2007:246 CIV

MASTER'S THESIS

Proof of Concept Implementation of UMTS Long Term Evolution

Jesper Hågstrand
Pär Karlsson
Olle Silverplats

Luleå University of Technology

MSc Programmes in Engineering
Computer Science and Engineering
Department of Computer Science and Electrical Engineering
Division of Computer Engineering

2007:246 CIV - ISSN: 1402-1617 - ISRN: LTU-EX--07/246--SE

Preface

This Master Thesis was written by Jesper Hågstrand, Pär Karlsson and Olle Silverplats in 2007 and is our final examination of the MSc Programme in Computer Science and Engineering. The project was carried out at TietoEnator in Ursviken, Skellefteå.

We would like to show gratitude for our supervisors Christer Åhlund, Erik Hellberg, Martin Larsson and our examiner Jingsen Chen. We would also like to thank TietoEnator for the opportunity to work on this project as well as supplying us with the needed resources. Thanks to Johan Forsman for his technical support during the project. And finally, thanks to Robert Brännström, Christian Nilsson, Axel Alatalo and others for reviewing our report.

Abstract

In recent years, mobile communication has evolved rapidly and demand for mobile devices with new and higher quality services is increasing. The existing 3G standard, Universal Mobile Telecommunication System (UMTS), is currently being upgraded with High Speed Packet Access (HSPA) to meet current demands. However, in the longer term this will not be sufficient.

The 3rd Generation Partnership Project (3GPP) has investigated the long term evolution of UMTS (LTE) to meet future demands and ensure the competitiveness of the standard. The objective is a radio access technology with higher data rates, lower latencies and optimal support for packet services such as multimedia, games and internet services. Telephony shall be supported by means of Voice over IP (VoIP) with at least as good quality as circuit switched telephony.

This thesis will cover the technology and architecture of LTE. The focus is on Evolved Universal Terrestrial Radio Access (E-UTRA) Layer 2 functionality and a proof of concept LTE system. The purpose is to demonstrate how user plane traffic is transferred through the system. The specifications of LTE are not yet finalized. Therefore, the implementation is based on work in progress specifications and meeting protocols from 3GPP work groups.

The prototype resulted in a system capable of handling multiple user equipments with streams in both uplink and downlink. In Layer 2, the prototype has a complete implementation of user plane data transfer in unacknowledged mode, featuring segmentation, concatenation and multiplexing. The prototype also supports retransmissions and basic scheduling.

Table of Contents

List of Figures

List of Tables

1 Concepts and Abbreviations

1.1 Concepts

3GPP LTE	3GPP project to improve the 3rd generation mobile standard UMTS.
Circuit switching	A communications paradigm in which a dedicated circuit between nodes and terminals are established.
Control plane	Parts of the system that handles control functionality.
E-UTRA	Radio access to E-UTRAN.
E-UTRAN	The radio access network for LTE.
Packet switching	A communications paradigm in which packets are routed between nodes over data links.
PDU	The packets produced by a sub layer, usually a header and one or more SDUs.
SDU	Packets served by a sub layer.
UMTS	3GPP 3rd generation cellphone technology standard.
User plane	Parts of the system that handles user generated traffic.

1.2 Abbreviations

3GPP	Third Generation Partnership Project
ACK	Acknowledgement
aGW	Access Gateway
AL	Alcatel-Lucent
AM	Acknowledge Mode
AMR	Adaptive Multi-Rate
AMBR	Aggregate Maximum Bit Rate
ARQ	Automatic Repeat Request
BCCH	Broadcast Control Channel
BCH	Broadcast Channel
BER	Bit Error Rate
CCCH	Common Control Channel
C-RNTI	Cell Radio Network Temporary Identifier
DCCH	Dedicated Control Channel
DDI	Data Description Indicator
DFT	Discrete Fourier Transform
DL	Downlink
DL-SCH	Downlink Shared Channel
DTCH	Dedicated Traffic Channel
DVB	Digital Video Broadcasting
eNB	E-UTRAN Node B
EPC	Evolved Packet Core
E-UTRA	Evolved Universal Terrestrial Radio Access
E-UTRAN	Evolved Universal Terrestrial Radio Access Network
GBR	Guaranteed Bit Rate.
GNU	GNU's Not Unix
GTP	GPRS Tunnelling Protocol
GTP-U	GPRS Tunnelling Protocol – User plane
HARQ	Hybrid ARQ
HSDPA	High Speed Downlink Packet Access

HSPA	High Speed Packet Access
ICC	InterDigital Communication Corporation
ID	Identity
IMIX	Internet Mix
IP	Internet Protocol
LI	Length Indicator
LTE	Long Term Evolution
MAC	Medium Access Control
MBMS	Multimedia Broadcast Multicast Service
MBR	Maximum Bit Rate
MCH	Multicast Channel
MCS	Modulation and Coding Scheme
MIMO	Multiple Input Multiple Output
MME	Mobility Management Entity
MTCH	Multicast Traffic Channel
NACK	Negative Acknowledgement
NDI	New Data Indicator
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
PBR	Prioritised Bit Rate
PCH	Paging Channel
PDCP	Packet Data Convergence Protocol
PDU	Packet Data Unit
PHY	Physical layer
PRB	Physical Resource Block
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase-Shift Keying
RACH	Random Access Channel
RLC	Radio Link Control
ROHC	Robust Header Compression
RRC	Radio Resource Control
RTT	Round Trip Time
S1-U	S1 for the user plane
SAE	System Architecture Evolution
SC-FDMA	Single Carrier – Frequency Division Multiple Access
SCTP	Stream Control Transmission Protocol
SDU	Service Data Unit
SDK	Software Development Kit
SN	Sequence Number
SNR	Signal-to-Noise Ratio
SUSE	Software– und System-Entwicklung
TEID	Tunnel Endpoint Identifier
TM	Transparent Mode
TTI	Transmission Time Interval
UDP	Universal Datagram Protocol
UE	User Equipment
UL	Uplink
UL-SCH	Uplink Shared Channel
UM	Unacknowledge Mode

UMTS	Universal Mobile Telecommunication System
VLC	VideoLAN Client
VoIP	Voice over IP
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network

2 Introduction

In a near future, mobile users equipped with a cellular phone or other mobile device shall be able to use high delay sensitive services, with a good and satisfying user experience. Such services include streaming video, IP telephony, teleconferencing and interactive games. Along with the delay sensitive services more elastic applications such as file sharing, email and Web services shall also be supported.

With UMTS, users are now offered new services through the introduction of the packet switched domain. In order to fulfil future user demands for the next ten years and beyond, 3GPP have investigated concepts for the UMTS Long Term Evolution (LTE). The objective is a packet optimized radio access technology with high data rates and low latencies. Focus lies on supporting packet switched services.

Currently deployed UMTS depends on circuit switching for voice telephony. With a lower latency and the overall higher performance of the packet switched service, circuit switched telephony is no longer needed. Instead VoIP is used in LTE.

LTE introduces a completely new radio access network (E-UTRAN) with major differences compared to general UMTS protocol architecture concepts, where the target is to significantly reduce system complexity.

2.1 Problem Description

A key challenge for E-UTRA is the new aspects for user and control plane. To highlight some of these new aspects, our work was to demonstrate how traffic is transferred through a LTE system in user plane. The focus lies on E-UTRA Layer 2, which introduces new services, functions and PDU structures. Some control plane aspects have been added such as limited scheduling and support for retransmission.

2.2 Purpose

The main purpose of this master thesis is to design, analyze and implement a proof of concept LTE system. The prototype demonstrates how data is transferred through the UE, the eNB and the aGW. The focus is on E-UTRA Layer 2 between the UE and the eNB. For All-IP network aspects tunnelling between the eNB and the aGW is supported. The prototype supports multiple UEs with multiple streams in both uplink and downlink.

Besides the LTE prototype, further studies are included in this thesis aiming at deeper insight in a particular LTE related subject. The parts of the further studies are PDU structure optimizations, HARQ characteristics including HARQ/ARQ interactions and scheduling schemes.

The thesis is mainly based on 3GPP LTE specifications and discussions.

2.3 Scope

This thesis covers E-UTRA Layer 2 and a proof of concept LTE of 3G system featuring user plane traffic in both uplink and downlink supporting multiple UEs with streams per UE.

The work with both the prototype and further studies mainly deals with E-UTRA Radio Protocol Layer concepts, where the focus of the prototype is on the user plane and E-UTRA Layer 2 while the further studies include more control plane concepts. For the constraints of LTE Prototype, see Chapter 4.1.

2.4 Methodology

The methodologies for the following phases were:

- **Studies**
  The studies are based on 3GPP LTE related work. The studies were performed during the entire project starting with a feasibility study before the design phase. Considerations to undergoing meetings and updated specifications have been taken alongside the design and implementation of the prototype.

- **Design**
  The design phase started after the feasibility studies where the main architecture was defined and then refined through iteration during the implementing phase as 3GPP specifications got updated.

- **Implementation**
  The prototype is implemented and debugged in Eclipse SDK V1.3.3 using C/C++ and GNU Compiler Base packet under the SUSE Linux operating system.

- **Testing**
  Testing was performed through out the development of the prototype with a dedicated testing phase at the end.

2.5 Thesis Outline

The thesis begins with an introduction of LTE in Chapter 3, *UMTS Long Term Evolution*. Chapter 4, *LTE Prototype*, presents the proof of concept LTE system, from design and implementation to test results and conclusions. In Chapter 5, *Further Studies*, we will go a bit deeper into three aspects of LTE, PDU structure optimizations, HARQ and scheduling schemes. We conclude with discussions in Chapter 6, *Discussion and Future Work*.

3 UMTS Long Term Evolution

With the deployment of UMTS packet data transfers are now operational with telephony services. UMTS is in line with the direction towards an evolved system fulfilling requirements for voice, video telephony, streaming media, interactive media, and lossless services such as file transfer and e-mail.

Currently, UMTS networks are being upgraded with High Speed Packet Downlink Access (HSDPA), which will increase data rate and capacity in downlink. The next step Evolved High Speed Packet Access (HSPA+) will support multiple antenna support, new modulation schemes and system protocol improvement [1].

Even with HSPA+, further improvements will continue for the next 10 years and beyond to ensure competitiveness of UMTS, and therefore concepts to fulfil these demands have been investigated for LTE. Main objective is a, high data rate and low latency radio access technology optimized for packet services.

LTE will introduce new access schemes on the air interface for uplink and downlink; i.e. OFDMA in downlink and SC-FDMA in uplink [2]. LTE will use these new transmission schemes and MIMO to support peak data rates of 100 Mbps in downlink and 50 Mbps in uplink within a 20 MHz spectrum with 2 receive antennas and 1 transmit antenna at the UE. Considering bandwidth usage compared to HSPA, the average throughput for LTE per MHz in downlink is 3 to 4 times better and in uplink 2 to 3 times better. Furthermore, VoIP shall be supported with at least as good radio and backhaul efficiency as voice traffic over UMTS circuit switched networks.

LTE introduces the network E-UTRAN and E-UTRA for radio access. E-UTRAN consists of base stations named eNBs providing protocol terminations towards the UE in user and control plane. Radio Protocol architecture for E-UTRA is very different from UMTS protocol concepts and aims at being simplified and less complex. The eNBs are interconnected with each other and are also connected to gateways (SAEs) and mobility management entities (MMEs) located in EPC.

The impact of the LTE system concerning overall network architecture is being investigated by 3GPP in the context of System Architecture Evolution (SAE). The study includes a vision for an all-IP network where heterogeneous network access shall be supported. The EPC is the backbone that will interconnect existing technologies through evolved interfaces and core entities.

LTE is challenged by other technologies such as Worldwide Interoperability for Microwave Access (WiMAX) [4], which is an alternative solution to fulfil future user demands and heterogeneous network access.

3.1 Orthogonal Frequency Division Multiple Access

The idea behind Orthogonal Frequency Division Multiple Access (OFDMA) is to split up one high data rate stream into several low data rate streams and transmit them in parallel [1][3]. Each low data rate stream doesn't necessarily need to be assigned to the same UE.

The digital multi-carrier modulation scheme OFDM divides the available spectrum in multiple closely spaced orthogonal sub-carriers. Each sub-carrier can then individually be assigned its own modulation scheme. In LTE the downlink modulation schemes are QPSK, 16QAM and 64QAM.

OFDM supports overlapping of sub-carriers without any interference between the carriers, by doing this almost 50% of the bandwidth is saved due to the orthogonality of the code compared to a non overlapping technique. See Figure 1.

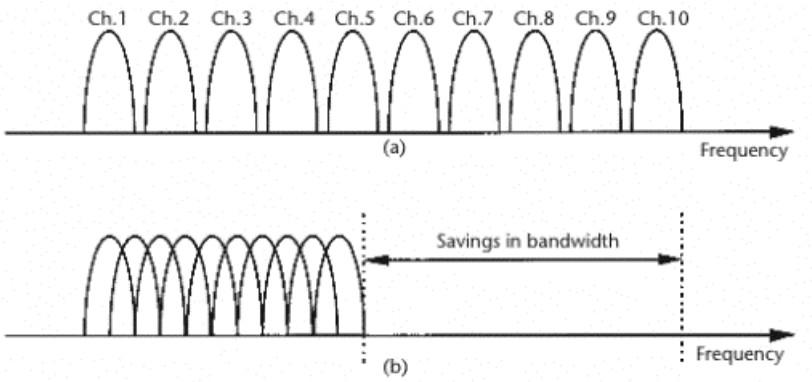

Figure 1. OFDMA modulation. [5]

Some benefits with OFDM are that it makes single frequency networks possible (good for broadcast applications), has MIMO support (see Chapter 3.3) and is very robust against multipath fading. Multipath fading is the result of radio signals reaching its target by two or more paths which can be caused by reflection from objects like buildings and mountains.

OFDM has become a popular scheme for wideband digital communication systems. Other known technologies using OFDM are WLAN, WiMAX and broadcast services like Digital Video Broadcasting (DVB).

OFDMA allows multiple UEs to share the same bandwidth. This is done by assigning a subset of sub-carriers to different UEs allowing multiple low data rate streams to different UEs at the same time.

3.2 Single Carrier Frequency Division Multiple Access

SC-FDMA is a modified version of OFDMA. As in OFDMA, the transmitters in an SC-FDMA system uses orthogonal sub carriers to transmit information symbols. However, they transmit the sub carriers sequentially and not in parallel in contrast to OFDMA signals. Relative to OFDMA, this arrangement reduces considerably the envelope fluctuations in the transmitted waveform [6]. Therefore, SC-FDMA signals have inherently lower peak-to-average power ratio than OFDMA signals.

The transmitter of an SC-FDMA system converts a binary input signal to a sequence of modulated sub carriers. The signal processing operations have much in common with the ones for OFDMA where the difference is the use of the Discrete Fourier Transform (DFT) in the SC-FDMA transmitter and the Inverse DFT in the SC-FDMA receiver. Because of this major difference SC-FDMA is sometimes referred to as DFT-spread OFDMA.

3.3 Multiple Input Multiple Output

In a Multiple Input Multiple Output (MIMO) system both sides of the communication link has multiple receiving and transmitting antennas. Each antenna uses the same time-frequency space. When a data stream is transmitted it can be divided between the multiple antennas to increase the transfer rate of the stream. MIMO can also be used to allow multiple UEs transmitting and receiving simultaneously.

In LTE, the basic configuration are considered to be two receiving and two transmitting antennas per cell on the eNB while there are two receiving and one transmitting antennas per UE. Support for up to 4x4 (4 receiving, 4 transmitting) antennas is also considered.

3.4 System Architecture

The overall architecture of UMTS LTE [3] consists of multiple eNBs providing the UEs access to E-UTRAN by means of E-UTRA. The eNBs are interconnected with each other by means of an X2 interface (see Figure 2). MME/SAE Gateways provides the connection between the eNBs and the EPC and is done through a S1 interface. Both the X2 and S1 interface supports a many-to-many relation. The eNBs are responsible for routing of user plane data between the UE and SAE Gateway. They also provide dynamic scheduling of UEs and data streams. Other important services the eNBs provide are header compression and encryption of user data streams.

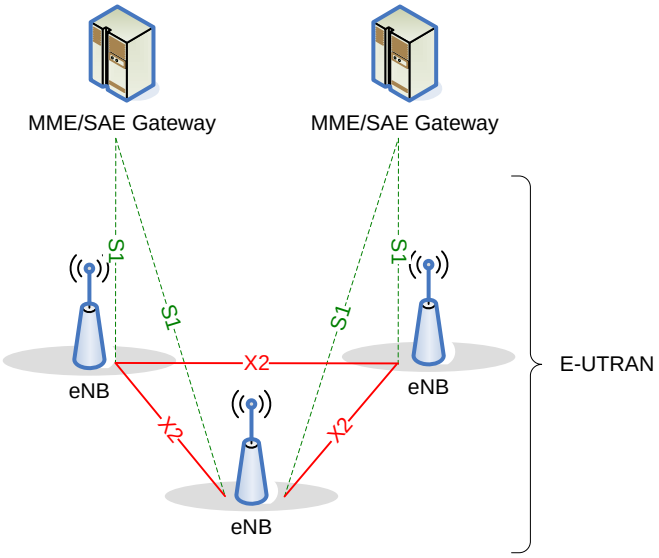

Figure 2. E-UTRAN overview. [11]

3.5 Layers for E-UTRA

Layer 1 refers to the physical layer and offers information transfer services to the higher layers by means of transport channels. They describe how and with what characteristics data is transferred over the radio interface. There are different transport channels in uplink and downlink.

Downlink Transport Channels:

- **Broadcast Channel (BCH)**
  Can be used to broadcast messages to all UEs within the coverage of a cell.
- **Downlink Shared Channel (DL-SCH)**
  A channel that is shared between all UEs within a cell. The channel has support for HARQ and dynamic as well as semi-static resource allocation.
- **Paging Channel (PCH)**
  A channel that is characterized by support for broadcast and UE discontinuous reception to save battery life.
- **Multicast Channel (MCH)**
  Broadcast channel with support for Single Frequency Network combining of MBMS transmission on multiple cells.

Uplink Transport Channels:

- **Uplink Shared Channel (UL-SCH)**
  A channel that is shared between all UEs within a cell. The channel has support for HARQ and dynamic as well as semi-static resource allocation.
- **Random Access Channel (RACH)**
  A channel that is used by UEs to Access E-UTRAN. The channel is characterized by limited control information and collision risk.

Layer 2 is divided into three sub layers, Medium Access Control (MAC), Radio Link Control (RLC) and Packet Data Convergence Protocol (PDCP). Above the PDCP sub layer in Layer 3 we find the Radio Resource Control (RRC) sub layer.

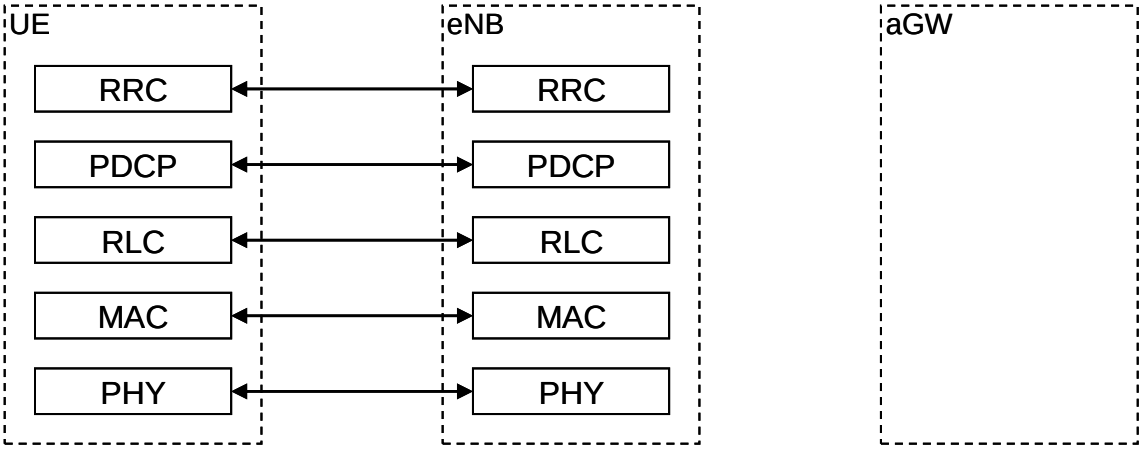

Figure 3. E-UTRA Layers.

While the transport channels describe how and with what characteristics data is transferred, the logical channels between the MAC and RLC sub layers tells what is transferred. Each logical channel type is defined by what kind of information that is transferred. The logical channels can be divided into two groups, control channels and traffic channels. The control channels are used for transfer of control plane information and the traffic channels are used for transfer of user plane information.

Control Channels:

- **Broadcast Control Channel (BCCH)**
  A downlink channel for broadcasting system control information.
- **Paging Control Channel (PCCH)**
  A downlink channel that transfers paging information. This channel is used when the network does not know the location cell of the UE.
- **Common Control Channel (CCCH)**
  Uplink channel for transmitting control information between a UE and the network. This channel is used by UEs without an RRC connection with the network.
- **Multicast Control Channel (MCCH)**
  A point-to-multipoint downlink channel used for transmitting MBMS control information from the network to UEs for one or several MTCHs. This channel is only used by UEs that receive MBMS.
- **Dedicated Control Channel (DCCH)**
  A point-to-point bi-directional channel that transmits dedicated control information between a UE and the network. Used by UEs having an RRC connection.

Traffic Channels:

- **Dedicated Traffic Channel (DTCH)**
  This channel is dedicated to one UE and can be used in both uplink and downlink.
- **Multicast Traffic Channel (MTCH)**
  A one to many channel for transmitting traffic data from the network to many UEs, this channel is only for downlink traffic.

The mapping between the logical channels and the transport channels are performed in the MAC sub layer. The mapping in uplink and downlink is illustrated in Figure 4 and Figure 5. In Figure 5 grey means the mapping is not decided yet.

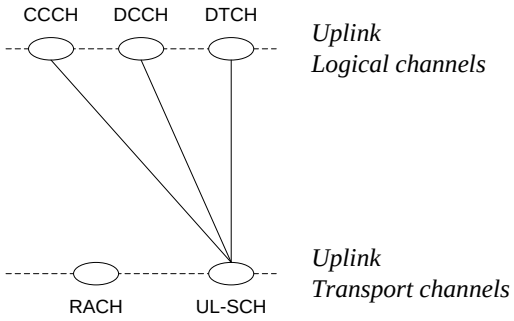

Figure 4. Mapping between logical and transport channels in uplink. [11]

In uplink all logical channels used for uplink transmission, CCCH, DCCH and DTCH, are mapped to the transport channel UL-SCH.

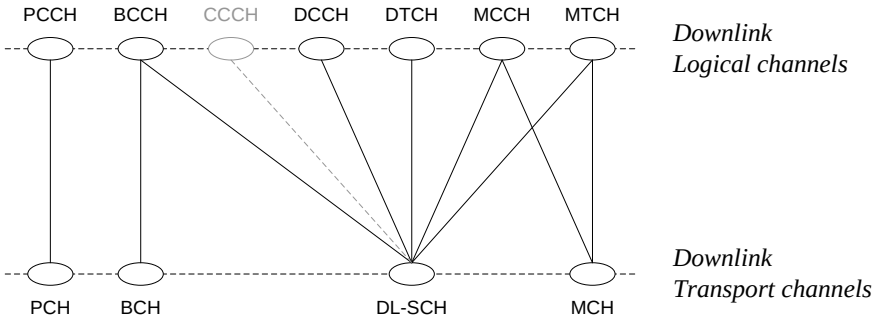

Figure 5. Mapping between logical and transport channels in downlink. [11]

For downlink the logical channels associated with multicast, MCCH and MTCH, can be mapped to MCH or DL-SCH. The broadcast logical channel BCCH can also be mapped to DL-SCH or to the broadcast transport channel BCH. DCCH and DTCH are mapped to DL-SCH, and PCCH is mapped to PCH.

3.5.1 Medium Access Control

The main functional blocks in the MAC sub layer are scheduling/priority handling, multiplexing and HARQ. Multiplexing is done for SDUs belonging to separate or the same logical channels. The HARQ located in the MAC sub layer ensures delivery between peer

entities at physical layer. HARQ support error correction when delivery of data fails. If the error correction also fails retransmissions in both downlink and uplink are performed.

Main services and functions provided by the MAC sub layer:

- Mapping between logical and transport channels.
- Multiplexing/Demultiplexing of SDUs.
- Traffic volume measurement reporting.
- Error correction through HARQ.
- Priority handling between logical channels of one UE.
- Priority handling between UEs by means of dynamic scheduling.
- Transport format selection.

3.5.2 Radio Link Control

The RLC sub layer supports three types of data transmission modes, Acknowledge Mode (AM), Unacknowledged Mode (UM) and Transparent Mode (TM). For AM, Automatic Repeat Request (ARQ) is used for retransmissions. ARQ can also be used for status report signalling and for resetting the transmitting and receiving RLC entities.

RLC sub layer also supports segmentation and concatenation of SDUs. When a RLC PDU does not fit entirely into a MAC SDU the RLC SDUs will be segmented into smaller pieces. Resegmentation of PDUs can be performed when a retransmitted PDU does not fit into a MAC SDU. The number of resegmentations is unlimited. SDUs and segments of SDUs are concatenated into PDUs.

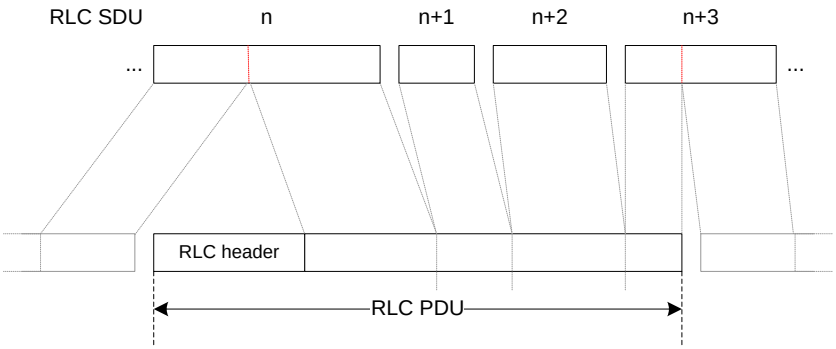

Figure 6. RLC SDU segmentation. [11]

Main services and functions provided by the RLC sub layer:

- Transfer of upper layer PDUs supporting AM, UM and TM.
- In sequence delivery of upper layer PDUs except at handover in the uplink.
- Duplicate detection.
- Segmentation for dynamic PDU size, no need for padding.
- Resegmentation of PDUs that need to be retransmitted.
- Concatenation of SDUs for the same radio bearer.
- Error Correction by retransmissions through ARQ.
- Protocol error detection and recovery.
- SDU discard.
- Reset.

3.5.3 Packet Data Convergence Protocol

The PDCP sub layer can perform robust header compression (ROHC) to improve transmission for latency sensitive data such as VoIP and video telephony. It also has ciphering capabilities for security.

Main services and functions provided by the PDCP sub layer:

- Transfer of upper layer data from NAS to RLC sub layer and vice versa.
- Header compression/decompression, ROHC only.
- Ciphering.
- Duplicate detection of lower layers SDUs.

3.5.4 Radio Resource Control

The RRC sub layer handles the control signalling of Layer 3 between the UE and the eNB. It makes handover decisions based on measurement reports from the UE and executes transmission of the UE context from the source eNB to the target eNB during the handover. The RRC sub layer is also responsible for setting up and maintenance of radio bearers.

Main services and functions provided by the RRC sub layer:

- Broadcast of System Information.
- Paging.
- Security functions including integrity protection for RRC messages.
- Establishment, configuration, maintenance and release of point to point Radio Bearers.
- Inter-cell handover.
- UE cell selection and reselection and control of cell selection and reselection.
- Context transfer between eNBs.
- QoS management functions.
- UE measurement reporting and control of the reporting.

3.6 S1 Interface

The S1-U interface provides a non guaranteed user plane data transfer between the eNB and the aGW. On the network layer UDP/IP is used and on top of that GTP-U is used to transport user plane data between the eNB and the aGW as illustrated in Figure 7.

The S1 control plane interface (S1-MME) is defined between the eNB and the MME. The control plane protocol stack of the S1 interface is built on IP transport, similarly to the user plane but for the reliable transport of signalling messages SCTP is added on top of IP.

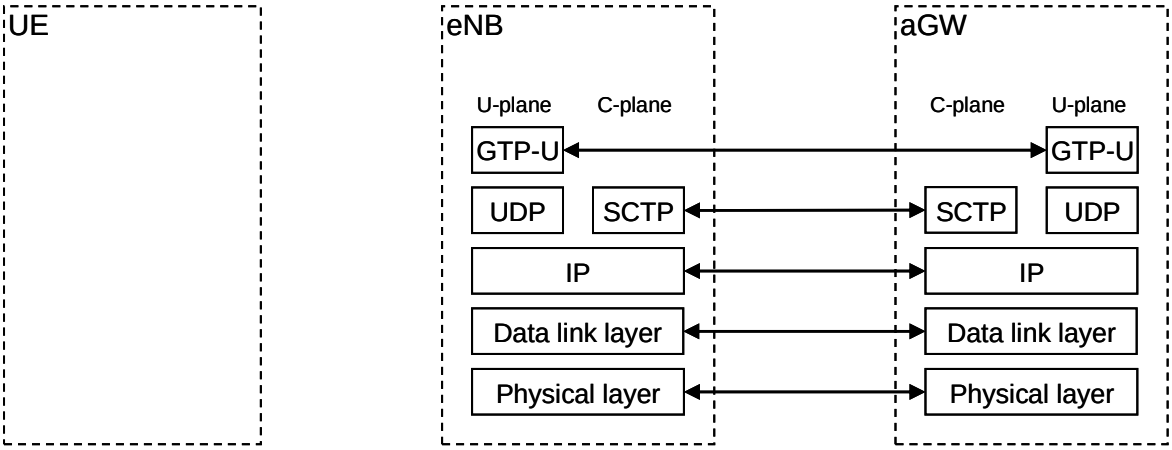

Figure 7. S1 Interface for both the user plane and control plane.

3.6.1 GPRS Tunnelling Protocol for User Plane

The GTP-U carries SDUs and signalling messages through tunnels between GTP-U Tunnel Endpoints. A Tunnel Endpoint Identifier (TEID) in the GTP-U header indicates which tunnel a specific SDU belongs to. By using the TEID multiplexing/demultiplexing of different UEs, different packet protocols and different Quality of Services (QoS) are supported by the GTP-U layer between a given pair of Tunnel Endpoints.

4 LTE Prototype

The primary goal with the prototype is to implement the user plane functionality of Layer 2. The prototype is also built with further development toward a complete LTE system in mind. The core of the system is PCs running Linux for flexibility, low cost and fast development. The prototype is implemented in C++ using an object oriented approach.

4.1 Constraints

The prototype is limited to a single eNB and aGW to simplify the prototype setup and allow us to focus on the user plane operation. Each Protocol in Layer 2 is limited to the user plane functionality.

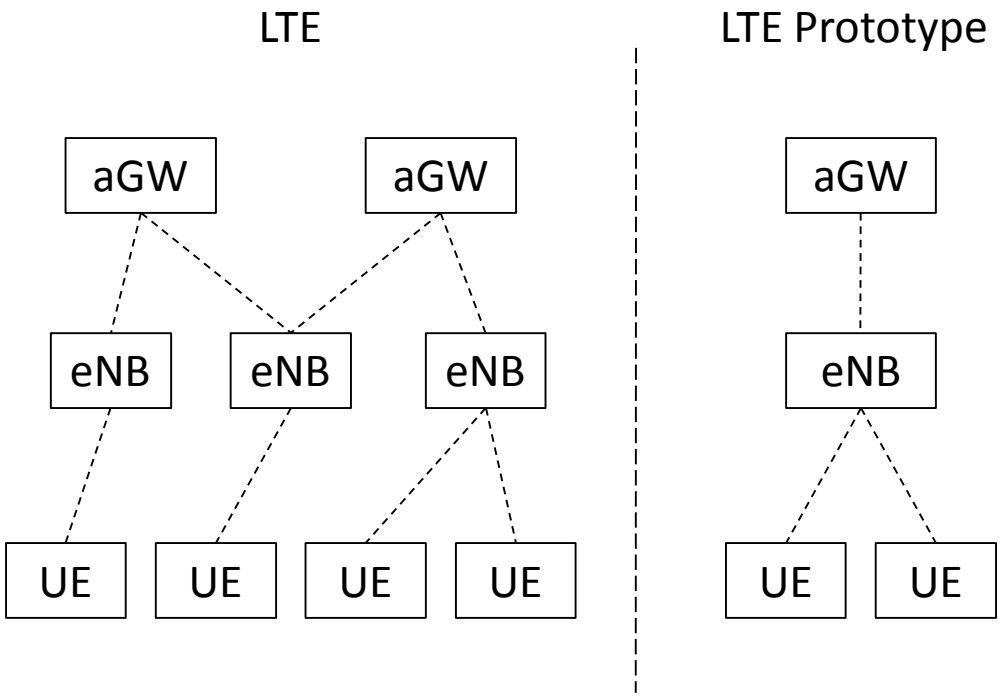

Figure 8. Comparison between a full LTE system and our prototype.

The PDCP sub layer adds a sequence number and forwards the data. No header compression is done. The RLC sub layer is a limited Unacknowledged Mode implementation meaning there is no ARQ or other control signalling done. The RLC sub layer has in order delivery, duplicate detection and segmentation but no resegmentation. The MAC sub layer has multiplexing and a limited HARQ with retransmissions. The MAC sub layer has static round robin scheduling between UEs to allow the system to handle more than one UE. The GTP-U layer between the eNB and the aGW is limited to tunnelling data between one eNB and one aGW.

Since the focus of the project is on Layer 2 the physical layer called PHY in the prototype is a stub and does not fulfil 3GPP specifications.

4.2 Timing

Since the MAC sub layer is on top of a radio link physical layer it needs to transfer Transport Blocks down to the physical layer at a determined interval called Transmission Time Interval (TTI).

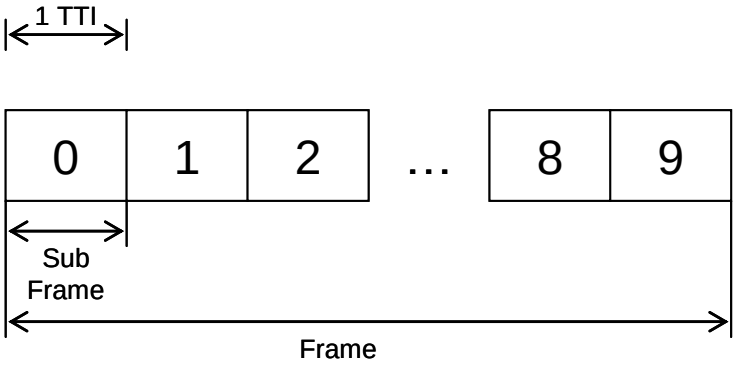

Figure 9. Frame structure.

The time is structured as a series of frames. Each frame consists of ten sub frames and each sub frame is one TTI long. In each sub frame two transport blocks can be transmitted in both uplink and downlink. The frame structure is based on the generic frame structure as specified in [3].

To have the MAC sub layer transmit transport blocks on a per TTI basis we need a timer to trigger the transmissions. Since the timer available in Linux only allows one timer per process we use it to run a TTI Clock. The TTI Clock runs in its own thread and is triggered by the Linux timer with an interval of one TTI. Each object that needs to be triggered can then register itself as a timer listener by creating a TTI Timer object. As parameters to the TTI Timer constructor it supplies itself (as a TTI Timer Listener), the interval in TTIs and priority. The priority is used to determine which objects trigger-function to call first when multiple timers get triggered the same TTI. The function call is done in the TTI Clock thread so it's up to the listener to do the work in a different thread if needed. This is done as we want to minimize delays when triggering time critical objects as we will be operating with a TTI close to the precision of the OS process scheduling.

4.3 Buffering

To utilize the memory efficiently while handling the payload during segmentation and concatenation we have a buffer implementation that allows those operations without copying the data. The buffer is a circular buffer that allocates all the memory needed at start up. Once the system is up and running the buffer class virtually allocates memory for new data within the pre-allocated space as seen in Figure 10.

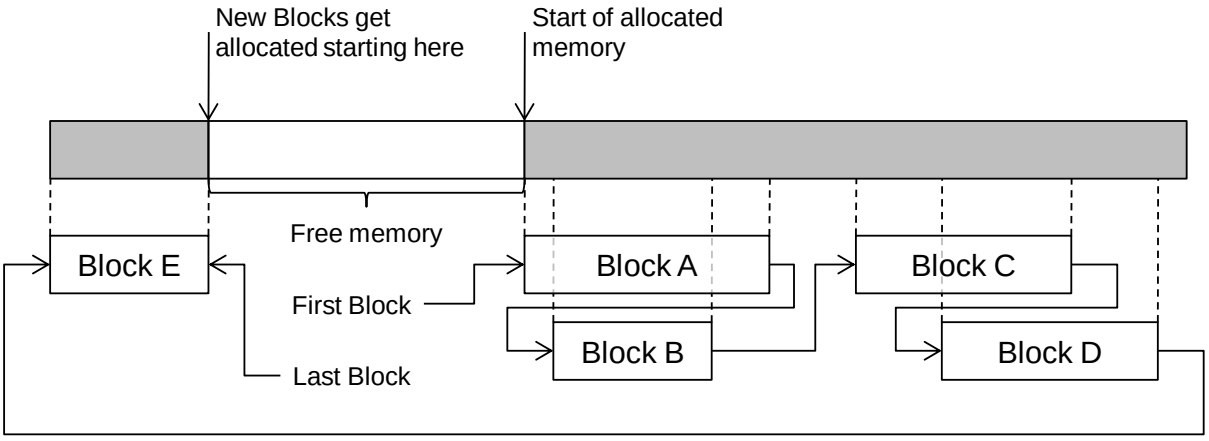

Figure 10. Buffer and virtual block allocation.

Virtual allocation is stored as virtual blocks in a linked list starting at "First Block" and ending at "Last Block" in Figure 10. Each block has a pointer to where in the buffer it is located and a length. The linked list is sorted by the pointers position in the buffer. New allocation is done after the last block. However only blocks where new data is going to be stored is added at the end. Blocks can also be created as part of another block or as a duplicate of a whole block in which case the block is added at the appropriate position in the linked list.

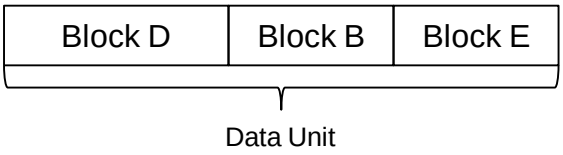

Figure 11. A data unit consisting of the three blocks D, B and E.

To form the complete sets of data that the layers in the prototype work on blocks are grouped into virtual data units. A data unit can consist of any number of blocks located at any position in the buffer. Allocation of new memory is done by creating a new data unit object. The data unit will construct a block which gets allocated on the buffer. To make a copy or extract parts of a data unit a different constructor is used to reference the copied data unit. The data units can be concatenated using a concatenation operator implemented in the data unit. Data units allow all the needed operations for working on the payload without directly manipulating the blocks.

Once a data unit is no longer needed, destroying the data unit will remove its blocks from the buffer. Note however that all data units that occupy the same memory in the buffer needs to be destroyed before it is considered unallocated and unallocated memory can only be used if it is located after the last block and before the first block. This allows fast allocation and de-allocation of new data as new allocation is simply added at the back while de-allocation just removes the block from the linked list.

4.4 System Architecture

This chapter describes the architecture of the overall system and the nodes illustrated in Figure 12 below.

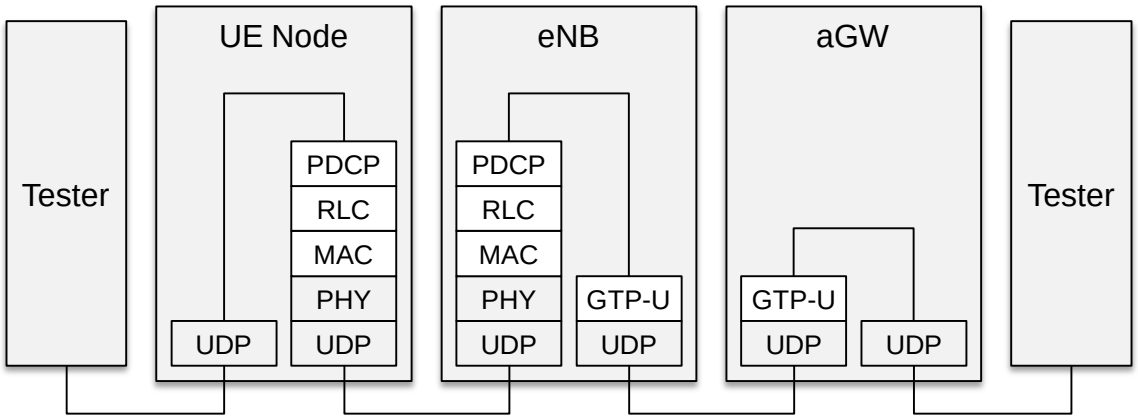

Figure 12. LTE prototype overview.

The prototype is divided into three main applications, one UE node that runs several UE entities, one eNB and one aGW. Each application can be run on the same or different machines. We run the system on an All IP Network where the radio link between the UE and the eNB has been replaced with UDP and GTP-U is used on top of UDP between the eNB and the aGW. Once the system is up and running data streams can be transferred through the system by transmitting data to the UDP interfaces on the UE or the aGW node and receiving them at the other end.

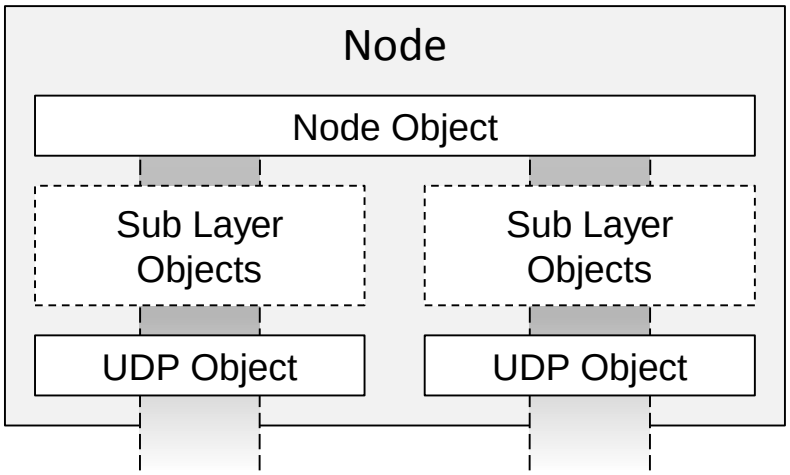

Figure 13. Node layer objects structure.

Each node has a similar structure with a main node object that maps the incoming channels to outgoing channels. This object is also responsible for constructing all the layers at start up.

The UDP objects maps channels to UDP ports. All communication between nodes and external application through these UDP objects. The UDP objects have two threads each. One that listens on the assigned UDP ports and one that runs the incoming data through the upper layers. In between the UDP objects and node object we find the sub layer objects which contain the implementations of the sub layer functionality. Each of these objects has references to the objects of the sub layers it needs to communicate with, commonly the sub layer objects above and below it.

4.4.1 UDP Object

We have an UDP object that provides the upper layers with channels mapped to UDP sockets. Since the number of channels and even the number of objects vary between the

different nodes it can have any number of channels as well as any number of objects above it. Each channel is mapped to one UDP socket and all upper layer objects receive all the data.

It has two threads, one for listening on the UDP sockets and one that delivers the incoming data to the upper layers through a function call. Transmission of data is done by any external thread calling the send function. The external thread may be the working thread of another UDP object. On the transmitting side upper layers transmit data on the UDP channels which is then transmitted on the associated UDP socket. On the receiving end the listener thread puts the incoming UDP data in the buffer, queues it and notifies the work thread that new data is available. The work thread takes data from the queue and delivers it up to the upper layers. Once the upper layers are done processing the data the work thread will continue with the next item in the queue or go to idle if it's empty. This means that unless one of the other layers hands the payload over to another thread all the processing of the payload in the node will be executed by the work thread of the receiving UDP object.

4.4.2 Node Operation

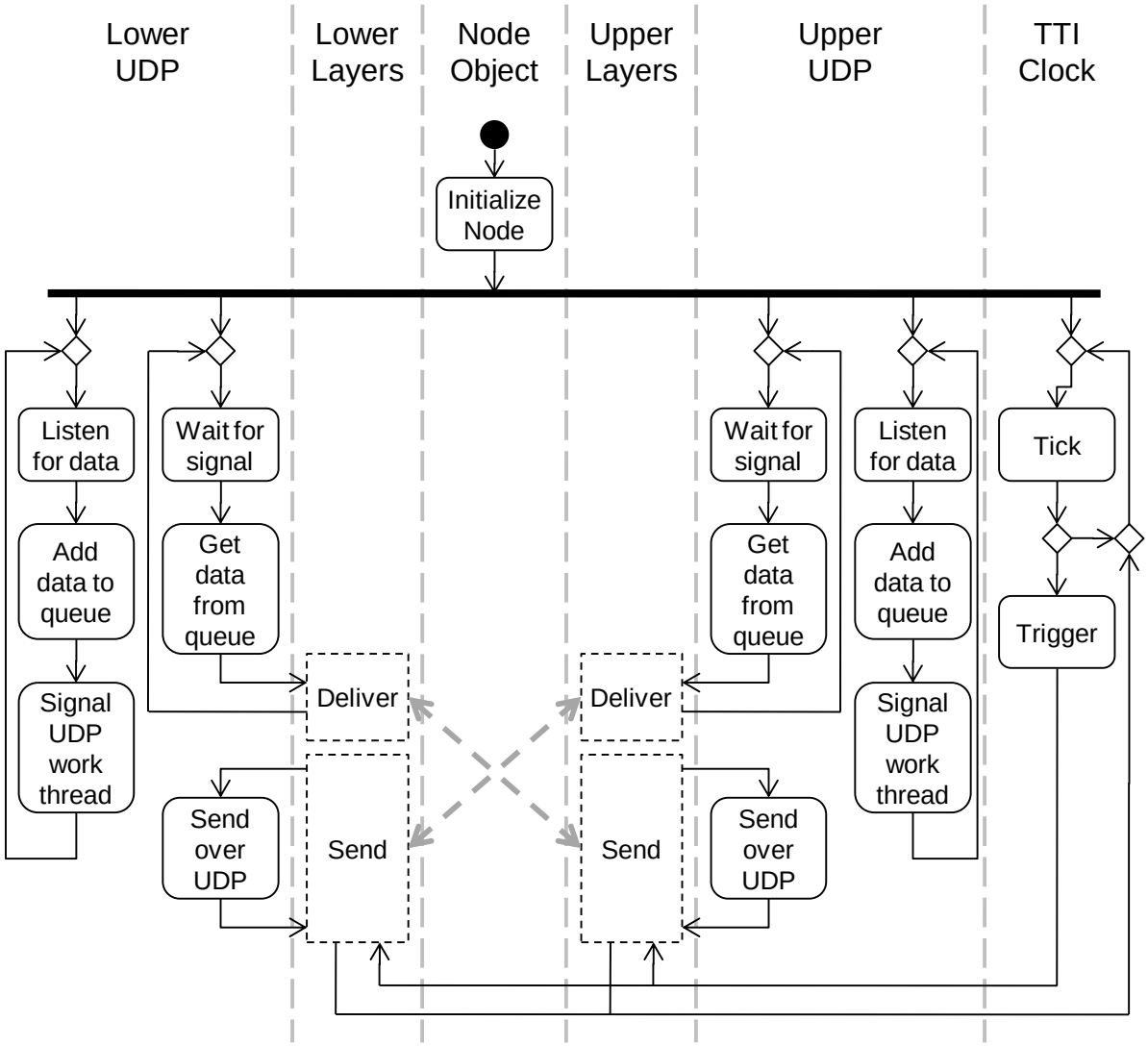

Figure 14. Generic Node operation.

As the node is started all objects are created and all threads are started. After the initialization of the node all the threads, except the TTI Clock is idling waiting for incoming data. The TTI Clock is constantly counting through the frames and calling any timers if they time out. Once data is received at one of the UDP objects its listen thread will store the incoming data on the buffer and signal the working thread of that same UDP object. The working thread calls the upper layers deliver method to hand over the incoming data. The layers then do their part of processing the data. At some point the data will be received at the node object through its deliver function call. The node object will determine where it's supposed to be sent and call the uppermost layer objects send function to have it sent. Just like the delivery chain will move the data up through the layers the send chain moves the data down through the layers until it reaches the UDP object at the bottom which transmits the data over UDP to the next node.

The layers may continue the processing in the working thread or queue it for another thread to continue the work. Transmission of data can also occur without any incoming data if the TTI clock thread triggers a function who initiates the transmission of data. One example of this is the synchronization messages sent from the eNB to the UE where one of the layers gets triggered to transmit the message.

4.5 User Equipment Node

The UE node runs the UEs in the system. The UE node can be seen in Figure 15 below.

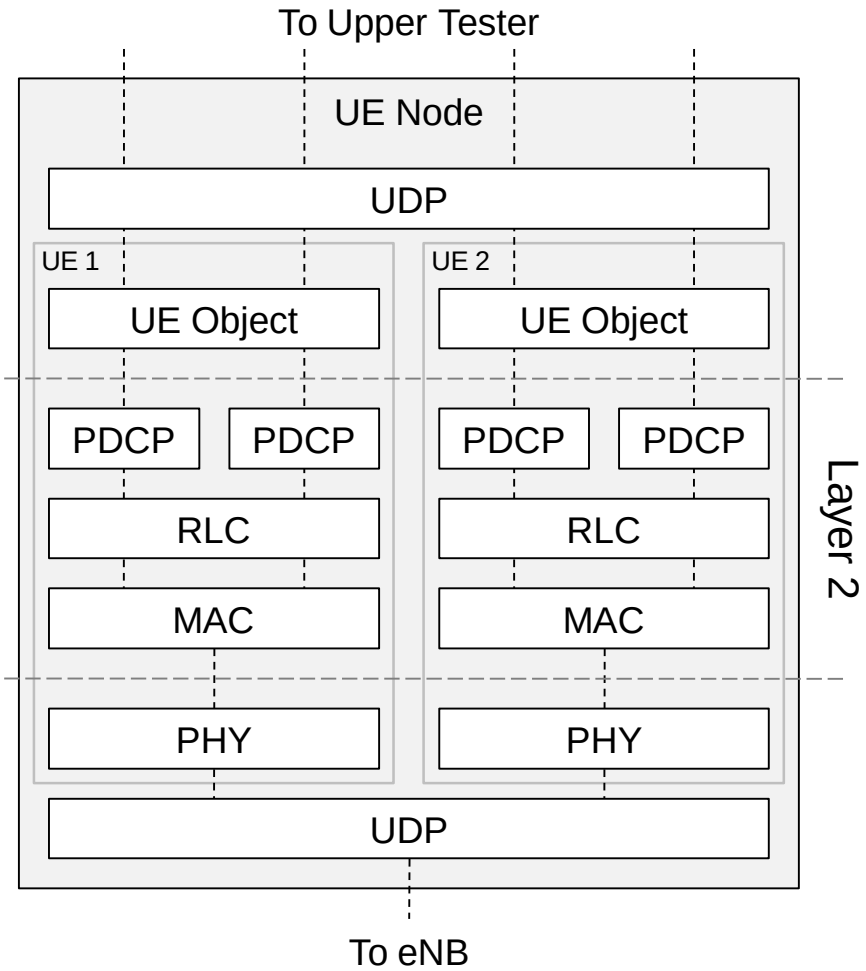

Figure 15. UE node layer objects.

The UDP towards the eNB forwards all data to all UEs and the PHY object determines if incoming data is relevant to the UE. The UE node also has one TTI clock for the whole node with each UE registering its own timers. Other than the UDP each UE has its own entities of all the layers. This makes each UE independent at Layer 2. The PHY object is responsible for getting the Transport Blocks and all the control signals transmitted to and from the eNB through the UDP interface instead of a real physical layer. It is however not a complete simulation of a real physical layer.

At Layer 2 each UE has one MAC and RLC object which takes care of all the respective sub layer functionality. At the PDCP sub layer there is one PDCP object for each logical channel. The same object takes care of both uplink and downlink on that channel. To map the streams to and from the lower tester against the logical channels we have an UE object for each UE. The UE object is also responsible for setting up all the UE specific objects at start up.

4.6 E-UTRAN Node B

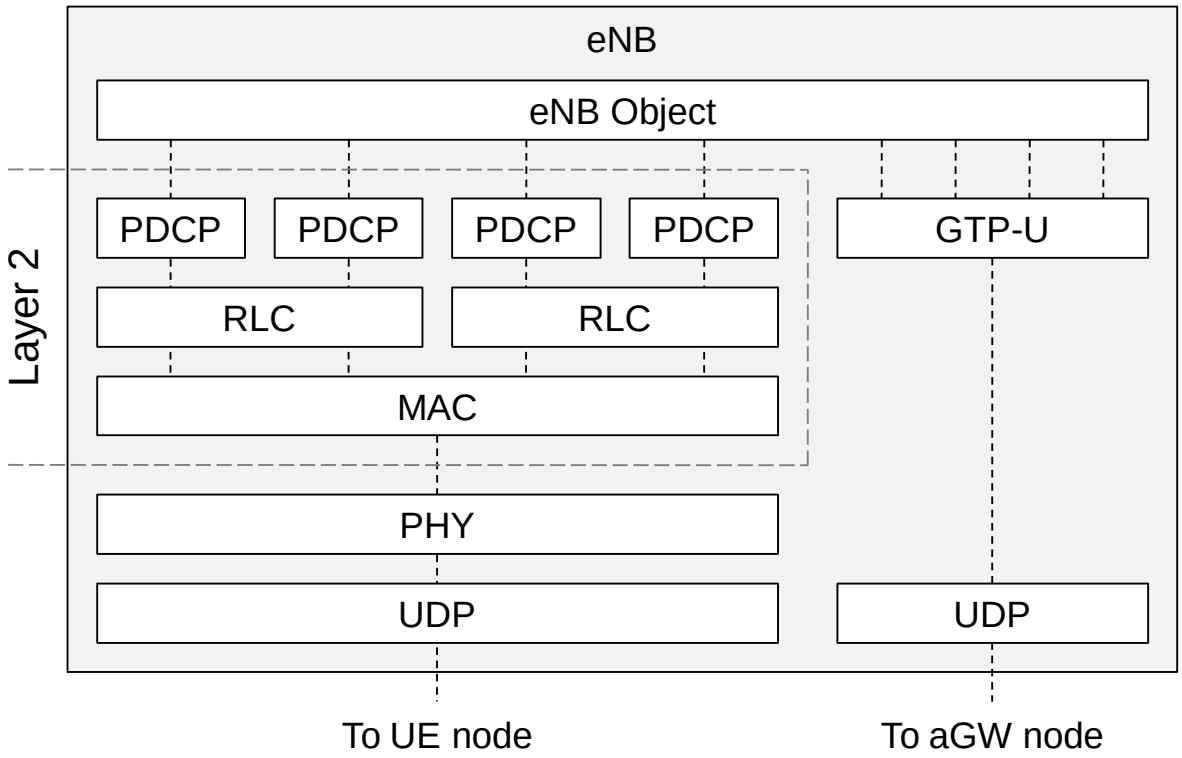

Figure 16. eNB layer objects.

The eNB has one UDP towards the UE node and one towards the aGW node. The eNB objects are illustrated in Figure 16. On the eNB side we have only one MAC object. The scheduler in the MAC schedules the UEs in both uplink and downlink. Just like on the UE side we have one RLC object for each UE and one PDCP object per logical channel. Towards the aGW it has the GTP-U object.

4.7 Access Gateway Node

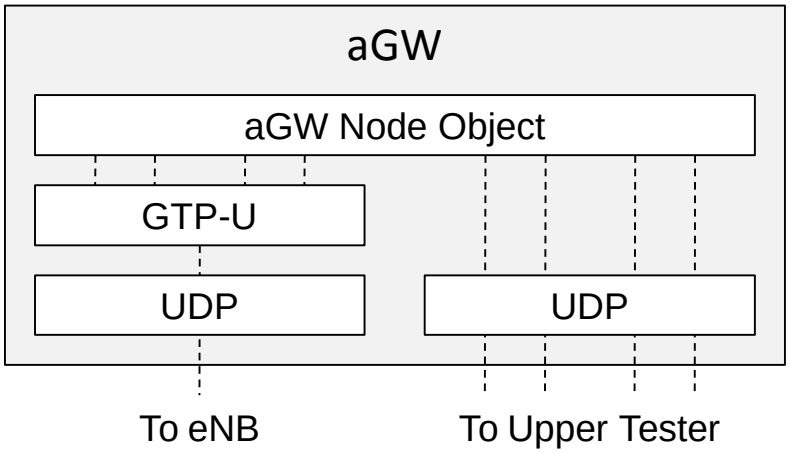

Figure 17. aGW node layer objects.

Since the aGW only has the GTP-U protocol it is not as complex as the UE and the eNB. The aGW objects are illustrated in Figure 17. It has the GTP-U towards the eNB whose streams are mapped to ports on the UDP towards the upper tester.

4.8 Layer 1 for E-UTRA

The main purpose of Layer 1 is to bridge the UE and the eNB. We have one sub layer of Layer 1 called PHY that takes care of all the control functionality needed to support Layer 2 in user plane. The interface between the MAC and the PHY sub layer has methods for each of the control signals as well as transmitting and receiving transport blocks.

4.8.1 Physical Sub Layer

The PHY sub layer takes the transport blocks and control signals delivered by the MAC sub layer and puts them in messages delivered to the UDP sub layer. It reads messages delivered by the UDP sub layer and delivers transport blocks and control signals found in messages up to the MAC sub layer. It is also responsible for synchronizing the TTI clock between the UE and the eNB.

The PHY sub layer provides the MAC sub layer with these services:

- Transfer of transport blocks.
- Signaling of grants.
- Signaling of acknowledgements.

Two UDP channels are used by the PHY sub layer, one for data messages and one for control messages. There is one data message which is used to transfer the transport blocks and three control messages, one for each type of signaling, synchronization, acknowledgement and scheduling grants.

4.8.1.1 Data Message

The data message is used to transfer user plane data between the UE and the eNB. In downlink, data needs to be received at the correct UE and the HARQ process. In uplink the transport block needs to be associated with the correct UE. This is achieved by adding a header with a UE ID and HARQ Process ID to the Transport Block. The same message format seen in Figure 18 is used in both uplink and downlink.

Figure 18. PHY data message.

The HARQ Process ID and the UE ID are signalled by the MAC sub layer as it hands over the transport block.

4.8.1.2 Synchronization Message

The PHY sub layer is responsible of transmitting and receiving synchronization messages to keep the TTIs as close to synchronized as possible on the two nodes. Synchronization messages are sent at the beginning of each frame and the UE node adjusts its TTI clock if the synchronization message is not received when expected. The precision is limited by the scheduling in the OS.

4.8.1.3 Acknowledgement Message

The PHY object is also responsible for transferring the control signalling needed to support HARQ retransmissions and scheduling. For HARQ retransmissions, the ACK/NACK message is sent when transport blocks are received. Figure 19 shows what the message looks like.

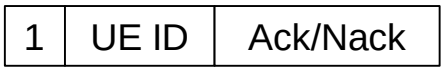

Figure 19. PHY ACK/NACK message.

4.8.1.4 Grant Message

For scheduling in downlink we don't need any extra messages since the UE finds all that is needed in the header added to the transport block and the transmitting eNB is the one who makes the scheduling decisions. In uplink however we need to transmit a grant message so the UE node knows who is allowed to transmit each TTI. Figure 20 shows the grant message.

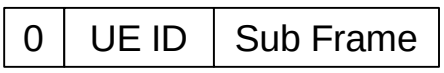

Figure 20. PHY grant message.

4.9 Layer 2 for E-UTRA

In Layer 2 all the main LTE functionality of the prototype is found. At the PDCP sub layer there is one PDCP object per logical channel. The RLC sub layer has one RLC object per UE

at both the UE and the eNB node while the MAC sub layer has one MAC object per UE on the UE node and one MAC object for all UEs on the eNB.

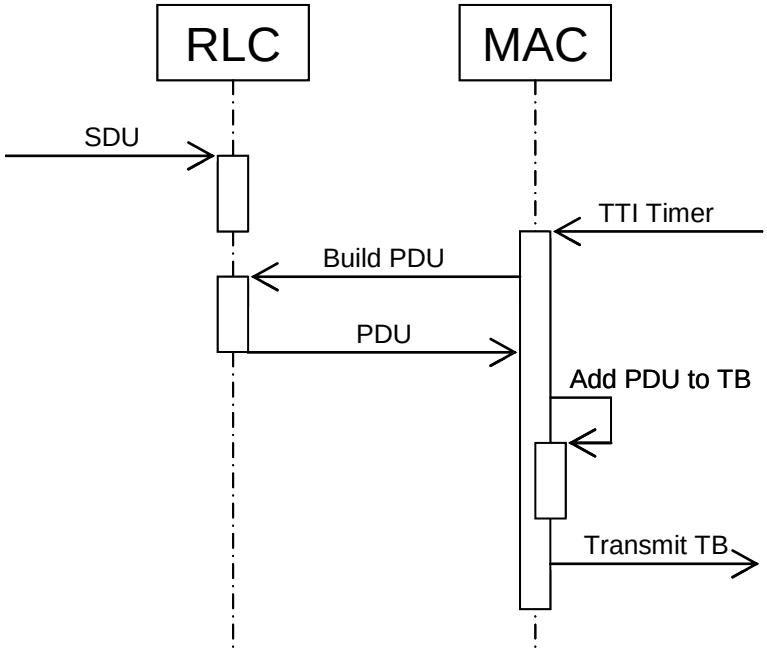

Figure 21. MAC initiated RLC/MAC interaction approach.

Incoming data is processed and handed over in a linear fashion from layer to layer. The interaction between RLC and PDCP sub layer works in the same way when transmitting data. The interface between the RLC and the MAC sub layer is more complicated though as the MAC sub layer only delivers data down to the PHY sub layer a predefined number of times each TTI.

Between the RLC and MAC sub layer we instead queue all the RLC SDUs at the RLC sub layer and let the MAC decide when it's time to build PDUs from them (Figure 21). The MAC requests PDUs from the RLC sub layer when it is scheduled to transmit. Since the RLC has all the SDUs on queue it takes as much data as possible up to the defined size specified by the MAC sub layer in the request and builds a PDU out of it. The MAC sub layer can then after receiving each PDU decide to request more PDUs or add padding if there is more space left in the transport block. See Figure 21 for a simplified sequence diagram of the interactions.

4.9.1 Packet Data Convergence Protocol

The PDCP sub layer in the prototype transfer data between the RLC sub layer and the node object. When receiving data from the node object a PDCP header is added consisting of a two byte long sequence number before delivering the packet to the RLC sub layer. When data is delivered from the RLC sub layer to the PDCP sub layer the PDCP header is removed before the packet is delivered to the node object.

Since the RLC sub layer handles in sequence delivery to the PDCP sub layer and the prototype doesn't support handover, the sequence number is a stub. The PDCP sub layer is based on [3] and [7].

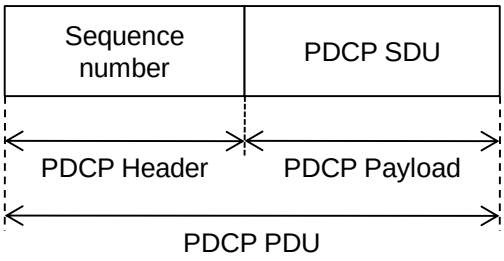

Figure 22. PDCP PDU structure.

4.9.2 Radio Link Control

Each RLC object can handle up to 16 uplink and downlink streams simultaneously. The RLC sub layer uses dynamic PDU size to build each PDU accordingly to the requested size by the lower layer. Each PDU can have multiple SDUs and segmentation of SDUs is supported for an optimal fit (no padding). The length indicator for each SDU is fixed at 11 bits and therefore the biggest possible upper layer SDU the RLC sub layer can handle is 2047 bytes. The RLC sub layer in the prototype only supports unacknowledged mode. RLC sub layer services and functions are based on [3].

Main services provided by the RLC sub layer to upper layers:

- In sequence delivery of upper layer PDUs.
- Transfer of upper layer PDUs supporting UM.

Main services provided by the RLC sub layer to lower layers:

- Dynamic PDU size.

Main functions:

- Duplicate detection.
- Segmentation for dynamic PDU size, no need for padding.
- Concatenation of SDUs for the same radio bearer.

4.9.2.1 PDU Structure

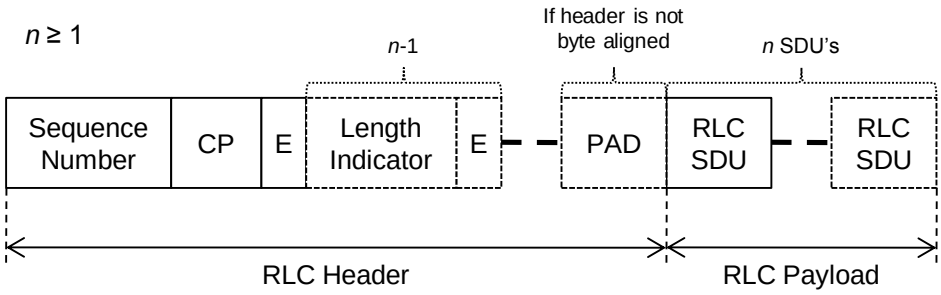

Figure 23. RLC PDU structure.

The RLC header always consists of a Sequence Number, Complete/Partial (CP) field and an Extension bit (E). More header fields may follow depending on the number of SDUs in each RLC PDU. For one SDU none of these extra fields will be present, but for every additional SDU one Length Indicator (LI) and one E bit are added.

The Sequence Number is 8 bits long and is needed for duplicate detection and in sequence delivery to the upper layer. The Complete/Partial field is needed to support segmentation and concatenation. The CP field is 2 bits long and the first bit indicates if the start of the first SDU is segmented while the second bit indicates if the end of the last SDU is segmented.

The E bit indicates if more header fields follow or if the rest of the PDU consists of SDUs. If there are more header fields a LI will follow to indicate where the first SDU ends and where the next SDU starts.

After the LI field another E bit follows. There will be one LI field and one E bit for every SDU in each RLC PDU except for the last SDU. The length of the last SDU can be calculated by subtracting the length of the RLC PDU with the sum of all present LIs.

Padding is added to the RLC header to byte align the payload if needed. The RLC PDU structure is based on [8].

4.9.2.2 Building and Transmission of RLC PDUs

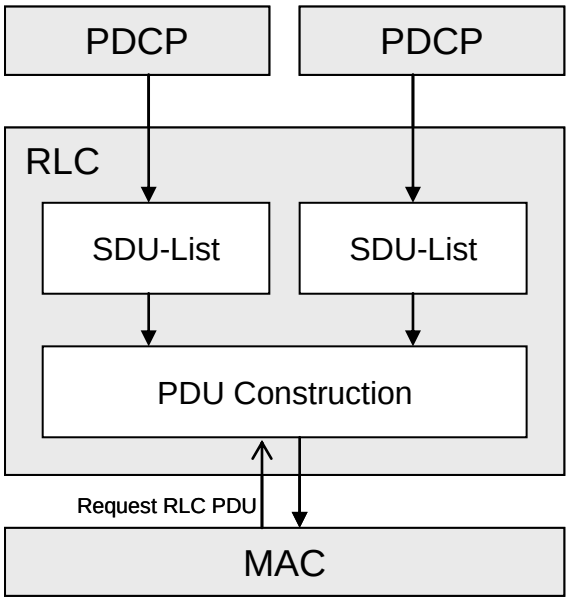

Figure 24. Overview of the RLC sub layer for downlink.

When receiving SDUs from the PDCP sub layer the RLC sub layer stores the SDUs in a SDU-List in the order they are received. All 16 channels in the RLC sub layer have its own SDU-List and work independently of each other. The SDUs are buffered in the RLC sub layer until the MAC sub layer requests data from the RLC channels. See Figure 24 for an overview of the RLC sub layer in downlink.

When the MAC sub layer requests data it will tell the RLC sub layer which channel and the maximum size of the RLC PDU that can be sent to the MAC sub layer. If the specific RLC channel has less data in the buffer than the requested size the RLC channel will put all the SDUs belonging to the specific channel in the same PDU, add a RLC header and deliver the RLC PDU to the MAC sub layer. If the specific RLC channel has enough data then a PDU of requested size is built using segmentation if needed (see Chapter 4.9.2.3).

4.9.2.3 Segmentation

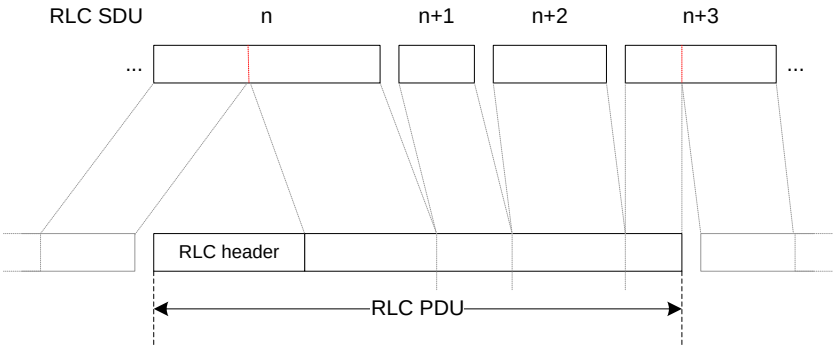

Figure 25. RLC SDUs may have to be segmented for a perfect fit in a RLC PDU. [3]

Segmentation is needed to build dynamically sized RLC PDUs without any padding. When the RLC sub layer receives the requested RLC PDU length from the MAC sub layer (See Chapter 4.9.2.2), the RLC sub layer may have to segment the last SDU in the PDU to be able to build a PDU with the requested length.

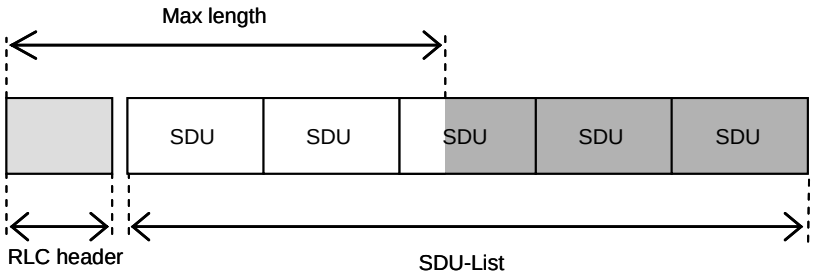

Figure 26. SDUs from the SDU-list are added to a RLC PDU.

Each SDU from the SDU-List is added to the RLC PDU until the requested length has been reached. Note that the requested length from the MAC sub layer also includes the length of the RLC header, and each SDU added to the PDU increase the RLC header with 12 bits (LI = 11 bits, E = 1 bit), which also is considered when adding the length of each SDU.

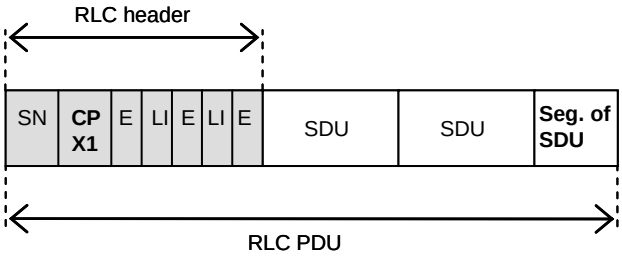

Figure 27. A complete RLC PDU with a SDU segment.

Unless the last SDU fits perfectly the last SDU will be segmented to fill the requested size. The first part of the segmented SDU is then added to the RLC PDU to be built and the second bit in the CP field in the RLC header is set to indicate that the last SDU in the PDU is a segment of an SDU (see Figure 27).

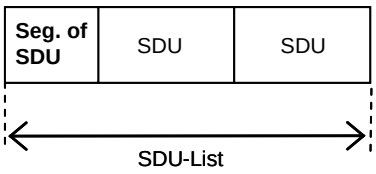

Figure 28. SDU-list with a segment.

The last segment of the segmented SDU will be left in the first position of the SDU-list (see Figure 28). That means the first SDU in the SDU-list actually is a segment of an SDU; this is indicated by setting the first bit in the CP field of the following PDU. When the next RLC PDU is constructed the first SDU will be the missing segment from the previous PDU and the CP field will already be set to indicate this (see Figure 29).

Figure 29. Next RLC header that has the last segment from an SDU.

4.9.2.4 Receiving of RLC PDUs

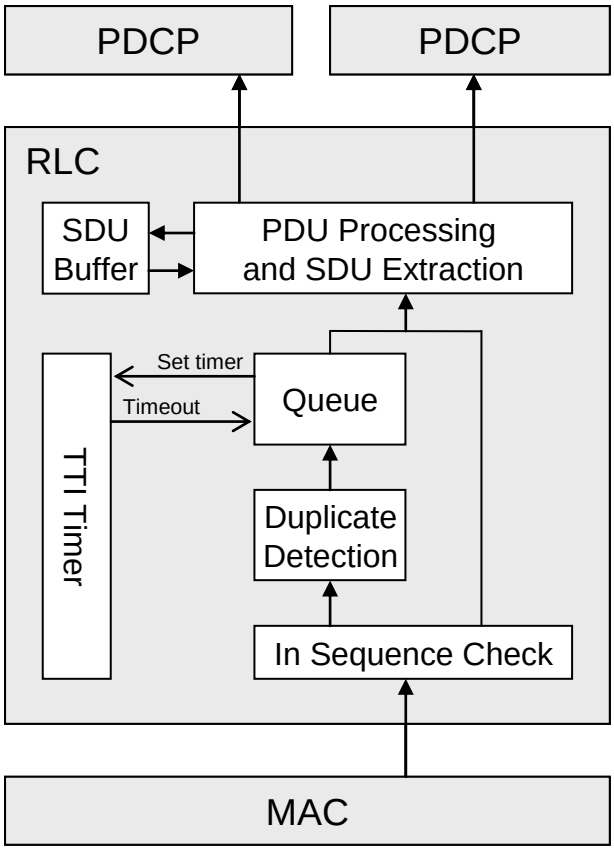

Figure 30. Overview of the RLC sub layer for uplink.

When the RLC sub layer receives a RLC PDU from the MAC sub layer, an in sequence check is performed to guarantee in sequence delivery of SDUs to the PDCP sub layer. The in sequence check is also needed to be able to rebuild segmented SDUs correctly. For more information regarding in sequence delivery see Chapter 4.9.2.7. If the received RLC PDU is the expected RLC PDU, processing and delivery of the RLC PDU (see Chapter 4.9.2.5) is

performed. Otherwise duplicate detection (see Chapter 4.9.2.8) is performed before the RLC PDU is put in a waiting queue. The RLC PDU will be delivered from the waiting queue when all the expected RLC PDUs before it, has been received (see Figure 30).

Each RLC PDU will only be stored in the waiting queue for a short period of time. When a RLC PDU reach a certain age a timeout will occur, and then the expected RLC PDU is considered lost and the waiting RLC PDU(s) from the queue will be delivered.

4.9.2.5 Processing and delivery of RLC PDUs

The first step when processing a RLC PDU is to pick out and count all the LIs from the RLC header and store them in a temporary list. The next step is to check if the first bit in the CP field is set, if so the first SDU in the RLC PDU is a segment of an SDU and rebuilding of the SDU is needed (see Chapter 4.9.2.6). If the first bit in the CP field is not set and we have more than one SDU (we can check this by counting the number of LIs) in the PDU, we know that the first SDU is a complete SDU and it will be delivered to the upper layer immediately.

If we have more LIs it means we have more completed SDUs which also will be delivered to the upper layer. Remember that the last SDU doesn't have an LI and only the first and the last SDU can be segmented so all SDUs in between are complete SDUs.

When we don't have any more LIs one last SDU remains in the PDU and if the second bit of the CP field is not set it means the last SDU is a complete SDU and will then be delivered to the upper layer. Otherwise if the second bit in the CP field is set the SDU needs to be rebuilt before delivery to the upper layer.

4.9.2.6 Rebuilding of SDUs

Rebuilding of SDUs is needed to restore SDUs that have been segmented (see Chapter 4.9.2.3). A segmented SDU is indicated by the CP field (see Chapter 4.9.2.1) in the RLC header. When the first part of an SDU is received the segment is stored in a SDU-buffer as illustrated in Figure 31. When the remaining segment has arrived the SDU is delivered to the upper layer (see Figure 32).

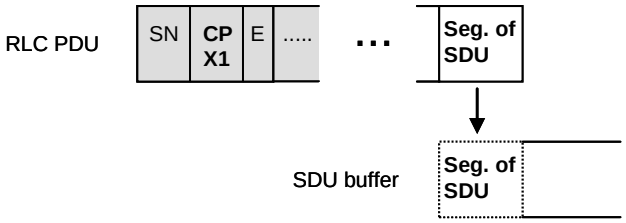

Figure 31. First segment from an SDU arrives and is stored first in the SDU-buffer.

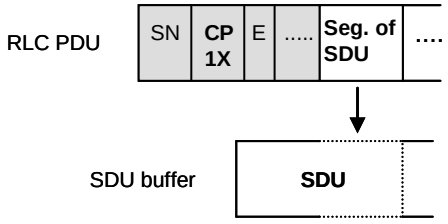

Figure 32. The last segment from an SDU arrives and the complete SDU can be rebuilt.

A timeout will occur if a RLC PDU doesn't arrive within a certain time, the RLC PDU is then considered lost. If a timeout has occurred then all segments from the SDU-buffer will be

deleted because a segment belonging to the SDU from the SDU-buffer has been lost and there is no way to rebuild that specific SDU. All upcoming segments belonging to the missing RLC PDU will also be skipped when they arrive. In Figure 33 a RLC PDU with sequence number 1 never arrives resulting in a timeout and all segments belonging to the missing PDU is deleted.

Figure 33. RLC PDU timeout example.

4.9.2.7 In Sequence Delivery for UM

Every RLC PDU has a sequence number indicating the order within a stream. On the receiving RLC entity an expected sequence number will be compared to the sequence number of each RLC PDU. If the sequence numbers are equal the RLC PDU will be processed (see Chapter 4.9.2.4) and the expected sequence number will be increased with one. If a sequence number doesn't match the expected sequence number the RLC PDU will be put in a waiting queue with a timestamp attached to it. The PDUs in the waiting queue will be delivered when all the missing sequence numbers up to the first in the queue has been received or the first PDU in the queue has timed out.

When a timeout occurs the expected sequence number will be set to the same as the sequence number from the first PDU in the queue and all PDUs from the queue matching the expected sequence number will be processed according to the procedure above.

4.9.2.8 Duplicate Detection

There are two types of duplicate detection in the prototype; the first one checks the gap between the incoming sequence number and the expected one. This check is hard coded to throw away all RLC PDUs where the gap is bigger than 191.

For example, if we receive a RLC PDU with sequence number 48 and the expected sequence number is 50 the gap is 254 (51, 52, … 254, 255, 0, 1, … 47, 48) and 254 is higher than 191 so we will throw away the PDU. This check detects if we recently have received and processed a PDU with the same sequence number as the incoming PDU.

The second duplicate detection checks when adding an RLC PDU to the queue if there already is a RLC PDU in the queue with the same sequence number, then the PDU in the queue will be replaced with the new one. This check detects if we have received but not processed a PDU with the same sequence number.

4.9.3 Medium Access Control

The main functional blocks in the MAC sub layer are scheduling, priority handling, multiplexing and limited HARQ. The prototype consists of two MAC objects, where one objects is located in the UE and one in the eNB. The objects are functionally similar except scheduling which is controlled from the eNB side.

The transport channels supported by the prototype are the Downlink Shared Channel (DL-SCH) and Uplink Shared Channel (UL-SCH) and the supported logical channels are DTCHs.

There are 16 downlink and uplink DTCHs which are mapped onto the DL-SCH and UL-SCH respectively. The MAC services and functions are based on [3].

Services provided by the MAC sub layer to upper layers:

- Data transfer.

Functions supported by the MAC sub layer:

- Mapping between logical channels and transport channels.
- Multiplexing of MAC SDUs.
- Strict priority handling between logical channels of one UE.
- Strict priority handling between UEs by means of Round Robin scheduling.

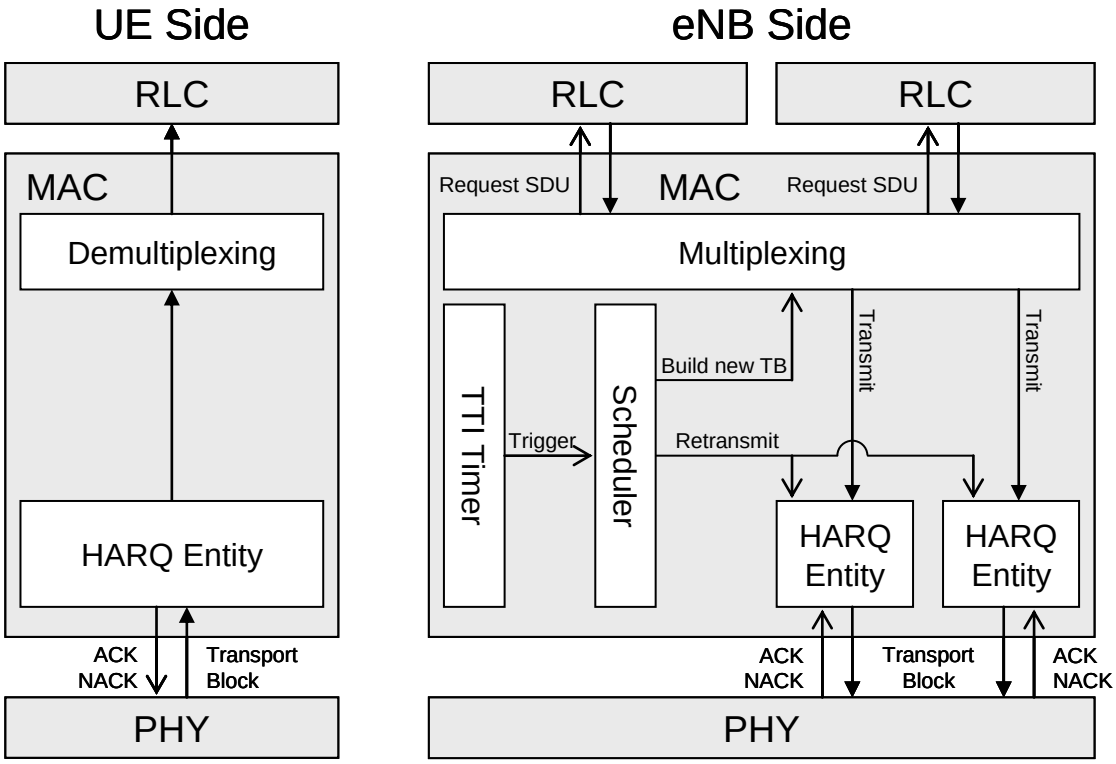

Figure 34. MAC downlink overview.

The MAC sub layer is triggered each TTI by means of a TTI Timer. The TTI Timer triggers the scheduler who does the scheduling decisions before instructing the MAC to either build new transport blocks or do retransmissions. If a new transport block is scheduled to be made the scheduled RLC object is instructed to build RLC PDUs based on the priority of the logical channels. The RLC PDUs are multiplexed into transport blocks and sent through the HARQ entity. Figure 34 shows how the internal parts of the MAC communicates and transports the data through the MAC sub layer in downlink while Figure 35 shows the uplink structure. The main difference is how the scheduler works as the eNB side is responsible of scheduling the UEs. The UE side scheduler is only used to keep track of when the UE is scheduled and take care of priority handling between the logical channels.

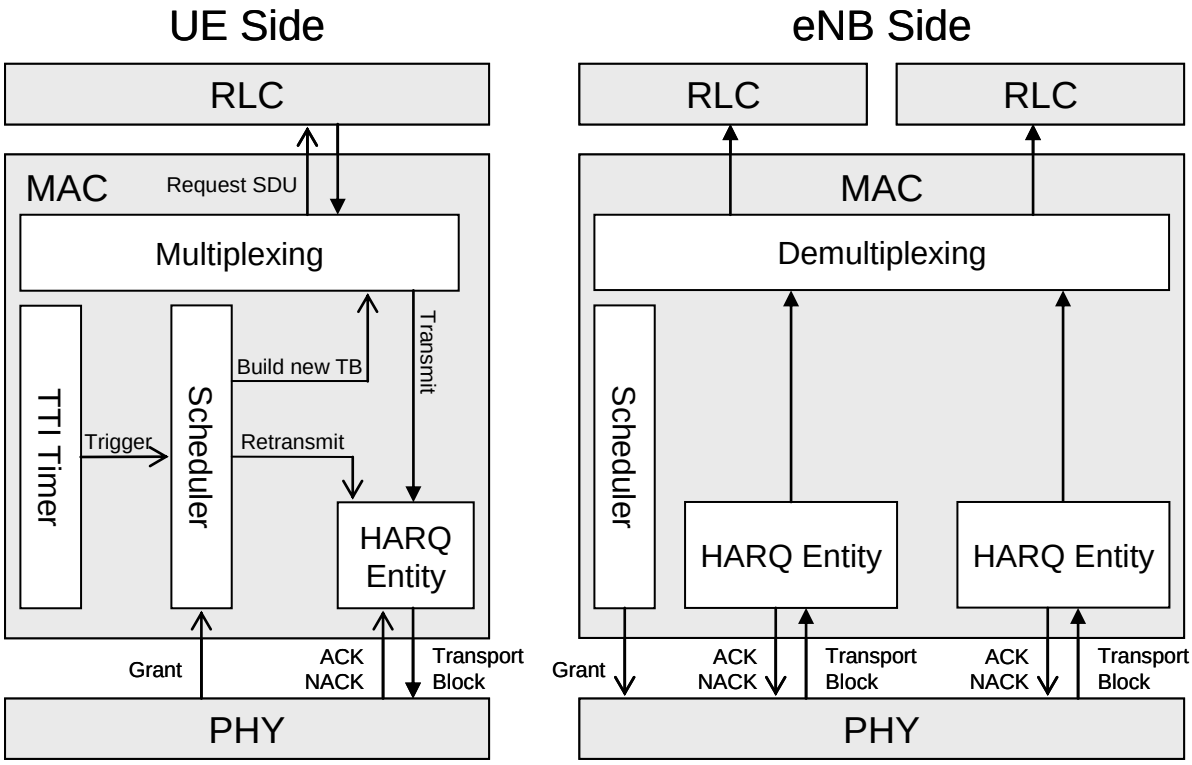

Figure 35. MAC uplink overview.

4.9.3.1 PDU Structure

The MAC PDU in the prototype consists of a header and payload. The payload includes at least one MAC SDU, i.e. RLC PDU, and may consist of several multiplexed SDUs and eventual payload padding.

In order to know which logical channel each SDU come from, and perform demultiplexing of the payload, there is a need to indicate corresponding logical channel. See Figure 4 and Figure 5 for uplink and downlink logical channels receptively where the prototype uses DTCHs. Besides SDUs there might be padding at the end of the MAC payload which also must be indicated. The Data Description Indicator (DDI) indicates existence of these payload elements.

MAC SDUs are of dynamic size and are built according to the size of the MAC PDU. The Length Indicators (LI) are present in the MAC header to indicate the length of corresponding SDU. MAC header extension bits are used for indication of whether header elements or payload is present in the next byte.

The MAC PDU structure for the LTE Prototype is primarily based on [9]. The prototype uses a byte aligned MAC header and the MAC PDU structure as illustrated in Figure 36.

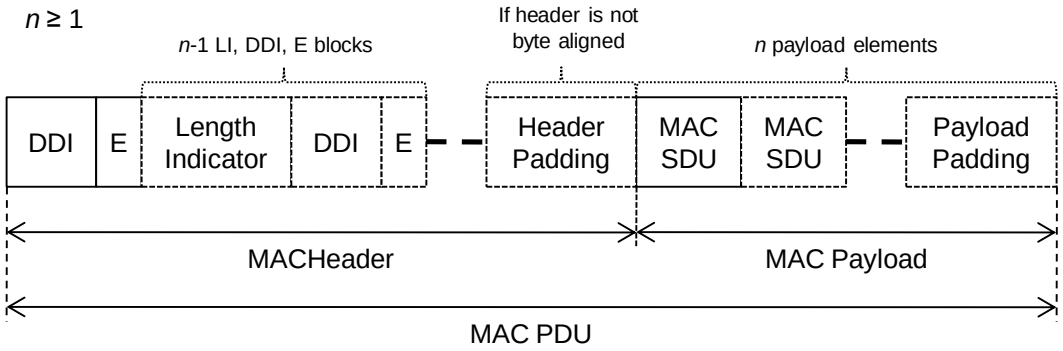

Figure 36. MAC PDU Structure.

The MAC header consists of one header block for each corresponding payload element where each block is defined as a combination of three header information elements. The first element is a Data Description Indicator and the second is an Extension bit. If the Extension bit is set to 1, the third field consist of a Length Indicator else it will hold eventual header padding bits.

Header Elements	Description
Data Description Indicator (DDI)	There is one DDI for each payload information element, which may indicate either an SDU belonging to a specific logical channel or padding. The Prototype supports 16 dedicated traffic channels (DTCH) used for user plane data.
Extension Bit (E)	The Extension bit indicates whether the next byte includes header elements or payload data. 0 indicates that MAC payload begins at next byte while 1 indicates that a LI and more header elements follow.
Length Indicator (LI)	There is one LI for each SDU if there is padding included. If padding is not included the length of the last included SDU is derived from the PDU size. The PDU may handle SDUs up to 2047 bytes.
Header Padding (PAD)	If the present header information elements all together are not byte aligned, 1 to 7 padding bits is used at the end of the header.

Table 1. MAC header elements.

Payload Elements	Description
MAC SDU	RLC PDU.
Payload Padding	Payload padding bits are inserted at the end of the MAC PDU if the SDUs do not fill the transport block size completely.

Table 2. MAC payload elements.

4.9.3.2 Scheduling and Priority Handling

All scheduling decisions are made on the eNB side and signalled to the UE using grant messages in the PHY sub layer in uplink and the PHY header of the data message in downlink (see Chapter 4.8.1). The scheduler uses a simple round robin algorithm to schedule the UEs. Each UE gets assigned one transport block during one TTI at a time in both uplink and

downlink. Since the primary role for the scheduling in the prototype is to use the shared channels correctly, all the UEs get the same amount of resources assigned whether they are using them or not. On the eNB side the scheduling is triggered each TTI. The scheduler schedules downlink for the current sub frame and uplink is scheduled one sub frame ahead. The uplink scheduling is signalled to the UEs through the PHY sub layer.

Once all scheduling decisions are made preparing of transport blocks addressed to the UEs scheduled in downlink, is initiated. On the UE side the scheduler checks if the UE is scheduled this sub frame. If it is scheduled then preparing of one transport block is started for each grant for the current sub frame. Retransmissions are always scheduled before any new transmissions.

The priority between logical channels is strict. The MAC uses the priority list provided by the scheduler each time a transport block is scheduled to be built. Since the focus of the prototype was not on scheduling the scheduler always provides the same static list. The MAC picks the channels in priority order requesting data from the RLC for each one until that channels queue in the RLC is empty. Once the transport block is full or all channels have been added to the transport block it's sent down to the HARQ for transmission.

4.9.3.3 Multiplexing

Multiplexing of SDUs into a transport block is performed when a user is scheduled for a new transmission (see Figure 37).

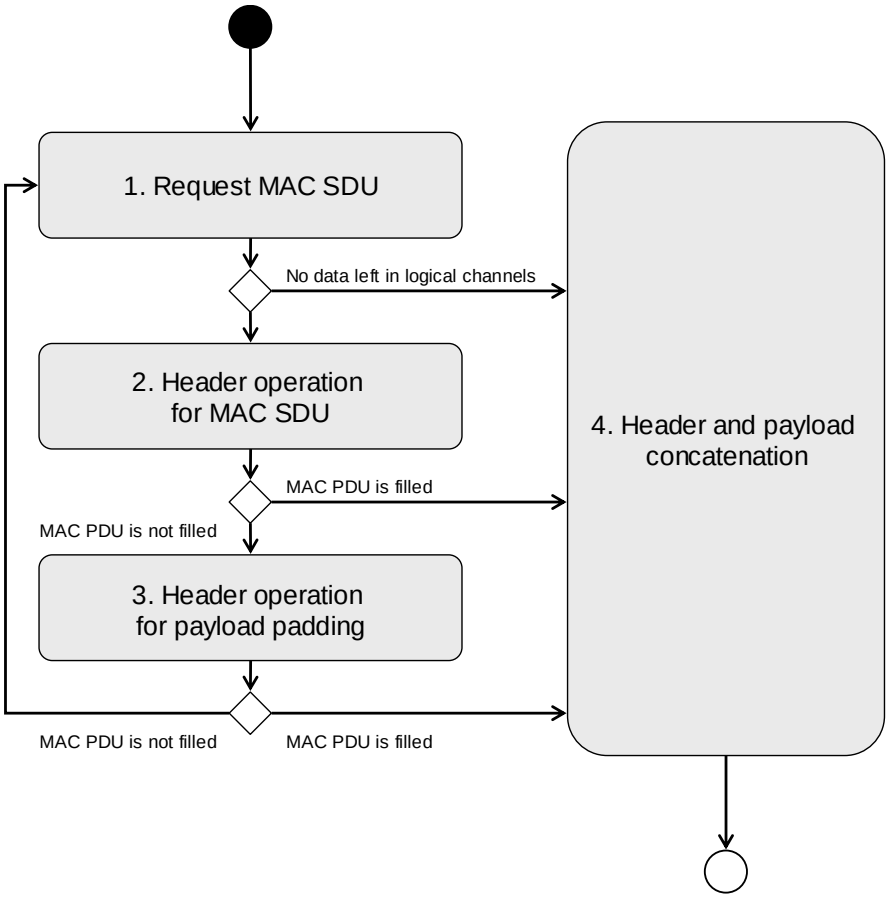

Figure 37. Multiplexing procedure.

Phase 1: Request MAC SDU.

During this first phase of the multiplexing procedure, an MAC SDU is requested from the RLC sub layer with requested size. The RLC sub layer aims at building this SDU according to the size requested, but may return a SDU with smaller size or indicate that there is no data in any of the logical channels. If there is no more data, phase 4 is invoked. The requested size is the empty space left in the transport block up to the maximum size of an SDU.

Phase 2: Header operation for MAC SDU.

When a requested SDU is returned from the prioritized logical channel, this logical channel is represented in the MAC header during this phase. In the first iteration for this phase, a MAC header multiplexing block is built in form of an initial block, which is illustrated in Figure 38.

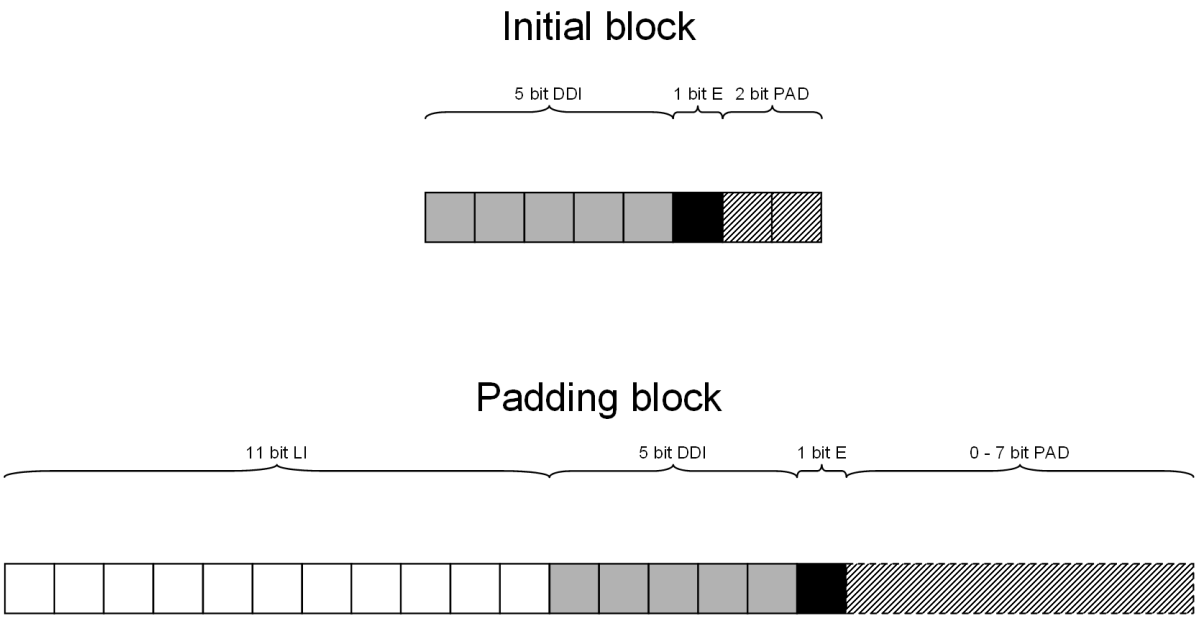

Figure 38. MAC header multiplexing blocks.

The initial block consists of a DDI, an unset E bit and header padding bits for byte alignment. In this phase, no more MAC header multiplexing block is built following the first iteration, where the initial block was built. For representing the DDI, without adding any new building blocks, the current MAC header is updated by overwriting the latest added DDI, to indicate the logical channel associated to the latest returned SDU. Before getting overwritten, the DDI indicates payload padding and belongs to the latest created MAC header multiplexing block, which was built as a padding block during phase 3. The location of this DDI varies relative to the lowest order MAC header bit, and is derived based on current header padding. The dynamic location of the DDI is related to where the padding block is inserted, which is illustrated in Figure 39.

It should be noted that the latest returned SDU isn't represented by length, i.e. LI, in this phase, which is instead added in phase 3. However, if the PDU is completely filled after this phase no LI is necessary for this SDU. It should also be noted that the latest returned SDU is concatenated together with previous returned SDUs, if any are present for this PDU, but these concatenated SDUs are however separated from the current MAC header in this phase.

Phase 3: Header operation for payload padding.

This phase follows phase 2, in case the PDU wasn't completely filled. Because the current MAC header does not have any indication of the empty space, it is managed in this phase. Furthermore, the latest returned SDU is not represented correctly in terms of size, and therefore a LI must be added to the header as well. Indication of the empty space and the LI for the latest returned SDU is both represented by building a MAC header multiplexing block in form of a padding block, which is illustrated in Figure 38.

Before the actual building, the location of the current E bit is identified based on header padding and must be set, as more header elements will follow this bit. The MAC header is then either shifted to the left by 2 bytes in case of current header padding or by 3 bytes in case of no header padding, i.e. byte aligned. The padding block is inserted in this newly created space, and there are no unused bits between the set E bit and the highest order bit for this padding block, implying that the N header padding bits in the shifted block was overlapped by the N highest ordered LI bits from the padding block.

The location of where the padding block shall be inserted is varying relative to the lowest order MAC header bit, and is derived based on current header padding. The dynamic location where the padding block may be inserted is illustrated in Figure 39, where a padding block occupies the 3 lowest order bytes in the MAC header. Header padding bits is related to the number of MAC header multiplexing blocks inserted. The initial block implies 2 header bits. All of the following block insertions results in one less header padding bit, except when there was no previous header padding. Then the header padding is set to a maximum of 7 bits.

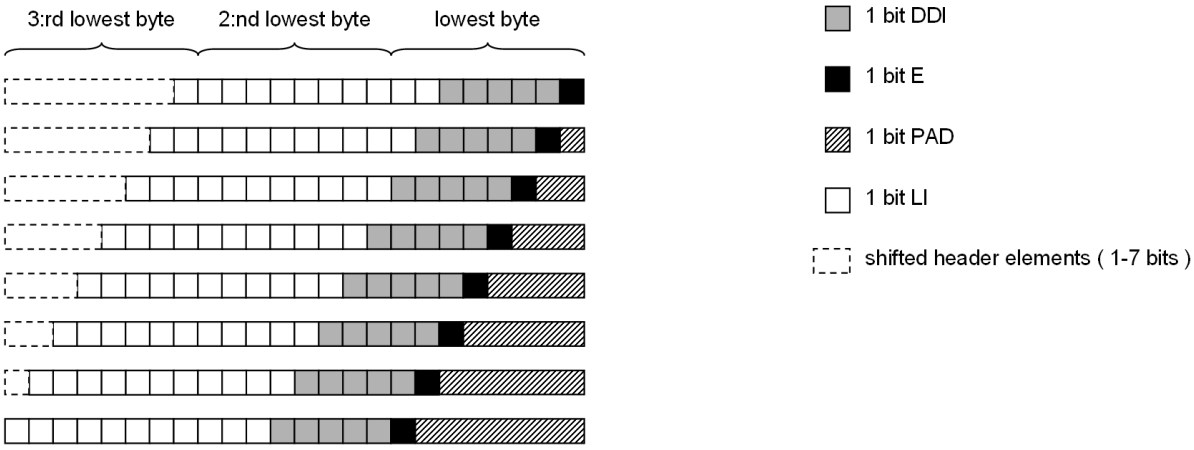

Figure 39. Location of padding block.

It should be noted that the MAC header and the payload are not concatenated in this phase. If the PDU is now completely filled it is ready for phase 4, else there will be a new iteration starting with another SDU request in phase 1.

Phase 4: Header and payload concatenation.

In this phase, the current MAC header will first be concatenated with previous concatenated SDUs for this PDU. If there is a need for padding, this will be added so the PDU is completely filled. When the concatenation is done, the transport block is delivered to the HARQ Entity.

4.9.3.4 Demultiplexing

Demultiplexing of SDUs from a transport block is performed when a transport block is delivered from the physical layer (see Figure 40).

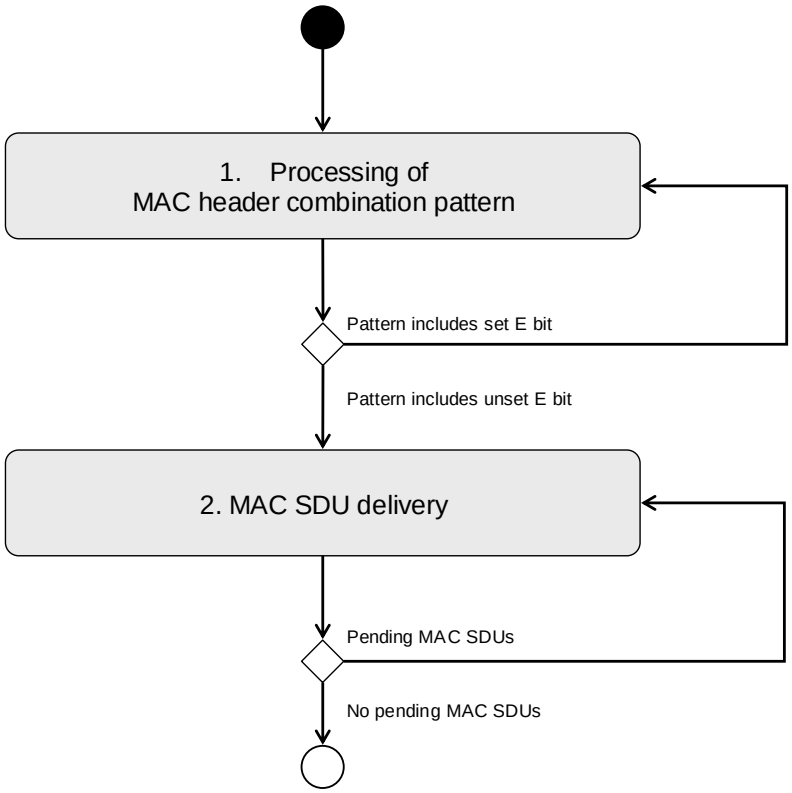

Figure 40. Demultiplexing procedure.

Phase 1: Processing of MAC header combination pattern.

Before each SDU included in the MAC payload may be delivered to the corresponding logical channel, which is performed in phase 2, the MAC header must first be processed that includes the LIs and DDIs, needed for the payload element identification. The iterative MAC header process is performed by processing the MAC header one byte at a time, from the header start to the payload start.

The MAC header pattern is repetitive and the approach for the bit identification, for the byte being processed, is by using a virtual repetitive MAC header pattern. This repetitive pattern consists of 17 positions, where position 0 to 4 points at DDI bits, 5 points at E bit, 7 to 16 points at LI / PAD where the first 7 positions may either relate to LI or header padding and the 4 remaining to LI only. A virtual bit pointer is moving in this pattern, synchronized with a physical data pointer which points to the actual data.

This phase is repeated until an unset E bit is found in the current processed MAC header combination pattern. Let the previous virtual bit pointer position be X, a MAC header combination pattern is defined as consisting of the values between pointing position X and (X+8) mod 17, where X is the position pointing at the highest bit value. As seen in Figure 41, each combination pattern consists of segmented parts of up to four separate header information elements. A DDI and LI occurs in up to 2 and 3 patterns respectively, where the resegmentation is performed by using the OR operator.

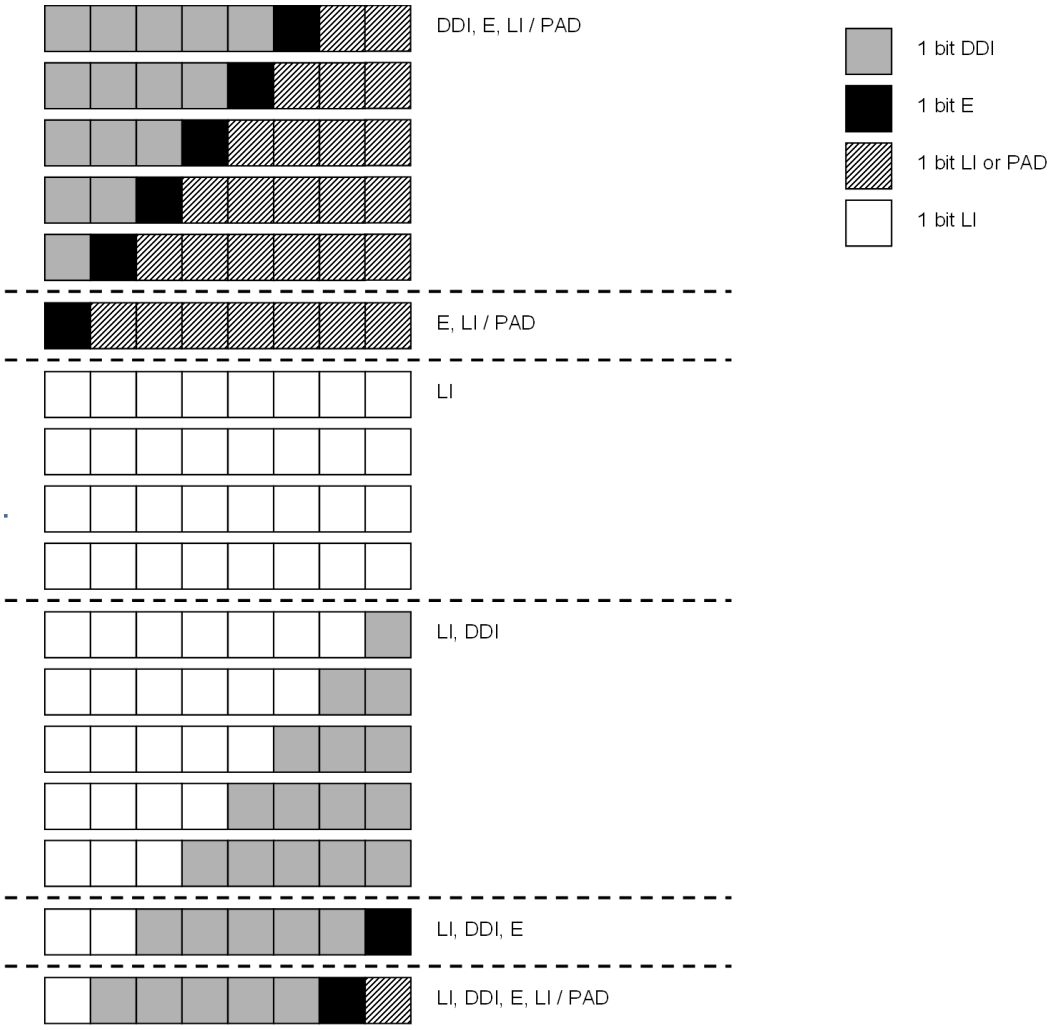

Figure 41. MAC header combination patterns.

Phase 2: MAC SDU delivery.

Based on the temporarily stored information elements (i.e. DDIs and LIs) for the SDUs, each SDU may be delivered to the corresponding logical channel. As described in phase 1, there is a physical data pointer indicating the operating point in the transport block. Each SDU is delivered to the logical channel indicated by the DDI, which was temporarily stored in phase 1. The appropriate cutting position of the payload is based on the physical data pointer, which moves a distance in the physical memory based on stored LIs or transport block size. The distance moved is equal to the size of a payload element. Notice that all cutting positions are based on LIs if payload padding is included in the PDU.

4.9.3.5 Hybrid Automatic Repeat Request

The HARQ located in the MAC sub layer ensures delivery between peer entities at the physical layer. There is a HARQ entity at the UE node and the eNB for each UE entity. A HARQ entity in the system has a configurable number of processes for parallel transmission, where each process is associated with a transmission buffer.

The limited HARQ is responsible for retransmissions in downlink and uplink respectively. Retransmissions will occur when the eNB or the UE do not receive transmitted data correctly. The prototype may simulate non successful decoding, meaning that negative feedback will be signalled (i.e. NACK) to the transmitter for the corresponding transport block, which

eventually will generate a retransmission. A transport block may be transmitted a configurable number of times before the transmitting HARQ process discards the packet. The prototype prioritizes transmissions of negative acknowledged data above generation and further delivery of new data.

The implemented HARQ do not use data combining at receiver and therefore transport blocks are not stored at the receiver if it is unsuccessfully decoded. At the HARQ receiving side a transport block is either delivered to demultiplexing or rejected, followed by generation and signalling of an ACK or NACK respectively.

When comparing the receiving and transmitting HARQ side, the receiving HARQ is less complex because there is no data combining functionality. During each TTI when the UE or the eNB MAC layer is triggered, which is based on valid uplink grants and downlink scheduling respectively, the HARQ entity searches the corresponding processes for pending data. In case a NACK has been received by a HARQ process the negative acknowledged data will be delivered to the PHY layer, else a new transport block may be generated and put in an empty HARQ process before delivery to the PHY layer.

The prototype HARQ signals the process identifier and the UE identifier along with each transport block, encapsulated in data messages. The HARQ uses a one byte acknowledgement, including process identifier (ACK) or an inversed process identifier (NACK).

A transport block belonging to a HARQ process at the transmitting side is stored in corresponding buffer until the active process has received an ACK, an ACK or NACK was lost or if the data has been transmitted the maximum number of times.

4.10 Layers for S1-User Plane Interface

4.10.1 GPRS Tunnelling Protocol for User Plane

GTP-U is used to transfer data between a pair of GTP-U tunnel endpoints, indicated by a unique Tunnel Endpoint Identifier (TEID) for each tunnel. In the prototype there is a tunnel per stream and one GTP-U entity per IP-Address, i.e. one GTP-U entity located in the eNB and aGW each. The system creates tunnels with configured unique TEIDs at start up, not by random dynamically TEIDs that are exchanged between tunnel endpoints as specified in [10].

Services provided by the GTP-U sub layer:

- Transmission services between a pair of GTP-U tunnel endpoints.
- Data transfer to lower and upper layers.

4.10.1.1 PDU Structure

The PDU Structure in the prototype is based on structure for GTP-U protocol version 1 without the optional fields (Figure 42). Message Type (MT) is always set to 255 which indicate a T-PDU in the GTP-U payload. The only dynamic fields in the GTP-U header are TEID and Length Indicator (L). The TEID is used for multiplexing and demultiplexing of different UEs and corresponding streams. The position for the T-PDU is derived from the Length Indicator. However, the static sized header in this limited PDU structure implies that position may always be derived from G-PDU size only.

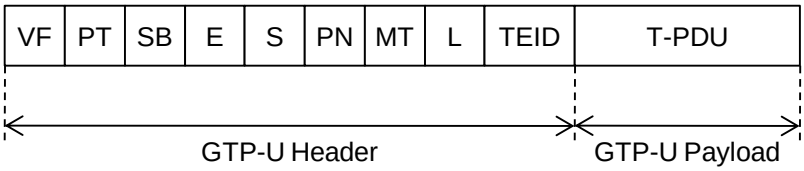

Figure 42. GTP-U Structure

Header Elements	Description
Version Field (VF)	This field is used to determine the version of the GTP protocol. Static
Protocol Type (PT)	This bit is used as a protocol discriminator between GTP (when PT is '1') and GTP' (when PT is '0'). Static
Extension Header flag (E)	This flag indicates the presence of a meaningful value of the Next Extension Header field. Static
Sequence number flag (S)	This flag indicates the presence of a meaningful value of the Sequence Number field. Static
N-PDU Number flag (PN)	This flag indicates the presence of a meaningful value of the N-PDU Number field. Static
Message Type (MT)	This field indicates the type of GTP message. Static
Length (L)	This field indicates the length in octets of the payload, i.e. the rest of the packet following the mandatory part of the GTP header (that is the first 8 octets). Dynamic
Tunnel Endpoint Identifier (TEID)	This field unambiguously identifies a tunnel endpoint in the receiving GTP-U protocol entity. Dynamic
Spare Bit (SB)	This bit is a spare bit. It shall be sent as '0'. Static

Table 3. GTP-U header elements.

4.11 Test Results

This chapter covers tests of the main functionalities in the prototype. The aim of these tests is to see how the different parts behave when data is transferred through the system. At the end we also present performance and stability tests.

4.11.1 Multiplexing, Concatenation and Segmentation

When testing multiplexing in the MAC sub layer and concatenation/segmentation in the RLC sub layer we used one UE with four active streams. Multiplexing and concatenation/segmentation are only used between the UE and the eNB so no aGW node was present in this test. Both the UE and the eNB where located on the same machine, an Athlon 64 3000+ (1.8 GHz) with 1 GB ram, running Linux. VLC was used for transferring video streams. Two different tests were performed, one with high bit rate (1374 kbs) streams and one with low bit rate (338 kbs) streams. TTI was set to 16 ms.

Parameters	Test 1	Test 2
No. of UEs	1	1
No. of streams per UE	4	4
Bit rate (kbs)	1374	338
TTI (ms)	16	16
RLC SDU size (bits)	1318	1318

Table 4. Parameters for testing multiplexing, concatenation and segmentation.

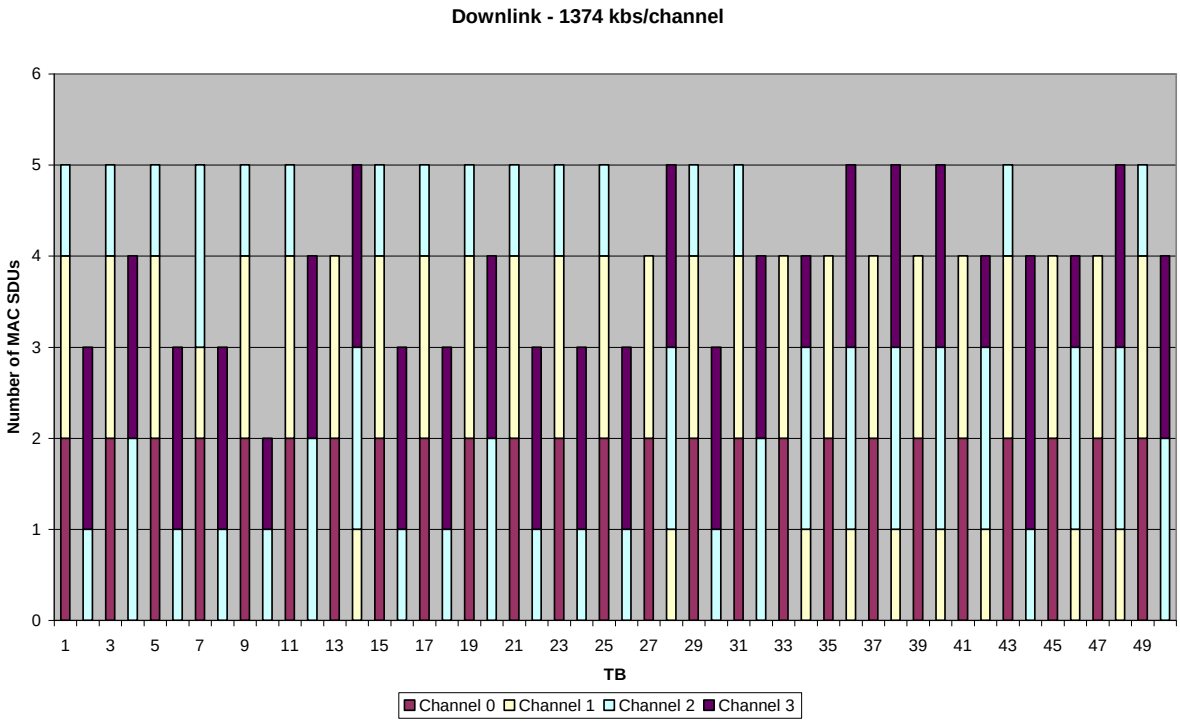

Figure 43. Multiplexing of 4 active RLC channels.

Studying the result for multiplexing from test 1 in Figure 43 we can clearly see the order each channel is multiplexed in each transport block. Channel 0 is followed by channel 1, channel 1 is followed by channel 2 and channel 2 is followed by channel 3. This is due to the strict priority handling described in 4.9.3.2. We can also see that more than one RLC PDU from the same channel is often multiplexed directly after each other when transferring high bit rate streams. That's because each RLC channel receives data in a high rate and all the data don't fit in a single MAC SDU.

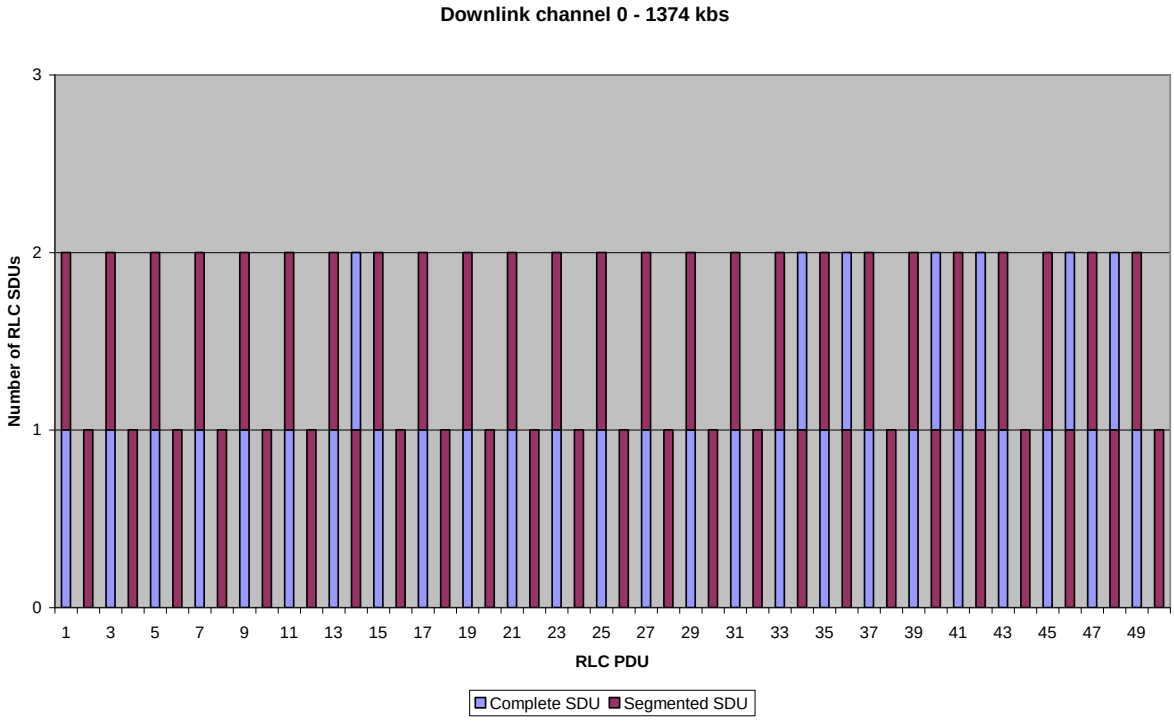

Figure 44. Concatenation and segmentation for channel 0 from test 1.

Figure 44 shows how the RLC SDUs is concatenated and segmented in RLC PDUs for channel 0. We can see that there seems to be max 2 RLC SDUs concatenated in each RLC PDU. That's because each SDU is 1318 bits large and the maximum size of a MAC SDU is 2047 bits. This will also result in lots of segmentation when transferring high bit rate streams, as we can see in Figure 44.

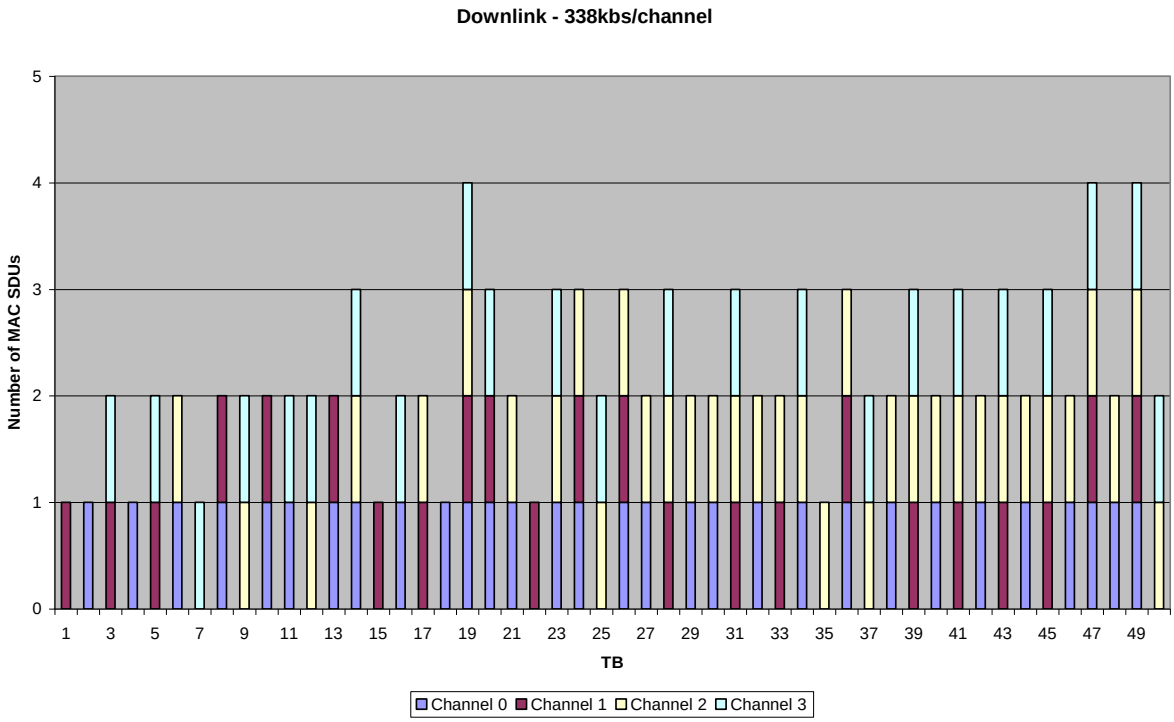

Figure 45. Multiplexing of 4 active RLC channels.

In Figure 45, we can see the result from the second test of multiplexing with lower bit rate streams. The difference here compared to the high bit rate test is that there are less multiplexing. The order of the channels is not as clearly visible in this test. The reason for this is that each RLC channel doesn't have as much data to transmit compared to test 1. There might not always be any data to transmit from a specific RLC channel. The MAC sub layer will then ask the next channel if it has anything to transmit. This will result in less multiplexing and that the order of the streams is harder to distinguish.

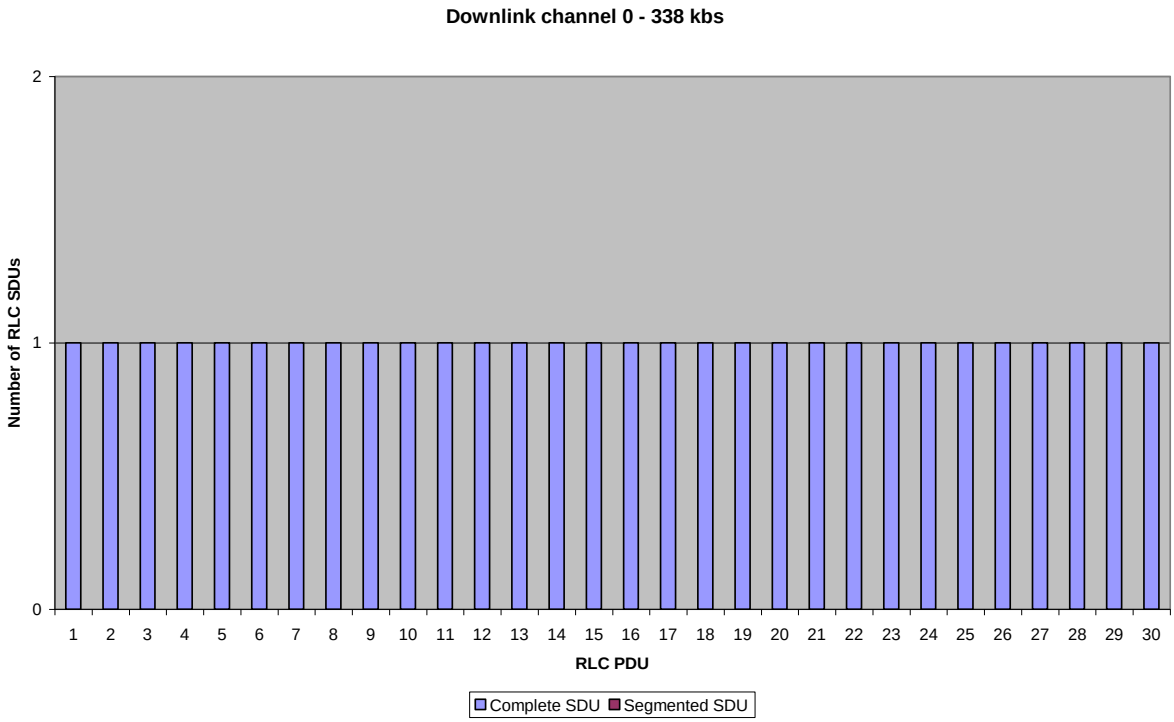

Figure 46. Concatenation and segmentation for channel 0 from test 2.

Figure 46 shows how many RLC SDUs that the 30 first RLC PDUs contain from channel 0 for test 2. For a low bit rate stream there will be less concatenation and segmentation compared to a high bit rate stream. We can see this in Figure 46, where no concatenation or segmentation is present. Each RLC PDU contains only one RLC SDU, as the MAC sub layer requests data at a much higher rate than the RLC sub layer receives data.

4.11.2 In Sequence Delivery for Unacknowledged Mode

In this chapter, test results for the in sequence delivery functionality for UM are presented. The decode failure probability is set to 25 % to get frequent retransmissions. NACKs at the transmitter are delayed long enough for new PDUs to be scheduled before the retransmissions. This results in frequent out of sequence delivery of RLC PDUs. The essential prototype parameters are shown in Table 5. The test is performed in uplink only as downlink is identical for in sequence delivery functionality and therefore excluded.

No. of UEs	1
No. of streams per UE	1
TTI	16 ms
Prob. Decode failure	25 %

Table 5. Parameters for test of in sequence delivery.

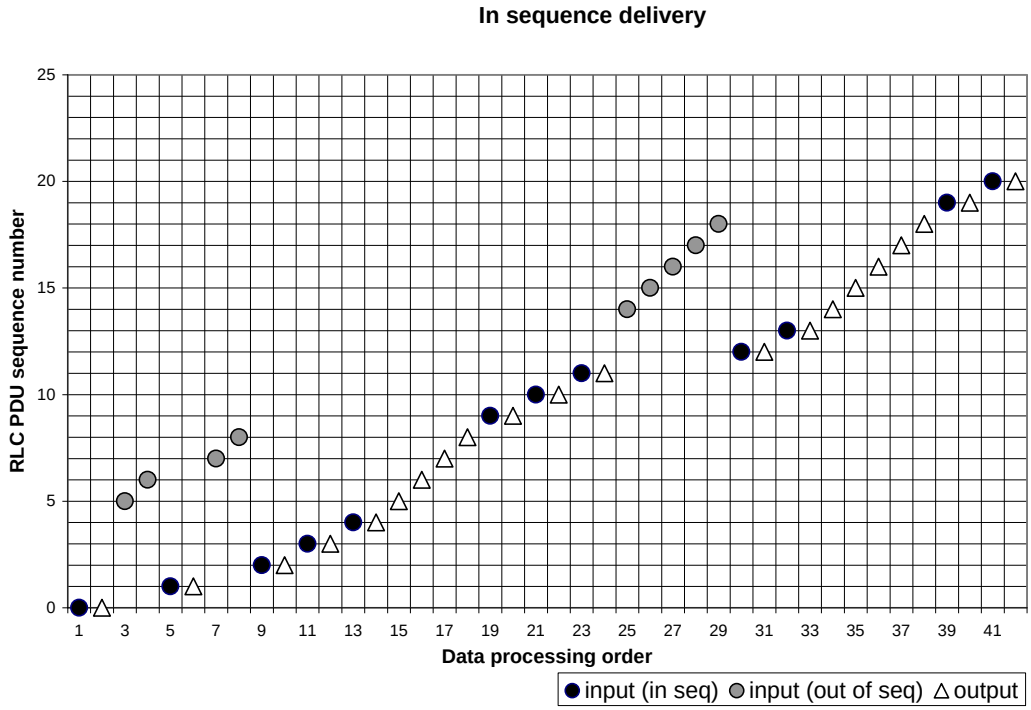

Figure 47. Test results for in sequence delivery.

In Figure 47 a sample for the first 21 incoming RLC PDUs where 9 is out of sequence is shown. If an incoming sequence number for an RLC PDU is out of sequence (grey dot), the RLC PDU will be directed to the waiting queue. If an incoming RLC PDU is in sequence (black dot), it is directed to "PDU processing and SDU extraction" along with in sequence RLC PDUs from the waiting queue if any (white triangle(s)). See Chapter 4.9.2.4 for a detailed description of the procedures. Note that a transport block may include multiple RLC PDUs. For example, the RLC PDUs with sequence number 2, 3 and 4 corresponds to one transport block which is retransmitted in this test.

4.11.3 Duplicate Detection

In this chapter, test results for the duplicate gap detection are presented. The decode failure probability is set to 0 % in the test and there are simulated ACK to NACK errors which will result in retransmissions of redundancy data. The essential prototype parameters are shown in Table 6. The test is performed in uplink only as downlink is identical for duplicate gap detection and therefore excluded.

No. of UEs	1
No. of streams per UE	1
TTI	16 ms

Table 6. Parameters for test of duplicate gap detection.

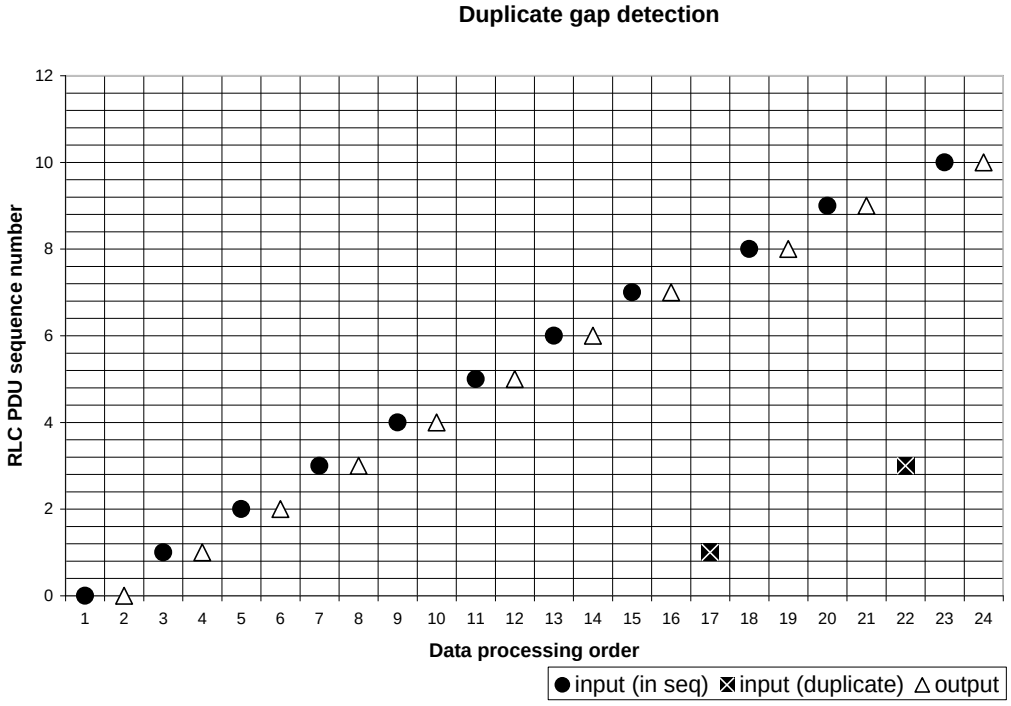

Figure 48. Test results for duplicate gap detection

In Figure 48, a sample of the first 13 incoming RLC PDUs where 2 are duplicates is shown. If an incoming RLC PDU is in sequence (black dot), it is directed to "PDU processing and SDU extraction" (white triangle). If an incoming RLC PDU is out of sequence and a duplicate (black rectangle with white cross), this duplicate will be detected in duplicate gap detection and will therefore be discarded. See Chapter 4.9.2.4 for a detailed description of the procedures.

As described in Chapter 4.9.2.8 there is a second form of duplicate detection to avoid duplicates in the RLC waiting queue. PDUs are stored in the waiting queue when they are received out of sequence. To achieve this, the decode failure probability is set to 15 % to get retransmissions. NACKs at the transmitter are delayed long enough for new PDUs to be scheduled before the retransmissions.

When the receiver has PDU(s) in the queue an ACK to NACK error is simulated, resulting in retransmission of already successfully transmitted data. This redundant data will be detected as a duplicate when trying to insert it in the waiting queue. The essential prototype parameters are shown in Table 7.

No. of UEs	1
No. of streams per UE	1
TTI	16 ms
Prob. Decode failure	15 %

Table 7. Parameters for test of duplicate in-queue detection.

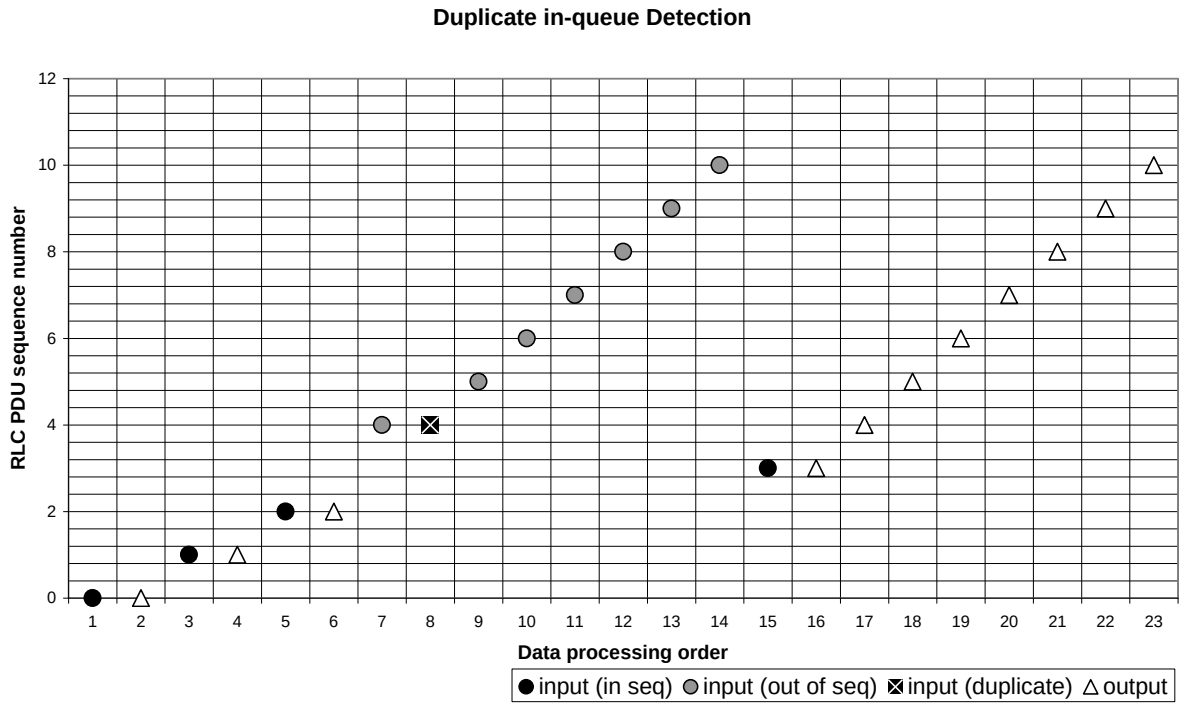

Figure 49. Test results for duplicate in-queue detection.

In Figure 49, a sample for the first 12 incoming RLC PDUs where one (Sequence number 4) is a duplicate is shown. If a sequence number for an RLC PDU is out of sequence (grey dot) the RLC PDU will be directed to the waiting queue. If an incoming RLC PDU is in sequence (black dot), it is directed to "PDU processing and SDU extraction" (white triangle). If an incoming RLC PDU is out of sequence and a duplicate (black rectangle with white cross), it will be detected when trying to insert it in the waiting queue and will therefore be discarded. See Chapter 4.9.2.4 for a detailed description of the procedures.

4.11.4 Performance and stability

To see how the system behaves and handles full capacity we tested the system by transferring video streams at varying bit rates. We then monitored different measurements to see that the system was stable and how it utilized the different resources. Video streams where sent and received using VideoLAN - VLC media player.

The following data are samples from the tests where we have plotted some of the monitored measurements in graphs. The test setup consisted of three PCs all running Ubuntu 7.04 Desktop Linux distribution with the low latency Linux kernel. Two of the PCs were dedicated to the eNB and the UE nodes respectively while the third PC was running the aGW node and VLC Media Player. Table 8 shows the PC hardware specifications.

	CPU	**RAM**
aGW, VLC	Athlon 64 x2 (2x2.0GHz)	2000 MB
eNB	Pentium 4 2.8 GHz	256 MB
UE	Pentium 4 2.8 GHz	256 MB

Table 8. Test PC hardware specifications.

The nodes were configured to run two UEs with two uplink streams each. The transport block size was set to 6250 bytes with a TTI of 2 ms and transmission of one transport block

per TTI. This leads to a max bit rate of 25 Mbit/s. This is about ¼ of a real LTE system where the TTIs are 1 ms and two transport blocks of the used size can be transmitted each TTI. Due to real time limitations in the used operative system, higher transmission rates are not possible with this test setup. For each stream a unique play list was constructed. Four different mpg movie clips with different bit rates were used in these play lists. Table 9 shows the play list for each stream.

Streams	First Clip	Second Clip	Third Clip
UE 1 Stream 1	~1500 kbit/s (5 min)	~7000 kbit/s (10 min)	~5000 kbit/s (10 min)
UE 1 Stream 2	~5000 kbit/s (10 min)	~5000 kbit/s (10 min)	~1500 kbit/s (5 min)
UE 2 Stream 1	~2500 kbit/s (7 min)	~7000 kbit/s (10 min)	~2500 kbit/s (7 min)
UE 2 Stream 2	~300 kbit/s (12 min)	~5000 kbit/s (10 min)	~1500 kbit/s (5 min)

Table 9. Stream play lists.

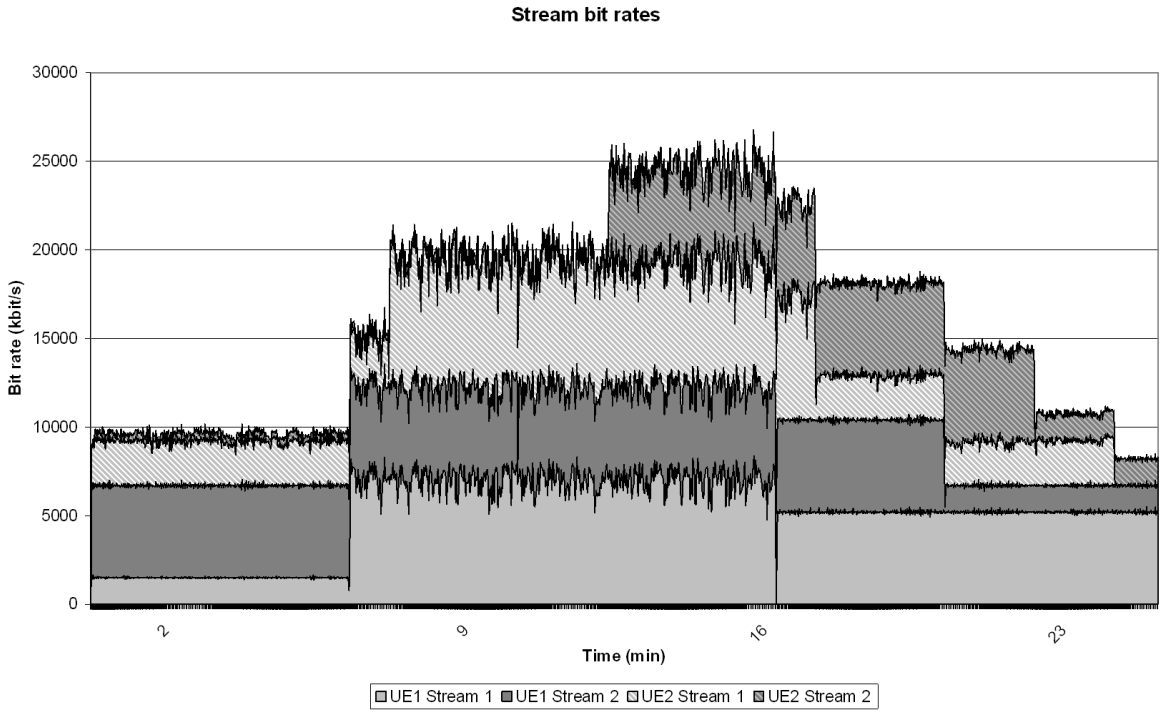

Figure 50. Stream bit rates.

In Figure 50 the bit rates of the streams as they were received at the UE are stacked on top of each other. The peak bit rate (~25 Mbit/s) seen from 12 to 15 minutes reaches the max bit rate the prototype should be able to handle in this configured state. Since we have two UEs and static equal scheduling each UE has a limit of 12.5 Mbit/s. This rate is reached by UE 1 when the first stream switches to the 7000 kbit/s video while the second UE reaches its peak at 15 minutes.

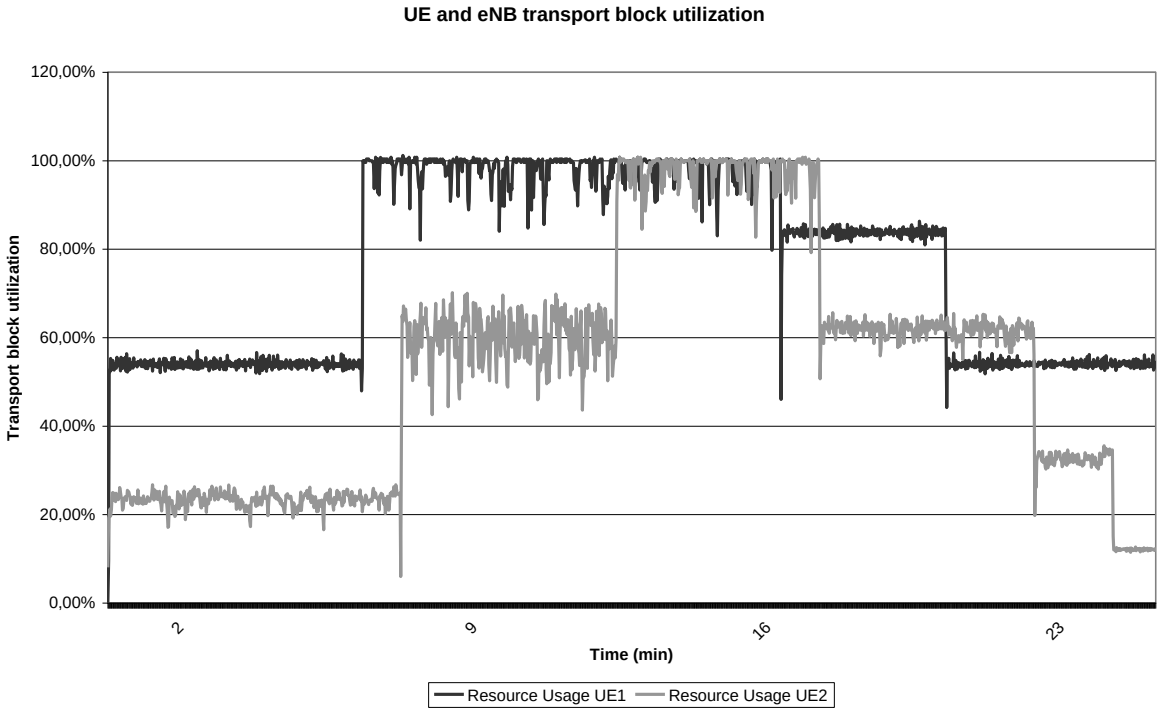

Figure 51. Transport block utilization.

It is clearly visible when the UEs reach their max bit rate in Figure 51 which shows how filled the transport blocks are.

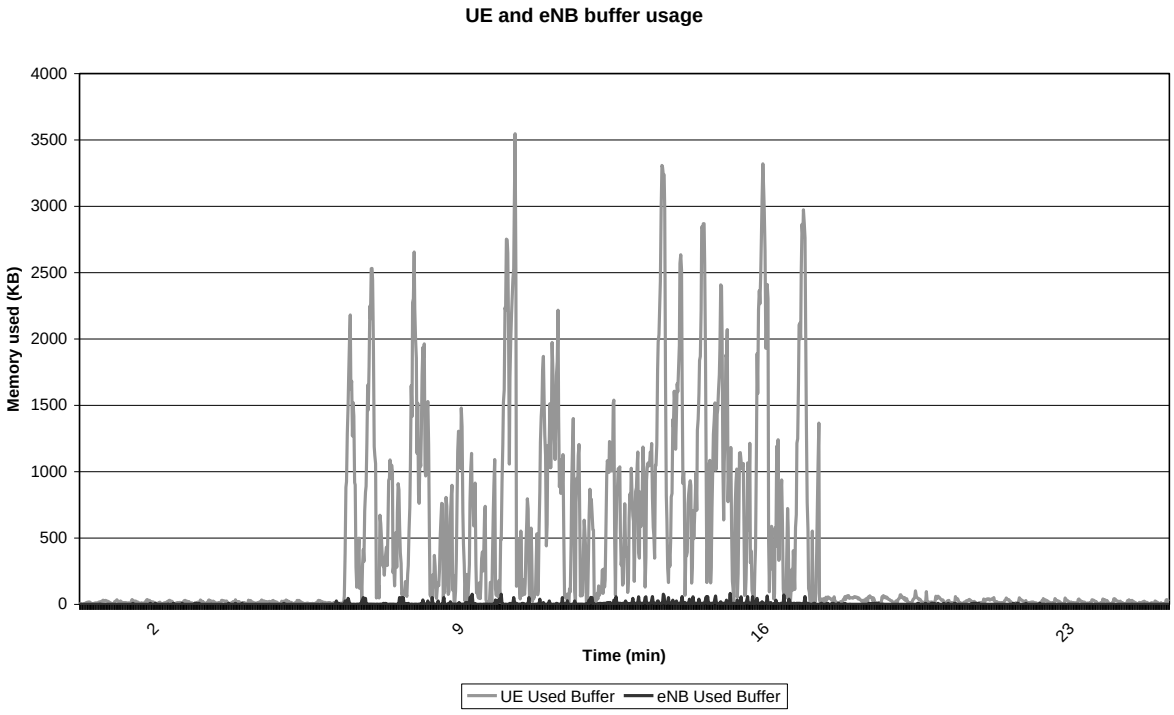

Figure 52. UE and eNB buffer usage.

In Figure 52, we see how much memory that is used in the buffer. The eNB that only receives data in this test is only using a few kilobytes since it can transmit the data to the aGW immediately. The UE buffer on the other hand is buffering up to about 3.5 MB during

the periods where one or both of the UEs have peak rates above the systems max bit rate. The buffer usage is sampled once each second, so peak values may be higher for shorter times than 1 second.

To test if the system is stable, the same test setup as above was used but with 4 steady streams in both uplink and downlink. This time the test was running for about 8 hours. Figure 53 and Figure 54 shows the streams bit rate as they exited the system at each end.

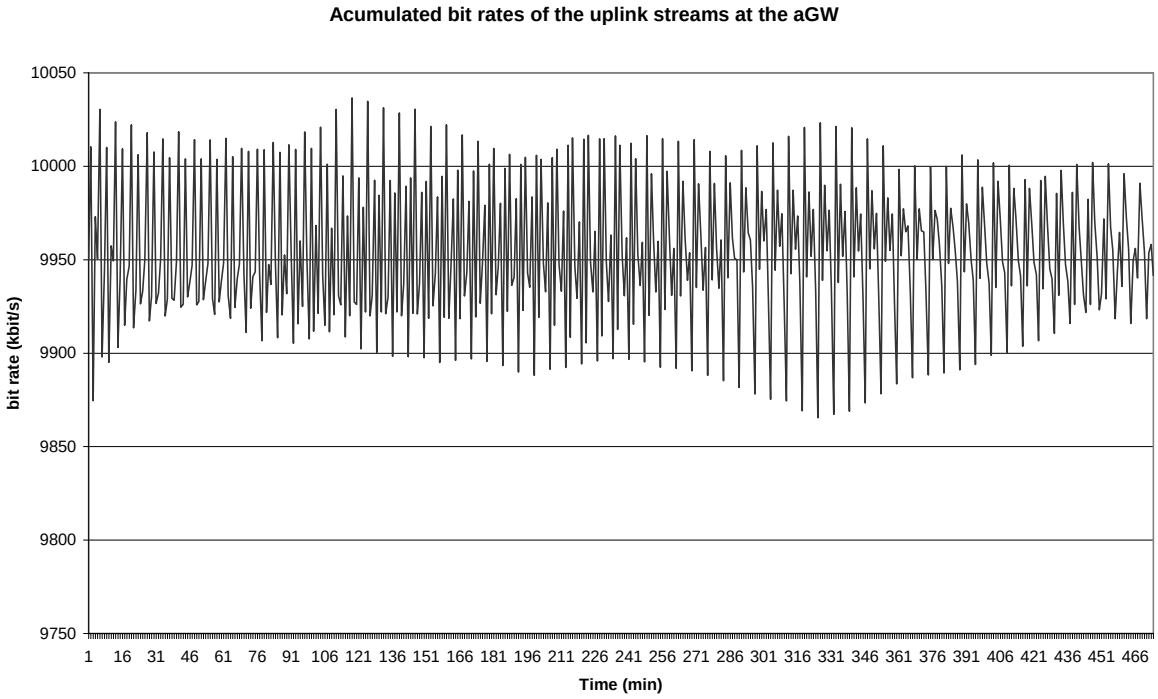

Figure 53. Accumulated uplink stream bit rates at the aGW.

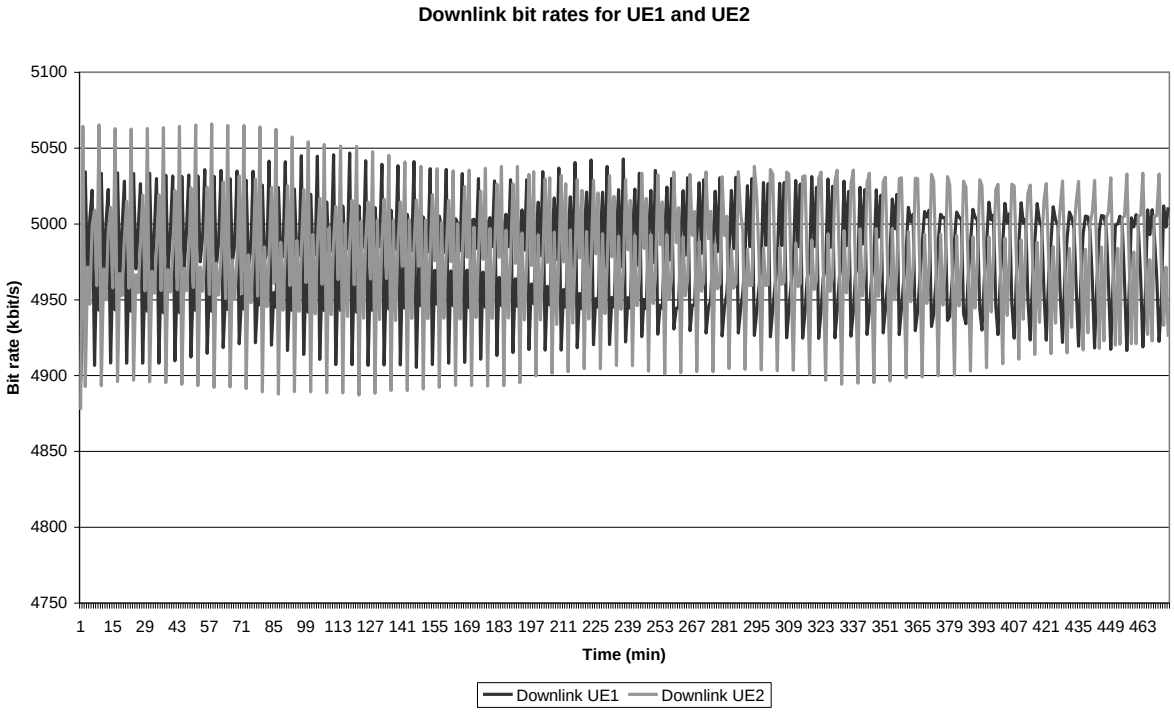

Figure 54. Downlink bit rates at the UE side.

4.12 Conclusion

We have described the design and implementation of an LTE user plane prototype consisting of a UE, eNB and aGW node. UDP data can be sent through the system in uplink and downlink. The system supports multiple UEs with multiple data streams for each UE. The main objective was to support two UEs with four streams but the system has been run successfully with hundreds of active streams.

In Layer 2 the prototype has a complete implementation of user plane data transfer in unacknowledged mode as specified in [3]. This means that we have multiplexing of logical channels in the MAC sub layer with dynamic RLC PDU sizes for optimal use of the transport block. The RLC uses segmentation and concatenation of RLC SDUs to effectively create dynamic sized PDUs without padding. RLC guarantees in sequence delivery to PDCP sub layer.

To allow multiple UEs and enable all the user plane functionality, the prototype also has some limited control plane functionality. For multiple UE support, the prototype has a static scheduling scheme supported by grant messages. MAC sub layer has limited HARQ functionality supporting retransmission using ACK and NACK messages.

5 Further Studies

In this chapter we will go a bit deeper into three aspects of LTE where there are some open issues or other considerations discussed. In the first part we cover the scheduling in LTE and continue with HARQ in the second part. In the third part of this chapter we present some of the suggested optimized alternatives on PDU structures for Layer 2. All three aspects are areas where future development of the prototype might be done which is discussed further in Chapter 6.

5.1 Scheduling

There are two schedulers in the eNB allocating physical resources, one for uplink and one for downlink. The schedulers grant the right to transmit on a per UE basis. The resource assignment consists of Physical Resource Blocks (PRB) and a Modulation and Coding Scheme (MCS). These resources are allocated for one or multiple TTIs.

A Physical Resource Block is a sub band of the frequency domain during one TTI in the time domain. This is illustrated in Figure 55.

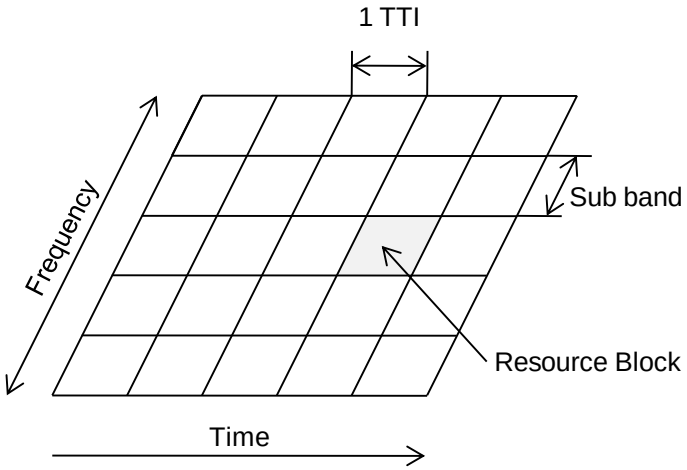

Figure 55. Resource Block in the time and frequency domain.

The baseline for both uplink and downlink is dynamic scheduling where the PRBs and MCS can be scheduled each TTI via the Cell Radio Network Temporary Identifier (C-RNTI) on the L1/L2 control channel(s). The UEs always monitor the control channel(s) in order to find any allocation of uplink or downlink resources when downlink reception is enabled.

Predefined resources can also be allocated which the UE can use if no C-RNTI is found on the control channel(s). In downlink this means that the UE does blind decoding of the predefined resources unless a C-RNTI is found in which case it overrides the predefined allocations for the TTI. In the downlink case the network decodes the resources predefined to the UEs unless the C-RNTI is present.

The scheduler should consider a number of factors when taking scheduling decisions. These factors include transport volume, QoS and measurements of the UE radio environment. In both uplink and downlink, measurement reports needs to be reported to the eNB. Radio environment measurement reports are yet to be specified.

In uplink, UE transmission buffer reports are provided to support QoS-aware scheduling. These reports refer to the data buffered in the logical channel queues on the UE and may be transmitted as a MAC control PDU. Detailed contents of the buffer reports are for further

study. The buffer reporting scheme should be flexible in order to support different types of data services and are configured using either MAC or RRC signalling.

5.1.1 Radio Bearer Priority and Rate Control

In downlink, the eNB enforces the MBR of radio bearers with a GBR and the AMBR of groups of Non-GBR bearers.

In uplink, the RRC controls the uplink rate by giving each bearer a priority and a PBR. For radio bearers with GBR, a MBR is also provided. The radio bearers are served in decreasing priority order up to their PBR. For any remaining resources the bearers are served again in decreasing priority order ensuring that the MBR is not exceeded. If all bearers have a PBR of 0, the first step is skipped and the bearers are served in strict priority order. The eNB ensures that the AMBR in uplink is not exceeded, by limiting the total amount of granted resources.

5.1.2 Real Time Services

The characteristics of real time services, such as VoIP with small packets and constant inter-arrival time, make the baseline scheduling alternatives less suitable. With dynamic scheduling the many small packages means a lot of signalling overhead. On the other hand, persistent scheduling uses less signalling but uses the bandwidth inefficiently during silent periods, which are common during voice communication. To better utilise the resources with less signalling overhead, different scheduling strategies have been suggested by members of the 3GPP in [12], [13] and [14]. The main approaches discussed are group scheduling and semi persistent scheduling. Both approaches take advantage of the silent periods which are frequent in VoIP and have less signalling overhead than dynamic scheduling.

Group scheduling is a semi dynamic scheduling approach where users are divided into groups that get scheduled dynamically. Within a group, the UEs are assigned resources using a set of predefined formats. The formats define how the resources are divided between the UEs. In Figure 56 a possible group grant structure is illustrated.

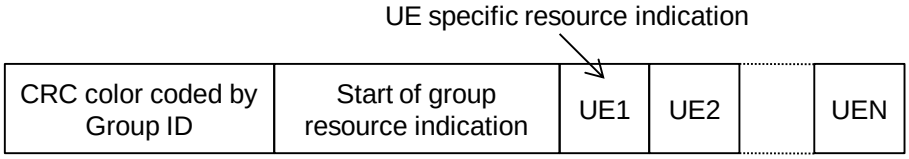

Figure 56. Possible group grant structure.

Semi persistent scheduling uses predefined allocations but switches dynamically between silent periods and talk-spurts. Retransmissions are scheduled dynamically on any available resources. In downlink, the eNB can avoid collisions by not transmitting to more than one UE at a time. This means transitions between talk and silent periods can be done without reassigning persistent resources. Dynamic scheduling of these resources can still be allowed for another UE during silent periods without collisions.

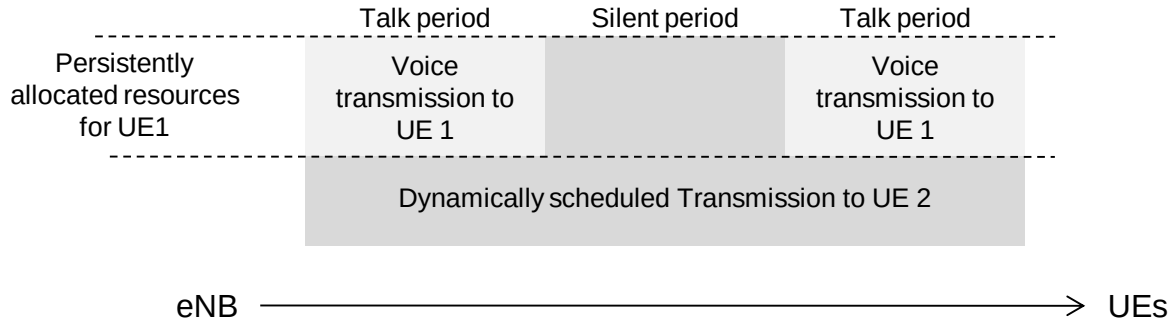

Figure 57. Example of eNB scheduling UE2 dynamically during UE 1's silent period.

In uplink however, the predefined resource can't be used by another UE since it's not known when the predefined resource will be used for transmission. Different methods to signal switching between silent periods and talk-spurts have therefore been suggested.

In [15] it is suggested that a fast Layer 1 indication should be studied to report voice activity status on the UE side. The resource used for the indication signalling is predefined and the indication can either be "switch to talk period" or "switch to silent period". It is further suggested that MAC signalling could be used for the start of silence period indication for example by adding an optional field to the MAC header. They also suggest that since voice communication is symmetric, the already needed resources for Layer 1 signalling such as ACK/NACK could be used. In [16], a similar indication is suggested but using a periodically dedicated out of band resource.

It has been agreed by the 3GPP, that semi-persistent scheduling shall be supported in both uplink and downlink.

5.2 Hybrid Automatic Repeat Request

The specification of HARQ is not finally set for LTE, and this chapter covers some of the aspects under discussion that haven't been agreed on yet and decided work that is being revised. Besides the hybrid ARQ in the MAC/PHY sub layer, there is an ARQ in the RLC sub layer that provides error correction by means of retransmissions in acknowledge mode at Layer 2. The two schemes may interact in LTE and these HARQ/ARQ Interactions are also covered in this chapter.

The LTE related references in this chapter is from [11] and meeting protocols mainly from 3GPP RAN WG2 Meeting 58 and 58bis.

5.2.1 Synchronous and Asynchronous HARQ

The HARQ protocol may either use asynchronous or synchronous mode for data transmission. In the synchronous mode, retransmissions occur at predetermined time instances. In contrast, when using the asynchronous mode, the retransmissions may occur at any time. Current working assumption for LTE is to retransmit transport blocks synchronous in uplink and asynchronous in downlink [11]. However, there may be optimizations for downlink where a synchronous approach is discussed as an option.

One property of synchronous HARQ is that there is no need for signalling the HARQ process identification number along with transmission or retransmission, in contrast to asynchronous HARQ where it is required. The reduced control signal overhead is particularly advantageous for uplink.

When using asynchronous mode, where retransmission can occur at any time, packet assignment is more flexible and that's an argument for using it in the downlink.

In [17], it is explained that asynchronous HARQ is more flexible in packet assignment compared to synchronous HARQ. A downlink scenario is illustrated in Figure 58, where both persistent scheduled users and normal scheduled users operate. When using synchronous HARQ in downlink some of the retransmissions are restricted to the UEs with highest priority. Assuming that persistent scheduled UEs are given higher priority than normal scheduled UEs, retransmission from the normal scheduled UEs is postponed when the TTI of the regular transmission timing is assigned to persistent scheduled users. The same methodology applies when the normal UEs are given higher priority, i.e. persistent retransmission is postponed. When using asynchronous HARQ these problems do not arise.

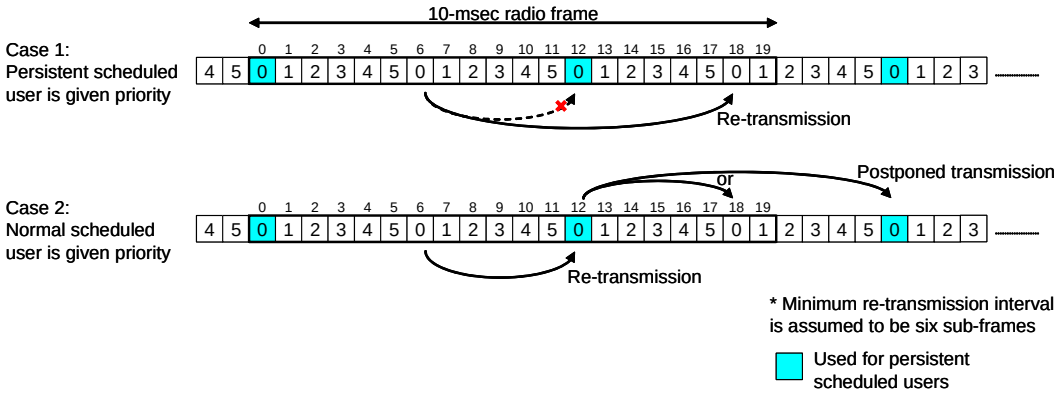

Figure 58. Synchronous HARQ. [17]

Another advantage in terms of flexibility in packet assignment is when adaptive TTIs are supported, i.e. variable TTI lengths. If assuming synchronous HARQ, it is mentioned that it may not be possible to retransmit after the retransmission interval when a different TTI length is accommodated within a specific radio frame.

5.2.2 Feedback Aspects

HARQ is based on positive and negative acknowledgements, which are needed to support retransmissions of unsuccessfully decoded transport blocks. If a transport block must be retransmitted, it would preferably be performed as soon as possible.

The HARQ Round Trip Time (HARQ RTT) is a configured value which indicates when a possible retransmission can be performed. As the transmission time over the wireless link is very fast, the processing delay in the UE and the eNB respectively determine how to configure the HARQ RTT. In [24], the processing delay is proposed to be 3 ms at minimum in both the UE and the eNB which fits a HARQ RTT of 8 TTIs. Considering different UE categories, it may be a drawback to have a common minimum processing delay, simply because some UEs may have a significantly shorter processing time than the configured time. The HARQ feedback timing may have an impact on those UEs when requiring the highest data rates.

A common minimum processing delay between UEs has benefits especially in terms of complexity. The scheduler only needs to support one HARQ timeline, in contrast to when having dynamic configuration between UEs which would result in more scheduler complexity. One argument for having a common minimum processing delay between the UE and the eNB is that an uplink transmission with a downlink reception for data and HARQ feedback may simultaneously operate when the nodes are awake, i.e. the sleep time will

increase [24]. This scenario is illustrated in Figure 59 where traffic is sent simultaneously in uplink and downlink.

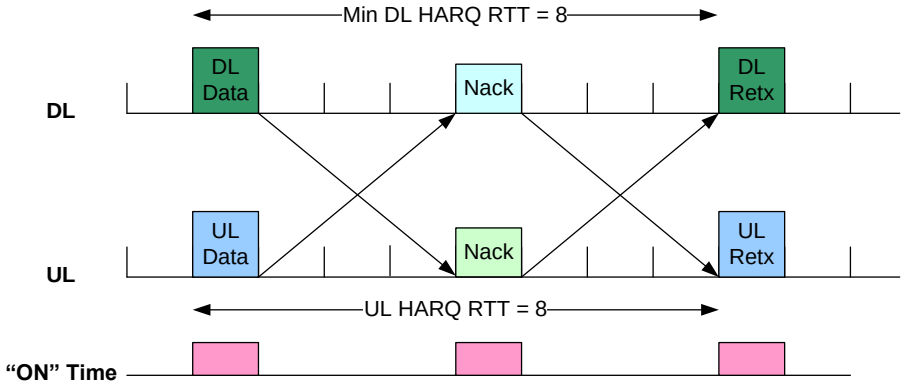

Figure 59. HARQ RTT. [24]

If signalling of both ACKs and NACKs are supported, these acknowledgements may be misinterpreted at the transmitter. An ACK may be misinterpreted as an NACK and vice versa. The ACK to NACK error can result in retransmissions of redundant data and the NACK to ACK results in packet loss if not handled by ARQ. The NACK to ACK error is covered in Chapter 5.2.5 HARQ/ARQ Interactions.

Assuming that one type of acknowledgement is signalled and the other is time-based, probability of misinterpreted acknowledgements and signalling overhead decreases. In case of explicitly signalled NACK and time-based ACK, the ACK to NACK error can be avoided and control signalling overhead decreases because there is no signalling of the ACK. Whether or not this may actually be advantageous in practice is out of scope of this paper.

When the ACK/NACK support is active for new transmissions or retransmissions, each transmission expects an ACK/NACK in general. In [25], this general approach is not followed by allowing multiple transmissions to have a common ACK/NACK for a special case. The case considers uplink traffic with power limitation which may occur at the cell borders. Assuming power limitation, an assigned transport block may be very small in size and thereby for example not hold even a VoIP packet. The discussed approach is to allow transmission of a single VoIP packet in more than one TTI by bundling TTIs, i.e. no segmentation of the VoIP packet is needed.

In response to the bundled TTIs, there is one single ACK/NACK response. The problem with staying within the synchronised HARQ process pattern may be solved by making one HARQ RTT idle. Some of the mentioned benefits are less overall header overhead as segmentation of the packet might be prevented, less grants and HARQ feedback signalled implying less probability for grant failures and ACK/NACK misinterpretations.

5.2.3 Synchronization of HARQ Processes

The process identification number is required to be signalled in conjunction with data when prioritizing new transmission over retransmission. This signalling occurs in downlink asynchronous HARQ, where flexible packet assignments are advantageous. In uplink synchronous HARQ, where retransmissions always have the highest priority, the process identifier is not required as it follows a HARQ process pattern known by both transmitter and receiver.

For supporting data combining based on revision versions, the revision version used by the turbo coding must be known for the decoding at the receiver. There are various ways to

approach this synchronization, where some require signalling and other do not. For uplink synchronous HARQ, it is possible to achieve this without explicit signalling. This option is stated in [27] where non-scheduled retransmission is assumed. However, the reference does not include explanation of the synchronization procedures.

One alternative is to simply use knowledge from the HARQ process pattern and perform proper synchronization of the HARQ process parameters. After each transmission, the transmitter increases the revision number by one, if the upper limit isn't reached. The number of transmission attempts is always increased by one after each transmission. These parameters are set to default at the transmitter if the corresponding HARQ feedback is an ACK or a NACK in case the maximum number of transmission attempts is reached.

At the receiver, the expected revision number is increased by one, if the upper limit isn't reached. The number of transmission attempts is increased by one if the decoding fails. However, in case of decode failure and when reaching maximum number of transmissions, the parameters are set to default. This will also be the case when the decoding is successful.

5.2.4 Link Adaptation

When data is transferred over the wireless link it is obvious that bit errors may occur quite often. This may result from wireless characteristics such as decreasing signal strength due to mobility and in cases when the signal passes through obstacles. Furthermore, bit errors may occur when other sources interfere with the signal and interference through time dispersion.

The Signal-to-Noise Ratio (SNR) is a relative measure of the received signal and background noise [19]. A larger SNR makes it easier for the receiver to extract the signal from the background noise and it is related to the bit error rate (BER). The higher the SNR, the lower the BER. If the BER is high, the transmitter may adapt to the conditions by choosing a MCS accordingly.

Asynchronous and synchronous HARQ may support adaptive and non-adaptive configuration. When using adaptive configuration, the transmission format may be changed between initial transmission and retransmission of a transport block. The control data which may be signalled for adaptive retransmissions is modulation scheme, channel coding rate and assigned resource blocks [11].

In [18], there is a discussion on slow link adaptation schemes for VoIP, but the discussion may be interesting for other services as well. It is mentioned that the impact of fast link adaptation is relative small when considering VoIP. The gain of successful transmissions is rather from HARQ operation than the link adaptation, because payload does not vary a lot in size when comparing to background services. One of the proposed adaptation schemes is based fully on NACK-based rate control and a second on NACKs and CQI. If the NACK rate is high, the transmission rate may fall back to a lower rate, and it can increase the transmission rate if the ACK rate is high.

Current working assumption for LTE is to support adaptive transmission parameters in downlink but optimizations might be added [11]. The choice for uplink is also for further study regarding the adaptive approach. Some proposed options are briefly covered in the following sub chapter.

5.2.4.1 Synchronous HARQ in uplink

During the RAN2 meeting #58 in Kobe, a decision was made to support both non-adaptive and adaptive synchronous HARQ in uplink. However, there are different opinions on the

adaptive approach. To generalise, all proposed options for uplink adaptation covered in this chapter solves resource fragmentation but differs a lot in other aspects.

The problem of resource fragmentation with synchronous non-adaptive HARQ is stated in [20], and an example is given assuming three different UEs being allocated in the same TTI with resource allocation 1, 2 and 3 respectively. Assuming that a retransmission is only needed for the UE corresponding to resource 2, it results in resource fragmentation. A single UE can then not use the whole available resource when the retransmission is performed. Only resource 1 or 3 may be used for a single UE and this could lead to inefficient usage of the bandwidth and limit throughput and system capacity.

In [21] a synchronous adaptive HARQ configuration for uplink is proposed, where each retransmission is scheduled. This solves the problem with resource fragmentation and it is not needed to signal explicit ACK/NACKs, since the scheduling of a retransmission is an implicit NACK.

An obvious concern is the increased control signalling overhead and in [22] there is a harmonized approach based on group scheduled retransmission and optional individual grants. The group scheduling is proposed to use a resource block bitmap which include retransmission information for a group of UEs.

A more control signal advantageous approach is discussed in [20]. It solves resource fragmentation by transferring explicit grants only when needed. If no explicit grant is sent, the resource allocation and MCS will be the same for a retransmission as for the original transmission. Notice that there is a need for ACK/NACK but there is no need to transfer explicit grants for each retransmission.

When using fast link adaptation, as in the papers proposing to schedule all retransmissions, channel quality information is instantaneous feedback to the transmitter which allows the transmitter to choose the optimal MCS for each retransmission. For VoIP it is reported that two retransmissions in average per packet would be needed [23]. There is however a major concern on the impact of the signalling overhead, as the overhead could reduce system capacity and increase link interference.

5.2.5 HARQ/ARQ Interactions

In this chapter, two vital HARQ residual errors are covered, which include a first case where maximum number of transmission has reached a defined limit and a second case where HARQ acknowledgement in form of a NACK is interpreted as an ACK at the transmitter, i.e. NACK to ACK error.

The recovery of these errors may be solved in different ways and the focus in this chapter lies on error recovery with HARQ assisted ARQ operation, i.e. by means of HARQ/ARQ interactions.

5.2.5.1 Maximum number of transmission attempts

As specified in [11], the first case may be solved by HARQ/ARQ interaction, i.e. if there is indication of maximum number of HARQ transmission attempts for a transport block, the relevant transmitting ARQ entities are notified and retransmissions and resegmentation can be initiated. A negative acknowledgement (NACK1) is sent from HARQ to ARQ when the transmitter receives an NACK corresponding to a MAC PDU which has reached the maximum number of attempts limit.

The maximum number of HARQ transmission attempts may either be dependent or independent of the served Radio Bearers Quality of Service class. In [26], it is argued that though it seems straightforward to let loss-tolerant services such as VoIP perform 1 or 2

attempts, in contrast to services such as file transfer which might target up to 5 or 6 attempts, there are some limitations to this dynamic approach. First of all, there is a concern regarding control signaling overhead, especially in the uplink operation. The UL scheduler in the eNB must be aware of upcoming HARQ retransmissions, and the maximum number of HARQ retransmission attempts must be signaled from the UE to the eNB, i.e. increased control signaling overhead. Another concern is that prior to each new transmission, the adaptation mechanism chooses a suitable transport format to approach the desired initial decoding probability. The link adaptation decision is based on channel quality estimates which are likely to be inaccurate. This implies that the inaccuracy of the channel quality estimates does not allow precise tuning of the HARQ residual error rate.

5.2.5.2 NACK to ACK error

Considering the second case, i.e. NACK to ACK error, it is for further study if the HARQ receiver is able to detect a NACK to ACK error by ARQ entity notification via explicit signaling.

In Figure 60 there is an example on how to detect and recover from an NACK to ACK error in downlink. Asynchronous adaptive HARQ is assumed and a redundancy version is included in the assignment, which implies that if an assignment is received, it is known to the UE whether new data is sent or not.

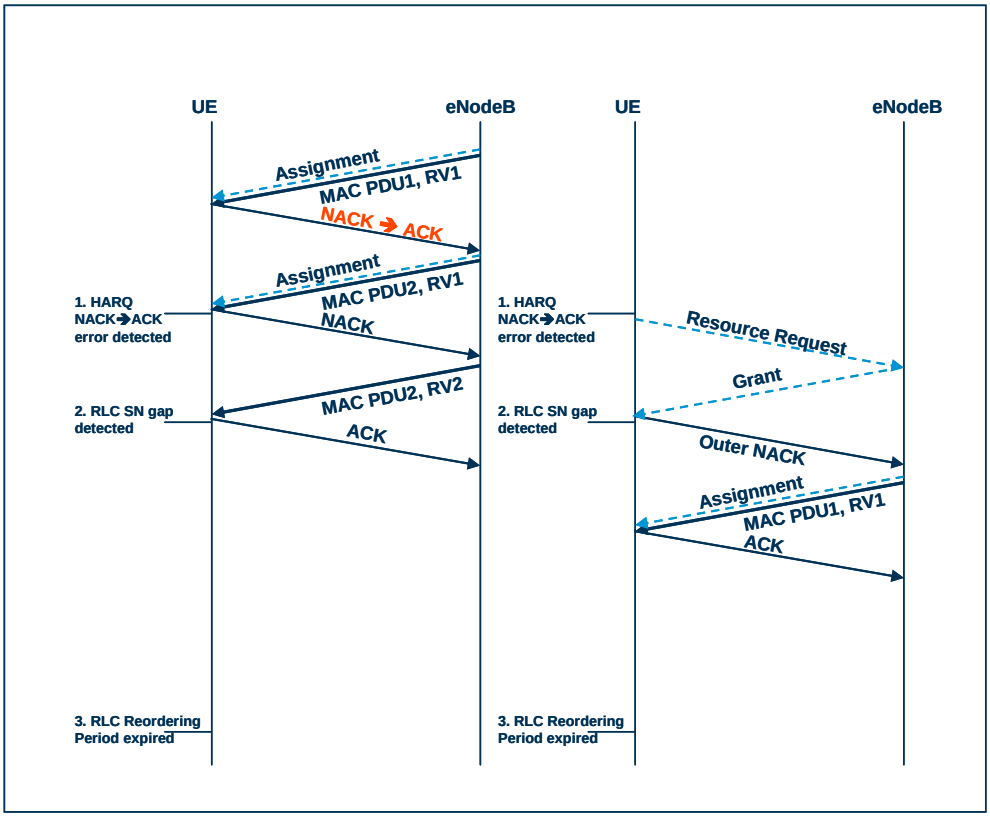

Figure 60. NACK to ACK error in downlink. [27]

As seen in the left part of Figure 60, the MAC PDU with revision number 1 is positive acknowledged at the eNB because it misinterpreted the NACK as an ACK. The HARQ error is detected when the UE receives MAC PDU2 because it's not the expected, i.e. an assignment is not expected with the data received. Then a resource request might be needed and a grant in response. When the grant is received at the UE a RLC sequence number gap is

detected where the missing RLC PDUs may be derived thereof. References to these missing segments may be reported to the eNB via an outer NACK, e.g. MAC control PDU. The eNB triggers the RLC transmitter based on the references from the outer NACK, i.e. trigger with a NACK2, and may retransmit the data or segments thereof if resegmentation is needed.

In Figure 61 there is an example on how to detect and recover from an NACK to ACK error in uplink. There are a predefined maximum number of transmission attempts which is common for all radio bearers. No uplink control signaling of neither redundancy version nor new data indicator (NDI) is sent in conjunction with data. The NDI-less HARQ operation [28] assumes explicit grants for new transmissions and implicit grants for retransmission, meaning that NACKs serves as grants.

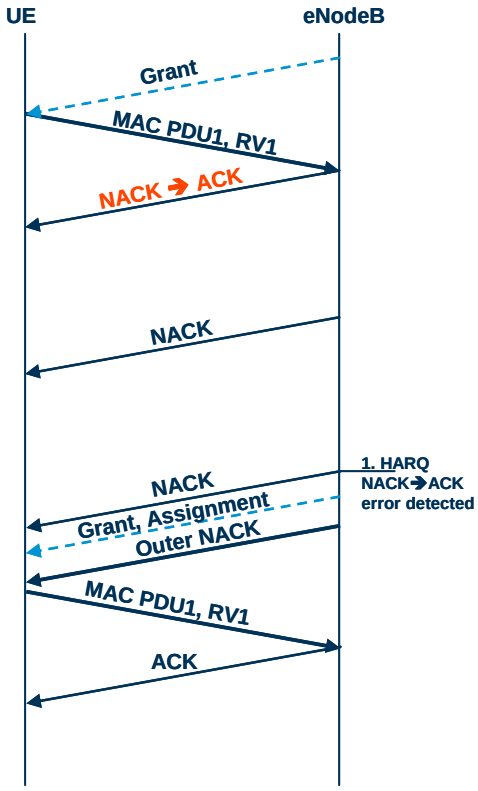

Figure 61. NACK to ACK error in uplink. [27]

As seen in Figure 61 the NACK to ACK error is detected when the eNB has reached up to the maximum number of receptions for a transport block, which is synchronized with maximum number of transmission attempts. Notice that eNB do not receive any transport blocks at all after the first transmission because the UE has been locally acknowledged, i.e. stops transmitting the transport block because it's interpreted as successfully decoded.

Upon error detection the eNB responds with an outer NACK that informs the UE that the previous transmission for a given process had failed. The recovery may then be performed by retransmitting the referenced data or segments thereof if resegmentation is needed, i.e. based on NACK2 references.

5.2.5.3 Support of NACK1 and NACK2

Since the HARQ/ARQ interaction for NACK to ACK error is not agreed on yet it is interesting to compare the two options in terms of performance gains, RLC simplifications and reduction of overall Layer 2 complexity.

Simulation results presented in [29] shows that performance gains with HARQ/ARQ interactions were small in terms of RLC recovery delay and file transfer download response time. There are however potential performance gains in reducing status reporting overhead by using NACK2. One may argue that potential performance gains and overhead is preferred as long as the added level of functionality is acceptable as mentioned in [30]. One drawback with adopting NACK2 may be when operating in UM because HARQ failure reports may be sent which is a waste of resources. However, this could in theory be solved by simply not report these errors. A rare scenario which results in unnecessary ARQ retransmissions is when an ACK to NACK error occur for the last transmission. According to [31] most companies assume that reaching the HARQ retransmission limit is in the range of 0.1 to 1 percent and the ACK/NACK error probability assumes to be 0.1 percent. This implies that this case occurs as seldom as in the range of 0.001 to 0.0001 percent.

According to [30] potential Layer 2 simplifications are the most compelling argument for supporting both NACK1 and NACK2. The added complexity to HARQ is supporting one NACK2 function and additional timer, but the ARQ may remove 5 functions and 4 timers. In Figure 62 which is based on [31] the overall simplification is shown.

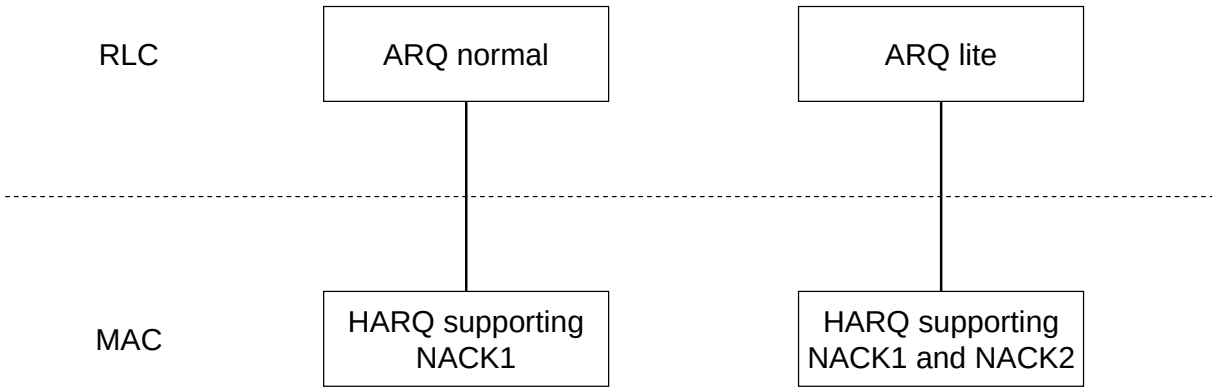

Figure 62. HARQ and ARQ alternatives

5.3 PDU Structure Optimizations

The PDU structures in LTE must be able to handle large IP packets of up to 1500 bytes and therefore large header fields are needed to support those lengths. But when the major part of the traffic are supposed to be VOIP calls and each VOIP packet is relative small, a reduction in the header fields for small packets could greatly decrease the bandwidth needed for services like VOIP. In the following chapters we will further discuss some different approaches to reduce the header overhead to get a more optimized PDU structure.

The final specifications for the PDU structures are not yet done but we will have a look at different proposals from different companies on their view regarding further optimizations for the PDU structure. We will also have a little discussing regarding the strengths and weakness of each proposal and what it could mean if they were implemented. We will be focusing on the user plane for Layer 2 sub layers (MAC, RLC and PDCP).

5.3.1 PDU optimisations for VoIP

In [32] Nokia proposes how to optimize the PDU structure of the MAC and RLC sub layer for small packets like VOIP.

Nokia propose a MAC PDU with two formats, one short and one long format. The long MAC PDU format is used for general data traffic when an SDU is over 128 bytes long. The short format has a reduced length field and can be used for VOIP and any RLC UM data that is shorter than 128 bytes. A format (F) flag is used to indicate which format is in use; this flag is also used to indicate the format (long or short) of the RLC PDU. When the short format (F=1) is in use a HE field will also be visible indicating which short format of the RLC PDU that is in use. The long and short formats are illustrated in Figure 63.

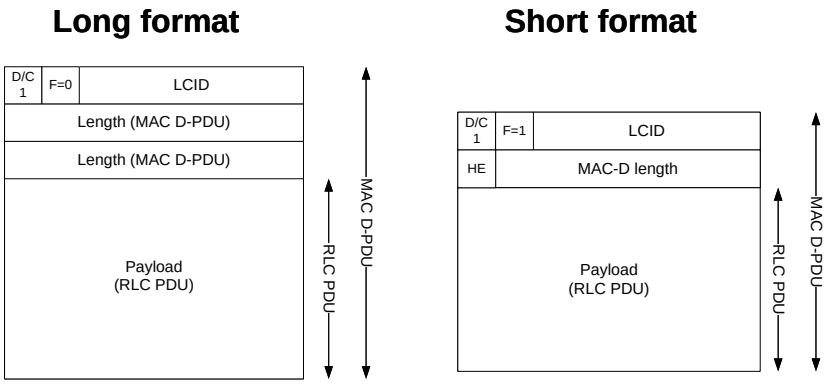

Figure 63. Long and short format for MAC PDU. [32]

For the RLC PDU, Nokia propose three formats, one long and two short formats as illustrated in Figure 64. The long RLC PDU format is used for general data traffic when an SDU is over 128 bytes long and supports in sequence delivery, retransmissions, segmentations, concatenation and polling. The short format is intended for UM traffic and therefore no retransmission and polling is needed, and when there are no retransmissions the RLC sequence number is not necessarily needed as the reordering may be done at the PDCP sub layer instead. In fact if the RLC PDU is a complete SDU no RLC header at all is needed, only the HE bit in the MAC header tells if a RLC header is present or not.

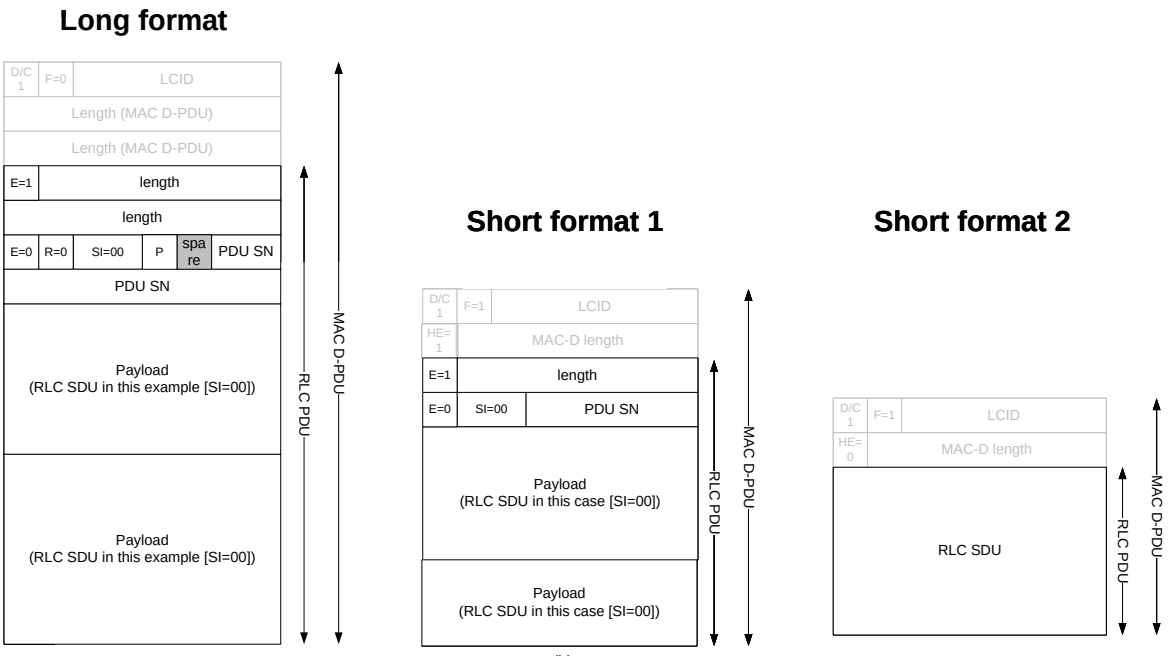

Figure 64. Long and short format for RLC PDU. [32]

5.3.2 Length Indicators

Ericsson [33] proposes a dynamic sized Length Indicator depending on the actual size of the corresponding data unit instead of a static sized LI based on the theoretical maximum.

When creating the MAC header on the transmitting side, the length of the LI will be based on the size of the transport block by using the following formula:

$$LI\_Size\_Opt = ROUNDUP(Log_2(MAC\_PDU\_Size_{byte}))$$

The LI for the RLC PDU will be based on the size of the RLC PDU in a similar way using the following formula:

$$LI\_Size\_Opt = ROUNDUP(Log_2(RLC\_PDU\_Size_{byte}))$$

The same rule is applied on the receiving side. The receiving MAC entity knows the length of the transport block from the transport format transmitted on a L1 control channel. And the size of the RLC PDU is then known from the LI in the MAC PDU.

By using these formulas each LI will be adapted to the actual size of the MAC and RLC PDU, meaning that smaller MAC and RLC PDUs will have smaller LIs than larger MAC and RLC PDUs.

5.3.3 Byte alignment

It has been agreed that all Layer 2 protocols (PDCP, RLC and MAC) shall be byte aligned to avoid any need of processing demanding byte shifting of Layer 2 payload. Ericsson [34] proposes that the RLC and MAC header together should be byte aligned instead of each header individually as illustrated in Figure 65. By doing this however all the MAC and RLC headers needs to be located in the beginning of the MAC PDU. Ericsson claims that by byte aligning the RLC and MAC header together instead of separately it will reduce the header overhead and also simplify header parsing in the receiver.

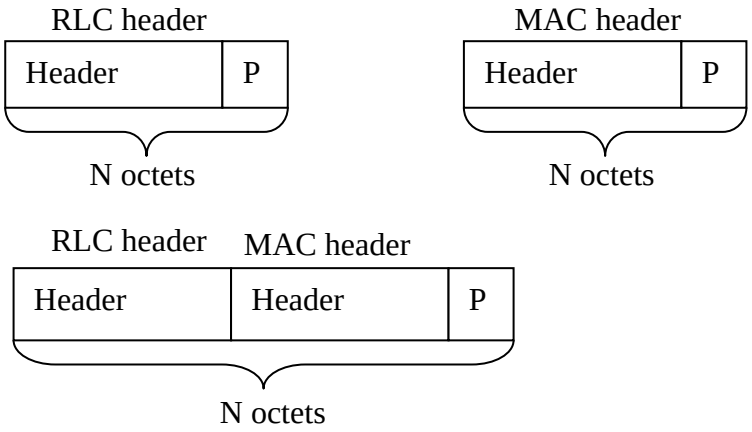

Figure 65. Byte aligning MAC and RLC header in start of MAC PDU (P=Payload). [34]

5.3.4 Reusing PDCP Sequence Number for RLC

InterDigital Communications Corporation (ICC) [35] proposes that the RLC should reuse the PDCP sequence number instead of a RLC PDU sequence number. By using one common Layer 2 sequence number ICC claims that it will not only reduce the header overhead but also make the system less complex as you only need to assign and maintain one sequence number instead of two. This method will also make it possible to remove the concatenated PDCP sequence numbers from a RLC PDU, in other words you only need one PDCP sequence number per RLC PDU (see Chapter 5.3.5 for a proposal regarding "*Removing of PDCP SN in RLC PDU payload*"). ICC has compared the two approaches and has come to the conclusion that reusing PDCP SN is superior. The result is summarized in Table 10.

Evaluation Criteria	Common L2 SN	Additional RLC PDU SN
Memory and operations required for SN assignment & maintenance	+ Single set of SN counters and SN operations	– Complexity increases due to duplicating operations
Reordering function per logical channel / Radio Bearer	+ Single reordering function for both PDCP/RLC	– Complexity increases due to duplicating functions
Sub-layer independence between the RLC and PDCP	Dependent sub-layers, but no issue or added complexity	Independent sub-layers, but no tangible benefit
Applicability to RRC and Other Traffic Types	Works for all traffic types	Works for all traffic types
Overhead for TCP traffic without ROHC-TCP compression	+ More efficient	– Increases overhead, reduces capacity (~2% worse on average for IMIX)
Overhead for TCP traffic with ROHC-TCP compression	+ Substantially more efficient	– Increases overhead, reduces capacity (~14% worse on average for IMIX)
Overhead for VoIP traffic with ROHC header compression	+ Significantly more efficient	– Increases overhead, reduces capacity (3.6% worse on average for AMR)

Table 10. Comparison between using additional RLC SN and reusing common SN. [35]

5.3.5 Removing of PDCP Sequence Number in RLC Payload

Alcatel-Lucent [36] proposes that the redundant PDCP sequence number is removed in the transmitting RLC entity and then resumed in the receiving RLC entity. The procedure of PDCP removal in the transmitting RLC entity is illustrated in Figure 66 and can be summarized in the following points:

- In case of concatenation:
  - All RLC SDUs' PDCP SN except the first one is removed and one RLC PDU only contains the PDCP SN corresponding to the first RLC SDU.

- If the first concatenated part is the last segment of a RLC SDU, only the PDCP SN corresponding to the second concatenated part is included.

- In case of segmentation the PDCP SN is included only with the first segment and then if the RLC PDU only contains one segment which is not the first segment of its original RLC SDU, no PDCP SN is needed.
- In case of no segmentation and no concatenation the two independent PDCP SN and RLC SN are kept per every PDU.

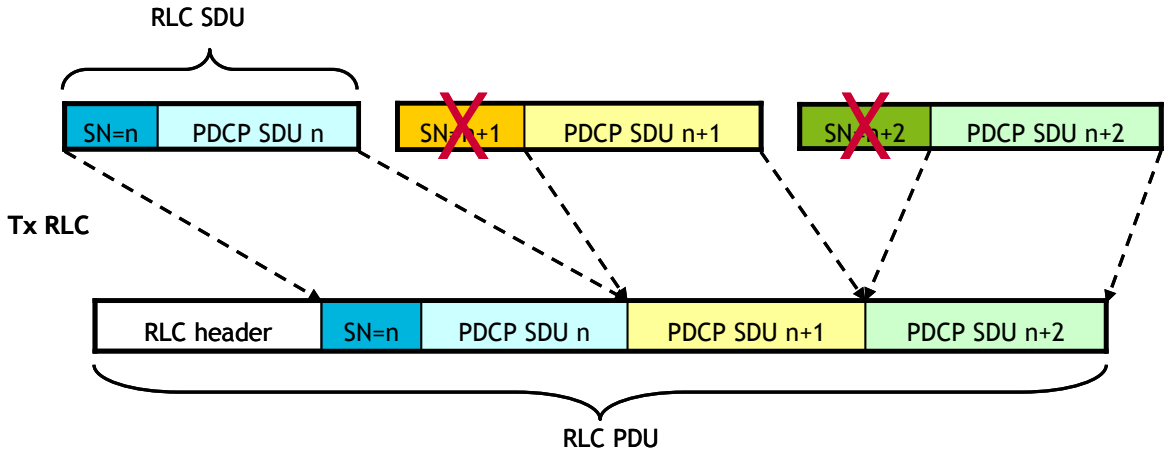

Figure 66. Concatenation in RLC where the redundant PDCP SN are removed. [36]

5.3.6 Discussion

When deciding if an optimization is beneficial or not you have to take more into consideration than just how it affects the PDU overhead. Will the optimization make the system too complex? Does it require extra expensive equipments? How will it affect the power consumption? Etc. We will go through each of the proposals and try to determine what their strengths and weakness are. Let start with Nokia's *"PDU optimization for VOIP"*:

PDU Optimization for VoIP

Nokia say that the short format will be used when the payload is under 128 bytes otherwise the long format will be used. That should mean that the short format has a 7 bit long length indicator. When no segmentation or concatenation is needed Nokia mention that no RLC header at all is needed, and when there is a need for concatenation/segmentation the length indicator as in the MAC PDU for the short format should be reduced. They also mention that the sequence number in the RLC PDU could be reduced for short format.

In Table 11 you can see a comparison between the header overhead in the prototype and Nokia's proposal. The comparison is for one stream with either small or large SDUs. The prototype uses a general PDU structure as described in Chapter 4.9 where no specific optimizations are used.

Cases	Prototype (RLC/MAC/PDCP) overhead (bits)	Nokia (bits)	Bits saved
1 small* PDCP PDU	33	24	9
1 large** PDCP PDU	33	34	-1
N small* PDCP PDUs	25 + 8 + (N-1)*28	27 + Red. RLC SN + (N-1)*24	-2 + Red. RLC SN + (N-1)*4
N large** PDCP SDUs	25 + 8 + (N-1)*28	26 + 8 + (N-1)*28	-1
*Small refers to RLC SDU that is smaller than 128 bytes. **Large refers to RLC SDU that is larger than 128 bytes.			

Table 11. Comparison of header overhead between Nokia's proposal and the prototype.

By looking at the comparison with the prototype we can clearly see that Nokia's method will reduce the header overhead for small SDUs. The worst case is when using large SDUs than it will actually increase the header overhead. We can also see that the biggest impact on the header overhead for one small SDU is the removing of RLC Sequence Number (8 bits).

(Note that the reduced RLC Sequence Number is not specified in Table 11, that's because the length for it wasn't specified in the proposal)

Length Indicator Optimizations

An advantage with this solution is that it doesn't add any extra fields to the different headers and therefore doesn't complicate the header structure. This method will work best when there are very few SDUs in a PDU. That's because the size of the Length Indicators is defined by the total size of the PDU and not for each SDU. For payloads where the total sum of the length of all SDUs are equal or bigger than the maximum size of the Length Indicator this method will have no affect at all. A good thing with this method is that there isn't any need for any extra communications between the Layer 2 sub layers that could affect the flexibility of the system.

Byte Alignment of RLC and MAC

This method doesn't add any extra fields to any headers either but with this method all MAC and RLC headers needs to be located in the beginning of the MAC PDU. That would require a little different approach when parsing the headers compared to if a header always follows by the payload. The problem that would arise is that the MAC sub layer can determine how many RLC PDUs the transport block contains by parsing the MAC header, but it doesn't know how long each RLC header is (it only knows the combined length of the RLC header and RLC payload). The RLC sub layer on the other hand can determine how long each RLC header is by parsing them but doesn't know how many RLC headers that would follow. So there has to be some kind of communication between the MAC and RLC sub layers, or to give MAC sub layer the ability to parse RLC headers.

Reusing PDCP Sequence Number for RLC

Cases	Prototype (RLC/MAC/PDCP) overhead (bits)	ICC (bits)	Bits saved
1 small* PDCP PDU	33	25	8
1 large** PDCP PDU	33	25	8
N small* PDCP PDUs	33 + (N-1)*28	25 + (N-1)*28	8
N large** PDCP PDUs	33 + (N-1)*28	25 + (N-1)*28	8

**Small refers to RLC SDU that is smaller than 128 bytes. **Large refers to RLC SDU that is larger than 128 bytes.*

Table 12. Comparison of header overhead between ICC's proposal and the prototype.

This method will decrease the overhead for all RLC PDUs with 8 bits but you will also lose the independence between the PDCP sub layer and the RLC sub layer. On the other hand you will only need one common reorder function compared to two if both the sub layers has there own sequence number. In LTE the PDCP sub layer has been moved to the same node as the RLC sub layer, so there might be an idea to further investigate if both the PDCP and the RLC sub layer still need its own sequence number and reordering function. If both aren't needed, this method could be a solution.

Removing of PDCP Sequence Number in RLC Payload

Cases	Prototype (RLC/MAC/PDCP) overhead (bits)	Alcatel-Lucent (bits)	Bits saved
1 small* PDCP PDU	33	33	0
1 large** PDCP PDU	33	33	0
N small* PDCP PDUs	33 + (N-1)*28	33 + (N-1)*12	(N-1)*16
N large** PDCP PDUs	33 + (N-1)*28	33 + (N-1)*12	(N-1)*16
**Small refers to RLC SDU that is smaller than 128 bytes. **Large refers to RLC SDU that is larger than 128 bytes.*			

Table 13. Comparison of header overhead between AL's proposal and the prototype.

This method only affects RLC PDUs that has more than one SDU and require the RLC sub layer to be able to parse PDCP headers. The weakness with this method is that if the structure of the PDCP header is changed it may not only affect the PDCP sub layer but may also affect the parser in the RLC sub layer. The strength with the method is that it can significantly reduce the bandwidth needed when there are many concatenated SDUs and especially for small packet it could really decrease the overhead percent.

When looking at all these proposals it seems that many of them will increase the communication between the Layer 2 sub layers in order to reduce the overhead. This will prevent the sub layers from being independent of each other and a small change in one sub layer may have affect on all sub layers. So before implementing these optimizations you have to consider which "road to walk". Do we want a clean and non complex system or a more optimized one? How important is the flexibility and complexity of the system compared to a more optimized PDU overhead? You may say that each of these proposals may not increase the complexity of the system any significantly but adding them all together and with further optimizations in mind you have to draw the line somewhere.

6 Discussion and Future Work

The approach on the prototype development was to first design a very basic system with one UE and one stream, implement it and build a more complete system around it. The resulting basic system turned out to be too basic when new functions not planned for it had to be added. This led to some of the architecture getting redesigned when it became clear that we needed HARQ and scheduling to properly allow multiple UEs and streams. Together with changes in the overall node structure, this redesign resulted in adding a PHY sub layer to transfer control information between the UE and the eNB. The major functionality implemented in the Layer 2 sub layers were however mostly unaffected by the changes. Some time could have been saved by designing the first architecture as a temporary framework to run data through the layers. Then design the full system once it was more clear what functionality was needed.

One of the main questions for the nodes was how the work should be split up into threads and processes and how they should communicate with each other. We started with a design where each sub layer was running in its own thread. From there we reduced the number of threads processing the data to one in downlink and one in uplink, along with one clock thread and one thread monitoring the UDP sockets per UDP object.

The UEs could have been implemented as one node for each UE. It would have given us a prototype where each UE is truly independent like in a real life system, but it would not scale as well with an increasing number of UEs. The main problem is that we don't have enough control of the OS scheduling to effectively run a large number of UEs on the same PC. Therefore we decided to use one process and run all the UEs within that process.

The connection between the node objects, PDCP sub layer and RLC sub layer was rather straight forward since we only needed to hand over the data, both at the transmitting and the receiving end. Because the MAC sub layer only transmits data a predefined number of times each TTI and has to handle the multiplexing of logical channels, we ended up trying two approaches for the RLC/MAC interaction.

The first approach we used was to let the RLC ask the MAC for available space in a transport block being built at the MAC sub layer and build its PDU to fit that size. Even though it may seem like the most straightforward approach, a number of problems arose. Since we were queuing up RLC PDUs at the MAC sub layer as a work in progress transport block and it would not be done synchronized with the TTIs, we ran into problems when it was time to transmit the transport block. In the worst case, the RLC would be working on a PDU to put into the transport block when it was time to transmit it. Should we block and wait for it to get built and risk missing the TTI deadline for transmitting the transport block? The other option would be to leave the in progress RLC PDU behind, but then the RLC might make it unnecessarily small and waste resources. There is also the issue of when the RLC should decide that it's time to build a PDU, as there is no way of predicting how much more data is on the way. A simplified sequence diagram of the interactions in this approach can be found in Figure 67.

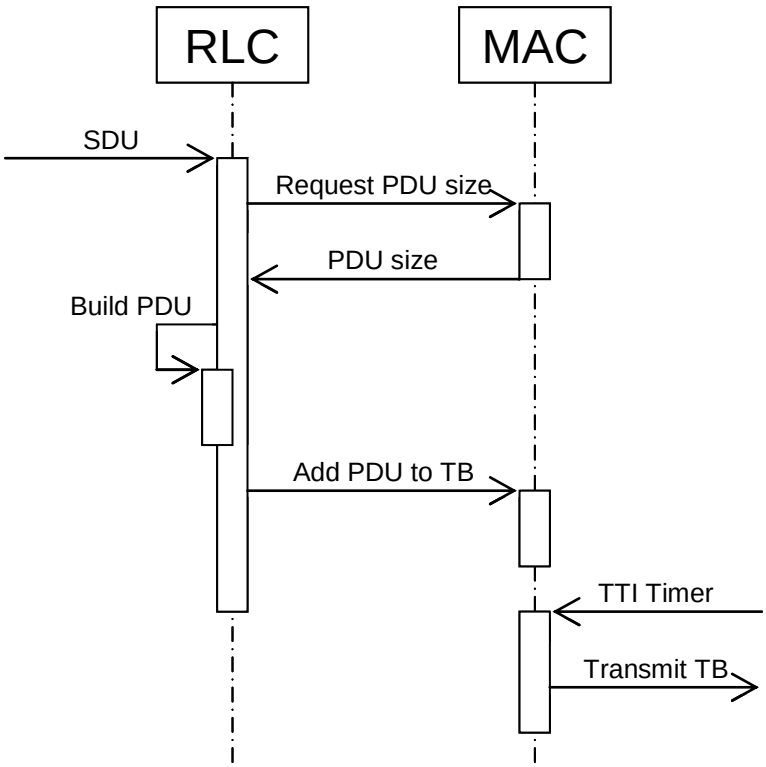

Figure 67. RLC initiated RLC/MAC interaction approach.

This RLC initiated approach was abandoned and the approach described in Chapter 4.9 was adopted instead. This approach entirely removed the need for the RLC to decide if it should build a PDU or wait for more SDUs. It also allows more flexibility for implementing priority handling between different logical channels, as all control of which channels to use and how much data from them is allowed, is handled at the MAC sub layer where the scheduling is done.

The prototype resulted in a system that can be used to simulate user plane traffic in uplink and downlink through the UE, the eNB and the aGW. The next step would be to add control plane functionality to the prototype. With the addition of Radio Bearer priority and bit rate control, the system will have fully realized multiplexing of different types of services. The system has support for many UEs and streams per UE and with the addition of a more sophisticated and true to specification scheduler it would be well suited for testing different scheduling scenarios and algorithms. For a more complete RLC, the Acknowledge Mode could be implemented by adding ARQ and resegmentation. If ARQ support is added, the HARQ/ARQ interaction can be an interesting study area.

A step further could be to add another eNB where the X2 interface between eNBs would be introduced, including control signalling to support the handover procedure. If simulating handover from the source to the target eNB, a GTP-U supporting tunnelling between eNBs is required.

During the development of the prototype there were different areas that caught our interest and we decided to study them further, those areas considered PDU structures, HARQ and scheduling. The specific areas of interest were chosen during the development of the prototype as we got more familiar with the functionality of LTE. The first two areas, scheduling and HARQ, are areas where the prototype only has limited functionality. During the design phase of the prototype, the PDU structures where still being discussed with several different suggestions on optimizations, but we opted to go for PDU structures with no optimizations for specific services. Studying these areas deeper could be useful for further

development of these parts in the prototype, as well as being interesting and educational in general.

7 References

[1] Rohde & Schwarz, *UMTS Long Term Evolution (LTE) Technology Introduction*, Rohde & Schwarz, March 2007. [Online]. Available: http://www.rohde-schwarz.com

[2] 3GPP, *Requirements for Evolved UTRA (E-UTRA) and Evolved UTRAN (E-UTRAN)*, 3GPP, TS 25.913 V.7.3.0, March 2006. [Online]. Available: http://www.3gpp.org

[3] 3GPP, *Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN)*, 3GPP, TS 36.300 V.1.0.0, March 2007. [Online]. Available: http://www.3gpp.org

[4] WiMAX Forum, WiMAX Home, WiMAX Forum, 2007. [Online]. Available: http://www.wimaxforum.org

[5] Prasad Ramjee, *OFDM for wireless Communications system*, Artech House Publishers, August 2004.

[6] Hyung G. Myung, Junsung Lim, and David J. Goodman, *Single Carrier FDMA for Uplink Wireless Transmission*, IEEE Vehicular Technology Magazine, vol. 1, no. 3, September 2006.

[7] Nokia, *PDCP Header Structure*, 3GPP TSG-RAN WG2, R2-070018, January 2007. [Online]. Available: http://www.3gpp.org

[8] LG Electronics Inc., *Discussion on RLC PDU Structure*, 3GPP TSG-RAN WG2, R2-070721, February 2007. [Online]. Available: http://www.3gpp.org

[9] NTT DoCoMo, NEC, *MAC PDU structure for LTE*, 3GPP TSG-RAN WG2, R2-071314, March 2007. [Online]. Available: http://www.3gpp.org

[10] 3GPP, *General Packet Radio Service (GPRS); GPRS Tunnelling Protocol (GTP) across the Gn and Gp interface*, 3GPP, TS 29.060 V.7.5.1, March 2007. [Online]. Available: http://www.3gpp.org

[11] 3GPP, *Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN)*, 3GPP, TS 36.300 V.8.0.1, June 2007. [Online]. Available: http://www.3gpp.org

[12] Nokia, *Uplink Scheduling for VoIP*, 3GPP TSG-RAN WG2, R2-070476, February 2007. [Online]. Available: http://www.3gpp.org

[13] Motorola, *Uplink VoIP scheduling*, 3GPP TSG-RAN WG2, R2-072002, May 2007. [Online]. Available: http://www.3gpp.org

[14] Motorola, *Uplink VoIP Performance*, 3GPP TSG-RAN WG2, R2-071483, March 2007. [Online]. Available: http://www.3gpp.org

[15] Research In Motion, *Uplink VoIP scheduling with Fast Indication*, 3GPP TSG-RAN WG2, R2-071961, May 2007. [Online]. Available: http://www.3gpp.org

[16] LG Electronics Inc., *Transition indicator for VoIP in UL*, 3GPP TSG-RAN WG2, R2-071449, March 2007. [Online]. Available: http://www.3gpp.org

[17] NTT DoCoMo, Fujitsu, Mitsubishi Electric, NEC, Sharp, Toshiba Corporation, *Hybrid ARQ Scheme for E-UTRA Downlink*, 3GPP TSG-RAN WG2, R2-060840, March 2006. [Online]. Available: http://www.3gpp.org

[18] Research In Motion Ltd, *Link adaptation overhead reduction for VoIP*, 3GPP TSG-RAN WG2, R2-072775, June 2007. [Online]. Available: http://www.3gpp.org

[19] James F. Kurose, Keith W. Ross, Computer *networking : a top-down approach (4th edition)*, Addison Wesley, March 2007.

[20] Ericsson, *Handling of HARQ retransmissions for LTE uplink*, 3GPP TSG-RAN WG2, R2-072580, June 2007. [Online]. Available: http://www.3gpp.org

[21] Nokia, *Synchronous adaptive HARQ for E-UTRAN UL*, 3GPP TSG-RAN WG2, R2-071744, May 2007. [Online]. Available: http://www.3gpp.org

[22] Qualcomm Europe, *On UL Scheduling for VoIP*, 3GPP TSG-RAN WG2, R2-071995, May 2007. [Online]. Available: http://www.3gpp.org

[23] Qualcomm Europe, *Impact of HARQ Termination Statistics on UL VoIP Capacity*, 3GPP TSG-RAN WG2, R2-071994, May 2007. [Online]. Available: http://www.3gpp.org

[24] Qualcomm Europe, *Number of HARQ processes*, 3GPP TSG-RAN WG2, R2-072671, June 2007. [Online]. Available: http://www.3gpp.org

[25] Ericsson, *HARQ operation in case of UL Power Limitation*, 3GPP TSG-RAN WG2, R2-072630, June 2007. [Online]. Available: http://www.3gpp.org

[26] Ericsson, *HARQ Configuration for LTE*, 3GPP TSG-RAN WG2, R2-071838, May 2007. [Online]. Available: http://www.3gpp.org

[27] Ericsson, *HARQ-ARQ Interactions for NACK to ACK error*, 3GPP TSG-RAN WG2, R2-072565, June 2007. [Online]. Available: http://www.3gpp.org

[28] Ericsson, *NDI-less HARQ operation*, 3GPP TSG-RAN WG2, R2-070577, February 2007. [Online]. Available: http://www.3gpp.org

[29] Samsung, *Simulation results on HARQ assisted ARQ scheme*, 3GPP TSG-RAN WG2, R2-062772, October 2006. [Online]. Available: http://www.3gpp.org

[30] InterDigital Communications Corporation, *On HARQ-ARQ interactions*, 3GPP TSG-RAN WG2, R2-072471, June 2007. [Online]. Available: http://www.3gpp.org

[31] Samsung, *Lite RLC vs. Normal RLC*, 3GPP TSG-RAN WG2, R2-072467, June 2007. [Online]. Available: http://www.3gpp.org

[32] Nokia, Nokia Siemens Networks, *PDU Optimisations for VoIP*, 3GPP TSG-RAN WG2, R2-072408, June 2007. [Online]. Available: http://www.3gpp.org

[33] Ericsson, *Length Indicator Optimization*, 3GPP TSG-RAN WG2, R2-072571, June 2007. [Online]. Available: http://www.3gpp.org

[34] Ericsson, *Byte alignment for user plane protocols in LTE*, 3GPP TSG-RAN WG2, R2-072558, June 2007. [Online]. Available: http://www.3gpp.org

[35] InterDigital Communications Corporation, *On RLC SN: PDU-based versus reusing PDCP SN?*, 3GPP TSG-RAN WG2, R2-072473, June 2007. [Online]. Available: http://www.3gpp.org

[36] Alcatel-Lucent, *SN removal in LTE data transmission*, 3GPP TSG-RAN WG2, R2-072692, June 2007. [Online]. Available: http://www.3gpp.org

8 Appendix A. Scenarios

Scenario 1: Uplink, one user, one stream

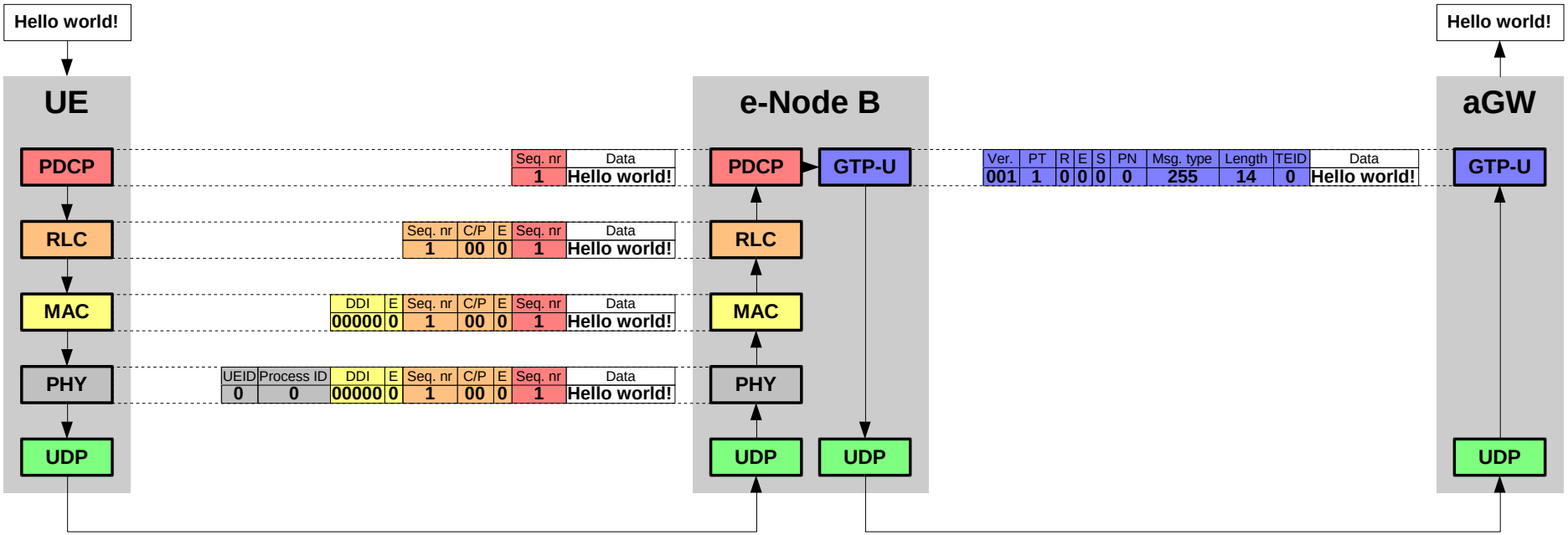

Figure 1, Sending data "Hello world!".

When data "Hello world!" arrives at the PDCP layer in UE it adds a header to the packet that consist of a sequence number. The sequence number increases with one for each packet that arrives at the PDCP layer. In the RLC layer another header is added to the packet that consist of a new sequence number, a C/P (Complete/Partial) field and a E (Extension) flag. The C/P field identifies whether the start of the PDU is the start of an SDU and whether the end of the PDU is the end of an SDU. Value 00: Start of the PDU is the start of the SDU and the end of the PDU is the end of the SDU. In other words the payload is a complete packet, no segmentation is used. E flag identifies whether next field is a LI (Length Indicator) or data.

When the packet arrives at the MAC layer a MAC header is added that uses a DDI field to identify the type of the corresponding information elements (i.e. logical channel type and identity of MAC SDU, MAC control block, or padding bits). The E flag indicates whether additional MAC header information elements (i.e. combination of LI, DDI and E) follows in the MAC header. Padding in the MAC header is used for byte-aligning the payload. The PHY layer adds a header with 2 fields. UEID identifies the UE and the Process ID identifies the sending HARQ process.

When the message is received at the eNB the PHY layer delivers the TB and control information to the MAC layer. At the MAC layer decoding of the TB is attempted. If decoding is successful an positive acknowledgment is sent to the corresponding UE and process. The payload is delivered to the channel indicated by the DDI field to the RLC layer. The RLC layer then remove the RLC header and uses the C/P field to see if the payload is a complete PDCP PDU or if it has to wait for more RLC PDU:s to reassemble the PDCP PDU. When it has a complete PDCP PDU the packet is delivered in sequence to the PDCP layer. Then the PDCP layer remove the PDCP header and deliver the SDU to the GTP-U layer that adds a GTP-U header before sending the packet to the aGW node. The TEID in the GTP-U header identifies the tunnel endpoints for the corresponding stream. In this scenario the TEID is set to 0 identifying the stream.

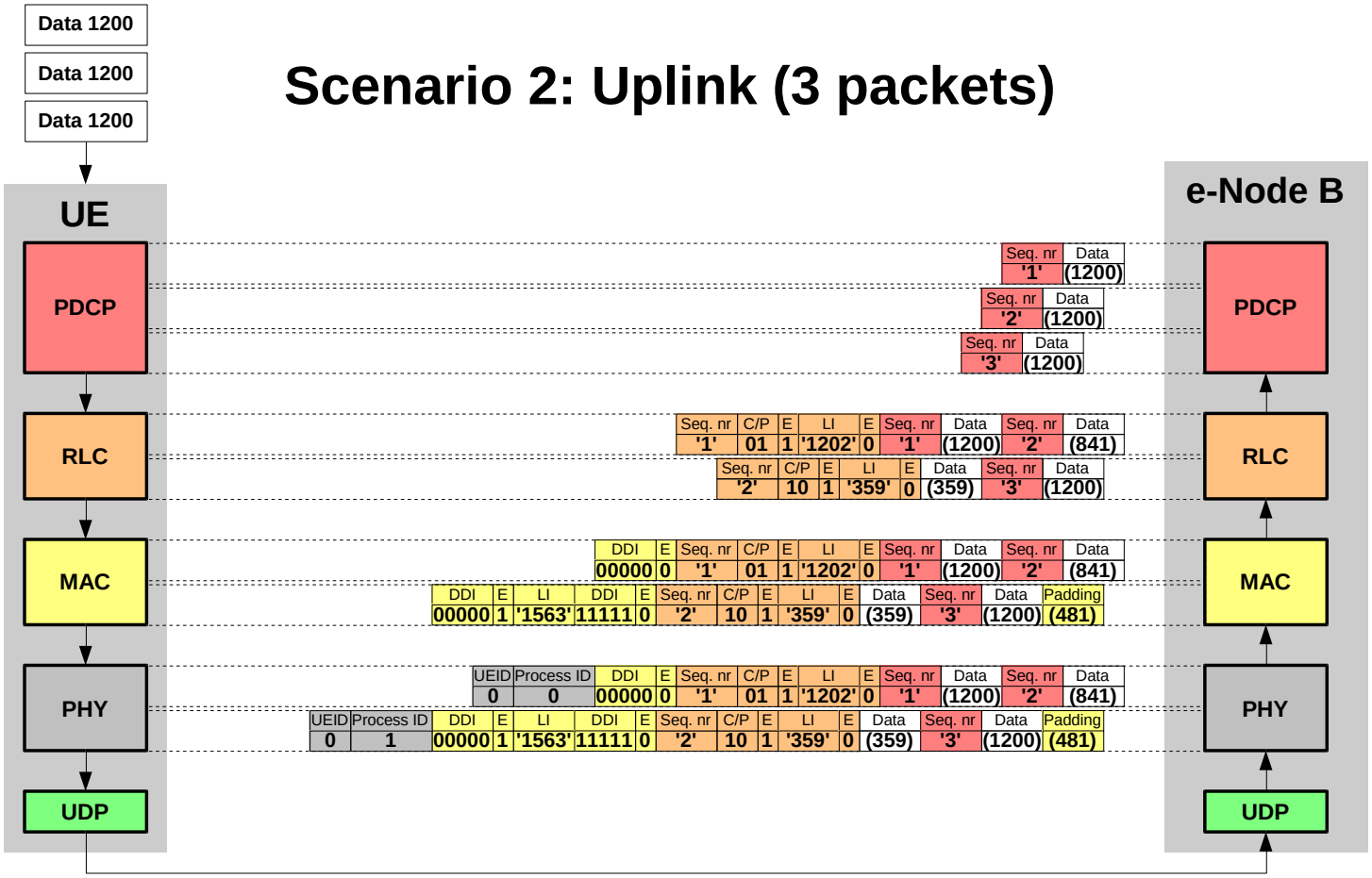

Figure 2, Sending three data packets.

In this example three packets, each of size 1200 bytes, are sent from the UE to the e-Node B. When the packets arrives at the PDCP layer each packet gets an incremented sequence number and are then forwarded to the RLC layer.

The RLC layer tries to build each RLC PDU as big as possible to reduce header data. Maximum size of each RLC PDU depends on the available size of the transport block. We have used 2048 bytes as the transport block size in this example. To build the largest possible RLC PDU, the RLC tries to fit more than one PDCP PDU in each RLC PDU and segments the data if it gets to big. In this example the entire first PDCP PDU fits and 843 bytes (Data+Seq. nr = 841+2 Bytes) of the second PDCP PDU in the first RLC PDU. We then set the C/P field (01) to indicate that the start of the RLC PDU is the start of an SDU and that the end of the PDU is not the end of an SDU. The second RLC PDU contains the remainder of the second PDCP PDU and also the entire third PDCP PDU. The C/P field (10) now indicate that the start of the PDU is not the start of an SDU and that the end of the PDU is the end of an SDU. For both RLC PDU:s we need a LI (Length Indicator) to know where the first SDU ends and where the last SDU starts.

When the first RLC PDU arrives at the MAC layer the RLC PDU fits perfectly in one MAC PDU and only the DDI, E and some padding bits are needed in the MAC header. When the second RLC PDU arrives at the MAC layer the RLC PDU is a bit smaller then the fixed size of the MAC PDU and padding bits are used at the end of the MAC PDU to fill the empty space. Then we need a LI in the MAC header to indicate where the RLC PDU ends and where the padding begins. The second DDI (11111) indicate that the second SDU is padding. The PHY header specifies the source UE and HARQ process used.

When the message is received at the eNB the PHY layer delivers the TB to the MAC layer. At the MAC layer decoding of the TB is attempted. If decoding is successful an positive acknowledgment is sent to the corresponding UE and process. The payload is delivered to the channel indicated by the DDI field to the RLC layer. The RLC layer then remove the RLC header and uses the C/P field to see if the payload is a complete PDCP PDU or if it has to wait for more RLC PDU:s to reassemble the PDCP PDU. When it has a complete PDCP PDU the packet is delivered in sequence to the PDCP layer.

Scenario 3: Uplink, one user, two streams

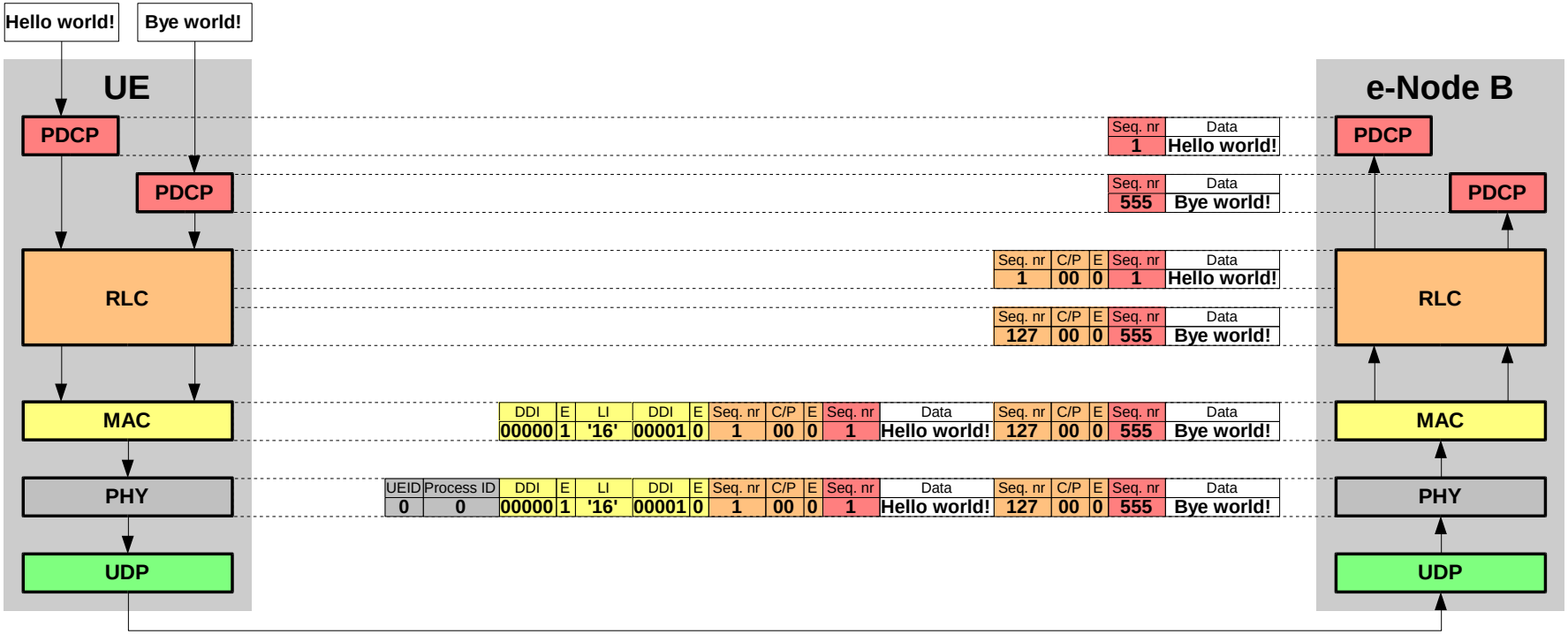

Figure 3, Sending two data streams "Hello world!" and "Bye world!".

In this example one user sends two data streams to e-Node B. Each stream arrives to its own PDCP entity and each PDCP entity is connected to one of the 16 logical channels (this example only have two logical channels) at the RLC layer. All channels are kept separated through the RLC layer and arrives separately to the MAC layer. At the MAC layer all streams are concatenated into one stream. The DDI field in the MAC-header tells which channel each packet belongs to.

When the packet arrives at the MAC layer in e-Node B, each SDU is delivered to the corresponding channel indicated by the DDI field to a RLC entity. There is one RLC entity for each user. The RLC layer then removes the RLC-header from the packet and deliver the PDCP PDU to the PDCP that is connected to the corresponding channel.

Scenario 4: Uplink, two users

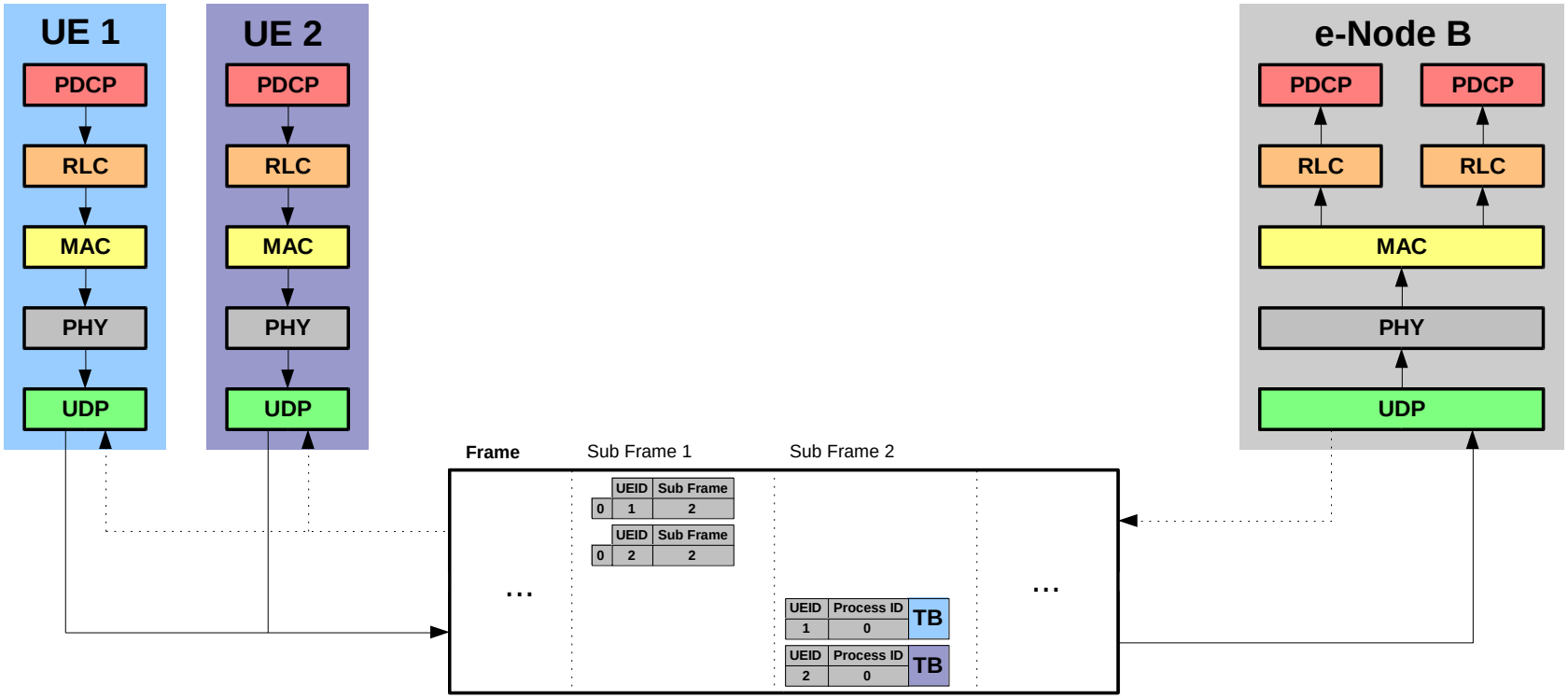

Figure 4, Sending data from two users.

This example shows two UE's sending data to the same eNB. The eNB schedules the transmission by sending grants to the UE's. The UE's sends data in the sub frames of the frame using the sub frame assigned to it by the grant message.

Here the eNB grants UE 1 and UE 2 in sub frame 1 to send in sub frame 2.

Scenario 5: Downlink, two users

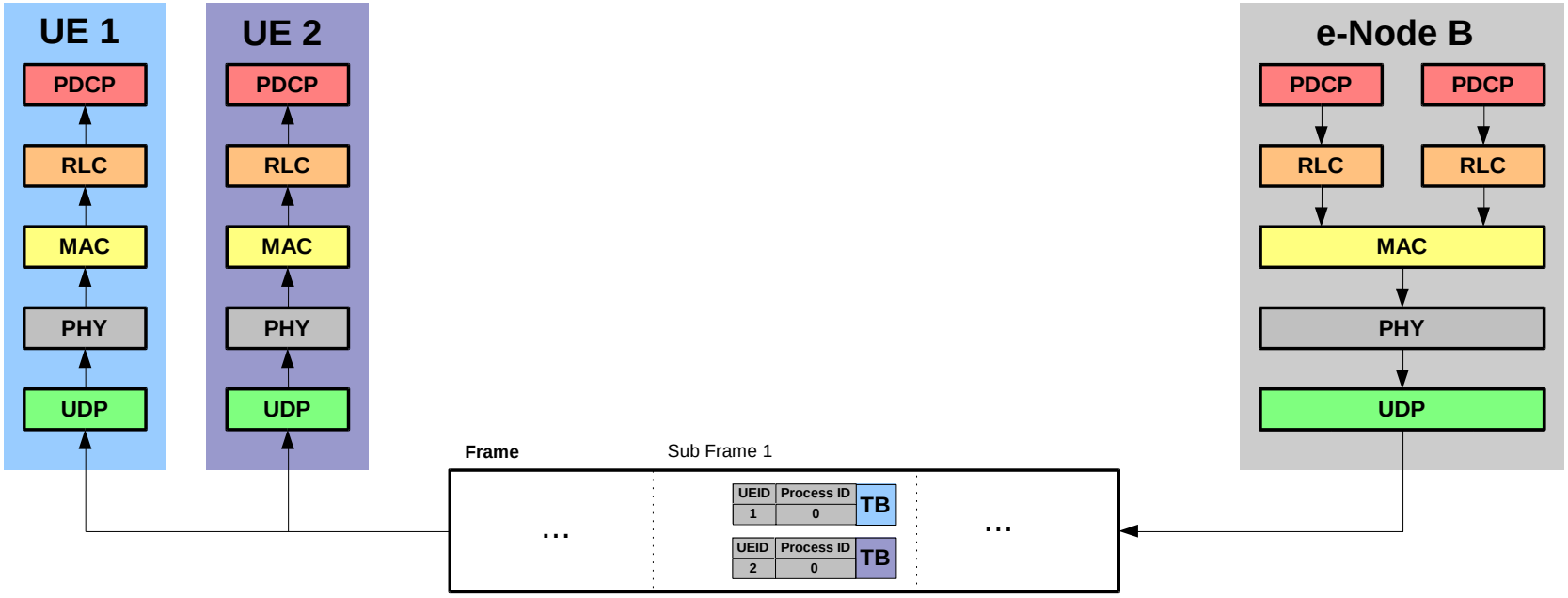

Figure 5, Sending data to two users.

This example shows two UE's receiving data from the same eNB. The eNB signals the scheduling by the UEID field telling the UE's who's data is in the TB of the same message.

Scenario 6: Downlink, one user, one stream

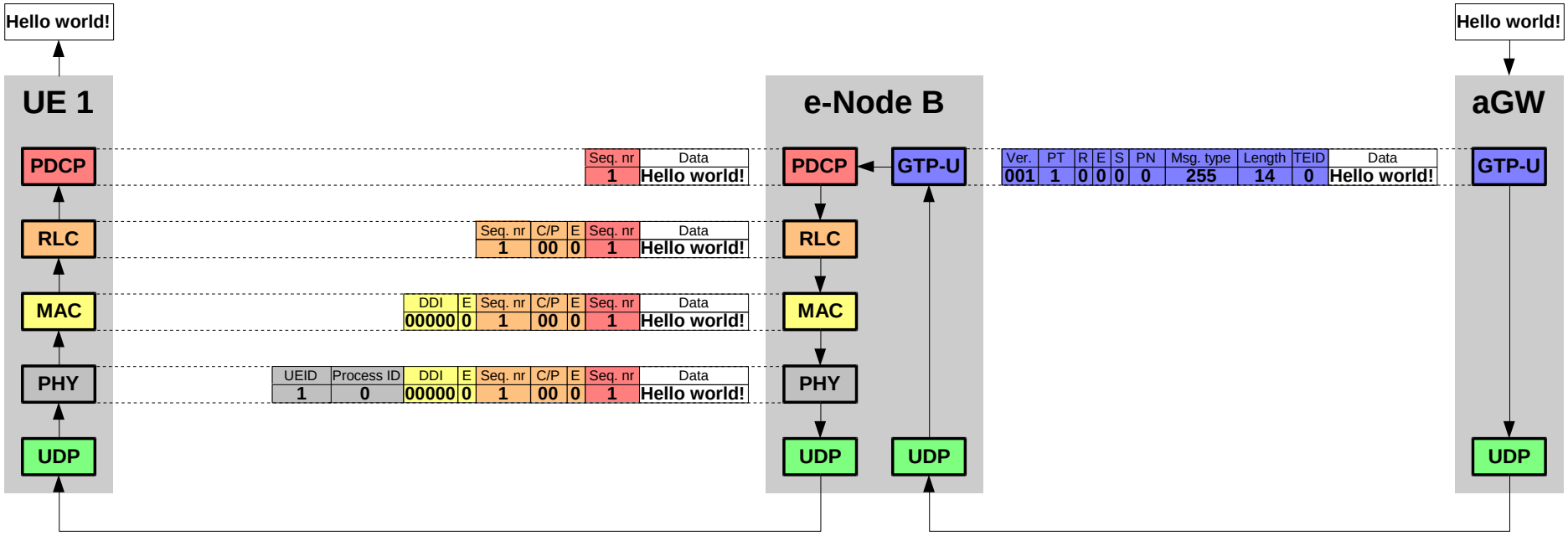

Figure 6, Receiving data "Hello world!".

In this example data "Hello world!" is sent from the aGW node to UE 1. When the data packet arrives at the GTP-U layer in the aGW, the GTP-U layer adds a GTP-U header to the packet before the UDP layer sends the packet to e-Node B. The TEID in the GTP-U header identifies the tunnel endpoints for the corresponding stream. In this scenario the TEID is set to 0 identifying the stream.

When the packet arrives at the GTP-U layer in e-Node B the GTP-U header is removed and the SDU is sent to the PDCP layer. At the PDCP layer a PDCP header is added to the packet that consist of a sequence number. The sequence number increases with one for each packet that arrives at the PDCP layer. In the RLC layer another header is added to the packet that consist of a new sequence number, a C/P (Complete/Partial) field and a E (Extension) flag. The C/P field identifies whether the start of the PDU is the start of an SDU and whether the end of the PDU is the end of an SDU. Value 00: Start of the PDU is the start of the SDU and the end of the PDU is the end of the SDU. In other words the payload is a complete packet, no segmentation is used. E flag identifies whether next field is a LI (E=1) or data (E=0). When the packet arrives at the MAC layer a MAC header is added that uses a DDI field to identify the type of the corresponding information elements (i.e. logical channel type and identity of MAC SDU, MAC control block, or padding bits). The E flag indicates whether additional MAC header information elements (i.e. combination of LI, DDI and E) follows in the MAC header.

The PHY layer adds a header with 2 fields. UEID identifies the UE and the Process ID identifies the sending HARQ process.

When the message is received at UE 1 the PHY layer checks if UEID belongs to this UE. The PHY layer then deliver the TB to the MAC layer. At the MAC layer decoding of the TB is attempted. If decoding is successful an positive acknowledgment is sent to the corresponding UE and process. The payload is delivered to the channel indicated by the DDI field to the RLC layer. The RLC layer then remove the RLC header and uses the C/P field to see if the payload is a complete PDCP PDU or if it has to wait for more RLC PDU:s to reassemble the PDCP PDU. When it has a complete PDCP PDU the packet is delivered in sequence to the PDCP layer. Then the PDCP layer remove the PDCP header and the data "Hello world!" has reached its destination.

Scenario 7: Downlink (3 packets)

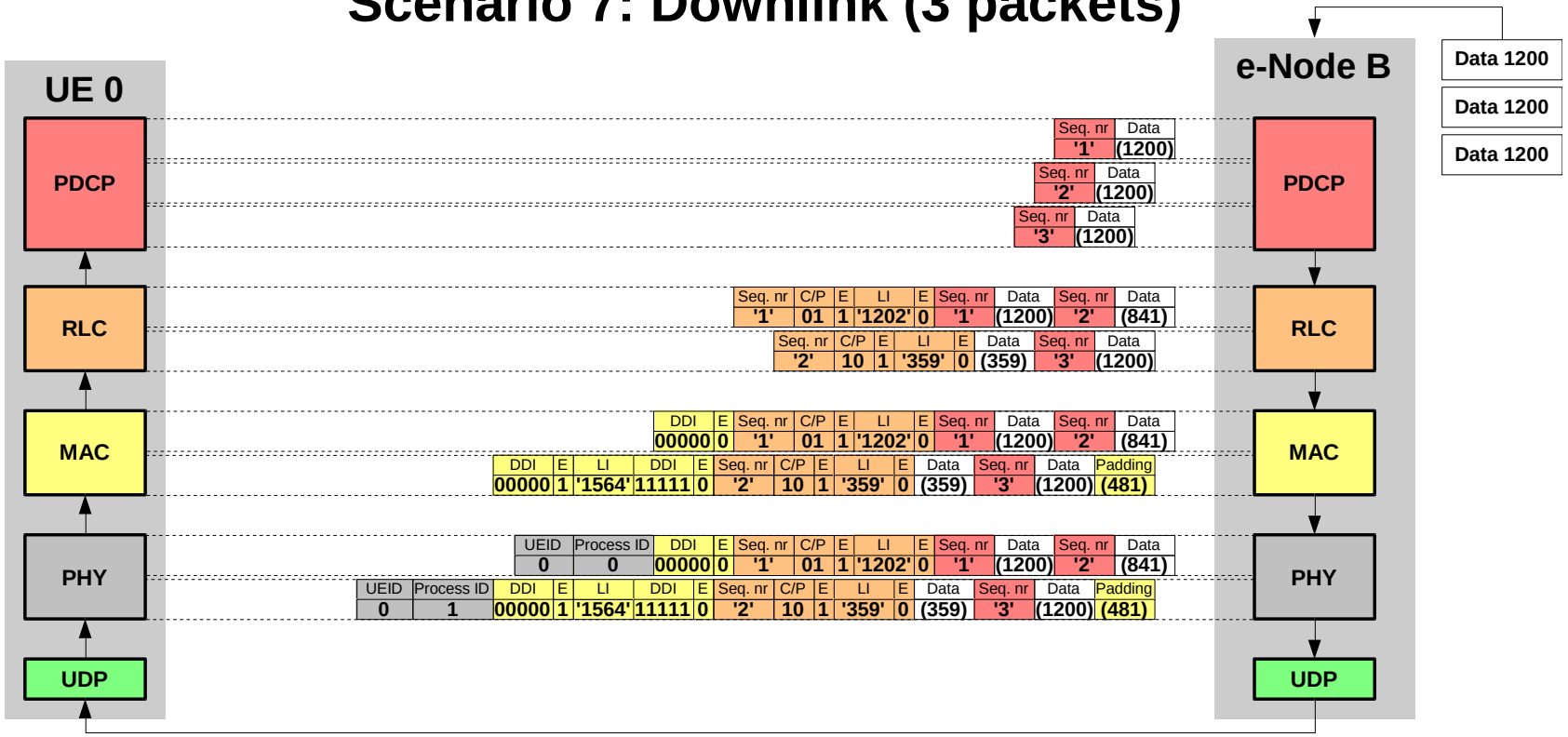

Figure 7, Receiving three data packets.

In this example three packets, each of size 1200 bytes, are sent from the eNB to the UE. When the packets arrives at the PDCP layer each packet gets an incremented sequence number and are then forwarded to the RLC layer.
The RLC layer tries to build each RLC PDU as big as possible to reduce header data. Maximum size of each RLC PDU depends on the size of the MAC header and the transport block. We have used 2048 bytes as the transport block size in this example. To build the largest possible RLC PDU, the RLC tries to fit more then one PDCP PDU in each RLC PDU and segments the data if it gets to big. In this example the entire first PDCP PDU fits and 843 bytes (Data+Seq. nr = 841+2 Bytes) of the second PDCP PDU in the first RLC PDU. We then set the C/P field (01) to indicate that the start of the RLC PDU is the start of an SDU and that the end of the PDU is not the end of an SDU. The second RLC PDU contains the remainder of the second PDCP PDU and also the entire third PDCP PDU. The C/P field (10) now indicate that the start of the PDU is not the start of an SDU and that the end of the PDU is the end of an SDU. For both RLC PDU:s we need a LI (Length Indicator) to know where the first SDU ends and where the last SDU starts. When the first RLC PDU arrives at the MAC layer the RLC PDU fits perfectly in one MAC PDU and only the DDI, E and some padding bits are needed in the MAC header. When the second RLC PDU arrives at the MAC layer the RLC PDU is a bit smaller then the fixed size of the MAC PDU and padding bits are used at the end of the MAC PDU to fill the empty spaces.

We then need a LI in the MAC header to indicate where the RLC PDU ends and where the padding begins. The second DDI (11111) indicate that the second SDU is padding. The PHY layer adds a header with 2 fields. UEID identifies the UE and the Process ID identifies the sending HARQ process.

When the message is received at UE 0 the PHY layer checks if UEID belongs to this UE. The PHY layer then deliver the TB to the MAC layer. At the MAC layer decoding of the TB is attempted. If decoding is successful an positive acknowledgment is sent to the corresponding UE and process. The payload is delivered to the channel indicated by the DDI field to the RLC layer. The RLC layer then remove the RLC header and uses the C/P field to see if the payload is a complete PDCP PDU or if it has to wait for more RLC PDU:s to reassemble the PDCP PDU. When it has a complete PDCP PDU the packet is delivered in sequence to the PDCP layer. Then the PDCP layer remove the PDCP header and the data has reached its destination.

Scenario 8: Downlink, one user, two streams

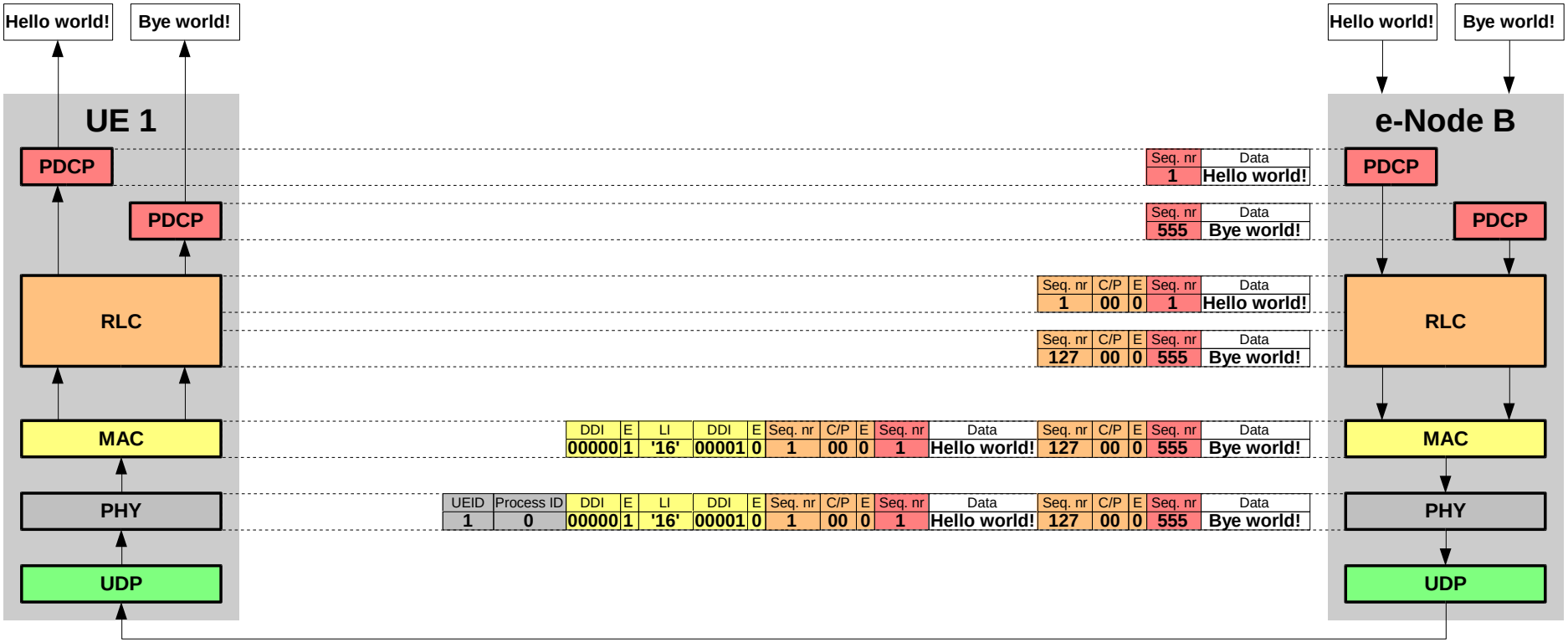

Figure 8, Receiving two data streams "Hello world!" and "Bye world!".

In this example one user receives two data streams from e-Node B. Each stream arrives to its own PDCP entity and each PDCP entity is connected to one of 16 logical channels (this example only have two logical channels) at the RLC layer. All channels are kept separated through the RLC layer and arrives separately to the MAC layer. At the MAC layer all streams are concatenated into one stream. The DDI field in the MAC-header tells which channel each packet belongs to.

When the packet arrives at the MAC layer in UE 1, each SDU is delivered to the corresponding channel indicated by the DDI field to a RLC entity. There is one RLC entity for each user. The RLC layer then removes the RLC-header from the packet and deliver the PDCP PDU to the PDCP that is connected to the corresponding channel.