TXSDU.request` must be provided for the data associated with the MCH. The header should contain in the channelType field `PMCH` and only one codeword is needed for this channel type.

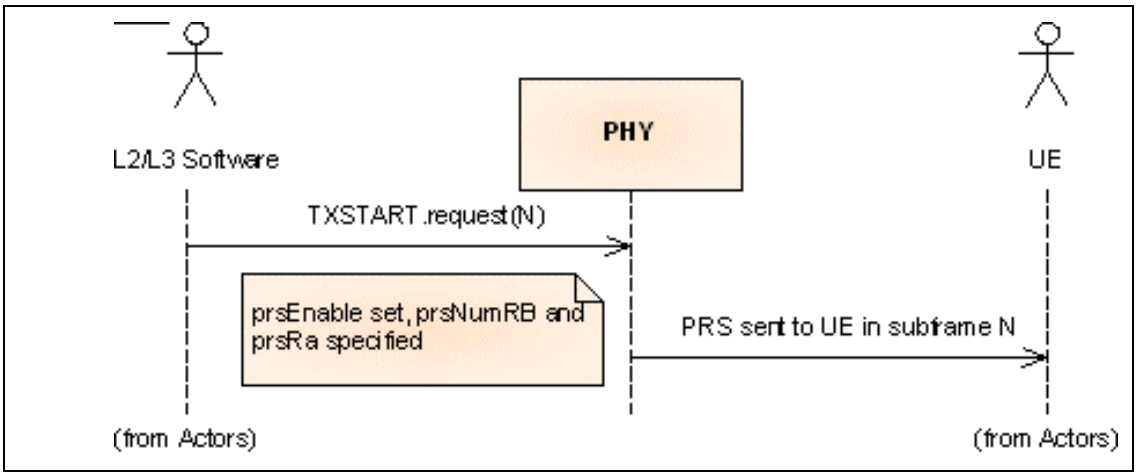
Figure 26. MCH Procedure



3.2.5.5 Positioning Reference Signals

Positioning reference signal (PRS) transmission was introduced in 3GPP release 9 to assist LTE positioning at a UE. PRS is transmitted over a PRS bandwidth and periodically in PRS occasions, where each PRS occasion comprises of a sequence of consecutive DL subframes. The L1 API supports the storage of the PRS configuration information in the MAC. The MAC L1 API is used to instruct the PHY when it should allocate a PRS transmission. The procedure for PRS transmission is shown in Figure 27.

Figure 27. Positioning Reference Signals Generation Process

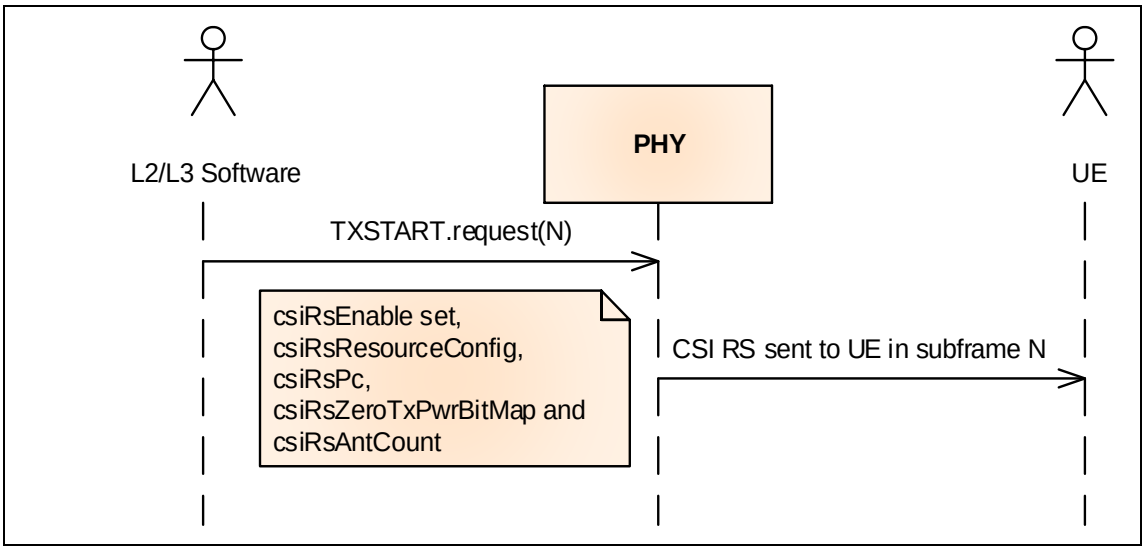


3.2.5.6 Channel State Indication Reference Signals

CSI reference signals were added in 3GPP release 10. `CSI-RSs` are meant to aid terminals to acquire channel state information (CSI). In frequency domain `CSI-RSs` are transmitted over the complete cell bandwidth and in the time domain CSI-RSs are transmitted over a time period varying from 5-80 ms.

The MAC stores the CSI-RS configuration information and needs to pass five parameters via the `TXSTART.request` with the following information for the PHY to generate the CSI-RS signals `csiRsEnable`, `csiRsResourceConfig`, `csiRsPc`, `csiRsZeroTxPwrBitMap` and `csiRsAntCount` as illustrated in Figure 28.

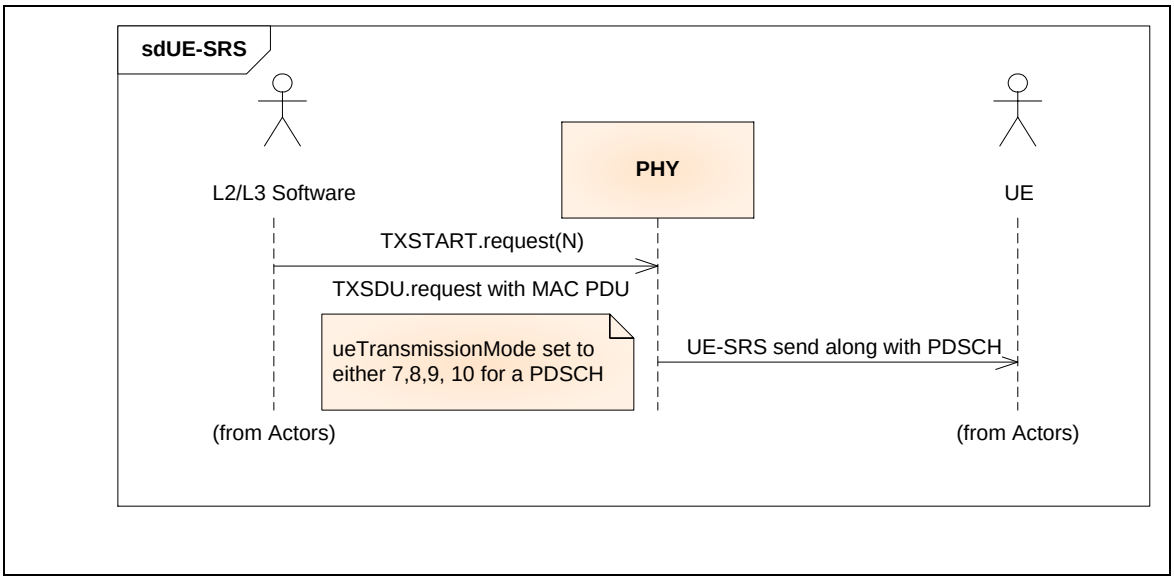
Figure 28. CSI RS R10 Procedure



3.2.5.7 UE Specific Reference Signals (UE-SRS)

UE Specific Reference Signals are generated when the Transmission Mode is one of the following 7, 8, 9, or 10 which are used for beamforming. The MAC L1 API is used to instruct the PHY based on a specific DL-SCH transmission mode when the `UE-SRS` should be generated. The procedure for UE-SRS transmission is shown in Figure 29.

Figure 29. UE-SRS Procedure



3.2.6 Uplink

This section describes the procedures relating to uplink reception.

3.2.6.1 RACH

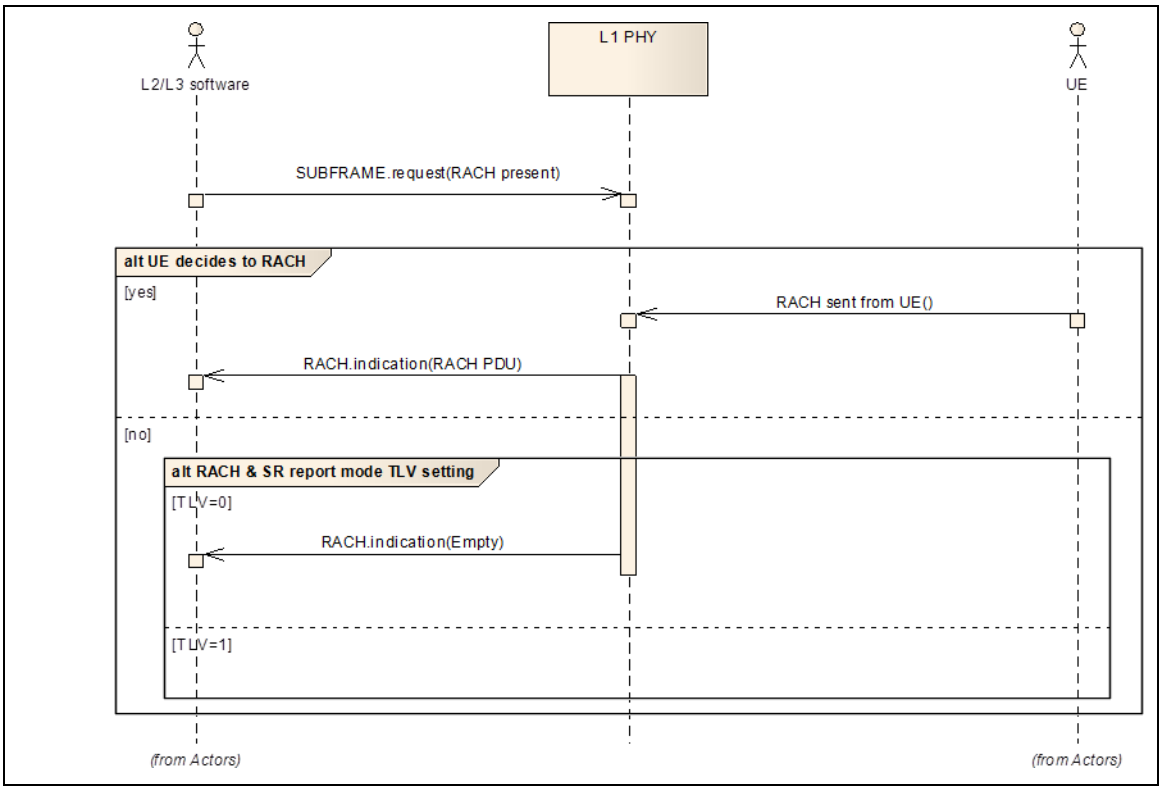
The `RACH` transport channel is used by the UE to send data to the eNB when it has no scheduled resources. Also, the L2/L3 software can indicate to the UE that it should initiate a RACH procedure. In the scope of the L1 API, the RACH procedure begins when the PHY receives a `RXSTART.request` message indicating the presence of a RACH.

The RACH procedure is shown in Figure 30. ..... RACH Procedureto configure a RACH procedure the L2/L3 software must provide the following information:

- In `RXSTART.request` the prachEnable bit must be set
- If a UE decides to initiate a RACH procedure, and a preamble is detected by the PHY:

– The PHY will issue a `RXSTATUS.prach.indication` message that contains all detected preambles. For no preambles detected the message states 0 events detected. There is also a configuration option in the PHY_INIT parameters in the `phycfg` field not to issue a RACH `RXSTATUS.indication` when no preamble has been detected in the current subframe.

Figure 30. RACH Procedure



3.2.6.2 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 31. To transmit an UL-SCH PDU the L2/L3 software must provide the following information:

- Within the `TXHIADCIUL.request` for subframe N, a DCI structure is defined and the DCI PDU included in a `TXDCIULSDU.request`. The DCI Format 0 PDU contains control information regarding the UL frame transmission being scheduled.
- In `RXSTART.request` for subframe N+K1, a ULSCH is defined. The timing of this subframe and value of K1 are explained in Section 0
-
- API Message Timing for FDD. For the case of TDD, the timing is also dependent on the DL/UL subframe configuration.

There are four possible ways to configure the receiver with the RXSTART.request depending on the expected scheduled data to be transmitted by the UE on the given subframe and listed below when the `simultaneousPucchPusch` option in the `PUCCHDEDCTL` (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) is disabled:

- ULSCH scheduled– is used if the UE is scheduled to only transmit data, in this case one `ULCHDESC` entry must be defined (Refer to Section 4.6.14 ULChannelDescriptorStruct) for this UE in the RXSTART.request.

- ULSCH_CQI_RI – is used if the UE is scheduled to transmit data and a CQI report. For this particular scenario it requires a ULCHDESC entry (Refer to Section 4.6.14 ULChannelDescriptorStruct) and also a PUSCHDED (Refer to Section 4.6.28 PuschConfigDedicatedStruct) entry that must contain the expected CQI and RI data input sizes.
- ULSCH_HARQ – is used if the UE is scheduled to transmit data and an ACK/NACK response. One ULCHDESC (Refer to Section 4.6.14 ULChannelDescriptorStruct) for this UE must be defined in the RXSTART.request and one PUSCHDED (Refer to Section 4.6.28 PuschConfigDedicatedStruct) entry for the same must have the nACK field set with the expected number of HARQ bits.
- ULSCH_CQI_HARQ_RI – is used if the UE is scheduled to transmit data, a CQI report and an ACK/NACK response. For this case also one ULCHDESC (Refer to Section 4.6.14 ULChannelDescriptorStruct) entry must be defined and one PUSCHDED (Refer to Section 4.6.28 PuschConfigDedicatedStruct) information entry with the expected number of CQI, HARQ and RI bits filled in the RXSTART.request.

When the simultaneousPucchPusch option is enabled (3GPP release 10 UEs) then the following additional scenarios are possible:

- ULSCH_UCI_HARQ – is used if the UE is scheduled to transmit data and ACK/NACK response using PUSCH and PUCCH respectively. In this scenario the L2/L3 software needs to provide a ULCHDESC entry (Refer to Section 4.6.14 ULChannelDescriptorStruct) for PUSCH and a PUCCHDEDCTL entry (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) for the PUCCH.
- ULSCH_UCI_CSI – is used if the UE is scheduled to transmit data and a CSI report using PUSCH and PUCCH respectively. In this scenario the L2/L3 software needs to provide a ULCHDESC entry (Refer to Section 4.6.14 ULChannelDescriptorStruct) for PUSCH and a PUCCHDEDCTL entry (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) for the PUCCH.
- ULSCH_CSI_UCI_HARQ – is used if the UE is scheduled to transmit data multiplexed with CSI over the PUSCH and UCI and HARQ info over the PUCCH. In this scenario the L2/L3 software needs to provide a ULCHDESC entry (Refer to Section 4.6.14 ULChannelDescriptorStruct) and a PUSCHDED (Refer to Section 4.6.28 PuschConfigDedicatedStruct) entry that must contain the expected CQI and RI data input sizes for PUSCH and a PUCCHDEDCTL entry (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) for the PUCCH.

The PHY returns the received uplink data in the RXSDU.indication message. The RXSDU.indication message provides the CRC information in a status field.

For the multiplexed control and data information on PUSCH, the PHY returns the control information on a RXSTATUS.indication with HARQ information provided first.

The ACK/NACK response must be submitted to the PHY using a TXHIADCIUL.request and a TXHISDU.request message in subframe N+K1+K2. The timing of this subframe, and value of K2, was explained in Section 0

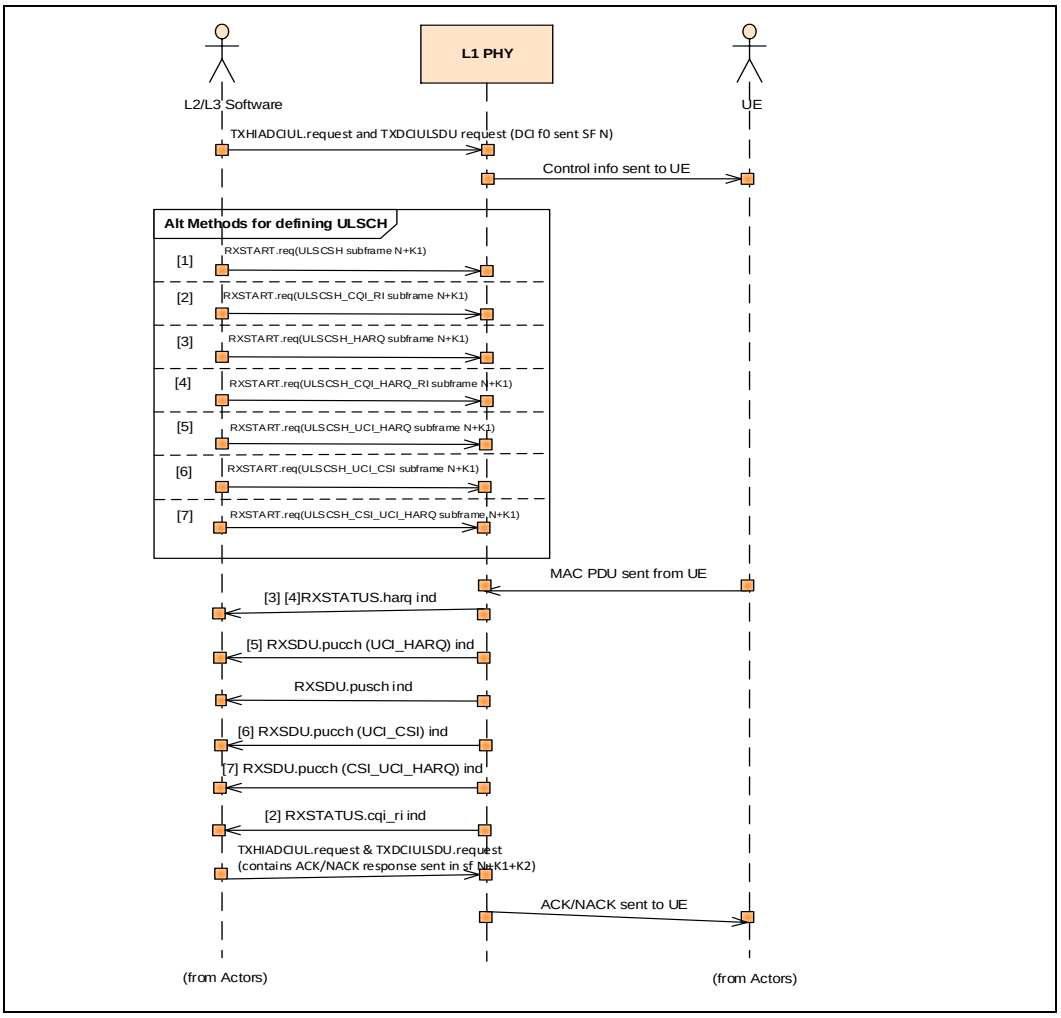
API Message Timing for FDD. For TDD the timing also depends on the DL/UL subframe configuration.

Section 0

API Message Timing explained the strict timing requirements in LTE. For FDD the L2/L3 software must analyze the CRC information status and generate the TXHIADCIUL.request message together with the TXHISDU.request within the timing constraints explained in the implementation section.

For 3GPP release 10 and later with UL MIMO support the PHY delivers two RXSDU.indications one per each codeword and the L1/L2 software is required to fill additional information in the RXVECTOR as stated in Section 4.6.16 ULSubChannelDescriptorR10.

Figure 31. ULSCH Procedure



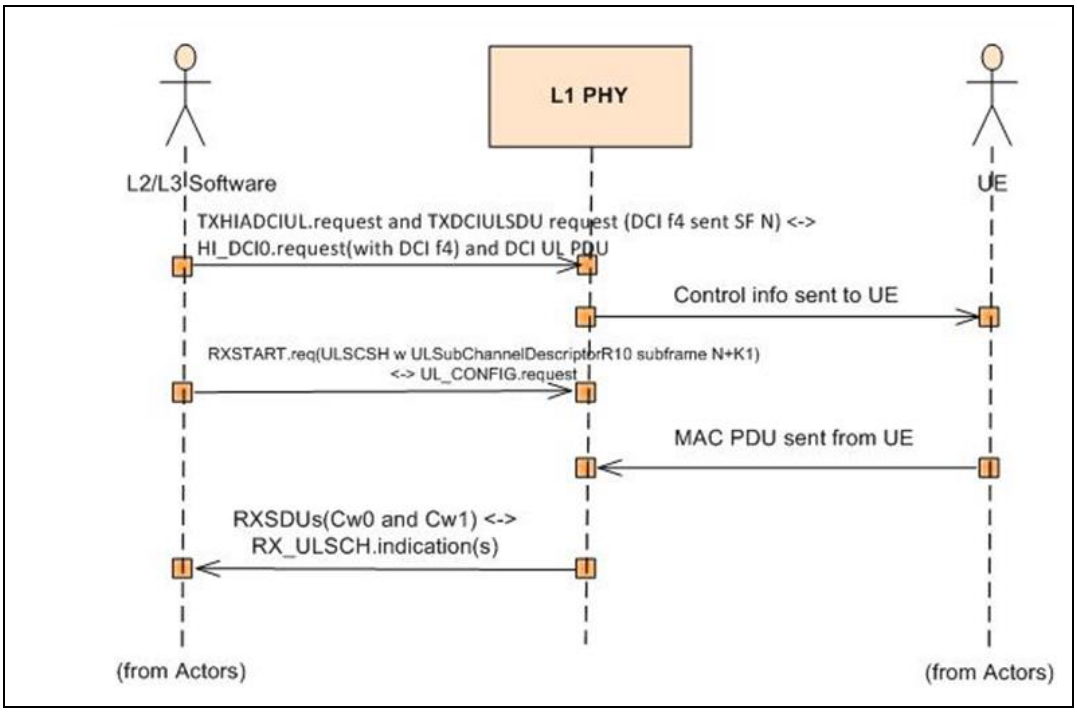
With DCI Format 4, the UL SCH channel is used to receive two data transport blocks from a UE, this requires a single DCI PDU, but two ULSCH PDUs. The procedure is shown in Figure 32. To initiate a transmission of two data transport blocks, the L2/L3 software must provide the following information:

- A `TXHIDCIUL.request` with a DCI Format 4 payload in a `TXDCIULDSU.request`.
- In `RXSTART.request` the ULCHANDESC for this UE needs the `ulRelease10` bit set and the information for both `ULChannelDesc` and `ULChannelDescR10` needs to be filled by the L3/L3 software.

The remaining behavior is identical to single-layer transmission.

Figure 32 shows the two-layer ULSCH reception process.

Figure 32. Two-Layer UL-SCH



3.2.6.3 SRS

The sounding reference signal (SRS) is used by L2/L3 software to determine the quality of the uplink channel.

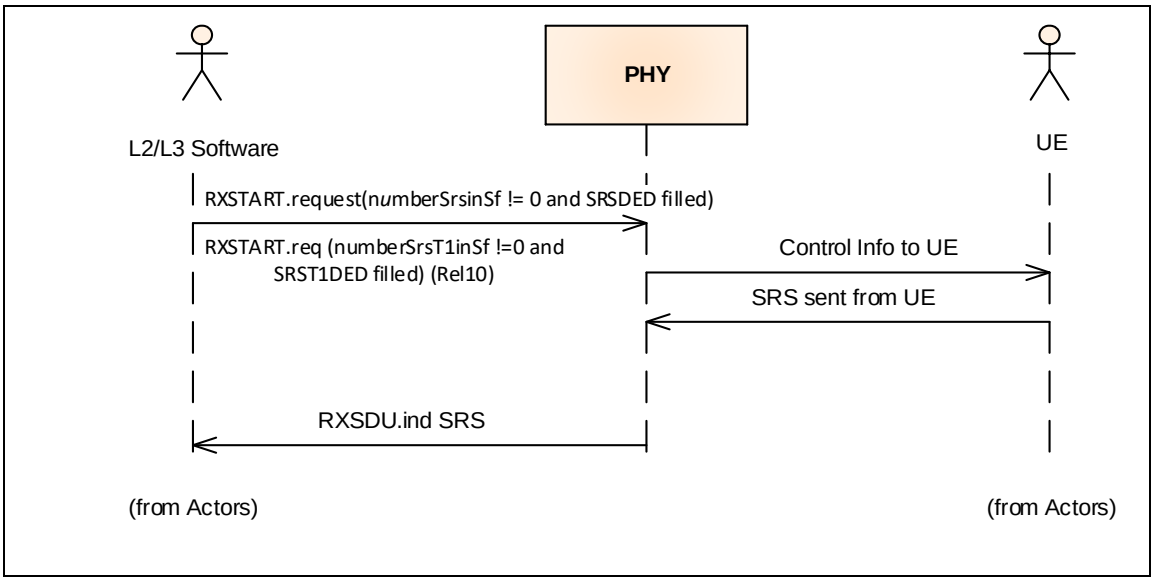
In 3GPP release 10, aperiodic SRS support was added where the L2/L3 software via the PDCCH informs the UE to send the SRS signal.

Figure 33 shows the SRS procedure. To schedule an SRS, the L2/L3 software must provide the following information:

- In `RXSTART.request` the `numberSrsinSf` must be set to the number of sounding UEs expected in the subframe and the corresponding `SRSDED` (refer to Section 4.6.26 SoundingRefDedicatedCtrlStruct) structures filled for each expected SRS.
- In the case of 3GPP release 10 and aperiodic SRS in `RXSTART.request` the `SRST1DED` (refer to Section 4.6.27 SoundingRefT1DedicatedCtrlStruct) structures need to be filled for each expected SRS in the subframe and the `numberSrsT1inSf` must be set to the number of aperiodic sounding UEs expected.

The PHY returns the SRS response to the L2/L3 software in the `RXSTATUS.srs.indication` message.

Figure 33. SRS Procedure



3.2.6.4 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:

- The RRC connection procedure the L2/L3 software will instruct the UE to transmit periodic CQI reports.
- The L2/L3 software can use the PDCCH to instruct the UE to transmit an aperiodic CQI report.

In both cases the PHY is only responsible for receiving CQI reports when instructed by the L2/L3 software.

The CQI reporting procedure is shown in Figure 34. To schedule a CQI report the L2/L3 software must provide the following information:

- For an aperiodic report a PDCCH channel is assigned in the `TXSTART.request` and the DCI format 0 data is included in the `TXSDU.request` payload. This instructs the UE to send a CQI report. For periodic CQI reports no explicit DCI information is sent.
- In the `RXSTART.request`, where the L2/L3 software is expecting a CQI report, the CQI description is included in the corresponding structure. There are eight possible ways that the CQI report could come on the uplink:

– ULSCH_CQI_RI – In this case the UE is scheduled to transmit data and a CQI report on the ULSCH so the `RXSTART.request` must have the `puschDed` (Refer to Section 4.6.28 PuschConfigDedicatedStruct) structure correctly setup by the L2/L3 code.

– ULSCH_CQI_HARQ_RI – In this case the UE is scheduled to transmit data, a CQI report and the ACK/NACK response on the PUSCH channel, so again the `RXSTART.request` must set the `puschDed` (Refer to Section 4.6.28 PuschConfigDedicatedStruct) structure associated with the UE with the CQI, HARQ and RI expected fields information.

– UCI_CQI – In this case the UE is just scheduled to transmit a CQI report on a PUCCH channel. So the `RXSTART.request` must have the `pucchDedCtl` (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure associated with the UE correctly set by the L2/L3 prior to decoding.

– UCI_CQI_SR – In this situation the UE is scheduled to transmit a CQI report on a PUCCH channel and there is also a Schedule Request opportunity. Since the L2/L3 does not know before time which one will

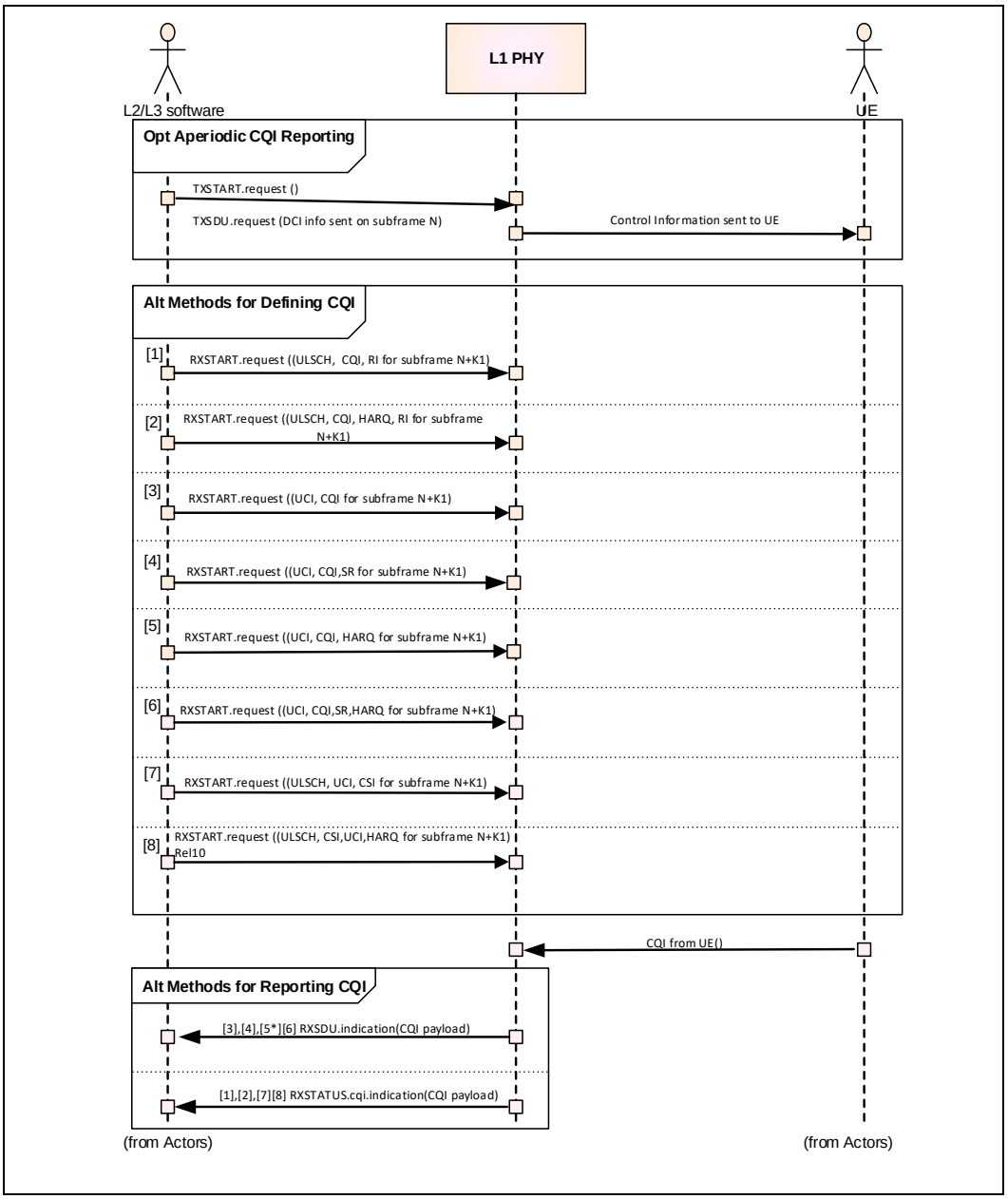
take place then it must set entries for both of these cases in the pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure associated with the UE.

- UCI_CQI_HARQ – In this case the UE is scheduled to transmit a CQI report and the ACK/NACK response on a PUCCH channel and the RXSTART request must have the pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure associated with the UE correctly set by the L2/L3 code.
- UCI_CQI_SR_HARQ – For this scenario the UE is scheduled to transmit a CQI report, an ACK/NACK response and it also has a Schedule Request opportunity over the PUCCH channel. The L2/L3 software must set the pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure associated with the UE to handle the possible cases.

Note: The next two scenarios apply only to 3GPP Release 10 when the simultaneous transmission on PUSCH and PUCCH has been enabled.

- ULSCH_UCI_CSI – In this case the UE is scheduled to send data over the PUSCH and a CSI report over the PUCCH. The L2/L3 software must set then both the pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure and the puschDed Refer to Section 4.6.28 PuschConfigDedicatedStruct) structure in the TXSTART request.
- ULSCH_CSI_UCI_HARQ – In this scenario the UE is scheduled to transmit data over PUSCH and a CSI report and an ACK/NACK response over the PUCCH. The L2/L3 software must set then both the pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) structure and the puschDed (Refer to Section 4.6.28 PuschConfigDedicatedStruct) structure in the TXSTART request.
  - Depending on whether the CQI is multiplexed with Uplink Data or not there are two ways for the PHY to provide the CQI report back to the L2/L3.
- If the report comes in the PUCCH then a RXSDU.indication is used with the payload being the CQI report. (For cases where CQI plus HARQ information are expected, refer to the appropriate UCI format type for the details on how CQI and HARQ are presented in the payload).
- If the report comes multiplexed with data on the PUSCH channel then the PHY issues a RXSTATUS.cqi.indication to the L2/L3 code.

Figure 34. CQI Procedure



3.2.6.5 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 PHY is only responsible for receiving SR information when instructed by the L2/L3 software.

Figure 35 shows the SR procedure. To schedule an SR the L2/L3 software must provide the following information:

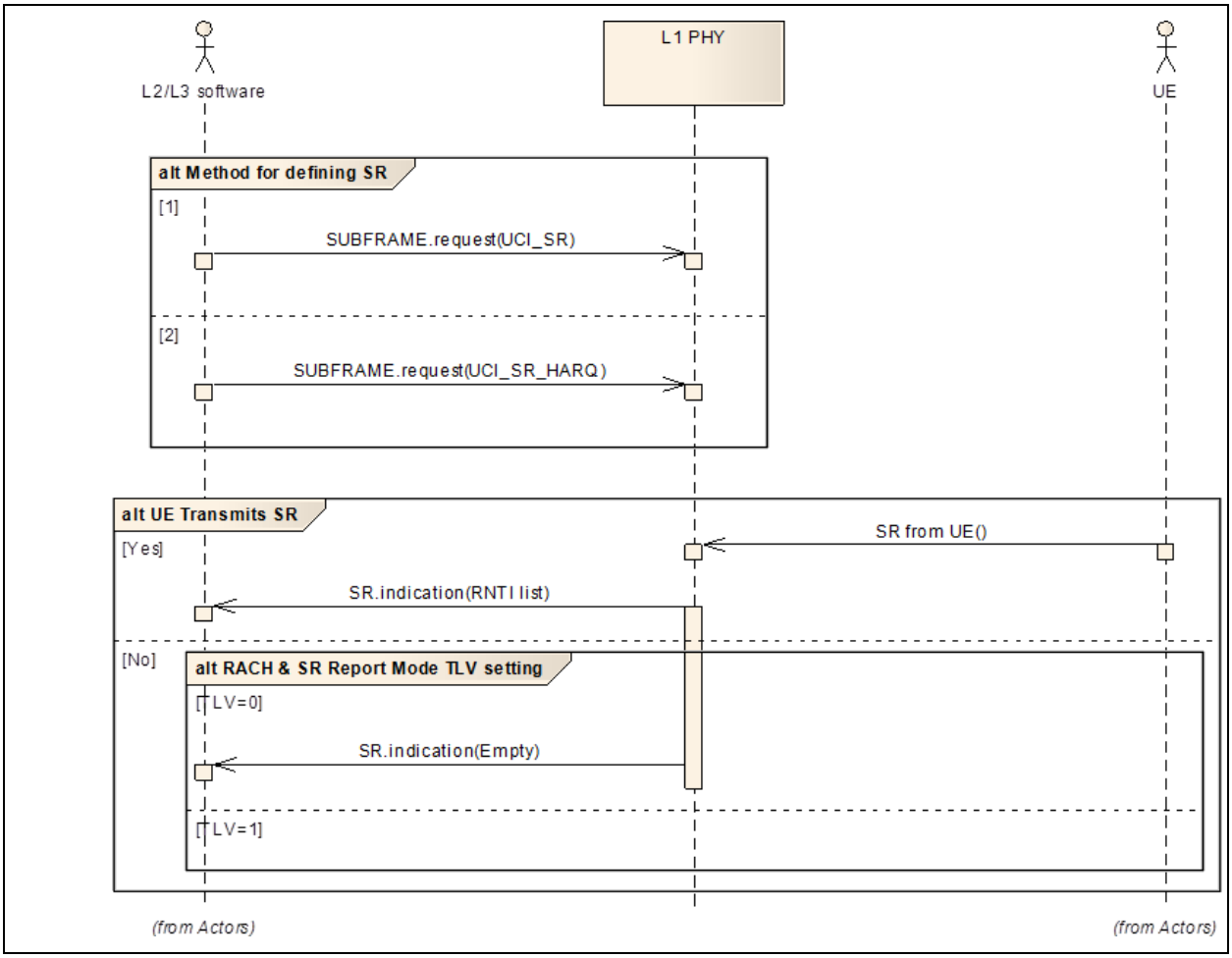
- In the RXSTART.request an entry for a PUCCH with its pucchDedCtl (Refer to Section 4.6.25 PucchDedicatedCtrlInfoStruct) entry for formatType set to correct format (see below) with the SR detection parameters correctly configured must be present.
- There are two possible ways that the SR can be sent by the UE on the uplink:

– If the UE is only scheduled to transmit a SR on PUCCH the formatType must be set to format1.

– If the UE is scheduled to transmit a SR and the ACK/NACK response, then depending on whether DL MIMO is enabled or not the formatType must be set to format1a (DL MIMO disabled) or format1b (DL MIMO enabled).

- If the UE transmits a positive SR and is received successfully, the PHY returns the SR to the L2/L3 software in a RXSDU.indication message with the header section of the message having the srdetected bit set and the pucchType reflecting the format set in the RXSTART.request for this PUCCH channel.
- If no positive SR is received, then a RXSDU.indication with the srdetected bit clear and the pucchType reflecting the format set in the RXSTART.request is issued. Also there is an option in PHY_INIT not to issue any RXSDU.indication when no SR has been received when bit 7 is set in the phyCfg field.

Figure 35. SR Procedure



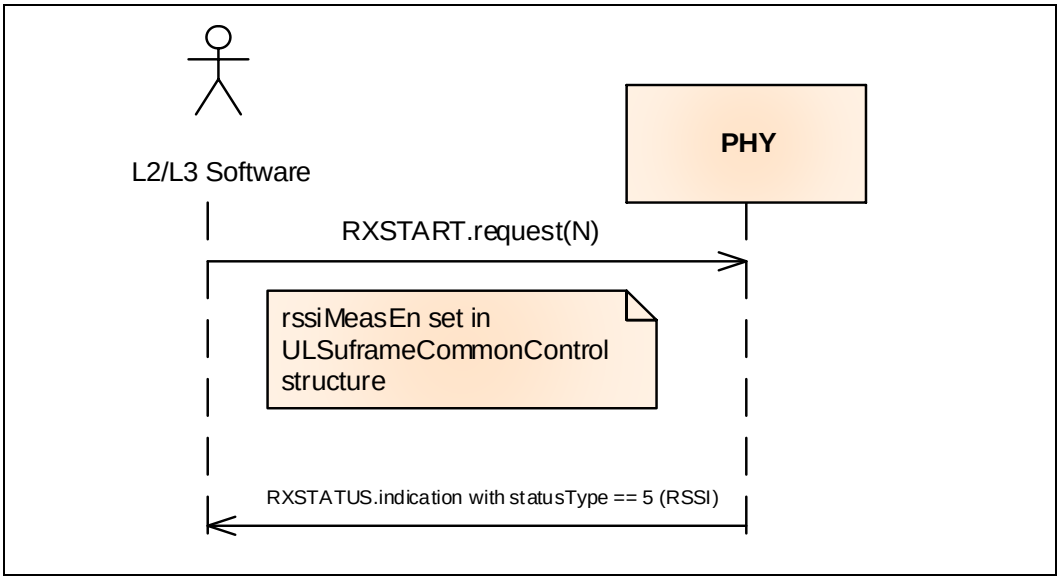
3.2.6.6 RSSI Measurements

There are two types of RSSI measurements available:

- A wideband RSSI
- A channel bandwidth based RSSI

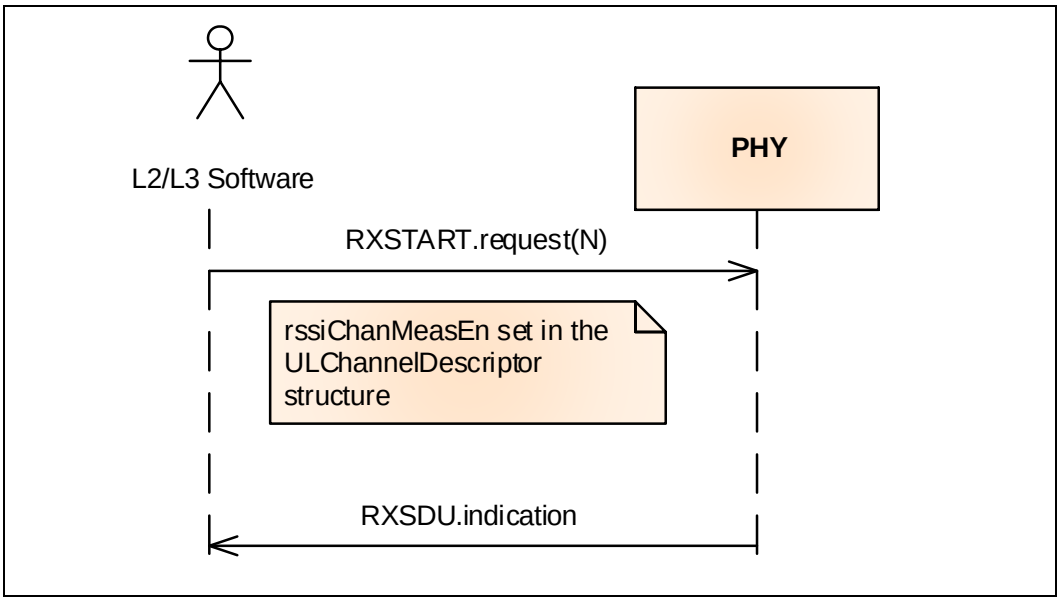
For the wideband RSSI, the L2/L3 software needs to set the `rssiMeasEn` bit in the `ULSubframeCommonControl` structure of the `RXVECTOR` prior to the subframe where the measurement is to be taken. The PHY reports the results using the `RXSTATUS.indication` as illustrated in Figure 36.

Figure 36. Wideband RSSI Procedure



For the channel band based RSSI the L2/L3 software needs to set the `rssiChanMeasEn` bit in the `ULChanDescriptor` structure for the subframe where the measurement is to be performed. The PHY reports the RSSI value using a `RXSDU.indication` with the RSSI value contained in the header as indicated in Section 4.3.15 PHY_RXSDU Indication. Figure 37 illustrates the procedure.

Figure 37. Channel Based RSSI Procedure

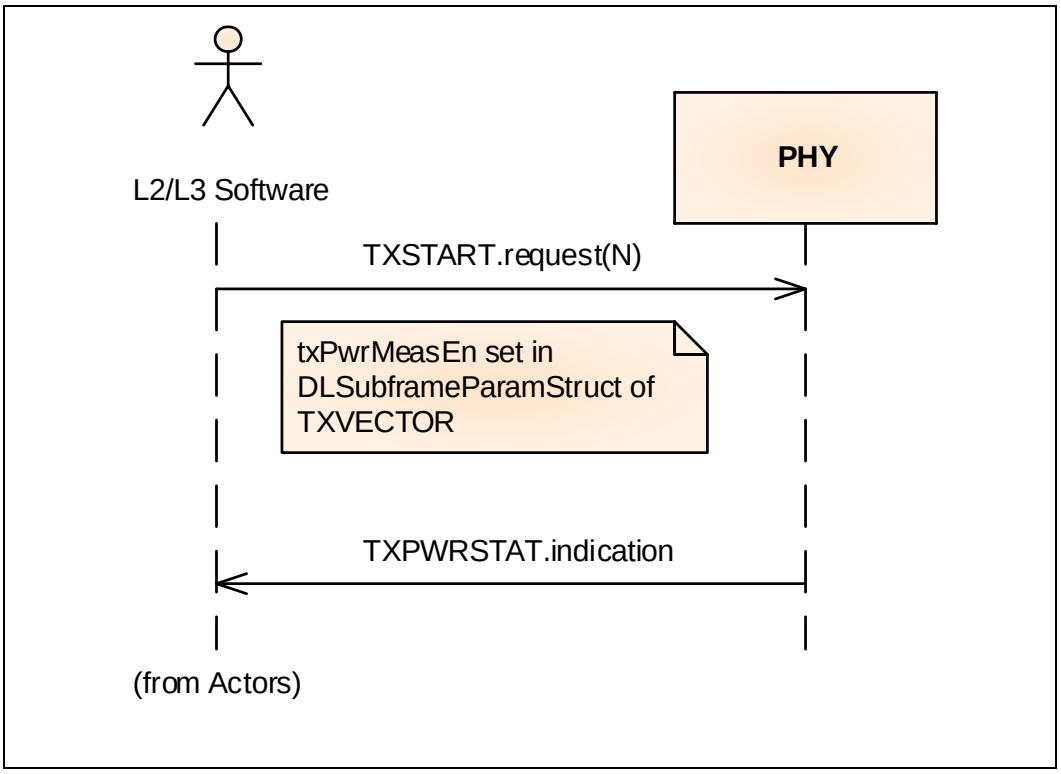


3.2.6.7 Transmit Power Measurement

To perform a Transmit Power Measurement the L2/L3 software needs to set the txPwrMeasEn bit in the DLSubframeParamStruct in the TXVECTOR and issue a TXSTART.request to the PHY. The PHY processes the request and issues at the completion of the subframe a TXPWRSTAT.indication with the measurement results. Refer to Section 4.3.18 PHY_TXPWRSTAT_IND.

The Transmit Power Measurement procedure is illustrated in Figure 38.

Figure 38. Transmit Power Measurement Procedure

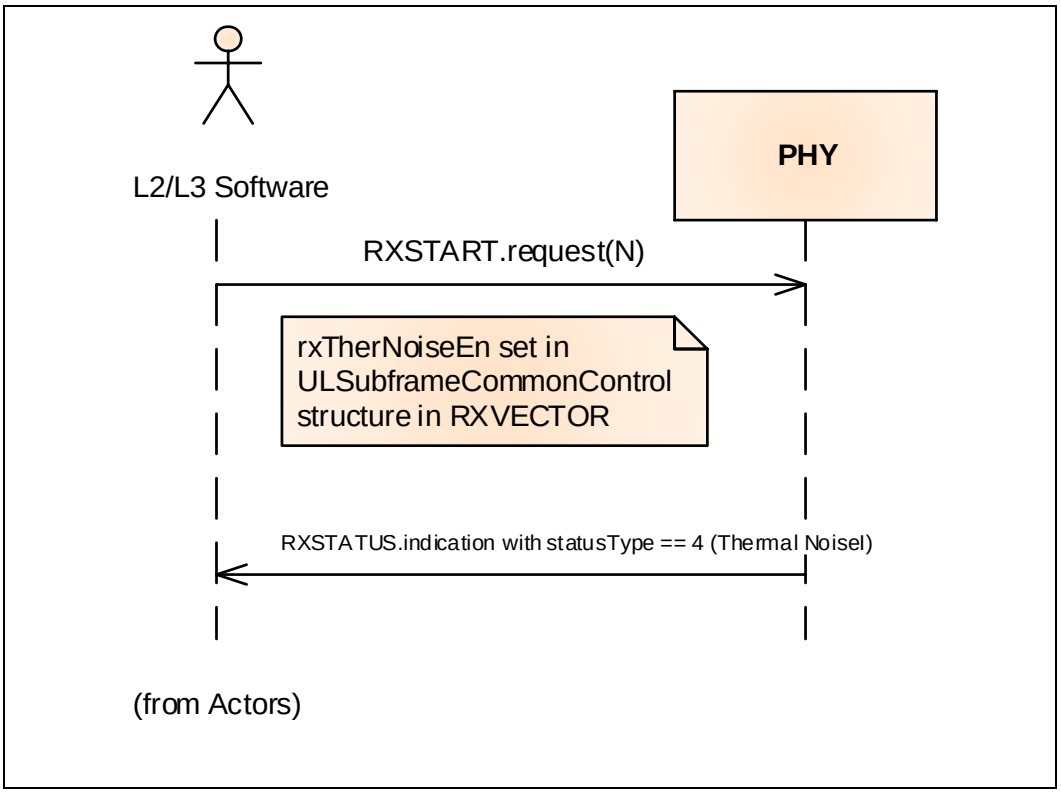


3.2.6.8 Thermal Power Measurement

To perform a thermal power measurement the L2/L3 software needs to set the rxTherNoiseEn bit in the ULSubframeCommonControl structure in the RXVECTOR and then it should issue the RXSTART.request command to the PHY. The PHY processes the request and issues a RXSTATUS.indication with StatusType 4 for Thermal Noise Power. Refer to Section Appendix 11.1.1.9 PHY_RXSTATUS for Thermal Noise Power for the details of the indication message.

The Thermal Power Measurement procedure is illustrated in Figure 39.

Figure 39. Thermal Noise Power Measurement

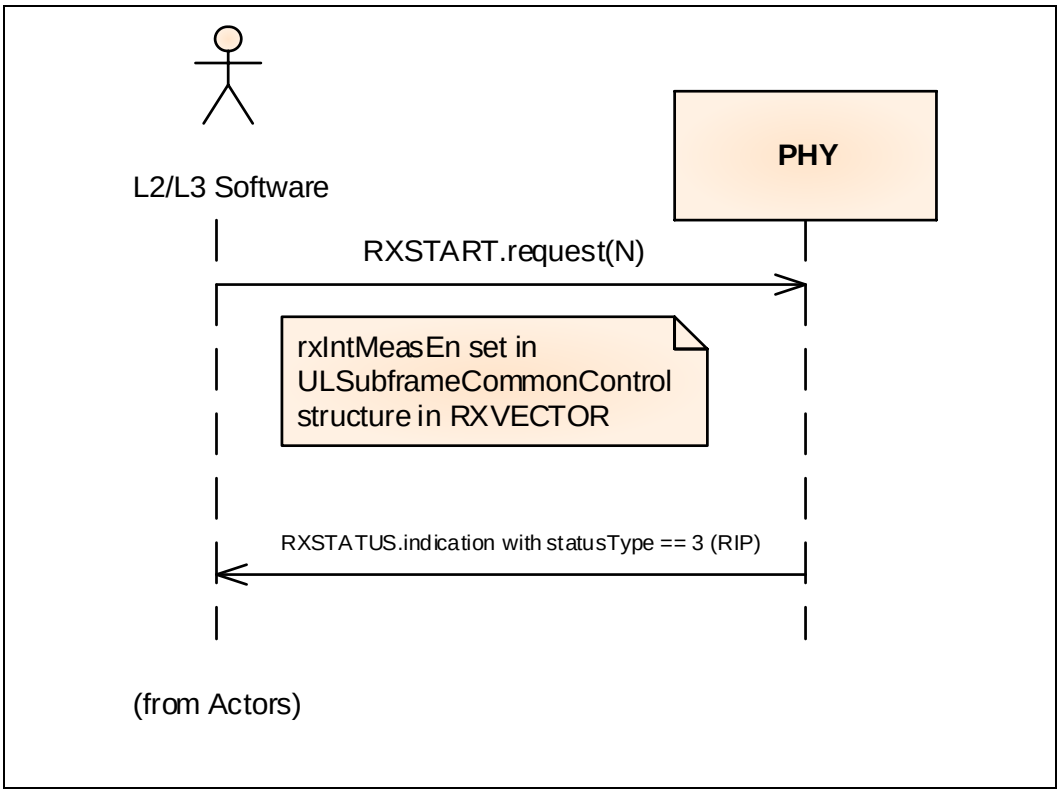


3.2.6.9 Receive Interference Power Measurement

To perform a Receive Interference Power Measurement the L2/L3 software needs to set the `rxIntMeasEn` bit in the `ULSubframeCommonControl` structure of the `RXVECTOR` and issue a `RXSTART.request` command to the PHY. After completing the processing of the subframe the PHY issues an `RXSTATUS.indication` with `statusType` set to 3 which means Receive Interference Power. Refer to Section Appendix 11.1.1.8 PHY_RXSTATUS for Received Interference Power for the details of the `RXSTATUS.indication` for this procedure.

The Receive Interference Power Measurement Procedure is illustrated in Figure 40.

Figure 40. Receive Interference Power Procedure



3.2.7 Cross-Carrier Scheduling

In 3GPP Release 10, carrier aggregation was introduced, where a UE can be connected to a primary cell and optionally secondary cells. This inclusion of multiple cell permits the concept of cross-carrier scheduling where a UE is signaled scheduling information on one cell but transmits or receives the data using a different cell. The cross-carrier scheduling procedure for both downlink and uplink is described below.

3.2.7.1 Downlink

In the downlink one cell may transmit scheduling information on a DCI with no corresponding `TXSDU.request` for a PDSCH. Similarly, a `TXSDU.request` may be present with no corresponding DCI.

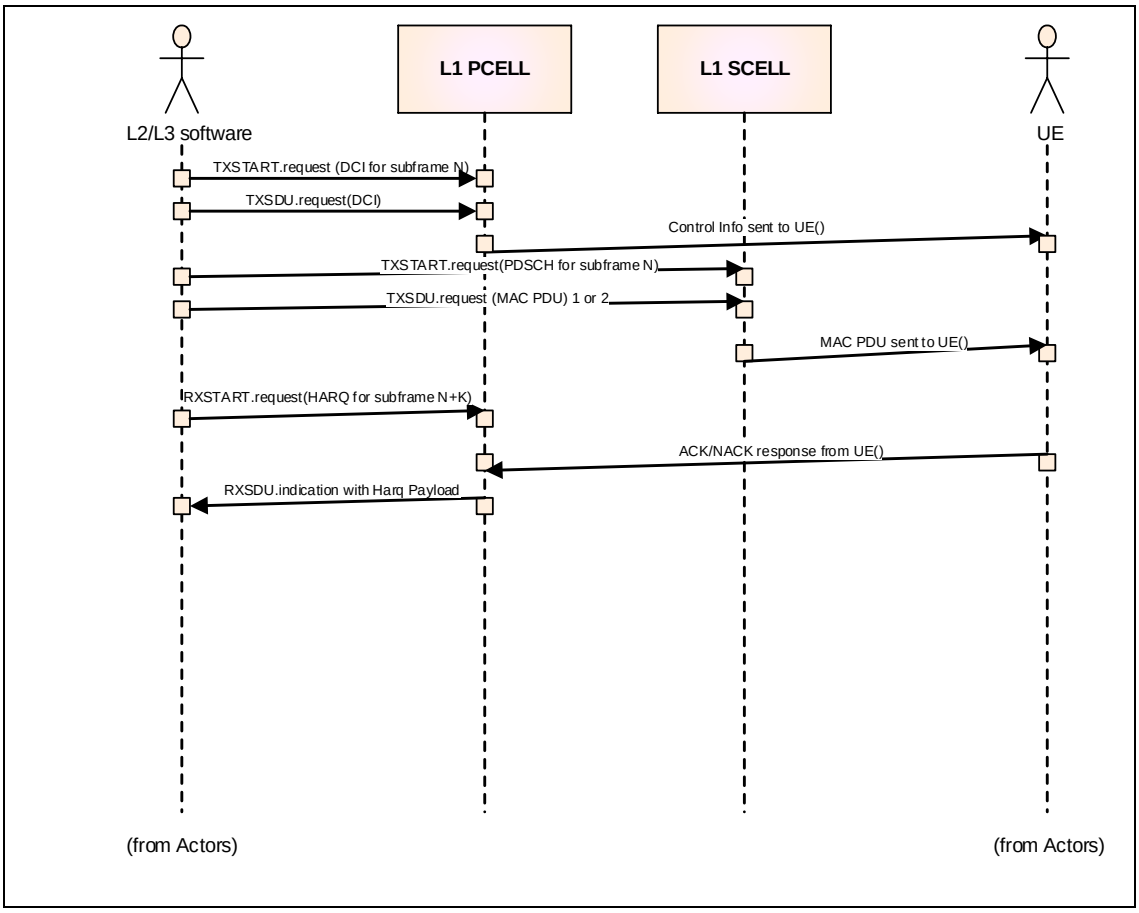
An example procedure for scheduling on the primary cell and data transmission on a secondary cell is given in Figure 41. The L2/L3 software must provide the following information:

- In the cell used for scheduling a `TXSTART.request` with `TXSDU.request` with PDCCH ChannelType and with the DCI payload is included. The DCI Payload contains control regarding the DL frame transmission.
- In the cell used for data transmission a `TXSTART.request` with a `TXSDU.request` for a PDSCH channelType and with the data payload is included.
- The cell used for the uplink HARQ signaling is dependent on whether the UE also has a scheduled ULSCH transmission. In this example no ULSCH transmission is scheduled resulting in the primary cell being used.
- A PUCCH HARQ Descriptor is included in a later `RXSTART.request`. The PHY will return the ACK/NACK response information in either a `RXSDU.indication` message if the HARQ information

came via the PUCCH or via a RXSTATUS.indication if the HARQ information came multiplexed in the PUSCH.

- The timing of this message is dependent on whether the system has been configured for FDD or TDD. In the case of TDD, it is also dependent on the DL/UL subframe configuration.

Figure 41. Cross-Carrier Scheduling DL



3.2.7.2 Uplink

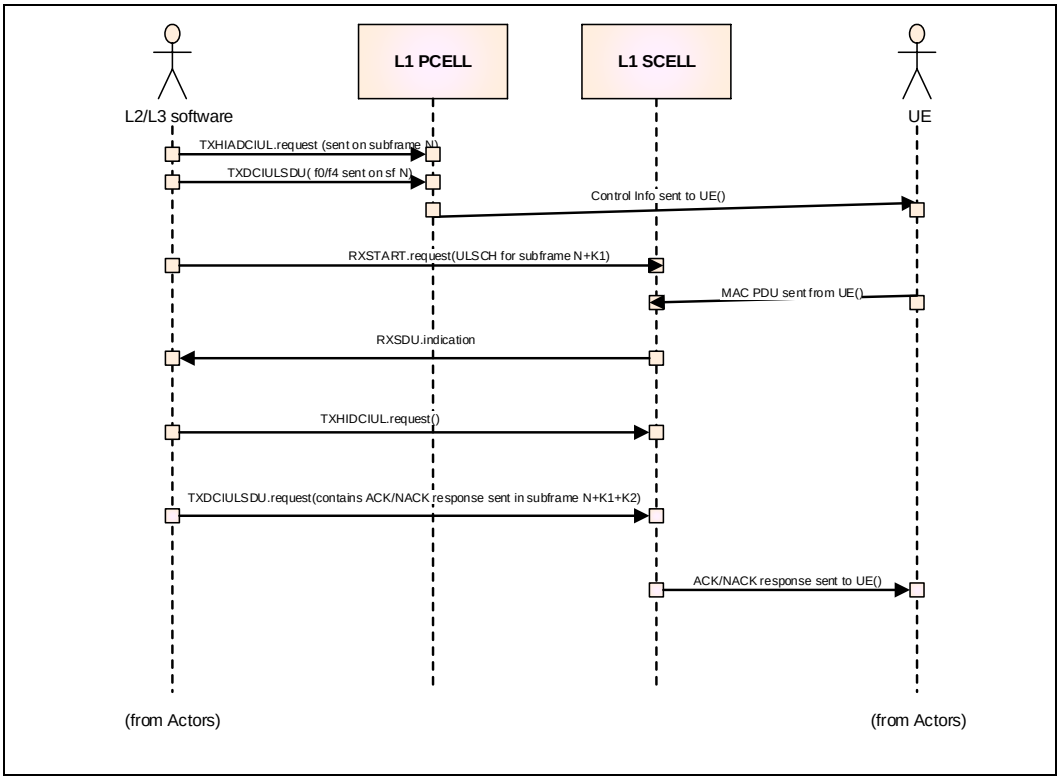
For uplink cross carrier scheduling one cell may transmit scheduling information on a DCI with no corresponding ULSCH information. Similarly, the ULSCH configuration information may be present with no DCI being sent on that cell.

An example procedure for scheduling on the primary cell and data transmission on a secondary cell is given in Figure 42; the L2/L3 software must provide the following information:

- In the cell used for scheduling a TXDCIULSDU.request for subframe N with the DCI control information defined in the HIADCIULMSGDESC (Refer to Section 4.3.13 PHY_TXDCIULSDU Request) is given by the L2/L3 software to the PHY, and it contains either DCI Format 0 or Format 4 information regarding the UL frame transmission being scheduled.
- In the cell used for transmission, a RXSTART.request for subframe N+K1 with an ULSCH PDU is included. The timing of this message depends on whether the system is configured for FDD or TDD. In the case of 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. The different ULSCH PDU options are explained in Section 3.2.6.2.2 UL-SCH.

- The PHY will return the received uplink data in the RXSDU.indication message. The RXSDU.indication message contains the CRC information.
- In the cell used for scheduling the ACK/NACK response must be submitted to the PHY using a TXHIADCIUL.request and a TXHISDU.request message in subframe N+K1+K2. The timing of this message is explained in Section 3.2.6.2.2 UL-SCH.

Figure 42. Cross Carrier Scheduling UL



3.2.8 Error Sequences

The PHY provides error indications to the L2/L3 software in the status field of the commands. Refer to Section 4.1.2 Error Field Coding for the details about the list of errors reported.

If robust vector operation is enabled, then if the TXSTART and /or RXSTART messages are late the PHY will use self-made default messages. These default messages have no channels defined, so only reference signals are generated in any downlink subframe. For subframe 0 and 5 the synchronization signals are generated and in subframe 0 the PBCH with the MIB payload is generated. When the late TXSTART and RXSTART are received, the PHY purges the TXSDUs messages, and on the receiver it generates RXSDUs messages with status Timeout on RXSTART.req and RXSTATUS messages, so the L1/L2 is aware that the data did not make it to the UE. This is similar to the UE experiencing interference on the air-interface and LTE error recovery mechanisms, such as HARQ and ARQ enable the L2/L3 software to recover. The operation of the robust mode depends on whether the system is configured for FDD or TDD. In the case of TDD it also depends on the DL/UL subframe configuration in use.

4.0 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.

The L2/L3 and the PHY communicate by exchanging PHY SAP primitives. These primitives allow the L2/L3 to pass control information and data to be transmitted to the PHY and allow the PHY to pass status information and received data to the L2/L3. The control information that is passed with these primitives consists mainly of the subframe descriptor structures.

The protocol is message-based, in which the maximum size of a message is specified as part of the configuration of the interface.

The general message format of the L1 API is shown in Table 5 where it can be seen that each L1 API message consists of a header followed by a message body.

The generic header consists of 4-bytes, including a message type ID and message body length. The current list of message types is given in Table 6. The L1 API messages follow a standard naming convention where:

- All `.request` messages are sent from the L2/L3 software to the PHY.
- All `.confirmation` 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 confirmation 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 4.1.1, Message Type Field Coding.
- All messages should be aligned to 32-bit boundaries and if necessary padding should be added to comply with this requirement.

A full description of each message body is given in the remainder of Section 4.2 L1 Configuration Messages.

4.1 Generic Message Header, Format, and Coding

All PHY SAP primitives are translated into messages by prefixing the message header, as defined in Table 5. Each message header contains an appropriate message type field, uniquely identifying the given PHY SAP primitive.

Table 5. LTE PHY SAP Message Header

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type
0	15:0	16	Message specific

The PHY entity (Instance) ID is supplied by the Control Plane during the initialization to both L2 and PHY layers. This identifier allows the L2 to communicate with multiple PHY instances. Normally the lower 16 bits of the Header contain the length of the message in bytes.

The message type values are known to both PHY and L2.

The convention used for all the primitives is based on a Little Endian format for all command fields. Only the actual TB payloads use a Big Endian or Network convention per 3GPP 36 Series standards. This API only supports Little Endian format at this time.

4.1.1 Message Type Field Coding

Table 6 specifies the Message Types associated with the different primitives.

Table 6. Message Type Description

Value	Description	Section with Message Format
0	Reserved	
1	PHY_TXSTART.request	Section 4.3.1 PHY_TXSTART Request
2	PHY_TXSTART.confirmation†	Section 4.3.2 PHY_TXSTART Confirmation
3	PHY_TXSTART.indication‡	Section 4.3.3 PHY_TXSTART Indication
4	PHY_TXSDU.request	Section 4.3.10 PHY_TXSDU Request
5	Reserved	
6	Reserved	
7	PHY_RXSTART.request	Section 4.3.5 PHY_RXSTART Request
8	PHY_RXSTART.confirmation†	Section 4.3.6 PHY_RXSTART Confirmation
9	PHY_RXSTART.indication‡	
10	PHY_RXSDU.indication	Section 4.3.15 PHY_RXSDU Indication
11	Reserved	
12	PHY_INIT.request	Section 4.2.1 PHY_INIT Request

Value	Description	Section with Message Format
13	PHY_INIT.indication	Section 4.2.2 PHY_INIT Indication
14	PHY_RXSTATUS.indication	Section 4.3.17 PHY_RXSTATUS Indication
15	PHY_RECONFIG.request	Section 4.2.9 PHY_RECONFIG Request
16	PHY_RECONFIG.confirmation	Section 4.2.10 PHY_RECONFIG Confirmation
17	PHY_START.request	Section 4.2.4 PHY_START Request
18	PHY_START.confirmation	Section 4.2.5 PHY_START Confirmation
19	PHY_STOP.request	Section 4.2.6 PHY_STOP Request
20	PHY_STOP.confirmation	Section 4.2.7 PHY_STOP Confirmation
21	PHY_STOP.indication	Section 4.2.8 PHY_STOP Indication
22	PHY_TXHIADCIUL.request	Section 4.3.8 PHY_TXHIADCIUL Request
23	PHY_TXHISDU.request	Section 4.3.12 PHY_TXHISDU Request
24	PHY_TXDCIULSDU.request	Section 4.3.13 PHY_TXDCIULSDU Request
25	PHY_ERROR.indication	Section 4.5.1 Error Sequences
26	PHY_INTRA_TTI.indication	Section 4.3.4 PHY_INTRA_TTI Indication
27	PHY_TXPWRSTAT.indication	Section 4.3.18 PHY_TXPWRSTAT_IND
28	PHY_BATCH_MSG.request	Section 4.4.1 PHY_BATCH_MSG Request
29	PHY_BATCH_MSG.indication	Section 4.4.2 PHY_BATCH_MSG Indication

Value	Description	Section with Message Format
30	PHY_ZBC_BLOCK_REQ	
31	PHY_ZBC_BLOCK_IND	
32-33	Reserved	
34	PHY_QUERY.request	Section 4.2.11 PHY_QUERY Request
35	PHY_QUERY.indication	Section 4.2.12 PHY_QUERY Indication
36	PHY_TXDCICATMSDU.request	Section 4.3.19 PHY_TXDCIULCATMSDU Request
37	PHY_SHUTDOWN.request	Section 4.2.13 PHY_SHUTDOWN Request
38	PHY_SHUTDOWN.confirmation	Section 4.2.14 PHY_SHUTDOWN Confirmation
37-255	Reserved	

Used only in non – real time early integration

Subframe delimiter indication for TXSTART and RXSTART indications are issued as PHY_TXSTART.indication for FDD and issued in TDD for UL only subframes.

4.1.2 Error Field Coding

In this document, the following error codes (ERRORCODE) are defined, to provide the cause of errors that occur in communications between the L2 and the PHY. The error code is generic and can be applied to applicable PHY SAP primitives. The ERRORCODE is an 8-bit value with the currently defined values listed in Table 7.

Table 7. Error Codes

Value	Description
0	Success
1	Primitive is not supported (for requests)
2	FEC code type is not supported
3	Overrun
4	Underrun
5	Transport Media Error

Value	Description
6	TX data size does not match TXVECTOR
7	Invalid TX VECTOR format
8	Invalid RX VECTOR format
9	RX CRC error
10	Transmit Timeout with missing TXSDU
11	Receiver did not match RXSDUs in RXVECTOR
12	Timeout on TXSTART.req
13	Timeout on RXSTART.req
14	No Custom Extensions Handler registered
15	L2/L3 rejection of CXR_STATE change command
16-255	Reserved

4.2 L1 Configuration Messages

The following sections define the available L1 Configuration Messages.

4.2.1 PHY_INIT Request

Table 8. PHY_INIT Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 12
0	15-0	16	PHY_INIT payload length (INITPARM structure)
...	...	...	Parameters

This primitive is issued by L2 to request the PHY to perform an initialization process based on the parameters provided in the INITPARMS portion of this primitive.

When carrier aggregation is used in either licensed or unlicensed spectrum mode the L2/L3 must issue this command to each PHY instance associated with a component carrier.

4.2.2 PHY_INIT Indication

Table 9. PHY_INIT Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 13

0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Reserved

This primitive is issued by the PHY to inform the L2 that it has completed the initialization process requested by the `PHY_INIT.request`.

4.2.3 PHY_INIT Parameters

The structure of `INITPARM` used in `PHY_INIT.request` message is given below.

Note: The carrier information for Downlink and Uplink is not specified as part of the PHY specification since the baseband processing does not require this information. Refer to [36.104] in Table 2 for the available frequency bands that need to be specified for the RF section.

To enable the optimized processing of `DL HARQ` Information and also of DL messages in a FIFO scheme refer to the `phyCfg` parameter below. The same applies for the Zero Buffer Memory copy scheme for L2 to L1 `TxData` for `PDSCH`.

For robust processing of commands from the L2 and fully autonomous `PBCH` generation by the PHY in addition to phyCfg also refer to the `phichResource, phichDuration, channelBandwith, radioFrameNumber`, and `subFrameNumber`.

Table 10. PHY_INIT Parameters - Structure of INITPARM

Parameter	Length, bits	Description, value
`channelBandwidth`	8	Channel Bandwidth 0 = 1.4 MHz 1 = 3 MHz 2 = 5 MHz 3 = 10 MHz † 4 = 15 MHz 5 = 20 MHz
`frtypeDuplexMode`	1	0 = FDD (default) 1 = TDD
`ulDlConfig`	3	For TDD UL and DL configuration (Not used in FDD mode)
`specialSubframeConfiguration`	4	For TDD special Subframe configuration (Not used in FDD mode)
`radioAccessMode`	8	0 = OFDMA (eNB) 1 = SC-FDMA (SS)
`physicalResourceBandwidth`	8	12 = 15 kHz carrier spacing (default) 24 = 7.5 kHz carrier spacing
`numResourceBlocksperSlot`	8	6, 15, 25, 50 (default), 75, 100
`phyLayerCellIdGroup`	8	0 (default) ... 167
`phyLayerId`	8	0 (default), 1, 2
`txAntennaPortCount`	8	1 (default), 2, 4, 8

Parameter	Length, bits	Description, value
fastFwdOrPreclockingNum	16	0 ... 1600 (default) ... 3000
fftSize	16	128, 256, 512, 1024 (default), 1536, 2048, 4096. For 15 Mhz channelBandwidth use 2048 fftSize
numberUsedSubcarriers	16	72, 180, 300, 600 (default), 900, 1200
nMaxDLRb	8	Maximum DL resource blocks 6, 15, 25, 50 (default), 75, 100
nMaxULRb	8	Maximum UL resource blocks 6, 15, 25, 50 (default), 75, 100
referenceSignalPower	16	Reference Signal Power in 1/256 dB units, 2's complement notation -32768 ... 0 (default) ... 1536 (-128 to 6 dB)
primarySyncSignalPower	16	Primary Synchronization signal power relative to the average subcarrier power in 1/256 dB units, 2's complement notation -32768 ... 0 (default) ... 1536 (-128 to 6 dB)
secondarySyncSignalPower	16	Secondary Synchronization signal power relative to the average subcarrier power in 1/256 dB units, 2's complement notation -32768 ... 0 (default) ... 1536 (-128 to 6 dB)
numDataRePerPRB	8	Number of data Resource Elements per Pairs of Resource Blocks, as a function of number of antennas used 80 = 1 antenna (default) 76 = 2 antennas 0 = not used 72 = 4 antennas
cyclicPrefixType	8	0 = Short (default) 1 = Long
rxAntennaPortCount	8	1 (default), 2, 4, 8
SyncAntSelection	1	0 Sync Signals only on Antenna 0 1 Sync Signals on both antennas
CarrierAggregationLevel	3	Range 0-4 where zero corresponds to one Carrier and the rest of the entries indicate the number of additional carriers being aggregated
reserved	11	Reserved leave as zero
rxSduLineMonitorInfoenable	1	Not used leave as zero
txStartConfDisable	1	Controls whether TXSTART confirmation is used or not 0 = Enable (default) (for compatibility with earlier specification) 1 = Disable

Parameter	Length, bits	Description, value
rxStartConfDisable	1	Controls whether TXSTART confirmation is used or not 0 = Enable (default) (for compatibility with earlier specification) 1 = Disable
pb	2	0 (default),1,2,3 Parameter P_B [36.213] Section 5.2 Cell Specific Power Control
customExtensionReportEn	1	0 No customer extensions in PHY_RXSTATUS 1 Customer extensions to PHY_RXSTATUS indications
srsMaxUpPts	1	Only used in TDD mode; for FDD leave as zero. In TDD specifies how the Sounding Reference Signals are generated in UpPts subframes per [36.211] Section 5.5.3.2. The $m_{SRS,0}$ value computation that affects the length of the SRS is controlled by this parameter. 0 Disabled 1 Enabled
txSduConfEnable	1	Controls whether TXSDU confirmation is used or not 0 = Disabled (default) 1 = Enabled
sduConfig	1	0 (default): uses contiguous data in TX_SDU and RX_SDU suiTable when the Host and PHY reside on different processors without shared memory 1: uses pointers to data in TX_SDU and RX_SDU suiTable when the Host and PHY have a shared memory and there is no need to copy the original data but simply pass a pointer.
radioFrameNumber	16	0 (default) ... 65535
subFrameNumber	8	0 (default) ... 9
slotNumber	8	0 (default), 1
srsBandwidthConfig	16	0 (default)...7 (see [36.211] tables 5.5.3.2.-1,-2,-3 and -4)
srsSubframeConfig	16	0...15 (default. see [36.211] Table 5.5.3.3-1)
srsSimultaneousANandSRS	16	0 (default),1 (see [36.213] Section 8.2)
prachConfigurationIndex	16	0 (default)...63 (see [36.211] Section 5.7.1)
prachFrequencyOffset	16	0 (default)...94 (see [36.211] Section 5.7.1)
prachHighSpeedFlag	16	0 (default),1 (see [36.211] Section 5.7.2)
prachCyclicShiftConfig	16	0 (default)...15 (Ncs in [36.211] Section 5.7.2)
prachRootSequenceIndex	16	0 (default) 837 (see [36.211] Section 5.7.1)
phichResource	16	This is the size of the PHICH allocation with the following possible values: 0 = 1/6 (default) 1 = 1/2 2 = 1

Parameter	Length, bits	Description, value
		3 = 2 N_g per [36.211] Section 6.9
phichDuration	16	This is the number of symbols that can be used for PHICH. Possible values are: 1 = Normal (default) 2 or 3 = Extended [36.211] Table 6.9.3-1
mbsfn_AreaId	32	MBSFN Area Id [36.211] Section 6.3.1
phyCfg	32	The phyCfg allows backward compatibility with previous releases. Currently LSB is used: **Bit 0 Set**: Use FIFO based MAC-PHY interface **Bit 1 Set:** Do automatic processing of PBCH within PHY ignoring MAC BCH messages. **Bit 2 Set:** Phy Robust processing Enabled. When enabled the PHY in the event that the MAC layer does not issue the TXVECTOR and RXVECTOR on time injects dummy TX and RX Vectors with zero channels and also when needed transmits the PBCH information. In the case of missing TX and RX vectors it issues to the MAC error codes TIMEOUT_RXSTART_REQ and TIMEOUT_TXSTART_REQ. Also on the receiver RXSDU's and RXSTATUS with this error are issued for every expected one in the late rxvector allowing the MAC to free any buffers that have been previously allocated. **Bit 3 Set:** Phy uses TXSDU pointer to data message instead of data lumped together with the txsdu request. In this case the PHY frees both the memory associated with the TxSduReq message and the memory associated with the payload of the message after the processing has completed. Using this scheme allows the MAC to save a memory copy. This setting can be overruled by the setting of bit 9. If bit 9 is zero the previously described behavior is true else bit 9 defines the contents of the pointer. (Supported in T2200/T3300 3.8.1 or later revisions). **Bit 4 Set:** Phy uses a prioritized HARQ ACK/NACK list for control data that is multiplexed on the PUSCH channel over the HARQ/RI information to minimize HARQ latency. The first one contains only the HARQ ACK/NACK information, the second list is for the PUSCH Received data and there even a third lower priority list for CQI and RI information. **Bit 5 Set:** Phy will not free next missed vector. This option is useful when the L2/L3 implements its own logic to detect a late vector and then it doesn't push to the FIFO the late messages.

Parameter	Length, bits	Description, value
		If the bit is clear the PHY after detecting a late message being delivered from the L2/L3 it will pop and free all those messages as invalid. **Bit 6 Set:** Use Intra TTI Indications. Intra TTI indications are issued in between Subframes to allow the L2/L3 to implement robust handling of late APIs from the PHY. **Bit 7 Set:** No SDUs are sent if SR or RACH are not detected. **Bit 8 Set:** Configures PHY for low latency path which in conjunction with matching configuration in L2/L3 stack allows for 8 ms DL HARQ Latency. **Bit 9 Set:** TxSDU pointer is a pointer to an array of pointers and lengths for channel Type PDSCH only. When clear bit 3 defines the meaning of the TxSDU pointer. Available only in T3300/T2200 3.8.1 or newer revisions. **Bit 10:** Reserved **Bit 11:** Reserved **Bit 12**: IRC global enable. When this bit is set a PUSCH channel can be configured individually to use the IRC instead of the MRC default scheme using the `ULChannelDescriptor ircEnable` bit. When this bit is clear the IRC algorithm is always disabled regardless of the state of the `ULChannelDescriptor ircEnable` bit. **Bits 13-31:** Reserved
`puschDtxThresh`	16	PUSCH DTX SINR Threshold reported as `ul_cqi` to Stack 112 = -8db. This is effective if and only if `PuschDtxEn` is set to 1 in the phycfg.xml Default value is 112 which correspond to a -8 db SINR.
`muxPuschAckNackDetThresh`	16	MUX PUSCH ACK NACK DTX SINR Threshold Increase/decrease fractional scale factor used in Calculation. Signal Detect = (((FractionScaleFactor - muxPuschAckNackDetThresh) * signal Power) > noise Power) A Positive number will increase fractional scale factor used to determine if decoded outputs are not random data, thus making it easier to detect. A Negative number will decrease fractional scale factor used to determine if decoded outputs are not random data, thus making it harder to detect. A value of zero implies that the PHY uses its internal default value else the number provided here is added to the default value.

Parameter	Length, bits	Description, value
muxPuschCqiDetThresh	16	MUX PUSCH CQI DTX SINR Threshold Increase/decrease fractional scale factor used in Calculation. Signal Detect = (((FractionScaleFactor - muxPuschCqiDetThresh) * signal Power) > noise Power) A Positive number will increase fractional scale factor used to determine if decoded outputs are not random data, thus making it easier to detect. A Negative number will decrease fractional scale factor used to determine if decoded outputs are not random data, thus making it harder to detect. A value of zero implies that the PHY uses its internal default value else the number provided here is added to the default value.
muxPuschRiDetThresh	16	MUX PUSCH RI DTX SINR Threshold Increase/decrease fractional scale factor used in Calculation. Signal Detect = (((FractionScaleFactor - muxPuschRiDetThresh) * signal Power) > noise Power) A Positive number will increase fractional scale factor used to determine if decoded outputs are not random data, thus making it easier to detect. A Negative number will decrease fractional scale factor used to determine if decoded outputs are not random data, thus making it harder to detect. A value of zero implies that the PHY uses its internal default value else the number provided here is added to the default value.
pucchRbStart	16	PUCCH Resource Block Start, first Resource block to DMA for PUCCH in slot 0 For example, if 0 is specified that means PUCCH can only reside on RB 0->4 and Rb 45->49 for 10 Mhz For example, if 10 is specified that means PUCCH can only reside on RB 10->14 and Rb 35->39 for 10Mhz Valid values = 0 - (half of total Rbs for configured bandwidth - 5) 3Mhz = 0 5Mhz = 0 - 7 (25/2 - 5) 10Mhz = 0 - 20 (50/2 - 5) 15Mhz = 0 - 32 (75/2 - 5) 20Mhz = 0 - 45 (100/2 - 5) Default value is zero
pucchF1DetThresh	16	PUCCH Formats 1, 1a, 1b detection threshold Measured/modified signal power must be at least XX dB above the measured noise power Value = 10 *Abs (log10 (value/32767)). Default is 13.98 dB i.e. PucchF1DetThresh= 1311

Parameter	Length, bits	Description, value
pucchF2sDetThresh	16	PUCCH Format 2, 2a, 2b detection threshold Increase/decrease fractional scale factor used in Reed Muller Confidence Calculation. Signal Detect = (((FractionScaleFactor - PucchF2sDetThresh) * signal Power) > noise Power) A Positive number will increase fractional scale factor used to determine if Reed Muller decoded outputs are not random data, thus making it easier to detect. A Negative number will decrease fractional scale factor used to determine if Reed Muller decoded outputs are not random data, thus making it harder to detect. A value of zero implies that the PHY uses its internal default value else the number provided here is added to the default value.
prachThresh	16	Prach Threshold Value for detection The value corresponds to the theshold needed to obtain 100% prach detection for a given target SNR. The equation provided is good for an AWGN channel. `SNR_in_dB = 10*log10(prachThr / (839-prachThr))` Some examples are: SNR_in_db = 13 requires prachThr= 800 SNR_in_db = 0 requires prachThr= 420 SNR_in_db = -8.7 requires prachThr= 100 SNR_in_db = -12 requires prachThr= 50 Default value is 100 which allows 100% detection for SNR = -8.7 db.
addFeatures[16]	16	For additional features, please consult release notes for the latest feature set and support

4.2.4 PHY_START Request

Table 11. PHY_START Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 17
0	15-0	16	Operational Mode (Based on Interrupt Source and Radiohead interface type) 0 – CPRI (Default) 1- Timer based (Used for Debugging Only)
1	31:0	32	CPRI: 0 Normal Mode Timer: Stop after Count subframes External: N/A

32-bit Word #	Bits	Size, bits	Description
2	31:0	32	Resolution (related to operational mode) CPRI: N/A Timer: period in ms (default is 1) External: N/A

This primitive is issued by the L2 to request the PHY to start or resume processing depending on whether a PHY_INIT or a PHY_RECONFIG command was issued earlier.

The purpose of this command is for the PHY to synchronize with the Radio Interface Frame clock and start issuing subframe indications to the L2 so the normal subframe command exchange described in Section 3.2.3 API Message Order can take place.

4.2.5 PHY_START Confirmation

Table 12. PHY_START Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 18
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	subFrameNumber

This primitive is issued by the PHY to acknowledge to L2 that it has started the synchronization with the Radio Interface as a result of the receipt of the `PHY_START.request` command.

The indication issued to the L2 that confirms that the synchronization has been achieved is the `TXSTART.indication`.

4.2.6 PHY_STOP Request

Table 13. PHY_STOP Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 19
0	15-0	16	Message Length

This primitive is issued by the L2 to request the PHY to stop subframe processing.

The purpose of this command is for the PHY to disable the processing of the Radio Interface timing derived interrupts to allow its reconfiguration with a `PHY_RECONFIG.request` command.

4.2.7 PHY_STOP Confirmation

Table 14. PHY_STOP Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 20

0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Reserved

This primitive is issued by the PHY to inform the L2 that it has completed the execution of the PHY_STOP.request command. Depending on the internal Phy state the PHY_STOP confirmation can be delayed until completion of the processing of the ongoing subframe.

4.2.8 PHY_STOP Indication

Table 15. PHY_STOP Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 21
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	SubframeNumber

The PHY_STOP Indication is sent when the PHY automatically stops processing according to PHY_START.request parameters. For example, if PHY_START.request specifies Count = 100 subframes and Mode = CPRI, then PHY will automatically stop after 100 loops and this indication will be sent.

4.2.9 PHY_RECONFIG Request

Table 16. PHY_RECONFIG Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 15
0	15-0	16	PHY_INIT payload length (INITPARM structure)
...	...	...	Parameters

This primitive is issued by the L2 to request the PHY to perform a reconfiguration process based on the parameters provided in the INITPARMS portion of this primitive.

This command is to be issued by the L2 once it has issued a PHY_STOP and received the PHY_STOP confirmation from the PHY.

4.2.10 PHY_RECONFIG Confirmation

Table 17. PHY_RECONFIG Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 16
0	15:8	8	Status 0 = Success

			otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Reserved

This primitive is issued by the PHY to inform the L2 that it has completed the reconfiguration process requested by the `PHY_RECONFIG.request` command.

4.2.11 PHY_QUERY Request

The `PHY_QUERY.request` is used by the L2/L3 software to find out the API version supported by the PHY. The version for any PHY release that incorporates new features, enhancements or changes that affect the API can be found in the release notes provided with the PHY release.

Table 18. PHY_QUERY Request

32-bit Word #		Bits	Size, bits	Description
0		31:24	8	PHY entity ID (Instance)
0		23:16	8	Message Type = 34
0		15-0	16	Message Length

4.2.12 PHY_QUERY Indication

The `PHY_QUERY.indication` primitive is issued by the PHY to the L2/L3 software with a text string that represents the version of the current PHY code.

Table 19. PHY_QUERY Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 35
0	15-0	16	Message Length
1	31:0	32	Version String Bytes 3..0
...		32	...
N	31:0	32	Version String Bytes n/4, n-1/4, n-2/4, n-3/4

If the version information is not a multiple of 4 bytes, bytes containing zero are appended to preserve 32 bits alignment.

4.2.13 PHY_SHUTDOWN Request

The `PHY_SHUTDOWN.indication` primitive is issued by the L2 to request the PHY to stop subframe processing.

The purpose of this command is for the PHY to disable the processing of the Radio Interface timing derived interrupts. All PHY resources and memory will be freed.

Table 20. PHY_SHUTDOWN Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 37
0	15-0	16	Message Length

4.2.14 PHY_SHUTDOWN Confirmation

The PHY_SHUTDOWN.confirmation primitive is issued by the PHY to inform the L2 that it has completed the execution of the PHY_shutdown.request command. Depending on the internal PHY state, the PHY_SHUTDOWN confirmation can be delayed until completion of the processing of the ongoing subframe.

Table 21. PHY_ SHUTDOWN Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 38
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Reserved

4.3 L1 Subframe Messages

4.3.1 PHY_TXSTART Request

Table 22. PHY_TXSTART Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 1
0	15:0	16	TXVECTOR Length
			TXVECTOR as described in Section 4.6.1 Downlink Transmitter Descriptor (TXVECTOR)

This primitive is issued by the L2 to request the eNB PHY to start the transmission of PHY PDUs in the nearest DL sub-frame, and carries all the control information needed for the local PHY as described in Section 4.6.1 Downlink Transmitter Descriptor (TXVECTOR).

As a result of the reception of this primitive, the PHY sends a confirmation primitive, and at the proper moment, starts transmission of the PHY PDUs or performs requested actions as defined by the control information contained in TXVECTOR. The TXVECTOR corresponds to the DL-Scheduling Information currently being sent to remote UE. The TXVECTOR contains only those DL-Scheduling Information parameters which are necessary for the PHY to perform the correct transmit actions.

4.3.2 PHY_TXSTART Confirmation

Table 23. PHY_TXSTART Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 2
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	SubframeNumber from TXVECTOR

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	reserved
1	0:15	16	frameNumber

This primitive issued by the PHY to confirm reception of `PHY_TXSTART.request` primitive and provides the command execution status to the L2. If the PHY was not able to process the `PHY_TXSTART.request` primitive it returns an error code (`ERRORCODE`) to indicate the issue. The `frameNumber` and `SubframeNumber` fields are used by the TX state machine to trace events regarding the given sub-frame.

Note: This message is no longer supported during real time execution, only in the c-reference code used to do initial host code integration debugging.

4.3.3 PHY_TXSTART Indication

Table 24. PHY_TXSTART Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 3
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Sub-frame number from TXVECTOR
1	31:16	16	Initial Frame Number (12 LSB's are used) System Frame Number
1	15:0	16	Reserved

This primitive is issued by the PHY to indicate to L2 that a downlink sub-frame has just started. It is sent once per sub-frame, so it allows synchronization between the PHY and the L2 on a sub-frame basis. In the real time execution case this primitive is used as a TTI delimiter for both TX and RX.

4.3.4 PHY_INTRA_TTI Indication

Table 25. PHY_INTRA_TTI Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 26
0	15:8	8	Index into the subframe 1 first intra TTI message in the subframe
0	7:0	8	Sub-frame number from TXVECTOR
1	31:16	16	System Frame Number
1	15:0	16	Reserved

4.3.5 PHY_RXSTART Request

Table 26. PHY_RXSTART Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 7

0	15:0	16	Length of RXVECTOR
...	...	...	RXVECTOR as described in Section 4.6.2 Uplink Receiver Descriptor (RXVECTOR)

This primitive is issued by L2 to request the eNB PHY to start the reception of the next uplink sub-frame. The RXVECTOR carries all of the control information required by the PHY to receive and decode the UL TB's. After receiving this primitive, the PHY sends a confirmation primitive and at the proper moment begins to receive and decode UL symbols. The RXVECTOR corresponds to the Uplink Scheduling Information sent to the remote Ues n-4 sub-frames earlier. It contains only those UL-Scheduling Information parameters which are necessary for the PHY to perform the correct receive actions.

Note: For the first sub-frames a RXVECTOR containing 0 channels is expected.

The eNB version of the RXVECTOR is formatted as specified in Section 4.6.2 Uplink Receiver Descriptor (RXVECTOR).

4.3.6 PHY_RXSTART Confirmation

Table 27. PHY_RXSTART Confirmation

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 8
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Sub-frame number from RXVECTOR

This primitive is generated by the PHY after it has received and processed a complete PHY_RXSTART.request primitive. It provides a confirmation of the reception of the request for command execution status to the L2. If the execution is not successful, the PHY returns an error code (ERRORCODE) to indicate the issue. The sub-frame number field is typically used by the RX state machine to trace events regarding the given sub-frame.

Note: This message is not issued in the real time execution of the PHY and it is available for c reference code based L2 – PHY integration debugging.

4.3.7 PHY_RXSTART Indication

Table 28. PHY_RXSTART Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 9
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7:0	8	Sub-frame number from RXVECTOR

This primitive is issued by the PHY to indicate to the MAC the actual start of the UL subframe. The PHY generates this primitive in normal operation mode (after receiving the first PHY_RXSTART.request from the L2). The Sub-frame number field is used by the RX state machine to trace events regarding the given sub-frame.

Note: This message is not issued in the real time execution of the PHY as the RXSTART and TXSTART are synchronous and TXSTART.indication is used as a subframe indication.

4.3.8 PHY_TXHIADCIUL Request

Table 29. PHY_TXHIADCIUL Request

32-bit word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 22
0	15:0	16	TXHIADCIUL Length
1	31:22	10	frameNumber [36.331]6.2.2 frameNumber ∈ {0, .., 1023}
1	21:20	2	reserved
1	19:16	4	subFrameNumber [36.331]6.2.2 subFrameNumber ∈{0,..,9}
1	15:8	8	Number of hi sdu's ∈{0,.., 255} in this subframe, i.e. number of PHY_TXHISDU.request messages.
1	7:0	8	Number of DCI UI's sdus∈{0,..,255} in this subframe, i.e. number of PHY_TXDCIULSDU.request messages. This number is not included in the TXVECTOR's numCtrlChannels entry .

4.3.9 Downlink Data

4.3.10 PHY_TXSDU Request

Table 30. PHY_TXSDU Request

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 4
0	15:0	16	ChannelId
1	31:16	16	Number of Bytes in the Transmit Block
1	15	1	cwId (0 or 1) (Codeword Identifier)
1	14:8	7	Reserved if channelType is not HI. For HI channelType it contains the PHICH sequence index
1	7:4	4	channelType (Transport Channel type refer to ChannelDescriptor channelType for allowed values)
1	3:1	3	For channelType PHICH these bits contain the transmissionMode right justified and zero padded as follows 0 Single Antenna 1 TransmitDiversity For other channelTypes these bits are reserved
1	0	1	For channelType HI this bit contains the ack/nack information. For channelType UCI and format1 this bit contains the SR information For other channel types this bit is zero
2	31:7	25	maxBitsperCw For Channel Type PDCCH (DCI) this field contains the number of bits in the payload. For other channel types this field is a don't care

32-bit Word #	Bits	Size, bits	Description
2	6:0	7	phichGrpNumber. No longer used. Set to zero.
3	31:0	32	PHY SDU pointer (TB), or PHY SDU Array pointer or SDU data depending on PHY_INIT configuration. If bit 9 of PhyCfg is one and this pointer is zero then the Array of pointers and lengths immediately follows in memory (Part of the payload). The PHY releases the storage associated with the command upon completion of the TxSdu processing. Since the L2 may have to retransmit the data for HARQ reasons then the MAC is responsible for releasing the memory associated with the data chunks. **Note:** only channel Type PDSCH uses the Zero memory copy scheme.

This primitive is issued by L2 to transfer a Transport Block (TB) to the PHY uniquely identified by its ChannelId using the parameters associated to this channel from the TXVECTOR. If there is a difference between the number of bits specified in the TXVECTOR for this channel and the number contained in the TB after the PHY performs the coding, modulation and redundancy check bits addition specified in the TXVECTOR, then an ERRCODE is given on a per PHY_TXSDU basis with a PHY_TXSDU confirmation if enabled.

Note: This error checking is not supported in the Real Time execution only in the c-reference code for early Host integration debugging.

For all payloads the convention used is that the most significant bit in the TXSDU payload corresponds to the first in time, or in 3GPP notation a_0 (The first bit to be processed by the PHY).

For information about DCI payload formats please refer to [36.212] Section 5.3.3.1.

For transmission modes where MIMO is enabled using Transmit Spatial Multiplexing, two Transport Blocks need to be provided by the L2/L3, meaning that two TXSDUs each with the cwId field set to identify the codeword being transmitted are provided per subframe to the PHY by the L2/L3.

4.3.11 Zero Buffer Copy for FlexRAN

The implementation of the zero memcpy interface (ZBC) involves replacing the current payload pointers in the PHY_TXSDU with a pointer to an array of descriptors that contain the data pointers and lengths for each data chunk. Because PBCH and PDCCH channel type payloads are small, there are no efficiency gains, so this scheme applies only to channel type PDSCH.

Note: The TXHISDU and TXDCIUSDU messages are not affected and still support data immediately attached to the header or a pointer to the payload. This is because the payloads for control information are quite small and there is no need to decompose them into multiple subblocks.

ZBC is the default mode of operation for the FlexRAN implementation for PDSCH payloads.

The L2+ stack sets up the array of pointers and lengths first and then fills the TXSDU request headers so the PHY handles internally the processing of the payloads.

Figure 43 illustrates the ZBC mode for the TxSDU API.

Case 1: pTxSdu=0

Case 2: for pTxsdu non-zero

Figure 43. TxSdu Scheme with Array of Pointers as Part of Payload

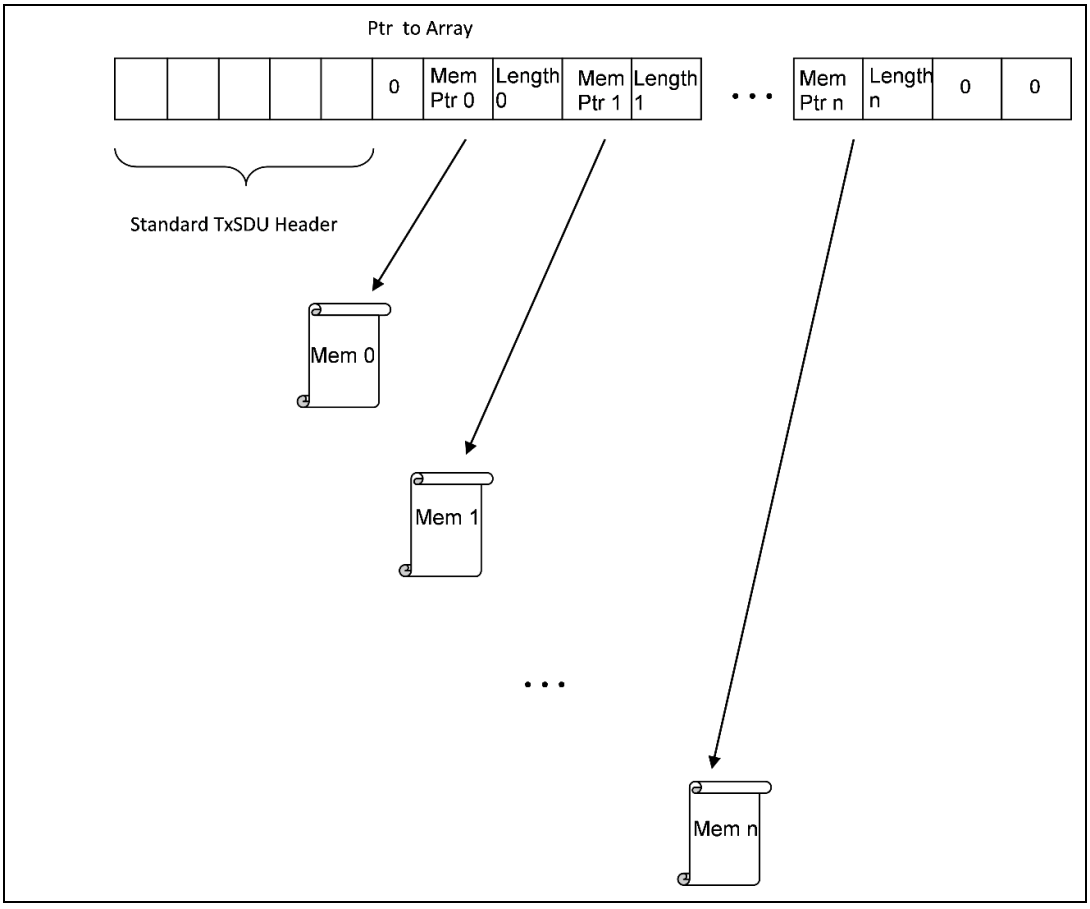
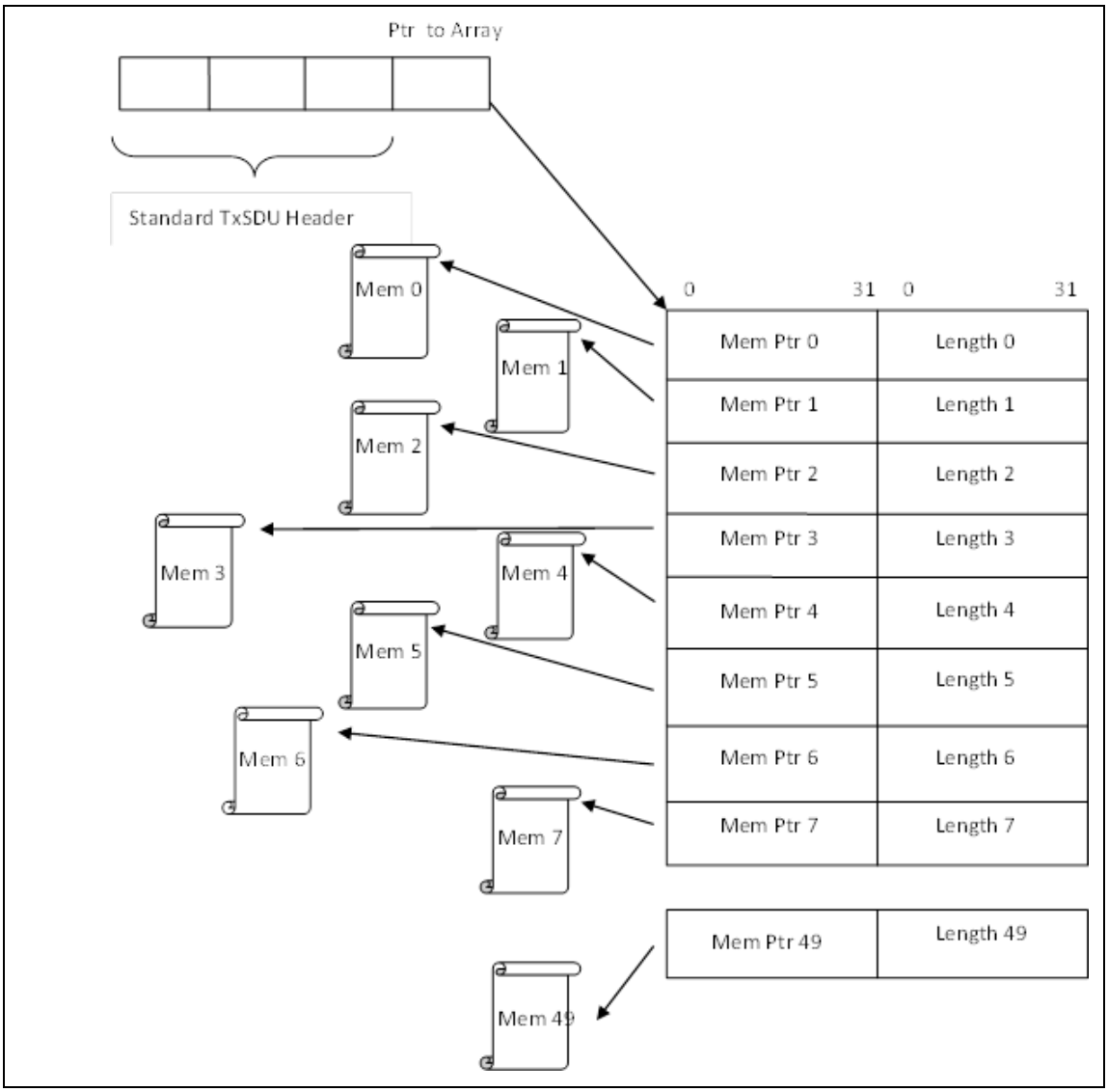


Figure 44. TxSdu Array Pointer Scheme with Separate Allocation



4.3.12 PHY_TXHISDU Request

Table 31. PHY_TXHISDU Request

32-bit word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 23
0	15:0	16	TXHISDU Length
1	31:16	16	ChannelId
1	15:11	5	PHICH Group Number
1	10:4	7	PHICH sequence index

32-bit word #	Bits	Size, bits	Description
1	3:1	3	Transmission Mode 1: Single Antenna 2: TransmitDiversity The transmissionMode must be right justified and zero padded
1	0	1	HI ack/nack information.
2	31:16	16	TX Power Control
2	15:11	5	PHICH Group Number Second TB (3GPP release 10)
2	10:4	7	PHICH sequence index Second TB (3GPP release 10)
2	3:1	3	Reserved
2	0	1	0: No second TB HARQ Info (3GPP release 10 support) 1: second TB info valid

4.3.13 PHY_TXDCIULSDU Request

Table 32. PHY_TXDCIULSDU Request

32-bit word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 24
0	15:0	16	TXDCIULSDU Length
1	31:24	16	Number of bytes in payload
1	15:0	16	Channel ID
2	31:24	8	StartCCE $\in$ [0,88]
2	23:16	8	numCCE or Aggregation level $\in$ {1,2,4,8}
2	15:0	16	RNTI $\in$ {1,..,65535}
3	31:16	16	txPowerControl This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. Depending on the type of channel the value in txpowerControl can be a relative value to the reference signal power for a PDSCH channel type. -32768 to 1536 (-128 to 6 dB) For RBs without UE specific RS, the ratio of PDSCH to pilot signal power depends on the PB, antenna port, CP type according to Table 5.2-2 in [36.213]. txPowerControl can be treated as. ρ_A. And Phy can determine ρ_B according to P_B configuration. For the RBs containing UE specific RS, the ratio of PDSCH to pilot signal power is kept constant as txPowerControl for all the OFDM symbols in one subframe.
3	15:0	16	transmissionMode The following transmission modes are possible for dci ul 1 = Single-antenna port; port 0 2 = Transmit diversity Right justified and zero padded
4	31:0	32	Pointer to DCI UL Payload (type U32*)

The fields defined in the DCI formats are mapped to the information bits a_0 to a_{A-1} as follows.

Each field is mapped in the order in which it appears in the description, including the zero-padding bit(s), if any, with the first field mapped to the lowest order information bit a_0 and each successive field mapped to higher order information bits. The most significant bit of each field is mapped to the lowest order information bit for that field, for example, the most significant bit of the first field is mapped to a_0. (In other words the Most Significant Bit in this description corresponds to the first in time to be processed at the PHY).

In the event that the payloads do not align to a byte boundary they should be zero filled, left justified. (Padding zeroes after the last in time bits).

4.3.14 Uplink Data and Status Indications

4.3.15 PHY_RXSDU Indication

The format for the `PHY_RXSDU.indication` is shown in Table 33.

Table 33. PHY_RXSDU Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 10
0	15:0	16	ChannelId
1	31:24	8	Status 0 = Success (valid data) otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
1	23:0	24	Number of bits received (including padding)
2	31:24	8	ChannelType as defined in ChannelDescriptor structure (Refer to Section 4.6.14 ULChannelDescriptorStruct)
2	23:16	8	Sub-frame number
2	15:0	16	Frame number (10 bits are used)
3	31-16	16	Timing Advance Information in two's complement notation. Value is in terms of Ta (16xTs). Range is 0 to 63 with a value of 31 means no change in UE timing is required
3	15:8	8	ulCQI in steps of 0.5 dB. Ul_CQI $\in$[0,255] which corresponds to SNR of -64 to +63.5 dB

32-bit Word #	Bits	Size, bits	Description
3	7:6	2	cqiPmiConf (Confidence Metric) Bit 7 HDQ Hard Decision Indicator (1=good, 0=bad) HDQ is a single status bit that indicates if the Reed Muller decoder outputs are most likely correct or not (0= erroneous, 1=correct). The assumption used for this indicator is that the UE is actually transmitting the proper Reed-Muller encoded CQI message with the possibility that the transmission is being corrupted by noise, fading or interference Bit 6 HDQRD Hard Decision Indicator assuming possible random data (1=good, 0=bad) HDQRD is a single status bit that indicates if the Reed-Muller decoder outputs are most likely correct or not (0=erroneous, 1=correct). However, the assumption used for this indicator is that the UE is possibly transmitting the proper Reed-Muller encoded CQI message or just random data. This can occur if the UE somehow gets misconfigured. In this case, the base station is expecting and decoding a CQI message but the UE never sends one. This indicator guarantees that the decoder inputs are really Reed-Muller encoded and the decoder outputs are correct. **Note:** The probability of false detection for this indicator is no worse than 1%. The L2 code should ensure that both HDQ and HDQRD are both "1=good" to use the CQI decoded bits otherwise they should be ignored. This criteria is used whenever CQI length is less than 12 bits which do not have a CRC attached
3	5:2	4	pucchType reflects format type as follows: 0 Format 1 1 Format 1A 2 Format 1B 3 Format 2 4 Format 2A 5 Format 2B 6 Format 3
3	1	1	chanDetected when set
3	0	1	Srdetected.This bit when set indicates that SR (schedule request has been detected for the case of simultaneous HARQ and SR or for SR alone)
4	31:0	32	mErrAvg[0] Obsolete
5	31:0	32	mErrAvg[1] Obsolete
6	31:27	3	Reserved
6	28	1	CodewordId: 0 or 1 Only for 3GPP Release 10 or later with UL MIMO support. Please check the release notes for the specific device for the availability of the support. For previous 3GPP releases this bit is zero
6	27:24	4	listType 1 UL Pusch 2 UL Pucch 3 Other 4 Single Entity (Prach/Srs/ any single message)
6	23:16	8	RSSI [dB] ∈ [0, 164] where 0 represents -82.0 dBFS and 164 represents 0 dBFS. Each step is 0.5 dBFS

32-bit Word #	Bits	Size, bits	Description
6	15:0	16	mErrExpo This value needs to be saved by the Host and restored via the RXVECTOR the next time this same channel is expected
7	31:0	32	Pointer to Payload or Actual payload depending on the value of sduConfig in the PHY_INIT parameters
…	…	…	ChannelType dependent indication data

The primitive is generated by the PHY to send the contents of a received PHY SDU from the PHY to the L2. The PHY generates this primitive in normal operation mode (after receiving first a `PHY_RXSTART.request` from the L2).

There are as many `PHY_RXSDU.indication` primitives issued per subframe as unique ChanId's are specified in the RXVECTOR.

1. PHY_RXSDU at eNB for UL-SCH Information

The `PHY_RXSDU` at the eNB for the `UL-SCH` uses the standard format described in Section 4.3.15 PHY_RXSDU Indication and at Word 7 there is either a pointer to the actual UL-SCH payload or the payload data itself depending on the value of sduConfig in the `PHY_INIT` parameters.

The channeld field in the indication matches the one set by the L2 in the `PHY_RXSTART.request RXVECTOR` and the data corresponds to one TB. In PHY versions compliant to 3GPP release 10 or later, check the `CodewordId` field to determine the specific TB being delivered.

2. PHY_RXSDU at eNB for RACH Information

Since the RACH process can be initiated either under the eNB control or by the UE on its own the RACH detection at the eNB corresponds to an indication There is no `PHY_RXSDU` at the eNB for RACH Information, only the `PHY_RXSTATUS.ind` of Section 0

PHY_RXSTATUS for Random Access Detection for all detection events that took place during the subframe.

3. PHY_RXSDU at eNB for UCI information

The UCI information at the eNB consists of SR, CQI, RI and PMI reports and HARQ ACK/NACK's. The details of the different payload formats are explained in [36.212] Sections 5.2.2.6, 5.2.2.6.1, 5.2.2.6.2, 5.2.2.6.3, 5.2.2.6.4 and [36.213] Sections 7.2, 7.2.1 and 7.2.2.

In the case of the `PHY_RXSDU` for SR, the number of bits in the indication message is zero and the indication is issued only if a Schedule Request was detected during the current subframe for the given channel id.

The PHY_RXSDU header message bit 0 in the third 32 bit word is set if the SR was detected.

The number of bytes reported for UCI information is always 2 and the information contained in them depends on the PUCCH format type as described below.

Table 34. PUCCH Format

Bit:	Byte 0								Byte 1							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
F1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
F1a	x	x	x	x	x	x	x	x	x	x	x	x	x	A1	x	x
F1b†	x	x	x	x	x	x	x	x	x	x	x	x	x	A1	A2	x
F2	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	x	x	x
F2a	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	A1	x	x
F2b	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	A1	A2	x

†For normal F1b. Convention for F1b with channel selection explained further below.

In this table:

- Cn = CQI or RI or HARQ ACK/NACK bits
- An = HARQ ACK/NACK Bits

The number of Cn bits can vary between 5-15.

For format 2x the location of the HARQ ACK/NACK bits depends on the number of CQI and RI bits used.

For format 1 the Schedule Request Information is contained in the RXSDU header as defined in Table 33. PHY_RXSDU Indication.

For format 1B the L2 knows whether a normal 1B or a channel selection 1B is in use and in the case of format 1B with channel selection the meaning of the 2 bytes is per [36.213] tables 10.1.2.2.1-2, 10.1.2.2.1-3 and 10.1.2.2.1-4.

Up to two bytes of information are used for F1B with channel selection depending on the number of resources, i.e the `numPucchResourcesHarq` selected by the L2 for the channel.

For up to 2 resources only 1 byte is used and it is split into 2 nibbles `harq-ack0` and `harq-ack1` where `harq-ack0` corresponds to the most significant nibble and `harq-ack1` corresponds to the least significant nibble.

For 3 or more resources 2 bytes are used where the second byte is split also in two nibbles `harq-ack2` and `harq-ack3` where `harq-ack2` corresponds to the most significant nibble and `harq-ack3` to the least significant nibble respectively.

The convention used for the HARQ-ACK data is as follows:

- 0000 = dtx
- 0001 = Ack
- 0010 = Nack
- 0100 = dtx
- 0110 = dtx/nack[36.213][36.213][36.213][36.213]

4.3.16 PHY_RXEND Indication

Table 35. PHY_RXEND Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 11
0	15:8	8	Status 0 = Success otherwise ERRCODE (Refer to Section 4.1.2 Error Field Coding)
0	7	1	Initial Radio Frame Sync Indicator This bit is only set after the PHY_START command and informs the L2 that it can issue the RX_START.req. When this bit is set the sub-frame number is a do not care and the Frame number is also a do not care
0	6:0	7	Sub-frame number
1	31:16	16	Frame number (10 bits are used)
1	15:0	16	Reserved

This primitive is issued by the PHY to indicate that no more `RXSDU.indication` messages sent from PHY to the L2 can be expected for this particular uplink sub-frame (as defined by the proceeding `PHY_RXSTART.request`). The status for this primitive applies to the entire UL subframe, not to any individual UL TB. The error code and the statistics for each UL `ChanId` are sent to the L2 in the `PHY_RXSDU.indication` primitives.

The Frame Number field and the Sub-frame number are provided in the indication to allow the RX state machine to trace events and resolve them in time.

4.3.17 PHY_RXSTATUS Indication

`RXSTATUS.indication` is used to report different detection, measurement and customer specific information computed or collected by the BS receiver PHY. For example, it can be used to report the total number of random access processes that were detected during the last subframe if the PHY_INIT parameter `rachReportMode` has been set. It can also be used for SRS channel Info results with the specific format for the reports to be defined later.

There are hooks provided in the `PHY_RXSTATUS.indication` message to provide customer specific reports by allocating reserved status types.

Table 36. PHY_RXSTATUS indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)

32-bit Word #	Bits	Size, bits	Description
0	23:16	8	Message Type = 14
0	15:0	16	Reserved1
1	31:24	8	Number of Events being reported
1	23:16	8	Subframe Number
1	15:0	16	System Frame Number
2	31:20	12	Reserved
2	19:16	4	listType 1 UL Pusch 2 UL Pucch 3 Other 4 Single Entity (Prach/Srs/ any single message)
2	15:8	8	Status Set to TIMEOUT_RXSTART_REQ if API delayed from MAC Set to SUCCESS otherwise
2	7:0	8	Status Type 0 Random Access Detector Results 1 SRS Channel Info Results 2 CQIRIHI Info Results 3 Received Interference Power 4 Thermal Noise Power 5 RSSI 6 NOISE FLOOR 7 NOISE DEMOD 8 SIGNAL LEVEL 9 NOISE DEMOD CELL WIDE 10-199 Reserved 200-255 For Customer Extensions
...	...	...	Event List

4. PHY_RXSTATUS for Random Access Detection

In the case of Random Access Detection events, the format of the event list is as follows:

Table 37. PHY_RXSTATUS for Random Access Detection

32-bit Word #	Bits	Size, bits	Description
1	31:26	6	prachPreambleId ∈ {0,..,63} per [36.211] Section 5.7.2.
1	25:20	6	prachConfigIdx ∈ {0,..,63} per [36.211] Section 5.7.1.

32-bit Word #	Bits	Size, bits	Description
1	19:16	4	frequencyIdx ∈ {0,..,5} corresponds of the detected PRACH within the subframe (TDD), the number corresponds to the frequency resource bit index in the prachTddFreqResEnabled variable. Please refer to f_id in [36.321] Section 5.1.4.
1	15:0	16	Timing Advance in terms of TA (16*Ts) complement , valid range ∈ [0,1282]

5. PHY_RXSTATUS for SRS Information

SRS Channel Info results events have a header as shown in Table 38:

Table 38. PHY_RXSTATUS for SRS Information

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Reserved filled with zeros
1	15:0	16	Length in bytes of the Status Message List
2	...	...	Status Message List

Each SRS event being reported has the following entries shown in Table 39:

Table 39. SRS Event Entries

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Srsindex ∈ [0.. numberSrsinSf] Corresponds to the index to one of the srs dedicated structures present in the RxVector for the subframe being processed
1	15:0	16	widebandSnr
2	31:16	16	Timing Advance in terms of TA (16*Ts) complement , valid range ∈ [0,1282]
2	15:8	8	Number of Rb's (Resource Blocks) used (For srs this number is always a multiple of 4)
2	7:0	8	RbStart (Index of first resource block used by this srs)
3	31:0	32	*psnr

For i= RbStart; i< (RbStart+Number of Rb's)/4; i+4

Table 40. SNR Status

32-bit Word #	Bits	Size, bits	Description
1	31:24	8	Snr (i+3) for Rb(i) ∈ [0, 255] with 0 corresponding to -64 dB and 255 corresponding to 63.5 dB, 0.5 dB steps
1	23:16	8	Snr(i+2) for Rb(i+1) ∈ [0, 255] with 0 corresponding to -64 dB and 255 corresponding to 63.5 dB, 0.5 dB steps

32-bit Word #	Bits	Size, bits	Description
1	15:8	8	Snr(i+1) for Rb(i+2) ∈ [0, 255] with 0 corresponding to -64 dB and 255 corresponding to 63.5 dB, 0.5 dB steps
1	7:0	8	Snr(i) for Rb(i+3) ∈ [0, 255] with 0 corresponding to -64 dB and 255 corresponding to 63.5 dB, 0.5 dB steps

6. PHY_RXSTATUS for CQI, RI, ACK/NACK Information

Table 41. PHY_RXSTATUS for CQI, RI ACK/NACK

32-bit Word #	Bits	Size, bits	Description
1	63:44	20	Reserved
1	43:38	6	ChannelId (For the ulSch)
1	37:36	2	riConf (Confidence metric for ri) 1=good, 0=bad
1	35:34	2	ackConf (Confidence metric for HARQ ACK/NACK) 1=good, 0=bad
1	33:32	2	cqiPmiConf (Confidence metric) Bit 17 HDQ Hard Decision Quality Indicator (1=good, 0=bad) HDQ is a single status bit that indicates if the Reed Muller decoder outputs are most likely correct or not (0= erroneous, 1=correct). The assumption used for this indicator is that the UE is actually transmitting the proper Reed-Muller encoded CQI message with the possibility that the transmission is being corrupted by noise, fading or interference Bit 16 HDQRD Hard Decision Indicator assuming possible random data (1=good, 0=bad) HDQRD is a single status bit that indicates if the Reed-Muller decoder outputs are most likely correct or not (0=erroneous, 1=correct). However, the assumption used for this indicator is that the UE is possibly transmitting the proper Reed-Muller encoded CQI message or just random data. This can occur if the UE somehow gets misconfigured. In this case, the base station is expecting and decoding a CQI message but the UE never sends one. This indicator guarantees that the decoder inputs are really Reed-Muller encoded and the decoder outputs are correct. **Note**: The probability of false detection for this indicator is no worse than 1%. The L2 code should ensure that both HDQ and HDQRD are both 1=good to use the CQI decoded bits otherwise they should be ignored
1	31:24	8	Length in bytes of the Status Message
1	23:8	16	Number of cqi/pmi bits. Not present when zero. The number of bits comes from RXVECTOR
1	7:5	3	Number of Ri bits. Ri not present when zero. The number of bits comes from RXVECTOR
1	4:0	5	Number of HARQ Bits. HARQ not present when zero. The number of bits comes from RXVECTOR
2 ..	...	...	Status Message Bytes

The format for the status message is CQI/PMI bytes first if available, followed by single byte of RI if available followed by single byte of HARQ if available. The information is left justified and starts at the most significant bit which corresponds to the last in time.

For CQI information there can be 1 or 2 bytes and the following is the format for the data (Number of bits is between 5 and 13).

Table 42. Byte 0 CQI Info

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D0	D1	D2	D3	D4	D5	D6	D7

Table 43. Byte 1 CQI Info

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D8	D9	D10	D11	D12			

For RI information there is only 1 byte. The format used is as follows:

Table 44. Byte 0 RI Info

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RI0	RI1	RI2	RI3	RI4	RI5	RI6	RI7

The confidence bit(s) for the RI information are on bits 25:24 and the number of RI bits being reported are in contained in bits 7:5. Bits not used are zero filled.

For HARQ information the number of bytes depends on the number of HARQ bits being reported which are provided in bits 4:0. The format used is described below with the MSB bit being ACK0. Bits not used are zero filled.

Table 45. Byte 0 Harq Info

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Ack 0	Ack 1	Ack 2	Ack 3	Ack 4	Ack 5	Ack 6	Ack 7

The confidence bit(s) for HARQ Information are located in bits 23:22.

7. PHY_RXSTATUS for Customer Extension Events

The generic structure for the Customer Extension events is shown below:

Table 46 PHY_RXSTATUS for Customer Extensions Events

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Reserved filled with zeros
1	15:0	16	Length in bytes of the Status Message
2 ..	...	...	Status Message List

8. PHY_RXSTATUS for Received Interference Power

This indication is issued by the PHY upon a rxIntMeasEn request coming from the L2 on `RXVECTOR`. Upon completion of the measurement the PHY issues a `PHY_RXSTATUS` message with Status Type 3 Received Interference Power.

Table 47. PHY_RXSTATUS for Received Interference Power 1

32-bit Word #	Bits	Size, bits	Description
1	31:0	32	Number of Resource Blocks

For i= 0; i< (Number of Resource Blocks)/2; i+2

Table 48. PHY_RXSTATUS for Received Interference Power 2

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	rxRIP_dBFS[i+1] Received Interference Pwr in dBFS for RB i+1 rxRIP_dBFS ∈ [0, 900] with 0 representing -90 dBFS and 900 0 dBFS, each step is 0.1 dBFS
1	15:0	16	rxRIP_dBFS[i] Received Interference Pwr in dBFS for RB i rxRIP_dBFS ∈ [0, 900] with 0 representing -90 dBFS and 900 0 dBFS, each step is 0.1 dBFS

For bandwidths that have an odd number of Resource Blocks the last entry is padded with 16-bits.

9. PHY_RXSTATUS for Thermal Noise Power

This indication is issued by the PHY when a rxTherNoiseEn request is received from the L2 in RXVECTOR. After completing the measurement the PHY issues a PHY_RXSTATUS message with Status Type set to Thermal Noise Power including the following information. This measurement is not currently supported in the PHY code.

Table 49. PHY_RXSTATUS for Thermal Noise Power

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Total number of resource blocks used in the computation
1	15:0	16	Thermal Noise Power [dB] ∈ [-255, 255] in 1/128 dB steps (Q7)

10. PHY_RXSTATUS for Received Signal Strength Indication (RSSI)

This indication is issued by the PHY when a rssiMeasEn request is received from the L2 in RXVECTOR. After completion of the measurement the PHY issues a PHY_RXSTATUS message with Status Type set to RSSI including the following information computed at the RF front end. The wideband RSSI measurement is not currently supported by the PHY code.

Table 50. PHY_RXSTATUS for RSSI

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	RSSI antenna 0 [dB] ∈ [0, -128] Only the lower 8 bits are used unsigned with the LSB representing 0.5 dB
1	15:0	16	Reserved

In addition there are RSSI information incorporated into the RxSdu indications for PUCCH and PUSCH.

11. PHY_RXSTATUS for Receive Noise at the Demodulator

This indication is issued by the PHY when the noiseDemodMeasEn is set by the L2 in the RXVECTOR. The receive noise measurement corresponds to the error vector taken after the IDFT by subtracting the received point from the closest ideal constellation point. At the completion of the measurement the PHY issues a PHY_RXSTATUS message with Status Type set to Noise Demod.

Table 51 shows the first the average complex values are reported, the values are in S16 format.

For i= start rb sl0; i< (Num rbs); i+1

Table 51. PHY_RXSTATUS for Receive Noise at the Demodulator

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Channel ID
1	15:0	16	Number of entries
2	31:0	32	Status/len
3	31:0	32	Start_rb_slot0[0]
4	31:0	32	Start_rb_slot0[1]
5	31:0	32	Num_rbs[0]
6	31:0	32	Num_rbs[1]
7	31:0	32	Start_rb_slot1[0]
8	31:0	32	Start_rb_slot1[1]

Table 52. Receive Noise at the Demodulator 1

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	AvRecvNoiseImagPart[i]
1	15:0	16	AvRecvNoiserealPart[i]
…			…
Num rbs	31:16	16	AvRecvNoiseImagPart[Num rbs]
Num rbs	15:0	16	AvRecvNoiserealPart[Num rbs]

Table 53 shows this entry is only available if the Frequency Hopping is enabled on PUSCH.

For i= start rb sl1; i< (Num rbs); i+1

Table 53. Receive Noise at the Demodulator 2

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	AvRecvNoiseImagPart[i]
1	15:0	16	AvRecvNoiserealPart[i]
…			…
Num rbs	31:16	16	AvRecvNoiseImagPart[Num rbs]
Num rbs	15:0	16	AvRecvNoiserealPart[Num rbs]

Table 54 shows the instantaneous values of the complex error vector i.e. receive noise at the demodulator are provided below:

For i= start rb sl0; i< (Num rbs); i+1

Table 54. Receive Noise at the Demodulator 3

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	RecvNoiseImagPart[i]
1	15:0	16	RecvNoiserealPart[i]
…			…
Num rbs	31:16	16	RecvNoiseImagPart[Num rbs]
Num rbs	15:0	16	RecvNoiserealPart[Num rbs]

Table 55 shows this information is only provided if Frequency Hopping is enabled in the PUSCH.

For i= start rb sl1; i< (Num rbs); i+1

Table 55. Receive Noise at the Demodulator 4

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	RecvNoiseImagPart[i]
1	15:0	16	RecvNoiserealPart[i]
…			…
Num rbs	31:16	16	RecvNoiseImagPart[Num rbs]
Num rbs	15:0	16	RecvNoiserealPart[Num rbs]

12. PHY_RXSTATUS for Signal Level Measurements

This indication is issued by the PHY when the noiseDemodMeasEn is set by the L2 in the RXVECTOR. At the completion of the measurement the PHY issues a RX_STATUS message with the Status Type set to Signal Level.

Table 56. PHY_RXSTATUS for Signal Level Measurements

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	Number of entries
1	15:0	16	Channel ID
2	31:0	32	Status_len
3	31:0	32	Number of Receive Antennas
4	31:0	32	Start_rb_slot0[0]
5	31:0	32	Start_rb_slot0[1]
6	31:0	32	Num_rbs[0]
7	31:0	32	Num_rbs[1]
8	31:0	32	Start_rb_slot1[0]
9	31:0	32	Start_rb_slot1[1]

The signal level values are reported in dBFS in the range of 0 to 900 with 0 representing -90 dBFS and 900 0 dBFS, each step is 0.1 dBFS and is reported for each used RB for each RxAnt. There is one RXSTATUS message per each UE for the signal level indication.

So the following information appears as many Number of Receive Antennas are being used i.e. in a loop for ant_idx in the range of 0 to Number of Receive Antennas.

Table 57. Signal Level Report 1

First the average values are reported.

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	AvgSignalLevel[ant_idx][start_rb_slot0+1]
1	15:0	16	AvgSignalLevel[ant_idx][start_rb_slot0]
…			…
Num rbs[ant_idx]/2	31:16	16	AvgSignalLevel[ant_idx][Num_rbs]
Num rbs[ant_idx]/2	15:0	16	AvgSignalLevel[ant_idx][Num_rbs-1]

Table 58. Signal Level Report 2

This entry only appears if Frequency Hopping is enabled in PUSCH.

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	AvgSignalLevel[ant_idx][start_rb_slot1+1]
1	15:0	16	AvgSignalLevel[ant_idx][start_rb_slot1]
…			…
Num rbs[ant_idx]/2	31:16	16	AvgSignalLevel[ant_idx][Num_rbs]
Num rbs[ant_idx]/2	15:0	16	AvgSignalLevel[ant_idx][Num_rbs-1]

Table 59. Signal Level Report 3

Next the instantaneous Signal Levels are reported.

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	SignalLevel[ant_idx][start_rb_slot0+1]
1	15:0	16	SignalLevel[ant_idx][start_rb_slot0]
…			…
Num rbs[ant_idx]/2	31:16	16	SignalLevel[ant_idx][Num_rbs]
Num rbs[ant_idx]/2	15:0	16	SignalLevel[ant_idx][Num_rbs-1]

Table 60. Signal Level Report 4

The following report is only present if the Frequency Hopping is enabled in PUSCH.

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	SignalLevel[ant_idx][start_rb_slot0+1]
1	15:0	16	SignalLevel[ant_idx][start_rb_slot0]
…			…
Num rbs[ant_idx]/2	31:16	16	SignalLevel[ant_idx][Num_rbs]
Num rbs[ant_idx]/2	15:0	16	SignalLevel[ant_idx][Num_rbs-1]

13. PHY_RXSTATUS for Receive Noise Floor

This indication is issued by the PHY when the `noiseFloorMeasEn` is set in the `RXVECTOR` by the L2. At the completion of the measurement the PHY issues a `PHY_RXSTATUS` message with Status Type set to Noise Floor.

Table 61. PHY_RXSTATUS for Receive Noise Floor

32-bit Word #	Bits	Size, bits	Description
1	31:0	32	Number of Resource Blocks

The second part of this PHY_RXTSTATUS message for the Receive Noise Floor is illustrated below. The values are in U16 format and reported in dBFS. The range is 0 to 900 with 0 representing -90 dBFS and 900 representing 0 dBFS. For Resource Blocks that are being used for PUSCH a value of 65364 decimal or 0xFFFF hex is reported so the L2 can ignore the particular Resource Block.

Table 62. Receive Noise Floor 1

For i= 0; i< (Number of Resource Blocks)/2; i+2

32-bit Word #	Bits	Size, bits	Description
1	31:16	16	RxNoiseFloor_dBFS[i+1]
1	15:0	16	RxNoiseFloor_dBFS[i]

For bandwidths that have an odd number of Resource Blocks the last entry is padded with 16-bits.

14. PHY_RXSTATUS for Noise Demod Cell Wide

This indication is issued by the PHY when the noiseDemodCellWideReport is set in the RXVECTOR by the L2. At the completion of the measurement the PHY issues a PHY_RXSTATUS message with Status Type set to Noise Demod Cell Wide.

Table 63. Noise Demod Cell Wide

32-bit Word #	Bits	Size, bits	Description
0	31:16	16	Noise bin for (-inf,-120dbm]
0	15:0	16	Noise bin for (-120dbm,-118dbm]
1	31:16	16	Noise bin for (-118dbm,-116dbm]
1	15:0	16	Noise bin for (-116dbm,-114dbm]
2	31:16	16	Noise bin for (-114dbm,-112dbm]
2	15:0	16	Noise bin for (-112dbm,-110dbm]
3	31:16	16	Noise bin for (-110dbm,-108dbm]
3	15:0	16	Noise bin for (-108dbm,-106dbm]
4	31:16	16	Noise bin for (-106dbm,-104dbm]
4	15:0	16	Noise bin for (-104dbm,-102dbm]
5	31:16	16	Noise bin for (-102dbm,-100dbm]
5	15:0	16	Noise bin for (-100dbm,-98dbm]
6	31:16	16	Noise bin for (-98dbm,-96dbm]
6	15:0	16	Noise bin for (-96dbm,-94dbm]

32-bit Word #	Bits	Size, bits	Description
7	31:16	16	Noise bin for (-94dbm,-92dbm]
7	15:0	16	Noise bin for (-92dbm,-90dbm]
8	31:16	16	Noise bin for (-90dbm,inf]

4.3.18 PHY_TXPWRSTAT_IND

This indication is issued by the PHY when a `txpwrMeasEn` request is received from the L2 in `TXVECTOR`. After completing the measurement the PHY issues a `TXPWRSTAT` indication with the following format. This measurement is not currently supported by the current PHY code.

Table 64. PHY_TXPWRSTAT_IND

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 27
0	15:0	16	Reserved
1	31:24	8	Transmit Power Level being reported in db ∈ [0, -90] in two's complement notation
1	23:16	8	Subframe Number
1	15:0	16	System Frame Number

4.3.19 PHY_TXDCIULCATMSDU Request

This message is used for MPDCCH channel.

Table 65. PHY_TXDCIULCATMSDU Request

32-bit Word #	Bits	Size, bits	Description
0	31:16	16	msgLength (TXDCIULCATMSDU Length)
0	15:8	8	msgType (Message Type)
0	7:0	8	PHY entity ID (instance)
1	31:16	16	Channel ID
1	15:8	8	numPayloadBytes (Number of bytes in payload)
1	7:0	8	numBitsDciUL

32-bit Word #	Bits	Size, bits	Description
2	31:24	8	setNum (Either 0 or 1)
2	23:18	6	localizedPortIndex
2	17:16	2	distributedAlloc if set
2	15:0	8	txPowerControl. This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. Depending on the type of channel the value in txpowerControl can be a relative value to the demodulation reference signal power for the associated MPDCCH channel type. -32768 to 1536 (-128 to 6 dB)
3	31:16	16	rnti
3	15:8	8	startRB
3	7:0	8	startCCE
4	31:24	8	nRbXm
4	23:16	8	numCCE
4	15:0	16	Reserved
5	31:16	16	Dmrs_txpowerControl
5	15:0	16	Reserved1
6	31:0	32	scramblerInit (for mpdcch setNum)
7	31:0	32	demodRSforMpdcchInitValue
8	31:0	32	Pointer to DCI UL Payload (type U32)

4.4 Batch API Delivery of Messages

The Batch API delivery of messages between the L2/L3 and the PHY reduces the overhead associated with the delivery of individual messages and allows multiple API messages to be exchanged using a Shared Memory Interface (SHM). The full description of this scheme is presented in Section 4.6.32 ULSubChannelDescriptorCatm.

4.4.1 PHY_BATCH_MSG Request

This primitive is issued by the L2/L3 to use the LTE API batch message delivery scheme described in Section 4.6.32 ULSubChannelDescriptorCatm for the messages from the L2/L3 to the PHY.

4.4.2 PHY_BATCH_MSG Indication

This primitive is issued by the PHY to use the LTE API batch message delivery scheme described in Section 4.6.32 ULSubChannelDescriptorCatm for the messages from the PHY to the L2/L3.

4.5 Error Indication

The following error indication message is issued from the PHY in the event that APIs are being delivered late by L2/L3 or under other conditions as stated in Table 7. .....Error Codes.

4.5.1 PHY_ERROR _IND

Table 66. PHY_ERROR.Indication

32-bit Word #	Bits	Size, bits	Description
0	31:24	8	PHY entity ID (Instance)
0	23:16	8	Message Type = 25
0	15:8	8	Error Code from Section 4.1.2 Error Field Coding
0	7:0	8	Sub-frame number from TXVECTOR
1	31:16	16	System Frame Number
1	15:0	16	Reserved

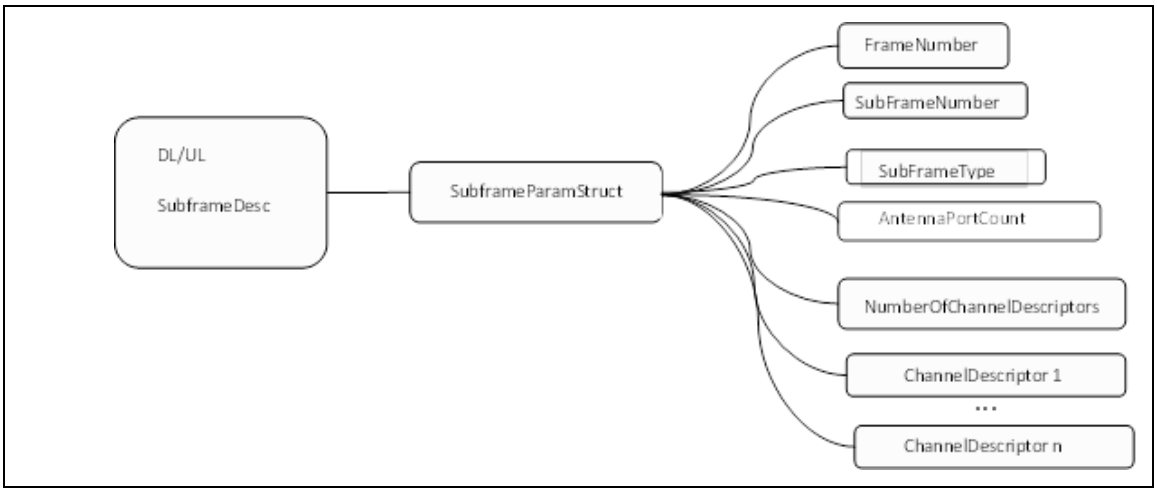
4.6 Frame Descriptors Hierarchy

This section describes from a top level the structures used in this API to convey the control plane information.

Since LTE resource scheduling is performed every sub-frame (1 ms) interval the descriptors for the Downlink and Uplink path are defined in terms of sub-frames with one entry that allows the unique identification of the sub-frame in terms of the overall frame (10 ms).

We use TXVECTOR to describe the Downlink sub-frame structure and RXVECTOR to describe the Uplink sub-frame structure.

Figure 45. Frame Descriptor Hierarchy



The cell specific reference signals are not included since they are always sent in the first slot of every sub-frame at predefined subcarriers and symbols that only require the Number of Antennas information and the cell-id (which is part of the `PHY_INIT` command) to uniquely identify the position. Also the primary and secondary synchronization signals are not included in the `TXVECTOR` and `RXVECTOR` since they also always occur in well defined resource elements of slot 0 for a given system Bandwidth.

Figure 46. DL TxVector Example

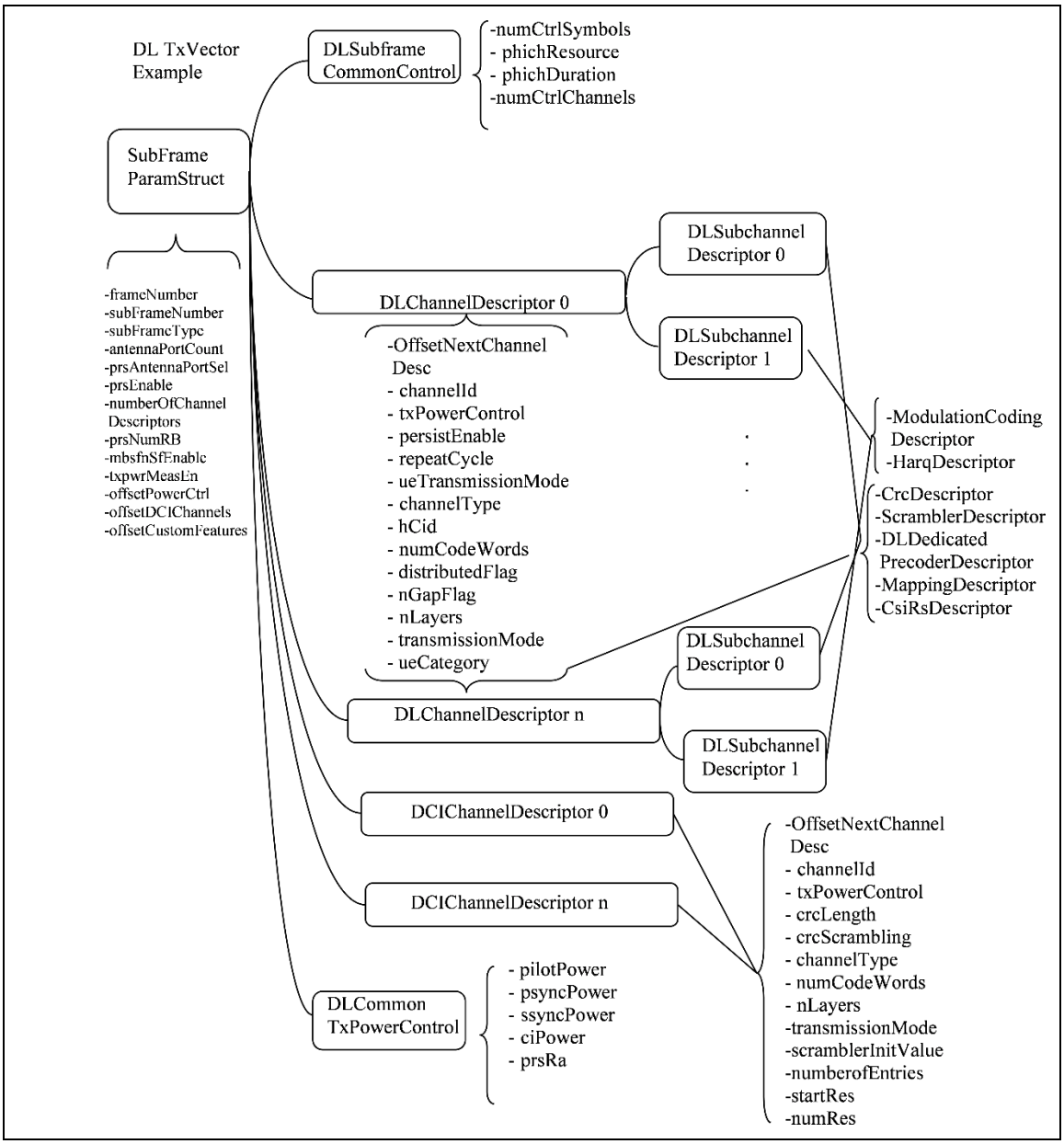
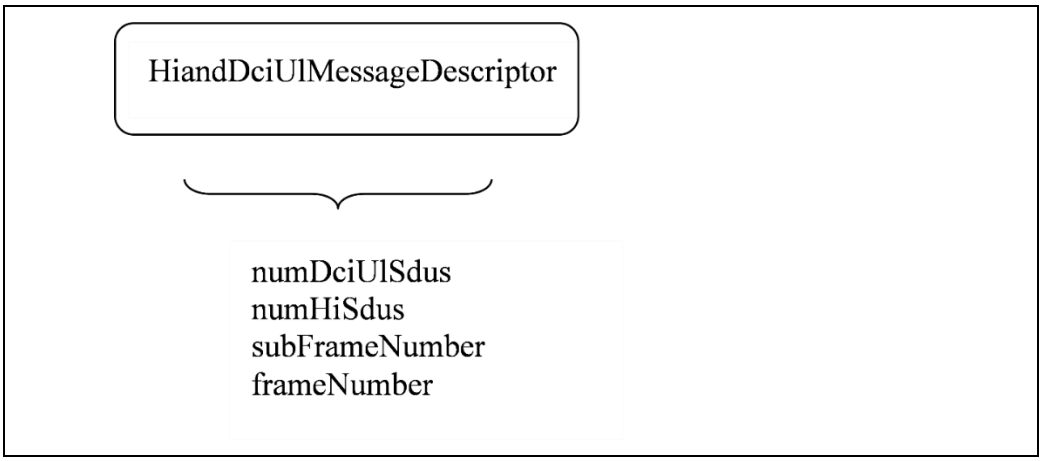


Figure 47. HiandDciUlMessageDescriptor



4.6.1 Downlink Transmitter Descriptor (TXVECTOR)

For the eNodeB `TXVECTOR` has `SubFrameType` 0, and the only permissible channel types are 0 through 4 (that is: DCI, DL-SCH, BCH, CFI and HI and in future releases 9 and 10 also which are assigned for PMCH and EPDCCH respectively).

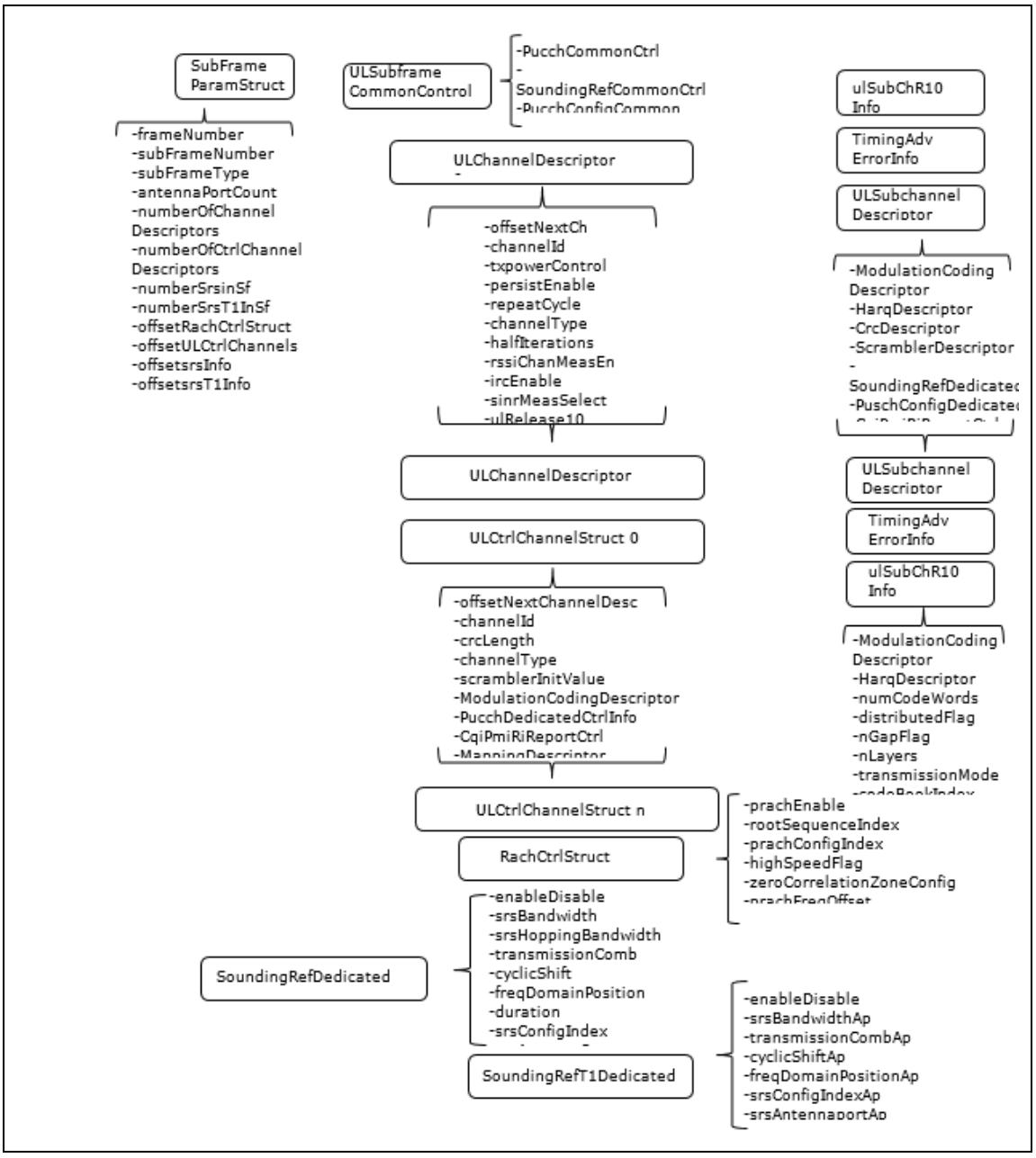
4.6.2 Uplink Receiver Descriptor (RXVECTOR)

For the eNodeB `RXVECTOR` has SubFrameType 1, and the only permissible channel types are 5 through 7 (that is, UL-SCH, RACH and UCI).

4.6.3 TxHarqDCIUlVector (TXHIADCIUL.request)

This is an optional addition to the `TXVECTOR` that is used to convey just in time HARQ and DCI control information for schedule grants.

Figure 48. UL RxVector Example



4.6.4 Parameter Structure Formats

The parameter structure formats are done on the basis of a Little Endian processor, so the entries are going from Least Significant Byte to Most Significant Byte as described in the following sections.

`BitField` entries always must align to 32-bit boundaries.

4.6.5 SubframeParamStruct

The SubframeParamStruct defines all the parameters for each of the expected TB's to be processed in the subframe. There are two different structures defined at the BS: one for the DL and a second one for the UL. The first eight bytes of these structures are common and the PHY_TXSTART.request and PHY_RXSTART.request commands contains the overall length of the corresponding TXVECTOR or RXVECTOR.

The possibility of customer defined parameters is considered here by using entries defining offsets from the beginning of the structure to the customer defined area, since the offset is always from the beginning of the structure an offset of zero means that the feature or optional entry is not present.

Table 67 lists the values of the general form of a SubframeParamStruct.

Table 67. SubframeParamStruct Values

Parameter	Length, bits	Description, value	Reference
frameNumber	16	10 bits are actually used In the BaseStation Transmitter the L2 is the master for the SystemFrameNumber which is written in this entry. The SFN is right justified with the MSB's zero filled frame Number ∈ {0, .., 1023}	[36.331] Section 6.2.2
subFrameNumber	8	In the BaseStation Transmitter the L2 is the master for the subFrameNumber that is written here. The subFrameNumber is right justified with the MSB's zero filled subframe Number ∈{0,..,9}	[36.331] Section 6.2.2
subFrameType	8	0 = Downlink Transmit 1 = Uplink Receive 2 = Uplink Transmit 3 = Downlink Receive	
antennaPortCount	8	1, 2, 4	[36.331] Section 6.3.2
numberOfChannel Descriptors	8	Indicates the number of channel descriptors being used in the sub-frame	
reserved	16	Padding to allow 32 bit alignment for Common structures	
Offset to DL or UL Control/Optional entries	Depends on DL or UL case	See specific DL and UL information in the next sections	

Parameter	Length, bits	Description, value	Reference
ChannelDescriptor Entries	Variable	See specific DL and UL information in the next sections. This information is channel type context sensitive. See the examples given for each channel type for the specific fields applicable to each one.	
OptionalControl Entries	Variable	See specific DL and UL information in the next sections	
OptionalCustomer Entries	Variable		

4.6.6 DLSubframeParamStruct

Table 68 lists the description values and details of the `DLSubframeParamStruct`.

Table 68. DLSubframeParamStruct Description and Values

Parameter	Length, bits	Description, value	Reference
frameNumber	16	10 bits are actually used In the BaseStation Transmitter the L2 is the master for the SystemFrameNumber which is written in this entry. The SFN is right justified with the MSB's zero filled frameNumber ∈ {0, .., 1023}	[36.331] Section 6.2.2
subFrameNumber	8	In the BaseStation Transmitter the L2 is the master for the subFrameNumber that is written here. The subFrameNumber is right justified with the MSB's zero filled subFrameNumber ∈{0,..,9}	[36.331] Section 6.2.2
subFrameType	8	0 = Downlink Transmit 1 = Uplink Receive 2 = Uplink Transmit 3 = Downlink Receiver	
antennaPortCount	5	1, 2, 4	[36.331] Section 6.3.2
prsAntennaPortSel	2	0 = No prs being used 1 = prs on Antenna 0 2 = prs on Antenna 1 3 = reserved Selects the antenna(s) where the Positioning Reference Signals are transmitted	[36.211] Section 6.2.1

Parameter	Length, bits	Description, value	Reference
prsEnable	1	0 no positioning reference signals are generated in this subframe 1 prs is generated in this subframe	[36.211] Section 6.10.4
numberOfChannel Descriptors	8	Indicates the total number of channel descriptors being used on the sub-frame.	
prsNumRB	8	Bandwidth available for prs generation	[36.211] Section 6.10.4
mbsfnSfEnable	1	MBSFN Subframe enable when set indicates to the PHY to generate an MBSFN subframe	
txpwrMeasEn	1	When set enables the Tx Power Measurement	[36.214] Section 5.2.1
lprime_mpdcch_start	6	First symbol that mpdcch will appear.	
offsetPowerCtrl	32	Offset from the beginning of the structure to the DLComonTxPowerControl structure.Since this entry is optional a value of zero means that this entry is absent for this subframe	
offsetDCIChannels	32	Offset from the beginning of the structure to the DCIChannels structures	
offsetCustom Features	32	Offset from the beginning of the structure to CustomFeatures section. An offset of zero means that the entry is absent	
.DLSubframe CommonControl Structure	32	See 4.6.7 DLSubframeCommonControlStructure for more details	
DLChannel Descriptors	1120*ndlch	See **Error! Reference source not found. Error! Reference source not found.** for more details	
DciChannel Descriptors	160*ndcich	See **Error! Reference source not found. Error! Reference source not found.** for more details	
DLCommonTx PowerControl	96	Optional entry depending on the value of the offset entry referenced above. See 4.6.8 DLCommonTxPowerControl for more details	

Parameter	Length, bits	Description, value	Reference
csirsInfo	32	See 4.6.24 CsiRsDescriptorStruct	
CustomFeatures	User Defined	Optional entry depending on the value of the offset entry above	

4.6.7 DLSubframeCommonControlStructure

Table 69. DLSubframeCommonControlStructure

Parameter	Length, bits	Description, value	Reference
numCtrlSymbols	3	This is the number of symbols in the subframe used for control information. The possible values are: 1, 2 or 3 for any Bandwidth except 1.4 MHz that uses 2, 3 or 4 (Corresponds to the payload for the PCFICH) In the case of an MBSFN subframe this field is also defined as non-MBSFNRegionLength in the [36.331] MBSFN-AreaInfoList Information Element	[36.211] Section 6.7
phichResource	2	This is the size of the PHICH allocation with the following possible values: 0 = 1/6 1 = 1/2 2 = 1 3 = 2	N_g per [36.211] Section 6.9
phichDuration	3	This is the number of symbols that can be used for PHICH. Possible values are: 1 = Normal 2 or 3 = Extended	[36.211] Table 6.9.3-1
Pmchstart	2	This is the PMCH start symbol in MBSFN subframe which is given by higher layer parameter *non-MBSFNregionLength*. The possible values are: *1 = PMCH is start from symbol 1.* *2 = PMCH is start from symbol 2.*	[36.331] Section 6.3.7
reserved1	6		
numCtrlChannels	16	This is the number of Control channels (DCI) present in this subframe and described in the TXVECTOR	

4.6.8 DLCommonTxPowerControl

Table 70. DLCommonTxPowerControl

Parameter	Length, bits	Description, value	Reference
pilotPower	16	Pilot signal Power Control 2's complement notation in Q8 dB steps (Range -128 to 6 dBr). Where 0 dBr is equal to a 2's complement rms level of 1 in the IQ samples of each antenna. Overall constraints for the system that need to be followed when changing the pilotPower are stated under psyncPower, ssyncPower, ciPower, prsRa and Section 4.6.8 under txPowerControl . Note: The IQ samples are stored as signed 16-bits I and 16-bits Q. So, if the fixed point format has 12 fractional bits (1:3:12)	
psyncPower	16	Primary Sync Power Control 2's complement notation in Q8 dB steps (Range -128 to 6 dB relative to pilotPower subject to pilotPower + psyncPower <= 6 dB)	
ssyncPower	16	Secondary Sync Power Control 2's complement notation in Q8 dB steps (Range -128 to 6 dB relative to pilotPower subject to pilotPower + ssyncPower <= 6 dB)	
ciPower	16	Control Indicator Power Control(PCFICH) 2's complement notation in Q8 dB steps (Range -128 to 6 dB relative to pilotPower subject to pilotPower + ciPower <= 6 dB)	
prsRa	16	Positioning Reference Signal Power ratio relative to Cell specific reference signal power. Control 2's complement notation in Q8 dB steps (Range -128 to 6 dB relative to pilotPower subject to pilotPower + prsRa <= 6 dB)	
Reserved	16	Zero filled	

4.6.9 DLChannelDescriptorStruct

Table 71. DLChannelDescriptorStruct

Parameter	Length, bits	Description, value	Reference
OffsetNext ChannelDesc	32	Offset to next channel descriptor from the beginning of this structure (bytes). An offset of 4 means that this is the last ChannelDesc. This entry allows customers to expand the ChannelDesc beyond what is currently defined in this document	
channelId	16	Uniquely identifies this channel among others of the same type and also allows a one to one map of TXSDU's and Resource element maps associated with this channel	

Parameter	Length, bits	Description, value	Reference
txPowerControl	16	This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. Depending on the type of channel the value in txpowerControl can be a relative value to the reference signal power for a PDSCH channel type. -32768 to 1536 (-128 to 6 dB). The value used for txPowerControl is subject to the constraint txPowerControl + pilotPower <= T dB. For RBs without UE specific RS, the ratio of PDSCH to pilot signal power depends on the PB, antenna port, CP type according to Table 5.2-2 in [36.213]. txPowerControl can be treated as ρ_A. And Phy can determine ρ_B according to P_B configuration. For the RBs containing UE specific RS, the ratio of PDSCH to pilot signal power is kept constant as txPowerControl for all the OFDM symbols in one subframe. Where T is 6 dB, respectively for single antenna port. For the multiantenna case the value of T is subject to the following constraints: When P_B is zero T=2 dB otherwise T=3 dB for PDSCH. For all other channel types T=6 dB regardless of the number of antennas.	[36.104] Section 6.3.1.1 for PDSCH, PDCCH [36.213] Section 5.2
persistEnable	1	Enable/Disable Persistent Resource allocation using repeatCycle 0 = Disable 1 = Enable This is not supported. Please always set this bit to 1	[36.321] Section 5.10.1
repeatCycle	2	This field is not used and is not supported.	[36.321] Section 5.10.1 [36.331] See SPS-Config
catMenabled	1	Indicates that catM is used in this PDSCH channel	

Parameter	Length, bits	Description, value	Reference
ueTransmissionMode	4	ueTransmissionMode chosen at RRCConnectionSetup or at RRCConnectionReestablishment or at RRCConnectionReconfiguration 1 = Single-antenna port; port 0 2 = Open-loopTransmit diversity 3 = Open-loop spatial multiplexing 4 = Closed-loop spatial multiplexing 5 = Multi-user MIMO 6 = Closed-loop Rank=1 precoding) 7 = Single-antenna port; port 5 8 = Dual Layer Beamforming 9 = Eight Layer Beamforming 10 = Eight Layer Beamforming Rel 11(not supported, placeholder only)	
channelType	4	0 = PDCCH 1 = PDSCH 2 = PBCH 3 = PCFICH (Reserved since the payload for this channel type is passed in the DLSubframeCommonControl structure) 4 = PHICH 5-7= Reserved 8 = Reserved 9 = PMCH 10 = EPDCCH 11 = MPDCCH 12-15 = Reserved Right justified and zero padded	
hCid	4	HARQ Process Number	[36.212] Section 5.3.3.1.2
numCodeWords	2	Specifies the number of codewords used in this channel. 1 = single layer and for Transmit Diversity 2 = Spatial Diversity Right justified and zero padded	
distributedFlag	1	0 = Resource Allocation of localized type 1 = Resource Allocation of distributed type	
nGapFlag	1	0 = Gap 1 1 = Gap 2 Only used if distributedFlag is set	
nLayers	4	Specifies the number of layers used by the channel. 1 = no diversity (Single Antenna mode) 2 and above = Transmit Diversity and Spatial Diversity schemes Right justified and zero padded	

Parameter	Length, bits	Description, value	Reference
transmissionMode	4	The following transmission modes are possible on a channel basis 1 = Single-antenna port; port 0 2 = Open-loopTransmit diversity 3 = Open-loop spatial multiplexing 4 = Closed-loop spatial multiplexing Right justified and zero padded	
ueCategory	4	ueCategory Integer ∈{[1,...,8] For 3GPP release 10 and later categories 6-8 else 5 is the upper bound for this field	[36.306] Section 4.1
SubchannelDescriptor	64	Descriptors for subchannels 0 and 1	
crcDescriptor	8	See 4.6.20 CrcDescriptorStruct	
scramblerDescriptor	24	See 4.6.21 ScramblerDescriptorStruct	
DLDedicatedPrecoderDescriptor	96	See 4.6.22 DLDedicatedPrecoderDescriptorStruct	
MappingDescriptor	832	See 4.6.23 MappingDescriptorStruct	
catMScramblerInitValue	32	This field Is initialized by the L2 when catMenabled is true whether repetitions are enabled or not.	

Depending on the `channelType` and the `numCodeWords` used there are as many `DLSubChannelDescriptors` as `CodeWords` being used, and the contents of the `DLSubChannelDescriptors` varies with the `channelType` as explained in the next sections.

4.6.10 DLSubChannelDescriptorStruct

Each `SubChannelDescriptor` has three elements. Table 72 lists these three elements.

Table 72. DLSubChannelDescriptorStruct

Parameter	Length, bits	Description, value
ModulationCodingDescriptor	16	Refer to Section 4.6.18 ModulationCodingDescriptorStruct
HarqDescriptor	8	Refer to Section 4.6.19 HarqDescriptorStruct
Pad	8	

4.6.11 DciChannelDescriptorStruct

Table 73. DciChannelDescriptorStruct

Parameter	Length, bits	Description, value	Reference
OffsetNext ChannelDesc	32	Offset to next channel descriptor from the beginning of this structure (bytes). An offset of 4 means that this is the last ChannelDesc. This entry allows customers to expand the DLDciChannelDesc beyond what is currently defined in this document	
channelId	16	Uniquely identifies this channel among others of the same type and also allows a one to one map of TXSDU's and Resource element maps associated with this channel	
txPowerControl	16	This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. -32768 to 1536 (-128 to 6 dB) The value used for txPowerControl is subject to the constraint txPowerControl + pilotPower <= 6 dB.	
crcLength	6	0 = None 16 = 16 bits	[36.212] Section 5.3.3.2
crcScrambling	2	0 = Disabled 1 = RNTI only 2 = UE Port 0 and RNTI 3 = UE Port 1 and RNTI	[36.212] Section 5.3.3.2
channelType	5	0 = DCI (goes to PDCCH) 10 = DCI (goes to EPDCCH) 11 = DCI (goes to MPDCCH)	
Reserved	3	Zero filled	
numCodeWords	4		
nLayers	4	Specifies the number of layers used by the channel. 1 = no diversity (Single Antenna mode) 2 and above = Transmit Diversity Right justified and zero padded Integer {1,..,4}	
transmissionMode	8	The following transmission modes are possible on a channel basis 1 = Single-antenna port; port 0 2 = Transmit diversity Right justified and zero padded	
catMinUse	1	This is required only if CatM is enabled and channelType is MPDCCH otherwise it is a don't care If enabled use catMmappingDescriptor	
distributedAlloc	2	Indicates if distributed allocation is used for MPDCCH	
localizedAntPortIndex	4	Index for 36.211 Table 6.8A.5-1 when localized allocation is used for MPDCCH	

Reserved1	9	Zero filled	
scrmblerInitValue	16	Scrambler Initialization value. If the dci is going to a specific UE use here the RNTI value else the PHY uses the information provided by the L2/L3 during initialization for the Ncell ID. For MPDCCH the value(s) need to be initialized in the MpdcchInfo entry	[36.211] Section 6.8.2
numberofEntries	16	Defines how many pairs of startRes and numRes are present for the current mapping. For PDCCH, it is one. For MPDCCH, it can be 1 or 2 depending on number of sets used with the MPDCCH.	
startRes	8	This is the starting block for an allocation. For the DCI channel this is the starting CCE.	
numRes	8	This input specifies how many consecutive CCE's are used starting at startRes.	
drms_txpowerControl	16	This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. -32768 to 1536 (-128 to 6 dB)	
Reserved2	16	Zero Filled	
MpdcchInfo[2]	192	This entry specifies StartCCE, NumCCE, StartRB and NumRbXm , demodRSforMpdcchInitValue for the 2 possible sets associated with an MPDCCH	

4.6.12 ULSubframeParamStruct

Table 74. ULSubframeParamStruct

Parameter	Length, bits	Description, value	Reference
frameNumber	16	10 bits are actually used In the BaseStation receiver the L2 is the master for the SystemFrameNumber which is written in this entry. The SFN is right justified with the MSB's zero filled. frameNumber ∈ {0, .., 1023}	[36.331] Section 6.2.2
subFrameNumber	8	In the BaseStation Transmitter the L2 is the master for the subFrameNumber that is written here. The subFrameNumber is right justified with the MSB's zero filled. subFrameNumber ∈{0,..,9}	[36.331] Section 6.2.2
subFrameType	8	0 = Downlink Transmit 1 = Uplink Receive 2 = Uplink Transmit 3 = Downlink Receive	
antennaPortCount	8	1, 2, 4	[36.331] Section 6.3.2
numberOfChannel Descriptors	8	Indicates the total number of channel descriptors being used in the sub-frame. (Shared Data and Uplink Control info)	

Parameter	Length, bits	Description, value	Reference
numberOfCtrl ChannelDescriptors	8	Indicates the number of Uplink Control Channel descriptors used in the sub-frame (PUCCH's)	
numberSrsinSf	8	Number of Sounding References expected in this subframe whose parameters are specified in the srsInfo descriptors. For 3GPP release 10 this number reflects the SRS trigger type 0 number.	
numberSrsT1inSf	8	Number of Sounding References T1 expected in this subframe.	3GPP release 10
Pad	24		3GPP release 10
offsetRachCtrlStruct	32	Offset from the beginning of the structure to the RachCtrl structure. Since this entry is optional a value of zero means that this entry is absent for this subframe.	
offsetULCtrl Channels	32	Offset from the beginning of the structure to the ULCtrlChannels structures.	
offsetsrsInfo	32	Offset from the beginning of the structure to the srsInfo structures.	
offsetsrsT1Info	32	Offset from the beginning of the structure to the srsT1Info structures.	
offsetCustom Features	32	Offset from the beginning of the structure to CustomFeatures section. An offset of zero means that the entry is absent.	
ULSubframe CommonControl Struct	96	See definition 4.6.13 ULSubframeCommonCtrlStruct	
ULChannel Descriptors	1216*nulch	See definitions 4.6.14 ULChannelDescriptorStruct.	
ULCtrlChannel Descriptors	288*nulctrlch	See definitions 4.6.17 ULCtrlChannelStruct.	
srsInfo dedicated descriptors	32*numsrs	For each descriptor see 4.6.26 SoundingRefDedicatedCtrlStruct.	
srsT1Info dedicated descriptors	32* numberSrsT1inSf	For each descriptor see 4.6.27 SoundingRefT1DedicatedCtrlStruct.	3GPP release 10
RachCtrlStruct	32	Optional entry depending on the value of the offset entry referenced see **Error! Reference source not found. Error! Reference source not found.**.	
CustomFeatures	User Defined	Optional entry depending on the value of the offset entry above.	

4.6.13 ULSubframeCommonCtrlStruct

This structure includes `PucchCommonControl`, `SoundingReferenceCommonControl`, `PuschConfigurationCommon` and `Measurements` control information.

Table 75. ULSubframeCommonCtrlStruct

Parameter	Length, bits	Description, value	Reference
deltaPUCCHShift	2	Parameter: Δ^{PUCCH}_{shift} Integer $\in$ {1, 2, 3}	[36.211] Section 5.4.1.

Parameter	Length, bits	Description, value	Reference
nRBCQI	7	Parameter: $N^{(2)}_{RB}$ Bandwidth in terms of resource blocks that are available for use by PUCCH formats 2/2a/2b transmission in each slot Integer ∈ {0, .. , 98}	[36.211] Section 5.4.
nCSAn	3	Parameter: $N^{(1)}_{cs}$ Number of cyclic shifts used for PUCCH formats 1/1a/1b in a resource block used for a mix of formats 1/1a/1b and 2/2a/2b Integer ∈ {0, .. ,7}	[36.211] Section 5.4.
n1PucchAN	11	Parameter: $N^{(1)}_{PUCCH}$ Determines the PUCCH resource to be used for transmission of HARQ-ACK Integer ∈ {0, .. , 2047}	[36.213] Section 10.1.
rxIntMeasEn	1	When set it enables the Received Interference Power Measurement in the current subframe	[36.214] Section 5.2.2.
rxTherNoiseEn	1	When set it enables the Thermal Noise Power Measurement in the current subframe	[36.214] Section 5.2.3
rssiMeasEn	1	When set it enables the RSSI measurement reported in RxStatus in the current subframe	
noiseFloorMeasEn	1	When set it enables the measurement of the noise floor	
noiseDemodCellWide Report	1	When set, it enables noise demod cell wide report to be sent to L2.	
srsMaxUpPts	1	When set enables reconfiguration of maximum value of $m_{srs,0}$ When clear disables reconfiguration of maximum value of $m_{srs,0}$ This parameter is only valid in TDD mode and it isnot used in FDD mode	[36.211] Section 5.5.3.2
Pad	3	Zero filled	
srsBandwidthConfig	3	Parameter: SRS Bandwidth Configuration. 0,1,2,3,4,5, 6 or 7 Actual configuration depends on UL bandwidth.	[36.211] Tables 5.5.3.2-1, 5.5.3.2-2, 5.5.3.2-3 and 5.5.3.2– 4.
srsSubframeConfig	4	Parameter: SRS SubframeConfiguration. 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14 or 15.	[36.211], Table 5.5.3.3-1 applies for FDD [36.211], Table 5.5.3.3-2 applies for TDD
ackNackSRSSimultaneousTransmission	1	Parameter: Simultaneous-AN-and-SRS	[36.213] Section 8.2.
nSB	2	Parameter: N_{sb} Number of subbands Integer ∈ {1, 2, 3, 4}	[36.211] Section 5.3.4.
hoppingMode	1	Parameter: Hopping-mode Value 0 means interSubFrame Value 1 means intraAndIntersubFrame	[36.211] Section 5.3.4.
puschhoppingOffset	7	Parameter: N^{HO}_{RB} Expressed in RB's Integer ∈ {0, .. , 98}	[36.211] Section 5.3.4.
enable64QAM	1	TRUE indicates that 64QAM is allowed FALSE indicates that 64QAM is not allowed.	[36.213] Section 8.6.1.

Parameter	Length, bits	Description, value	Reference
groupHoppingEnabled	1	Parameter: Group-hopping-enabled This applies to the Pusch Reference signals.	[36.211] Section 5.5.1.3.
groupAssignmentPUSCH	5	Parameter: Δ_{SS} Integer $\in$ {0, .. , 29}. This applies to the Pusch Reference signals.	[36.211] Section 5.5.1.3.
sequenceHopping Enabled	1	Parameter: Sequence-hopping-enabled Value of 1 means enabled else disabled. This applies to the Pusch Reference signals.	[36.211] Section 5.5.1.4.
cyclicShift	3	Parameter: cyclicShift Integer $\in$ {0, .. ,7} This applies to the Pusch Reference signals.	[36.211] Table 5.5.2.1.1-2.
Padding	3	Reserved	
noiseDemodAverScale	8	Scale factor for averaging of the noise demod cell wide report. Scale factor is a number between 0 and 255. The average NoiseM(n)= alpha*NoiseM(n-1) + beta*InstantNoise where alpha=noiseDemodAverScale/256 beta= 1-alpha	
padding1	24	Reserved	

4.6.14 ULChannelDescriptorStruct

Table 76. ULChannelDescriptorStruct

Parameter	Length, bits	Description, value	Reference
OffsetNext ChannelDesc	32	Offset to next channel descriptor from the beginning of this structure (bytes). An offset of 4 means that this is the last ULChannelDesc. This entry allows customers to expand the ULChannelDesc beyond what is currently defined in this document	
channelId	16	Uniquely identifies this channel among others of the same type and also allows a one to one map of RxSDU's and Resource element maps associated with this channel	
txPowerControl	16	This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. Depending on the type of channel the value in txpowerControl can be a relative value to the reference signal power. -32768 to 1536 (-128 to 6 dB)	[36.213] Section 5.1.1 for PUSCH and [36.213] Section 5.1.2 for PUCCH
persistEnable	1	Enable/Disable Persistent Resource allocation using repeatCycle 0 Disable 1 Enable Currently not supported by PHY don't care field	[36.321] Section 5.10.2

Parameter	Length, bits	Description, value	Reference
repeatCycle	7	If persistEnable is not zero then repeatCycle defines how often in sub-frames the current allocation re-occurs. For example a repeatCycle of 0 means that every 10 sub-frames we have the same resource allocation mapping for this channel and the L2 no longer needs to provide this channel information in the RxVector, only include it in the overall channel count. The PHY keeps track of the initial time that the enable took place and then stores the channel configuration on its internal database and uses the repeatCycle to determine when the same allocation reoccurs. To disable issue persistEnable with a value of 0 to this Channel repeatCycle ∈{10, 20, 32, 40, 64, 80,128, 160,320, 640} Mapped as {0, 1 , 2, 3, 4, 5, 6, 7, 8, 9} respectively Right justified and zero padded Currently not supported by PHY don't care field	[36.321] Section 5.10.2 [36.331] See SPS-Config
channelType	8	0 – 4 Reserved 5 = PUSCH 6 = PRACH 7= PUCCH	
halfIterations	8	Number of halfiterations used by the FEC Decoder. The default value is 16 and the valid range is {8,,9,10,11,12,13,14,15,16}	
rssiChanMeasEn	1	Enable rssi measurements for this channel only. The actual value is reported in the RXSDU.indication 0: Disable 1:Enable	
noiseDemodMeasEn	1	If set enables the measurement of noise at the Demodulator. This is calculated by finding the error between the received point and the closest ideal constellation point.	
sigLevelMeasEn	1	When set enables the Signal Level Measurement. The received signal level is calculated at each used resource block for each RxAnt. Both averaged and instantaneous values are returned for each RB for each attached UE. So the average values are UE specific. There are PUCCH and PUSCH independent controls for this measurement.	
ircEnable	1	When the PHY_INIT command has enabled the IRC global enable bit and this bit is set, the PUSCH receiver uses the IRC algorithm. When this bit is clear or the PHY_INIT IRC global enable bit is clear, the PUSCH receiver uses the MRC algorithm	

Parameter	Length, bits	Description, value	Reference
sinrMeasSelect	1	This bit allows the selection of where the sinr measurement takes place in the PUSCH receiver. When clear the sinr is measured before the equalizer and a value of one means that the sinr measurement is done after the equalizer.	
ulRelease10	1	If this bit is set then there is an additional ULSubChannelDescriptorR10 after the TimingAdvErrorInfo if UL MIMO support is present in the PHY If the bit is clear then no additional ULSubChannelDescriptorR10 is present Always zero for 3GPP pre-release 10 PHY versions	
ulCatmEn	1	This bit is set when this channel is a cat M channel and the L2 needs to write the information in the ulSubChCatmInfo. Otherwise, the ulSubChCatmInfo is a don't care.	
reserved	1	Zero filled	
noiseDemodAverScale	8	Scale factor for averaging of the noise at the PUSCH Demodulator. Scale factor is a number between 0 and 255. The average NoiseM(n)= alpha*NoiseM(n-1) + beta*InstantNoise where alpha=noiseDemodAverScale/256 beta= 1-alpha	
sigLevAverScale	8	Scale factor for averaging of the PUSCH Signal Level. sigLevAverValue(n)= alpha*sigLevAverValue(n-1) + beta*sigLevInstValue where alpha=sigLevAverScale/256 beta= 1-alpha	
mu_mimo	4	Set to non-zero value if this UE is doing Uplink Mu-Mimo. It must be paired with the other UE using the same resource blocks for Mu-Mimo. For example, if UE 0 and UE 1 are doing Mu-Mimo and are taking the same resource blocks, this field should have the same non-zero value.	
altCQITableUsed	1	Set to 1 to allow Uplink to use alternative CQI Table (3GPP 36.213, Table 8.6.1-3) to enable Qam256 demodulation.	
Padding	11		
ULSubChannelDescriptor	1024	Refer to Section 4.6.14 ULChannelDescriptorStruct.	
TimingAdvErrorInfo	96	Refer to Section 0 TimAdvErrInfCtrlStruct.	

Parameter	Length, bits	Description, value	Reference
ULSubChannelDescriptorR10	64	Refer to Section 4.6.16 ULSubChannelDescriptorR10.	
ulSubChCatmInfo	64	Refer to Section 4.6.32 ULSubChannelDescriptorCatm	

Currently, the persistent Resource allocation support at the PHY is not supported so the `persistEnable` and `repeatCycle` fields are `don't care`.

4.6.15 ULSubChannelDescriptorStruct

Table 77. ULSubChannelDescriptorStruct

Parameter	Length, bits	Description, value
ModulationCodingDescriptor	16	See 4.6.18 ModulationCodingDescriptorStruct for more details
HarqDescriptor	8	See 4.6.19 HarqDescriptorStruct for more details
CrcDescriptor	8	See 4.6.20 CrcDescriptorStruct for more details
ScramblerDescriptor	24	See 4.6.21 ScramblerDescriptorStruct for more details
Reserved	8	Zero filled
PuschConfigDedicated	64	See 4.6.28 PuschConfigDedicatedStruct for more details
CqiPmiRiReportCtrlStruct	64	See 4.6.30 CqiPmiRiReportCtrlStruct for more details
MappingDescriptor	832	See 4.6.23 MappingDescriptorStruct for more details

4.6.16 ULSubChannelDescriptorR10

Table 78. ULSubChannelDescriptorR10

Parameter	Length, bits	Description, value
ModulationCodingDescriptor1	16	See 4.6.18 ModulationCodingDescriptorStruct for more details
HarqDescriptor1	8	See 4.6.19 HarqDescriptorStruct for more details
Reserved	8	
numCodeWords	2	Number of code words 1 or 2
distributedFlag	1	0 Resource allocation of localized type 1 Resource allocation of distributed type

Parameter	Length, bits	Description, value
nGapFlag	1	0 Gap 1 1 Gap 2 Only valid if distributedFlag is set else it is a don't care
nLayers	2	Number of Layers minus 1.
transmissionMode	1	0 Single Antenna Port 10 1 Closed Loop Spatial Multiplexing
reserved1	1	Zero Filled
codeBookIndex	6	Codebook Index to be used for precoding Valid only when 2 codewords are being used 2 antenna ports ∈ [0,..,7] 4 antenna ports ∈ [0,..,63]
reserved2	18	Zero filled

4.6.17 ULCtrlChannelStruct

Table 79. ULCtrlChannelStruct

Parameter	Length, bits	Description, value	Reference
OffsetNext ChannelDesc	32	Offset to next channel descriptor from the beginning of this structure (bytes). An offset of 4 means that this is the last ULCtrlChannelDesc. This entry allows customers to expand the ULCtrlChannelStruct beyond what is currently defined in this document.	
channelId	16	Uniquely identifies this channel among others of the same type and also allows a one to one map of TXSDU's and Resource element maps associated with this channel.	
txPowerCtl	16	This field allows for power to be adjusted in steps of 1/256 dB using a two's complement notation. Depending on the type of channel the value in txpowerControl can be a relative value to the reference signal power. -32768 to 1536 (-128 to 6 dB)	[36.213] Section 5.1.2 for PUCCH
crcLength	8	0 = None 8= 8 bits	[36.212] Section 5.3.3.2
channelType	8	7 = UCI (comes from PUCCH or UL-SCH if UL-SCH data present)	
repetitionNumber	15	Repetition Number for PUCCH	
catMenabled	1	catMenabled when set indicates that this PUCCH is a catM	
noiseDemodMeasEn	1	When set it enables the noise measurement at the demodulator of the PUCCH.	
sigLevelMeasEn	1	When set it enables the signal level measurement for PUCCH.	
noiseDemodAverScale	1	Description Scale Factor for averaging of the PUCCH demodulator noise.	
sigLevelAverScale	1	Description Scale Factor for the PUCCH signal level averaging	
scrmblerInitValue	16	Scrambler initialization value used	[36.211] Sections 5.4.1 and 5.4.2

Parameter	Length, bits	Description, value	Reference
coding Descriptor	3	Refer to Section 4.6.18 ModulationCodingDescriptorStruct	
blockCodeConcatenation	1	Refer to Section 4.6.18 ModulationCodingDescriptorStruct	
modulationType	4	Refer to Section 4.6.18 ModulationCodingDescriptorStruct	
mcsType	6	Refer to Section 4.6.18 ModulationCodingDescriptorStruct	
PucchDedicated CtrlInfo	160	Refer to Section 4.6.25PucchDedicatedCtrlInfoStruct	
CqiPmiRiReport CtrlStruct	64	Refer to Section 4.6.30 CqiPmiRiReportCtrlStruct	
numberofEntries	16	For ULCtrlChannel this has a value of 1	
startResSr	8	First RB for an SR opportunity	
startRes	8	First RB for this allocation.When simultaneous SR and HARQ detection is enabled this entry corresponds to the F1A,F1B startRes and the PHY determines the SR startRes internally from the configuration information provided by the user	
numRes	8	Number of consecutive RB's used	
TimingAdvErrorInfo	96	See 0 TimAdvErrInfCtrlStruct	
pucchRfTuningSymbol	8	Symbol(s) to be ignored in the current subframe for PUCCH decoding as follows 0 Nothing 1 0 2 13	
pucchRepitions	8	Number of repetitions Є {1,2,4,8,16,32}	

4.6.18 ModulationCodingDescriptorStruct

Table 80. ModulationCodingDescriptorStruct

Parameter	Length, bits	Description, value	Reference
codingDescriptor	3	2 bits are used. Possible values are: 0 = Turbo Coder (PDSCH/PUSCH) coding rate 1/3 1 = Tail biting Viterbi Convolutional Code (PBCH or PDCCH), cl 7 and coding rate 1/3 2 = (32, 2) Block Code (CFI) 3 = (3, 1) Block Code (DL-HARQ) 4=(20,A) Code(UCI)	
blockCodeConcatenation	1	0 = Disabled 1 = Enabled (Disabled for Control, HARQ, and so on)	
modulationType	4	2 bits are used. Possible values are: 0 = BPSK (PUCCH, PHICH) 1 = QPSK 2 = 16-QAM 3 = 64-QAM 4 = 256-QAM	

Parameter	Length, bits	Description, value	Reference
mcsType	8	Corresponds to the Modulation and Coding Index that is used to determine the optimum Transport Block given a Bandwidth,	[36.213] Section 7.1.7.1

4.6.19 HarqDescriptorStruct

Table 81. HarqDescriptorStruct

Parameter	Length, bits	Description, value	Reference
Reserved	3	Zero filled	
altCQITableUsed	1	Higher layer parameter altCQI-Table-r12, for 256QAM support	[26.331] CQI-ReportConfig
nDi	1	New Data Indication. In the TxVector this field is don't care to the PHY. In the RxVector this bit reflects whether the incoming data is a retransmission or not, so it states whether the FEC needs to do HARQ combining or not. (The value toggles in normal operation)	[36.321] Section 5.4.1
rV	2	Redundancy Version	[36.212] Section 5.3.3.1.2
flushReq	2	Flush Request from L2 of the PHY buffers associated with the channel	[36.321] Section 5.4.2.1

4.6.20 CrcDescriptorStruct

Table 82. CrcDescriptorStruct

Parameter	Length, bits	Description, value	Reference
crcLength	6	0 = None 8 = 8 bits (CQI, RI, HARQ-ACK, PMI control info on PUSCH) 16 = 16 bits (BCH, DCCH,DCI) 24 = 24 bits (DL-SCH,UL-SCH)	[36.212] Section 5.2.2.6 [36.212] Section 5.3.1.1 [36.212] Section 5.3.3.2 [36.212] Section 5.3.2 [36.212] Section 5.2.2
crcScrambling	2	0 = Disabled 1 = Enabled	

4.6.21 ScramblerDescriptorStruct

Table 83. ScramblerDescriptorStruct

Parameter	Length, bits	Description, value	Reference
scramblerType	8	0 = DL Scrambler 1 = UL Scrambler with provisions for ACK/NACK and Rank Indication Handling	
scrInitValueInput	16	Depends on channelType and channelID. The L2 is responsible for writing here the particular initialization value portion that is generated in the upper layers and that is required by the PHY to generate the c_{init} value. Depending on channelType, the following values are used in this entry:	
		n_{RNTI} for PUSCH, PDSCH, PUCCH formats 2, 2a, 2b and PDCCH for UE specific info	
		N_{ID}^{MBSFN} for PMCH	
		N_{ID}^{cell} for PBCH, PCFICH, PHICH, PUCCH, and PDCCH	
		f_{ss}^{PUSCH} for group hopping of Uplink of demodulation or sounding reference signals	

Table 84. Examples for ScramblerDescriptor usage:

Equation	Description
$c_{init} = n_{RNTI} \cdot 2^{14} + \lfloor n_s/2 \rfloor \cdot 2^9 + N_{ID}^{cell}$	for PUSCH (N_{ID}^{cell} from initialization, so n_{RNTI} is written in this case by the L2 in the ScrInitValueInput).
$c_{init} = \begin{cases} n_{RNTI} \cdot 2^{14} + q \cdot 2^{13} + \lfloor n_s/2 \rfloor \cdot 2^9 + N_{ID}^{cell} \\ \lfloor n_s/2 \rfloor \cdot 2^9 + N_{ID}^{MBSFN} \end{cases}$	For PDSCH
	For PMCH
N_{ID}^{MBSFN}	needs to be written in the case of PMCH by L2
$c_{init} = N_{ID}^{cell}$	for PBCH (Already at the PHY from initialization)
$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^9 + N_{ID}^{cell}$	for PCFICH (same as above)

Equation	Description
$c_{init} = \lfloor n_s/2 \rfloor 2^9 + N_{ID}^{cell}$	for PDCCH (PHY takes care of this based on Initialization information from the L2/L3, for DCI destined to a specific UE the L2/L3 passes n_{RNTI} in the scrInitValueInput, if the PDCCH data is not specific to a UE then the crcScrambling parameter must be disabled
$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^9 + N_{ID}^{cell}$	for PHICH (same as above)
$c_{init} = N_{ID}^{cell}$ for PUCCH	(same as above)
$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^{16} + n_{RNTI}$	for PUCCH formats 2, 2a, 2b (N_{ID}^{cell} from initialization, so n_{RNTI} is written in this case by the L2 in the ScrInitValueInput). where: • q Indicates whether one code word (0) or two code words (1) are transmitted in the sub-frame • n_{RNTI} is the Radio Network Temporary Identifier that is associated with a UE once an RRC connection exists. • N_{ID}^{cell} Physical Layer Cell Identity = 3* PhysicalCellIDGroup + PhysicalLayerCellIdGroup • (See power-up initialization information). • N_{ID}^{MBSFN} MBSFN Area Identity • n_s slot number
For Reference signals that use a scrambler the following initialization values are required:	
$c_{init} = 2^{10} \cdot (7 \cdot (n_s + 1) + l + 1) \cdot (2 \cdot N_{ID}^{cell} + 1) + 2 \cdot N_{ID}^{cell} + N_{CP}$	For Downlink Cell Specific reference signals.
$c_{init} = 2^9 \cdot (7 \cdot (n_s + 1) + l + 1) \cdot (2 \cdot N_{ID}^{MBSFN} + 1) + N_{ID}^{MBSFN}$	For MBSFN reference signals
$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^{16} + n_{RNTI}$	For UE Specific reference signals
$c_{init} = \left\lfloor \frac{N_{ID}^{cell}}{30} \right\rfloor$	For group hopping of Uplink of demodulation or sounding reference signals

Equation	Description
$c_{init} = \left\lfloor \frac{N_{ID}^{cell}}{30} \right\rfloor \cdot 2^5 + f_{ss}^{PUSCH}$	for sequence hopping of Uplink demodulation or sounding reference signals where: • l is the OFDM symbol number • N_{CP} is 0 for normal CP and 1 for extended CP • f_{ss}^{PUSCH} is the sequence shift pattern • $f_{ss}^{PUSCH} = \left(f_{ss}^{PUCCH} + \Delta_{ss}\right) \bmod 30$, where $\Delta_{ss} \in \{0,1,\ldots,29\}$ is configured by higher layers.

4.6.22 DLDedicatedPrecoderDescriptorStruct

The following section provides the `DLDedicatedPrecoderDescriptorStruct`.

Table 85. DLDedicatedPrecoderDescriptorStruct

Parameter	Length, bits	Description, value	Reference
cddType	1	Cyclic Diversity Delay 0 = No CDD; 1= Large CDD	[36.211] Section 6.3.4.2.2
codeBookIdx	4	Codebook index	[36.211] Section 6.3.4.2.
AntennaPort5	1	Port selection of UE-specific RS for TM7 0=No ue ref rs; 1=port5	[36.211] Section 6.10.3
AntennaPort7to14	4	Port selection of UE-specific RS for TM8/9. 0=No ue ref rs; 1=port7; 2=port8; 3=port7-8; 4=1 codeword up to 4 layers, port7-10 depending on nLayers; 5=2 codewords up to 8 layers, port7-14 depending on nLayers;	[36.211] Section 6.10.3
nSCID	1	nSCID scrambles identity field according to Table 6.10.3.1 of 36.212	[36.211] Section 6.10.3
CSIreportenabled	1	When set it enables the codebook based precoder to be used based on CSI report.	
mu_mimo	1	When set it enables the multi user MIMO mode.	
codeBookidx2	4	Code book index for CSI reporting code book of 8 antennas	
dl_delta_power_offset	1	For TM5 mu-mimo. Set according to the DCI Format 1D Downlink power offset field. If Downlink power offset field in DCI Format 1D is set, this bit should be set. Will notify L1 that to adjust power settings according to 3GPP 36.213 Table 7.1.5-1.	
Reserved	14	Padding	
codebookSubsetRestriction	64	Bitmap where 0 = Disabled 1 = Enabled Depending on the transmission mode 0 to 64 bits are used. Codebook subset restriction is supported for open-loop spatial multiplexing, closed-loop spatial multiplexing, multi-user MIMO and closed-loop Rank=1 precoding	

The `DLDedicatedPrecoderDescriptor` contains information that allows the support from one set of I-Q samples stream to as many as the number of antennas present in the system. This descriptor is used to configure the precoder in support of a corresponding transmission mode such as Spatial Diversity (Space Time Coding), Spatial Division Multiplexing and Cyclic Delay Diversity.

For single layer the number of codewords is 1 and the second `SubChannelDescriptor` information is a `don't care`, but it is advisable to be zero filled.

4.6.23 MappingDescriptorStruct

Table 86. MappingDescriptorStruct

Parameter	Length, bits	Description, value
numberofEntries	16	Defines how many pairs of startRes and numRes are present for the current mapping
startRes[0]	8	This is the starting block for an allocation. For data channels it corresponds to the starting Pair of Resource Blocks (where the allocation is spans slot 0 and slot 1). For a control channel other than PCFICH this is the starting CCE. **Note**: For a PCFICH or a PBCH channel the allocation is implied and the PHY will ignore this entry
numRes[0]	8	For a data channel this input specifies how many consecutive Pair of RB's are used, starting at startRes. For a control channel other than PCFICH this input specifies how many consecutive CCE's are used starting at startRes. **Note**: For a PCFICH or a PBCH channel the allocation is implied and the PHY will ignore this entry
...		There are always 50 entries in this descriptor. The user only needs to fill in the RXVECTOR the information for up to the numberofEntries. The rest are do not care
startRes[49]	8	
numRes[49]	8	
Pad	8	

4.6.24 CsiRsDescriptorStruct

Table 87. CsiRsDescriptorStruct

Parameter	Length, bits	Description, value
csiRsEnable	1	0 csiRs disabled 1 csiRs enabled this subframe The L2/L3 software keeps track of the $I_{CSI\text{-}RS}$ per [36.211] Section 6.10.5.3.-1 and enables csiRS when required.
csiRsResourceConfig	5	Corresponds to CSI Reference Signal Configuration per [36.211] tables 6.10.5.2-1 and 6.10.5.2-2 0 <= csiRsResourceConfig <= 31
csiRsAntCount	2	number of antenna ports used for transmission of CSI reference signals. Possible values are 1,2,4 or 8* Not all values are supported in all devices please refer to release notes for device specific limitations
csiRsPc	8	csiRsPc is the assumed ratio of PDSCH EPRE to CSI-RS EPRE when UE derives CSI feedback and takes values in the range of [-8, 15] dB with 1 dB step size, where the PDSCH EPRE corresponds to the symbols for which the ratio of the PDSCH EPRE to the cell-specific RS EPRE is denoted by P_A Two's complement representation constrained to [-8,..,15]

Parameter	Length, bits	Description, value
csiRsZeroTxPwrBitMap	16	Corresponds to *ZeroPowerCSI-RS* per [36.211] Section 6.10.5.2. This is a 16 bit bitmap where the MSB corresponds to the Lowest CSI Reference Signal Configuration index.

4.6.25 PucchDedicatedCtrlInfoStruct

Table 88. PucchDedicatedCtrlInfoStruct

Parameter	Length, bits	Description, value	Reference
formatType	3	PUCCH (UCI)format type in use Coded as 0 Format 1 1 Format 1a 2 Format 1b 3 Format 2 4 Format 2a 5 Format 2b 6 Format 3	[36.211] Section 5.4. 3GPP release 10
ackNackRepetition	1	Parameter indicates whether ACK/NACK repetition is configured Value of 1 indicates configured Value of 0 indicates is not	[36.213] Section 10.1.
simSRHarq	1	Parameter indicates if simultaneous SR and HARQ are expected.in this subframe for this channel 0 No 1 Yes	[36.213] Section 7.3.
repetitionFactor	4	Parameter N_{ANRep} Integer ∈ {2, 4 or 6} (Padded)	[36.213] Section 10.1.
msPucchSrSel	1	Parameter used in ms tx to determine whether to use pucch resource index (0) or sr resource index(1). For Bs this bit should be zero	
n1PucchANRep	11	Parameter: $n^{(1)}_{PUCCHANRep}$ Integer ∈ { 0, .. , 2047} For 3GPP release 10 and later corresponds to Port0 i.e. $n^{(1,0)}_{PUCCHANRep}$	[36.213] Section 10.1.
cqiPUCCHResourceIndex	11	Parameter $n^{(2)}_{PUCCH}$ Integer ∈ {0, .. , 1185} $n^{(2,0)}_{PUCCH}$ for 3GPP release 10 and later (antenna port 0) Integer ∈ {0, .. , 1184}	[36.213] Section 7.2. [36.211] Section 5.4.
srPUCCHResourceIndex	11	Parameter: $n^{(1)}_{PUCCH,SRI}$. Integer ∈ {0, .. , 2047}	[36.213] Section 10.1.

Parameter	Length, bits	Description, value	Reference
		For 3GPP release 10 and later Parameter $n^{(1,0)}_{PUCCH,SRI}$ for Antenna Port 0 ∈{0,..2047} applies to F1,F1A,F1B	
dlCqiPmiSizeBits	8	Size of dlCqiPmi Information in bits ∈ {0,..,255}	[SCAPI] Table 58
numCqiPucchResources	1	Number-1 Integer ∈ {0,1} A value of 1 indicates that the UE uses two antenna ports	[SCAPI] Table 71 3GPP release 10 support
numSrPucchResources	1	Number-1 Integer ∈ {0,1} Avalue of 1 indicates that the UE uses two antenna ports	[SCAPI] Table 73 3GPP release 10 support
cqiPUCCHResourceIndexP1	11	Parameter $n^{(2,p)}_{PUCCH}$ for Antenna Port 1 ∈ {0, .., 1184}	3GPP release 10 support [36.213] Section 7.2
rssiChanMeasEn	1	Enable rssi measurements for this channel only. The actual value is reported in the RXSDU.indication 0: Disable 1:Enable	
numConfiguredServingCells	2	Number of configured serving cells 0 corresponds to one 1 corresponds to two or more	3GPP release 10 support [36.213] Section 10.1.2.1 and Section 10.1.2.2
numPucchResourcesHarq	2	Number of Pucch Resources used for the HARQ Information ∈ [1, ..,4] -> [0,..,3] If the HARQ are transmitted on two antenna ports then the resource for the second antenna is in ackPUCCHResourceIndex1	3GPP release 10 support [36.213] Section 10.1.2.2.1 and Section 10.1.2.2.2

Parameter	Length, bits	Description, value	Reference
harqSizebits	5	Number of bits used for ack/nack FDD integer ∈[1,10] TDD integer ∈[1,20]	3GPP release 10 support
srPUCCHResourceIndexP1	11	Parameter $n^{(1,1)}_{PUCCH}$ for Antenna Port 1 ∈{0,..2047} applies to F1,F1A,F1B	3GPP release 10 support [36.213] Section 10.1
harqAckFeedbackMethod	3	The interpretation of this field depends on whether TDD or FDD is being used For TDD the values are interpreted as: 0 Bundling 1 Multiplexing 2 F1b with Channel selection 3 Format 3 For FDD the values are interpreted as 0 Format 1a/1b 1 Don't Care 2 F1b with Channel Selection 3 Format 3	3GPP release 10 support [36.213] Section 10.1.2.2.1 and Section 10.1.2.2.2 [SCAPI] section 3.3.1.3.10
ackPUCCHResourceIndex1	11	corresponds to $n^{(1)}_{PUCCH,1}$ ∈ {0, ..,2047} for formats 1a/1b with channel selection Or to $n^{(3,1)}_{PUCCH}$ for format 3 ∈ {0,..,549}	3GPP release 10 support [36.213] Section 10.1
twoAntennaPortActivatedPucch Format1a1b	1	0 Only 1 Antenna is used 1 Two Antenna Ports are used for Format1a1b. It also applies to f1af1b transmission when format 3 is configured	3GPP release 10 support [36.213] Sections 10.1, 10.1.2.2.2 and 10.1.3.2.2
simultaneousPucchPusch	1	0 No simultaneous Pucch and Pusch is allowed 1 Simultaneous Pucch and Pusch is enabled	3GPP release 10 support [36.213] Sections 10.1 and 5.1.1
reserved2	16	Set to zero	3GPP release 10 support [36.213] 10.1
ackPUCCHResourceIndex2	11	corresponds to $n^{(1)}_{PUCCH,2}$ ∈ {0, ..,2047} for formats 1a/1b with channel selection Or to $n^{(3,2)}_{PUCCH}$ for format 3 ∈ {0,..,549}	3GPP release 10 support [36.213] Sections 10.1 and 5.1.1
ackPUCCHResourceIndex3	11	corresponds to $n^{(1)}_{PUCCH,3}$ ∈ {0, ..,2047} for formats 1a/1b with channel selection Or to $n^{(3,3)}_{PUCCH}$ for format 3 ∈ {0,..,549}	3GPP release 10 support [36.213] Sections 10.1 and 5.1.1
subframeMultiplexNum	10	M the number of elements in the set K defined in Table 10.1-1 of [36.213]. This field is only applicable when the harqAckFeedbackMethod is set to multiplexing feedback mode Integer∈ { 1,2,3,4}	[36.213] Section 10.1

4.6.26 SoundingRefDedicatedCtrlStruct

Table 89. SoundingRefDedicatedCtrlStruct

Parameter	Length, bits	Description, value	Reference
EnableDisable	4	A value of 1 enables Sounding Reference A value of 0 disables it	3GPP release 10 support
srsBandwidth	2	Parameter: B_{srs} or b 0,1,2,or 3	[36.211] Table 5.5.3.2-1.
srsHoppingBandwidth	2	Parameter: hopping bandwidth $\in${0,1,2,3} *hop b*	[36.211] Section 5.5.3.2
transmissionComb	1	Parameter: $k_{TC} \in \{0, 1\}$	[36.211] Section 5.5.3.2.
cyclicShift	3	Parameter: n^{CS}_{SRS}. Integer $\in$ {0,...,7}	[36.211] Section 5.5.3.1,
freqDomainPosition	5	Parameter: n_{RRC} Integer $\in$ {0,1,..,23}	[36.211] Section 5.5.3.2.
Duration	1	Parameter: Duration. FALSE corresponds to “single” and value TRUE to “indefinite”.	[36.213] Section 8.2.
srsConfigIndex	10	Parameter: I_{SRS} Integer $\in$ {0,..,1023}	[36.213] table8.2-1.
srsAntennaPort	4	Indicates the number of antenna ports used for periodic sounding reference signal transmission $\in${1,2,4}	[36.211] Section 5.5.3 3GPP release 10 support
Rnti	16	Radio network temporary identifier	
beamformingEnable	1	Enable direction of arrival calculation for beam forming on TM7-8	
Reserved	15		

4.6.27 SoundingRefT1DedicatedCtrlStruct

Table 90. SoundingRefT1DedicatedCtrlStruct

Parameter	Length, bits	Description, value	Reference
EnableDisable	4	A value of 1 enables Aperiodic Sounding Reference A value of 0 disables it	3GPP release 10 support
srsBandwidthAp	2	Parameter: Bsrs or b ∈ {0,1,2,or 3}	[36.211] tables 5.5.3.2-1, 5.5.3.2-2, 5.5.3.2-3 and 5.5.3.2-4
Pad	2	Set to zero	
transmissionCombAp	1	Parameter: k TC ∈ {0, 1}	[36.211] Section 5.5.3.2.
cyclicShiftAp	3	Parameter: n^{CS}_{SRS}. Integer ∈ {0,…,7}	[36.211] Section 5.5.3.1.
freqDomainPositionAp	5	Parameter: n_{RRC} Integer ∈ {0,1,..,23}	[36.211] Section 5.5.3.2.
pad1	6	Set to zero	
srsConfigIndexAp	5	Parameter ISRS for aperiodic sounding reference signal transmission. ∈ {0, .., 31}	[36.213] Table 8.2-5
srsAntennaportAp	4	Indicates the number of antenna ports used for aperiodic sounding reference signal transmission ∈{1,2,4}	[36.211] Section 5.5.3.

4.6.28 PuschConfigDedicatedStruct

Table 91. PuschConfigDedicatedStruct

Parameter	Length, bits	Description, value	Reference
betaOffsetACKIndex	4	Parameter: $I^{HARQ\text{-}ACK}{}_{offset}$ Integer ∈ {0, .. , 15}	[36.213] Table 8.6.3-1.
betaOffsetRIIndex	4	Parameter: $I^{RI}{}_{offset}$ Integer ∈ {0, .. , 15}	[36.213] Table 8.6.3-2.
betaOffsetCQIIndex	4	Parameter: $I^{CQI}{}_{offset}$ Integer ∈ {0, .. , 15}	[36.213] Table 8.6.3-3.
nDMRS2	4	Parameter $N^{(2)}{}_{DMRS}$, to generate cyclic shift for DMRS in PUSCH Integer ∈ {0, .. , 10}	[36.211] Section 5.5.2.1.
Nsymi	4	Initial number of SC-FDMA symbols per subframe excluding those used for pilots or SRS Integer ∈ {9,..,12}	[36.212] Section 5.2.2.6.
nACK	4	Number of HARQ ACK bits. Integer ∈ {0,1, 2}	[SCAPI] Table 50.
Nrbi	7	Number of Resource Blocks initially assigned to this Transport Block	[SCAPI] Table 52.
tddAckNackFeedbackMode	1	TDD ack/Nack feedback for UE that does not support more than one serving cell 0: Ack/Nack Bundling 1: Ack/Nack Multiplexing	[36.213] Section 10.1.3.
nRI	8	Number of RI bits. Integer ∈ {0,1, 2}	[SCAPI] Table 48.
nr1CQI	16	Number of CQI bits for Rank = 1. Integer ∈ {0,..., 504}	[SCAPI] Table 48.
nrg1CQI	16	Number of CQI bits for Rank > 1. Integer ∈ {0,..., 504}	[SCAPI] Table 48.
puschHoppingEn	1	Pusch Hopping Enable when set	[36.211] Section 5.3.4.
hoppingInfoBits	2	Hopping Information Bits	[36.213] Section 8.4.
Nbundled	5	For TDD AckNack Bundling mode only else it is a don't care. Integer ∈{1,..,6}	[36.212] Section 5.2.2.6. and [36.213] Section 7.3.
Reserved	16	reserved	

4.6.29 RachCtrlStruct

Table 92. RachCtrlStruct

Parameter	Length, bits	Description, value	Reference
prachEnable	1		
rootSequenceIndex	10	Parameter: RACH_ROOT_SEQUENCE Integer ∈ {0,..,837}	[36.211] Section 5.7.1.
prachConfigIndex	6	Parameter: prach-ConfigurationIndex Integer ∈ {0,..,63}	[36.211] Section 5.7.1.
highSpeedFlag	1	Parameter: High-speed-flag TRUE corresponds to Restricted set FALSE to Unrestricted set.	[36.211] Section 5.7.2.
zeroCorrelationZoneConfig	4	Parameter: N_{CS} configuration Integer ∈ {0,..,15}	[36.211] Section 5.7.2: Table 5.7.2-2 for preamble format 0,..,3 [36.211] Section 5.7.2: Table 5.7.2-3 for preamble format 4.
prachFreqOffset	10	Parameter: prach-FrequencyOffset or $\eta^{RA}_{PRBoffset}$, For TDD the value range is dependent on the value of *prach-ConfigIndex*. Integer ∈ {0,..,94}	[36.211] Section 5.7.1.
prachPreambleNumber	6	Used only in Mobile Station Transmitter Don't care in Base Station Receiver	
prachTddFreqResEnabled	6	Valid only in TDD mode. Don't care for FDD. In TDD this is a 6 bit-field that indicate which RACH frequency resources are used in the subframe. The frequency resources are in ascending order from LSB to MSB	[36.211] Section 5.7.1
Pad	20		

4.6.30 CqiPmiRiReportCtrlStruct

Table 93. CqiPmiRiReportCtrlStruct

Parameter	Length, bits	Description, value	Reference
cqiReportModeAperiodic	3	Parameter: *reporting mode.* PUSCH reporting modes Integer ∈ {0, .. , 4} Where 0 corresponds to Mode 1-2 1 corresponds to Mode 2-0 2 corresponds to Mode 2-2 3 corresponds to Mode 3-0 4 corresponds to Mode 3-1	[36.213] Section 7.2.1.
nomPDSCHRSEPREOffset	3	Parameter: Δ_{offset} Actual value = IE value * 2 [dB]. Integer ∈ {-1, .. , 6} mapped as 0, .., 7 respectively	[36.213] Section 7.2.3.
cqiReportPeriodicEnable	2	1 enables a periodic CQI report 0 disables a periodic CQI report	
cqiPUCCHResourceIndex	11	Parameter $n^{(2)}_{PUCCH}$ Integer ∈ {0, .. , 1185}	[36.213] Section 7.2.
cqipmiConfigIndex	10	Parameter: CQI/PMI Periodicity and Offset Configuration Index *ICQI/PMI* Integer ∈ {0,.. , 1023}	[36.213] tables 7.2.2-1A and 7.2.2-1C.
cqiFormatIndicatorPeriodic	1	Parameter: PUCCH CQI Feedback Type Depending on transmissionMode, reporting mode is implicitly given from the table. 0 Wideband CQI 1 Sub-band CQI	[36.213] Table 7.2.2-1
K	2	Parameter: K Integer ∈ {1,.. , 4}	[36.213] Section 7.2.2.
riConfigIndex	10	Parameter: RI Config Index I_{RI}. Integer ∈ {0, .. , 1023}	[36.213] Section 7.2.2-1B.
simultaneousAckNackAndCQI	1	Parameter: Simultaneous-AN-and-CQI TRUE indicates that simultaneous transmission of ACK/NACK and CQI is allowed.	[36.213] Section 10.1.
Padding	21		

4.6.31 TimAdvErrInfCtrlStruct

Table 94. TimAdvErrInfCtrlStruct

Parameter	Length, bits	Description, value	Reference
mErrAvgCh[0]	32	Obsolete.	
mErrAvgCh[1]	32	Obsolete.	
mSetDefaults	31:16, 16	Obsolete.	
mErrExpo	15:0, 16	Obsolete.	

4.6.32 ULSubChannelDescriptorCatm

Table 95. UlSubChannelDescriptorCatm

Parameter	Length, bits	Description, value	Reference
puschCEmodef	1	puschCEmode 0 Mode A 1 Mode	
puschRepetitionIndex	11	Є {0,2047}	
puschRftuningSymbols	4	Symbols to be ignored in the current subframe for pusch decoding 0 None 1 0 2 0 and 1 3 13 4 12 and 13	
puschMaxNumRepetitions	4	For Mode A 0 Not configured 1 16 2 32 For Mode B 0 Not configured 1 192 2 256 3 384 4 512 5 768 6 1024 7 1536 8 2048	
puschrepSetSelection	4	Selects 1 particular repetition from the set given by the Mode and the puschMaxNumrepetitions. Please refer to 3GPP 36.213 Tables 8-2b and 8-2c for {n1,n2,n3,n4} and {n1,n2,n3,n4,n5,n6.n7.n8} respectively	
harqProcessNumber	8	HARQ process number for PUSCH	
scramblerInitValue	32	Scrambler Init Value for PUSCH catM provided by the L2 and takes into account repetition number and other parameters	

Note: For catM support, the L2 is responsible for providing the correct cinit information in the scramblerInitValue using η_{RNTI}, Nacc, *NCell*ID and N_{PUSCH}abs.

5.0 Physical Implementation

5.1 Transport Mechanisms

5.1.1 DL Message Delivery

The use of a FIFO scheme for conveying messages from L2/L3 to PHY is recommended. Under the FIFO scheme the L2/L3 sends messages like TXSTART, RXSTART, TXSDUs, TXHIDCIUL, TXDCIULSDUs and TXHISDUs without any acknowledgement. The PHY peels out messages from the FIFO queue at the beginning of the subframe following the command timing guidelines of Section 0

API Message Timing.

In order to select the FIFO scheme of message delivery from L2/L3 to PHY, there is a bit to be set in the INIT.request message containing the INITPARM message coming from MAC to PHY during system startup. The field phyCfg in this message needs to have bit 0 set to enable this scheme.

5.1.2 UL Message Delivery

The PHY API splits the CQI and RI information that comes from the UE into separate lists and sends the HARQ ACK/NACK bits as soon as the PHY is able to decode them (because of the latency requirements in LTE). Therefore:

- PUCCH HARQ ACK / NACK, CQI and RI messages (RXSDU.indication) and Multiplexed PUSCH Control messages HARQ ACK / NACK (RXSTATUS.indication) are in Listtype 2 [Pucch].
- PUSCH messages (RXSDU.indication) are in Listtype 1[Pusch].
- Multiplexed PUSCH Control CQI and RI messages (RXSTATUS.indication) are in Listtype 3[Other] since we take a bit longer to process them and this is not time critical information for the L2/L3 software.
- SRS message (RXSTATUS.indication) is a single message containing all the srs information for the subframe[Single Entity].
- PRACH message (RXSTATUS.indication) is a single message for all PRACH events detected during the subframe[Single Entity].

Listtype 2 usually is sent out to the L2/L3 usually within 1 ms of arrival of UL IQ samples to eNodeB. Listtype 1 is sent to the L2/L3 as soon as all PUSCH channels have been decoded, usually within 1.5 ms of arrival of UL IQ samples to eNodeB. Listtype 3 is sent to L2/L3 as soon as the PHY is done with all the processing. In the worst case this is within 3 ms of arrival of IQ samples to eNodeB.

There is a RXEND.indication supplied at the end of Listtype 1 / Listtype 2 (depending on whether they are present and which one finishes first). There is no RXEND.indication supplied at the end of Listtype 3.

To enable the Listtype 3 messages from the PHY to the L2/L3, there is a bit to be set in the INIT.request message containing the INITPARM message coming from MAC to PHY during system startup. The field phyCfg in this message needs to have bit 4 set to enable this scheme. If bit 4 is not set all PUCCH and Mux PUSCH messages will be sent in one list and will take longer to arrive at L2. Therefore, if bit 4 is not set, disabling the expedited and prioritized HARQ delivery, the L2/L3 the system may not be able to meet the LTE HARQ latency requirements.

The assignment of this bit in the INITPARM structure was done to maintain backward compatibility and to allow the selection of the expedited delivery without re-compiling the PHY code.

When the Mux Control Bit is 1 in the phyCfg then the list processing is as follows:

1. PUSCH List.
2. PUCCH List for UCI format 1, 1a, 1b, 2, 2a, 2b or RXSTATUS for Multiplexed PUSCH HARQ Info. RXEND is appended to 1 or 2 depending on which one finishes last.
3. Other List RXSTATUS for CQI/PMI/RI.
4. Single Entity SRS/PRACH/etc.

When the Mux Control Bit is 0 in the phyCfg then the list processing is as follows:

1. PUSCH List.
2. PUCCH List PUCCH and RXSTATUS for multiplexed PUSCH. RXEND is appended to 1 or 2 depending on which one finishes last.
3. 3 Single Entity SRS/PRACH/etc.

5.2 API Batch Message Delivery Scheme

The API Batch message delivery scheme between the L2/L3 and the PHY has been designed to further reduce the overhead in the Shared Memory Interface (SHM) that is used for the exchange of these messages that is managed by the WLS.

The API batch message delivery option needs to be enabled in the phycfg.xml file in the section named API as shown below.

```
<API>
<mac2PhyBatchApi>1</mac2PhyBatchApi>
<phy2MacBatchApi>1</phy2MacBatchApi>
<API>
```

When the fields are set to 1, the API batch mode is used. For the FlexRAN implementation Batch API mode is the default and only supported mode.

5.3 Batch API Message Exchange

This message scheme reduces the overhead by batching multiple messages and their associated ListElementHeader into a single SHM Block to optimize the processing at the sending and receiving ends.

Table 96. Batch API Allowed Messages

Direction	Message Type	Message Struct Definition	Special Notes
MAC -> PHY	PHY_TXSDU_REQ (PDSCH)	TXSDUREQ with ZBCAPI	Only active when DL ZBC is enabled (bit PHYINIT_USE_SCATTERED_TXSDU_POINTER is set in PPHYINIT->phyCfg) (Default)

MAC -> PHY	PHY_TXSDU_REQ (PDCCH)	TXSDUREQ	
MAC -> PHY	PHY_TXHISDU_REQ	HIINFOMSGDESC	
MAC -> PHY	PHY_TXDCIULSDU_REQ	DCIULSDUMSG	
PHY -> MAC	PHY_RXSDU_IND (PUSCH)	RXSDUIND	Only active when UL ZBC is enabled (bit PHYINIT_USE_SCATTERED_RXSDU_POINTERS is set in PPHYINIT->phyCfg) (Default)
PHY -> MAC	PHY_RXSDU_IND (PUCCH)	RXSDUIND	
PHY -> MAC	PHY_RXEND_IND	MSGIND	
PHY -> MAC	PHY_RXSTATUS_IND (MUX PUSCH)	RXSTATUSIND	

All of the messages exchanged between the MAC and PHY have the following structure definition for the first 32 bits:

```
BIT_FIELD abc:16;
BIT_FIELD msgType:8;
BIT_FIELD xyx:8;
```

The msgType defines the information that follows these 32 bits. Because of the common structure among all messages it is easy to batch the processing into a single SHM block in the Batch API implementation.

5.3.1 Batch API Message Exchange Overview

The Batch API message scheme uses the following List Element header definition follows:

```
typedefstruct stMac2PhyApiQueueElem
{
    U16 frameNumber;
    U8 MessageType;
    U8 subframeNumber;
    U32 MessageLen;
    struct stMac2PhyApiQueueElem* Next;
    U32 align_res1;
    U8* MessagePtr;
    U32 align_res2;
    U16 NumMessageInBlock;
    U16 AlignOffset;
    U32 TimeStamp;
    U32 reserved[56];
} MAC2PHY_QUEUE_EL,*PMAC2PHY_QUEUE_EL;
```

In the above structure alternatively an ALIGN_32 macro can be used after struct `stMc2PhyApiQueueElem*` Next and after `U8* MessagePtr` when compiling for a 64 bit OS.

Each SHM block have this structure at the top defining the content of the block and uses the `Next` field to point to the next SHM block. The `MessagePtr` field is ignored and the user needs to set the entry to NULL.

The message or group of messages follows the ListElementHeader in the same SHM block.

The `NumMessageInBlock` field defines the number of messages that are present in this SHM block.

The AlignOffset field defines the number of bytes between messages in the block. Keep this offset in multiples of 64 bytes which corresponds to one cache line in the Intel® processor so it improves the efficiency of cache related operations. Examples of AlignOffset are shown for different message types in Table 97.

Table 97. Examples of AlignOffset

MessageType	Message Struct Size (bytes)	Recommended AlignOffset (bytes)
PHY_TXSDU_REQ(PDSCH)	16 + payload	2048 (to accommodate for 250 potential ZBC blocks per SDU)
PHY_TXSDU_REQ(PDCCH)	16 + payload	64
PHY_TXHISDU_REQ	12	64
PHY_TXDCIULSDU_REQ	20 + payload	64
PHY_RXSDU_IND(PUSCH)	8 + payload	64
PHY_RXSDU_IND(PUCCH)	8	64
PHY_RXEND_IND	2	64
PHY_RXSTATUS_IND (MUX PUSCH)	4 + payload	64

The AlignOffset value of 2048 for PDSCH TXSDU is only a recommendation. For example if the L2/L3 design constrains the maximum number ZBC blocks per SDU to 100 then a value of 832 can be used.

All the messages in the same SHM block must have the same offset as defined in the AlignOffset field. This is part of the requirements for the Batch API Transfer scheme.

For example if there are three TXSDUs messages of sizes 128, 256 and 512 bytes to be sent using the Batch API transfer then the AlignOffset field needs to be set to 512 and the messages are located at locations 0, 512 and 1024 respectively after the MAC2PHY_QUEUE_ELM in the SHM Block. The NumMessageInBlock needs to be set to 3. The gaps between the TXSDUs are ignored at the PHY.

If all the messages to be sent in an SHM cannot fit the requirements cited above, then the need to be split into multiple SHM blocks and chained.

The TimeStamp is a reserved field to be used by the WLS.

MessageType needs to be filled with the PHY_BATCH_MSG_REQ (For L2/L3 to PHY direction) and with PHY_BATCH_MSG_IND (For PHY to L2/L3 direction).

In the event that there is a single message in the SHM block the actual message type can be used and the MessagePtr is set as NULL.

As stated earlier the Batch API processing can be enabled independently for each direction i.e. L2/L3 to PHY and PHY to L2/L3 but for the FlexRAN implementation described in this manual it is mandatory to have both directions enabled.

Table 98 illustrates an example of the Batch API Message exchange.

Table 98. Example of the Batch API Message Exchange

SHM Block 1	
Relative Address to SHM Head in Bytes	**Data**
0	**MAC2PHY_QUEUE_EL** U16 frameNumber; U8 MessageType = PHY_BATCH_MSG_IND; U8 subframeNumber; U32 MessageLen; PMAC2PHY Next = NextShmBlock; U32 align_res1; U8* MessagePtr = NULL; U32 align_res2; U16 NumMessageInBlock = n; U16 AlignOffset = XYZ; U32 TimeStamp; U32 reserved[58];
A = Sizeof(**MAC2PHY_QUEUE_EL**)	**PhyMsg[1]** abc:16; msgType:8; xyz:8; ...
...	...
A + ((n-1)*XYZ)	**PhyMsg[n]** abc:16; msgType:8; xyz:8; ...

In this example we can notice the following facts about the Batch API message exchange:

- All messages that are to be sent are placed in a single SHM block managed by WLS.
- The first part of the message is the `MAC2PHY_QUEUE_EL` which has:

– MessageType set as `PHY_BATCH_MSG_IND` (Meaning Batch Mode messages from PHY to L2/L3).

– `NumMessageInBlock` set to n since there are n messages in this SHM Block.

– `AlignOffset` set as XYZ so the L2/L3 advances in multiples of XYZ in between messages until reading all the messages in this block. Where XYZ is a multiple of 64 bytes.

- The total size of the message is (`NumMessageInBlock * AlignOffset`) + sizeof (`MAC2PHY_QUEUE_EL`). This number has to be entered in the `MessageLen` field so the WLS can correctly perform the cache clean/invalidate operations for the SHM block.
- The messages themselves follow the `MAC2PHY_QUEUE_EL` and each have a maximum size of `AlignOffset` each including header and payload. If the messages are smaller than the maximum size the remainder bytes are ignored.
- The L2/L3 can potentially pack all `PHY_TXSDU_REQ(PDCCH), PHY_TXHISDU_REQ` and `PHY_TXDCIULSDU_REQ` into a single SHM block or it can send all the PHY_`TXSDU_REQ`, `PHY_TXHISDU_REQ` and `PHY_TXDCIULSDU_REQ` into 3 separate message blocks.

In a similar way the PHY can pack all the messages into a single SHM block.

The partitioning of the messages is left to the discretion of the implementer based on specific implementation constraints.

The basic Batch API message exchange framework allows that any combination of messages subject to the exchange rules outlined earlier can be combined into a single SHM block.

The maximum size of an SHM block is 16 Kbytes so everything must be smaller or equal to this once the offsetAlignment constraints are taking into consideration otherwise additional SHM blocks need to be used.

5.3.2 Error Handling

For Messages from the L2/L3 to PHY that use the Batch API message exchange and that may contain errors due to some information not being filled correctly the normal error indications from the PHY to the L2/L3 would be issued. So for erroneous UL channels allocated by the L2/L3 via RxVector a RXSDU(s) and/or RXSTATUS indication messages with a status field set to INVALID_RX_VECTOR. In the event of errors in the TxVector a PHY_ERROR_IND with status set to INVALID_TX_VECTOR is issued.

5.3.3 Example Batch API Message Exchange

In this example we consider a typical TTI processing case where the L2/L3 issues the following messages to the PHY:

- One TxVector (PHY_TXSTART_REQ)
- One RxVector (PHY_RXSTART_REQ)
- One MiniTxVector (PHY_TXHIADCIUL_REQ)
- Two PDCCH SDUs (PHY_TXSDU_REQ)
- Four PDSCH SDUs (PHY_TXSDU_REQ)
- Two DCI0 PDSCH SDUs (PHY_TXDCIULSDU_REQ)
- One PHICH SDU (PHY_TXHISDU_REQ)

Using the Batch API message exchange scheme the information is transferred as shown in Table 99.

Table 99. Example of the L2/L3 Using the Batch API Message Transfer Scheme

Address	SHM Block	Contents
0x10000000	SHM1	**QUEUE_EL[0]** = {Next=SHM2; MsgPtr=NULL;NumMessageInBlk=1; AlignOffset= sizeof(DLSUBFRDESC); MsgLen=sizeof(DLSUBFRDESC) + sizeof(MAC2PHY_QUEUE_EL); MessageType=PHY_TXSTART_REQ;} DLSUBFRDESC
0x10004000	SHM2	**QUEUE_EL[1]** = {Next=SHM3;MsgPtr=NULL;NumMessageInBlk=1; AlignOffset= sizeof(ULSUBFRDESC); MsgLen=sizeof(ULSUBFRDESC) + sizeof(MAC2PHY_QUEUE_EL); MessageType=PHY_RXSTART_REQ;} **ULSUBFRDESC**
0x10008000	SHM3	**QUEUE_EL[2]** = {Next=SHM4;MsgPtr=NULL;NumMessageInBlk=1; AlignOffset=sizeof(HIADCIULMSGDESC);MsgLen=sizeof(HIADCIULMSGDESC) + sizeof(MAC2PHY_QUEUE_EL); MessageType= PHY_TXHIADCIUL_REQ} **HIADCIULMSGDESC**
0x1000C000	SHM4	**QUEUE_EL[3]** = {Next=SHM5;MsgPtr=NULL;NumMessageInBlk=5; AlignOffset=64;

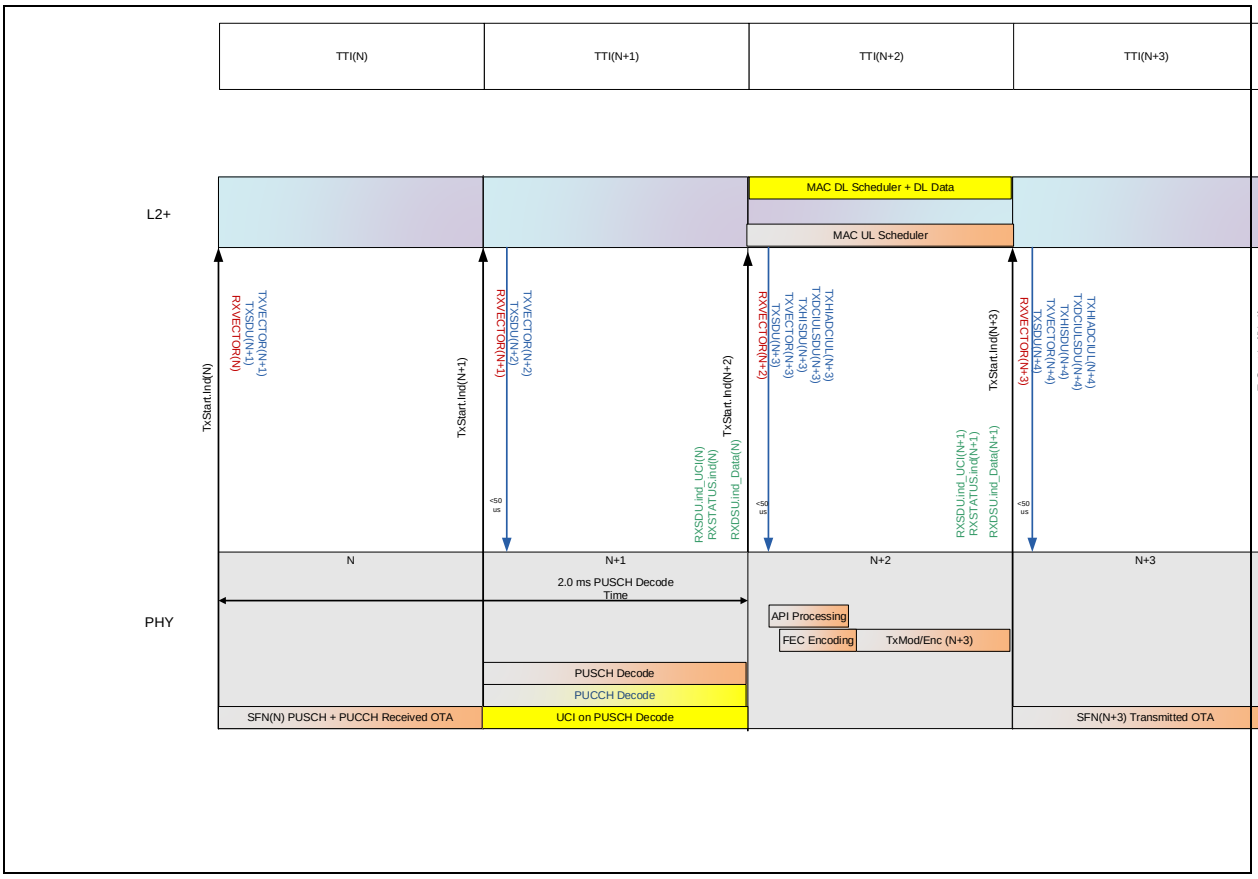
Address	SHM Block	Contents
		MsgLen= 2*sizeof(TXSDUREQ) + 2*sizeof(DCIULSDUMSG) + sizeof(HIIINFOMSGDESC) + sizeof(MAC2PHY_QUEUE_EL); MessageType = **PHY_BATCH_MSG_REQ**} **TXSDUREQ[0] at 0x1000C0100** **TXSDUREQ[1] at 0x1000C0140** **DCIULSDUMSG[0] at 0x1000C0180** **DCIULSDUMSG[1] at 0x1000C01C0** **HIIINFOMSGDESC[0] at 0x1000C0200**
0x10010000	SHM5	**QUEUE_EL[4]** = {Next=NULL;MsgPtr=NULL;NumMessageInBlk=4; AlignOffset=512; MsgLen= 4*AlignOffset + sizeof(MAC2PHY_QUEUE_EL); MessageType= **PHY_BATCH_MSG_REQ**;} TXSDUREQ[2] at 0x10010100 TXSDUREQ[3] at 0x10010300 TXSDUREQ[4] at 0x10010500 TXSDUREQ[5] at 0x10010700

This is just an illustrative example on how the L2/L3 can use the Batch API message transfer scheme. Other schemes are possible subject to the rules and limitations that have been outlined earlier.

5.4 DL HARQ Latency Processing Considerations

Figure 49 illustrates the DL/UL HARQ latency processing for the FDD case. For more details please refer to Table 1. .....Related Documentation *FlexRAN L1 Doxygen Code Reference Document*.

Figure 49. UL-DL HARQ Latency Diagram



5.5 Message Timing

5.5.1 API Messages Timing

The following time deadlines are imposed on the API commands and also the expected maximum time to post indications are summarized in Table 100.

Table 100. API Message Timing

Message	Time Delay from subframe (n) on the air beginning (ms) (min.)	Time Delay from subframe(n) on the air beginning (ms) (max.)
RXSDU.ind (PUSCH, PUCCH,CRC,SR)	1.9	2.0
RXSTATUS.ind HARQ,CQI,RI	1.9	2.0
RXSTATUS.Ind Prach	2.0	4
RXSTATUS.Ind Srs	2.0	3
Message	Window Start from TTI (ms)	Window End from TTI (ms)
PHYINIT.req	-0.50	+0.50
PHYRECONFIG.req	-0.50	+0.50
PHYSTART.req	-0.50	+0.50
PHYSTOP.req	-0.50	+0.50
PHYSTOP.ind	0.20	2.00
TXSTART.req	-0.95	+0.05
RXSTART.req	-0.95	+0.05
TXHIADCIUL.req	-0.95	+0.05
TXSDU.req	-0.95	+0.05
TXHISDU.req	-0.95	+0.05
TXDCIULSDU.req	-0.95	+0.05

5.6 Limitations

5.6.1 Limitations on Channel ID Assignments in the DL

The channelID for downlink control channels must start at zero and have a lower value than the channelID of any downlink data channel. All channelIDs for both downlink control as well as data channels must be consecutive and with no gaps.

5.6.2 Limitations on Channel ID Assignments in the UL

The channelID for the UL-SCHSCH must be consecutive and below the ChannelD of the uplink control channels and they must start at zero. The uplink control ChannelIDs must be consecutive with no gaps and start right after the last UL-SCHSCH ChannelID.

5.7 Customer Extensions Support

For customers who wish to add custom features to the PHY, the hooks have been provided in the PHY structures and messages to make this possible.

Within the Tx and Rx vector structures there are offset fields that allow new parameter groups to be added without having to define them at the top level.

The `PHY_RXSTATUS.indication` has reserved fields that allow for additional customer defined indications to be added.

Finally, functions are provided for the customer to register handlers with the PHY to deal with customer defined indications.

5.7.1 Phy_Register_custom_indications

The prototype for this function is the following:

```
int Phy_Register_custom_features(int Phy_instance_id, int custom_ext_entry,
void *custom_ext_handler);
```

where:

- `Phy_instance_id` is the unique phy instance identifier that is returned by the `Create_phy()` command.
- `Cust_ext_entry` corresponds to one of the `PHY_RXSTATUS.ind` entries in the range of 200 to 255.
- `custom_ext_handler` is the pointer to the customer provided function that handles `Cust_ext_entry`.

This function only been defined and not been implemented.