GRANT **AGREEMENT NO.: 732174**

Call: H2020-ICT-2016-2017
Topic: ICT-13-2016
Type of action: RIA

Orchestration and Reconfiguration Control Architecture

D3.1: First operational real-time SDR platforms

Revision: v.1.0

Work package	WP3
Task	T3.1, T3.2, T3.3
Due date	31/12/2017
Submission date	31/12/2017
Deliverable lead	NI
Version	1.0
Authors	Clemens Felber (NI), Vincent Kotzsch (NI), Wei Liu (IMEC), Xianjun Jiao (IMEC), Jan Bauwen (IMEC), Peter Ruckebusch (IMEC), Bart Jooris (IMEC), Ingrid Moerman (IMEC), Martin Danneberg (TUD), Roberto Bomfin (TUD), Andrea Guevara (KUL), Ana-Belen Martinez (TUD), Peter Neuhaus (TUD), Luiz Da Silva (TCD)

Reviewers	Sofie Pollin (KUL), Seyed Ali Hassani (KUL), Francisco Paisana (TCD)

Abstract	This deliverable will include the Year 1 implementation results on the available SDR platforms and the risk analysis whether all scenarios defined in WP2 are possible for real-time prototyping in the testbed facilities. This deliverable will also include a plan for functionality to be implemented in the upcoming year.
Keywords	real-time, SDR, data plane, PHY, MAC, higher layer, testbeds

Project co-funded by the European Commission in the H2020 Programme		
Nature of the deliverable:	R*	
Dissemination Level		
PU	Public, fully open, e.g. web	✓
CI	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to ORCA project and Commission Services	

** R: Document, report (excluding the periodic and final reports)*

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

OTHER: Software, technical diagram, etc

EXECUTIVE SUMMARY

ORCA wants to accelerate flexible end-to-end network experimentation by making open and modular software and hardware architectures available that smartly use novel versatile radio technology. More-specifically real-time Software Defined Radio platforms meeting the requirements in terms of runtime latencies, throughput, and fast reconfiguration and reprogramming are needed to fulfil this overall objective.

In this deliverable the focus is on essential data plane SDR functionality to achieve low latency and high throughput operation by allowing real-time operation. Therefore, ORCA extends the current state-of-the-art SDR capabilities by designing data-plane SDR solutions on heterogeneous hardware platforms that can combine high data rates or low latencies with fast design cycles and high versatility.

This deliverable will include the ORCA Year 1 implementation results on the available SDR platforms and the risk analysis whether all scenarios defined in WP2 are possible for real-time prototyping in the testbed facilities. This deliverable will also include a plan for functionality to be implemented in the upcoming year.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ACC	Accelerator
ADC	Analog Digital Converter
API	Application Programming Interface
ARM	Advanced RISC Machine
AP	Access Point
BS	Base Station
CE	Clock Enable
CDF	Cumulative Density Function
CFO	Carrier Frequency Offset
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DAC	Digital Analog Converter
DDR	Double data rate
DL	Downlink
DSSS	Direct-sequence spread spectrum
EBD	Electrical Balance Duplexer
EVM	Error Vector Magnitude
FCS	Frame Control Sequence
FD	Full Duplex
FFT	Fast Fourier Transform
FPGA	Field-programmable gate array
GUI	Graphical user interface
GFDM	Generalized Frequency Division Multiplexing
HDL	Hardware description language
HAL	Hardware Abstraction Layer
HALO	hardware in the loop
ICC	Inter-cluster combining
IoT	Internet of Things
IP	Internet Protocol
IQ	In-phase and Quadrature
LBT	Listen before Talk
LoS	Line of sight
LS	Least Square
MAC	Medium access control

MaMIMO	Massive MIMO
MIMO	Multiple-input multiple-output
MF	Matched Filter
ML	Machine Learning
NLoS	Non-Line of sight
NR	New Radio
OFDM	Orthogonal frequency-division multiplexing
OSI	Open Systems Interconnection
PL	Programmable Logic
PS	Particle Swarm
PS	Processing System
RAM	Random Access Memory
RAT	Radio Access Technology
ReLU	Rectified linear unit
RF	Radio Frequency
RMSE	Root Mean Squared Error
ROM	Read-only memory
Rx	Receiver
SIMD	Single instruction multiple data
SPI	Serial Peripheral Interface Bus
SQNR	Signal-to-quantization-noise ratio
SVS	Singular Value Spread
TAISC	Time Annotated Instruction Set Computer
TCP	Transmission Control Protocol
TDD	Time Division Duplexing
TDMA	Time-division multiple access
TS	Traditional Hierarchal
TSCH	Time-Slotted Channel Hopping
TTI	Transmission Time Interval
Tx	Transmitter
UART	Universal asynchronous receiver/transmitter
UE	User Equipment
UD	User Device
USRP	Universal Software Radio Peripherals
UL	Uplink
UPI	Unified Programming Interfaces

URLL	Ultra-Reliable Low-Latency Communications
VPN	Virtual Private Network
ZF	Zero Forcing

1 INTRODUCTION

ORCA wants to accelerate flexible end-to-end network experimentation by making open and modular software and hardware architectures available that smartly use novel versatile radio technology. More-specifically real-time Software Defined Radio (SDR) platforms meeting the requirements in terms of runtime latencies, throughput, and fast reconfiguration and reprogramming are needed to fulfil this overall objective. Further ORCA will offer experimental facilities, with SDR devices incorporating relevant software and hardware building blocks that allow easy design, implementation and programming, while also achieving low runtime delay allowing end-to-end networking experimentation.

In this deliverable the focus is on essential data plane SDR functionality to achieve low latency and high throughput operation by allowing real-time operation. Therefore, ORCA extends the current state-of-the-art SDR capabilities by designing data-plane SDR solutions on heterogeneous hardware platforms that can combine high data rates or low latencies with fast design cycles and high versatility. Flexible end-to-end reconfiguration facilities are described in detail in D4.1 [1].

This deliverable will include the ORCA Year 1 implementation results on the available SDR platforms and the risk analysis whether all scenarios defined in WP2 are possible for real-time prototyping in the testbed facilities. This deliverable will also include a plan for functionality to be implemented in the upcoming year.

1.1 Organization of the Deliverable

The remaining sections of the document are organized as follows:

- Section 2 summarizes the physical layer capabilities of the high throughput mmWave system and describes TUD/NI improvements on the hardware in the loop (HALO) setup and on the higher layer data plane
- Section 3 describes KUL' achievements on analog self-interference cancellation and the plan to employ full duplex and advanced collision detectors
- Section 4 summarizes KUL physical layer improvements and experiments on distributed Massive MIMO
- Section 5 describes TUD's improvements on Generalized Frequency Division Multiplexing (GFDM) physical layer which ensures full flexibility and runtime configuration
- Section 6 is dedicated to IMEC's improvements with focus on "PHY accelerator pool" and its control interfaces
- Section 7 describes TCD's signal classification framework, the general PHY functionality, its integration with upper layers
- Section 8 summarizes NI's improvements on the ns-3 based prototyping platform for RAT interworking that enables experimentation with LTE, Wi-Fi and 5G

Each section is then subdivided into: (i) summary of the implementation results obtained during Year 1 and these results are integrated into the testbeds, (ii) how the implemented solution is integrated in its respective Year 1 ORCA showcase, (iii) a risk analysis whether all scenarios defined in WP2 are possible for real-time prototyping in the testbed facilities (iv) and the extension plans currently envisioned by the partner for Year 2.

2 HIGH THROUGHPUT MMWAVE SYSTEM

The high throughput mmWave system as an outcome of the MiWaveS EU project [2] represents the baseline for further development and testbed integration within ORCA. In the following a short introduction is given to provide the reader some background based on information from MiWaveS Deliverable D6.5 [3].

Figure 1 shows the setup of the V-band access point and user device mmWave node, respectively. It comprises an integrated radio transceiver with an on-chip phased-array antenna. Note that access point (AP) and user device (UD) use identical radio transceiver and antenna arrays.

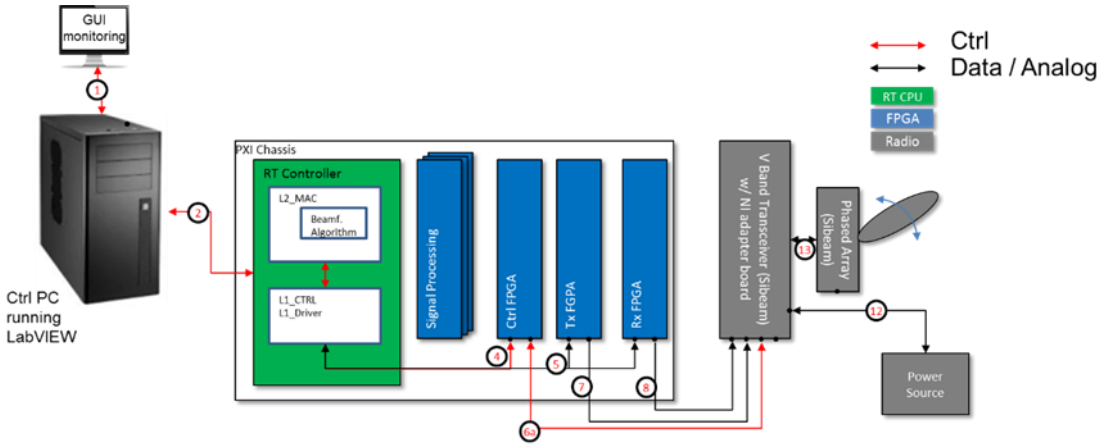

Figure 1: mmWave architecture for V-band access point and user device. [3]

The software implementation of each node follows a layered architecture. Figure 2 compares the OSI layered architecture to the layered architecture chosen for this implementation.

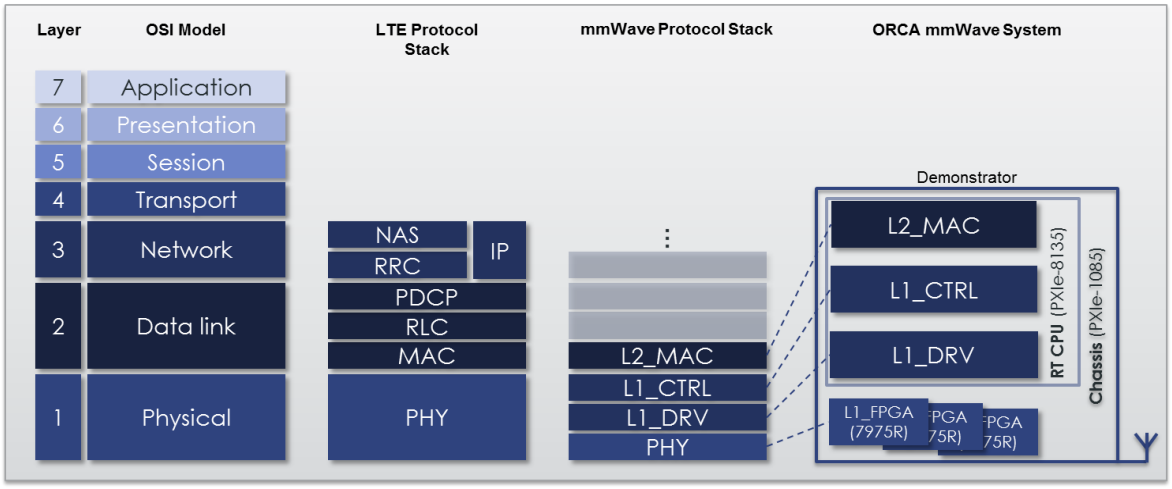

Figure 2: mmWave protocol stack in respect to LTE and OSI.

A brief overview, referring to the right-hand side diagram plot is given below

- Layer 1
  - Layer 1 FPGA: implements physical layer real time inner and outer transceiver signal processing, and comprises the FPGA based radio control interface.
  - Layer 1 driver: provides data, control, and monitoring interfaces between FPGAs and real-time controller
  - Layer 1 control: is responsible for channel measurements and for real time configuration of the physical layer and the radio based on parameters computed in the layer 2 MAC.

- Layer 2:
    - MAC layer: Implements the actual protocol including generation and evaluation of control and configuration messages, the beam steering algorithm and scheduling.

Furthermore, a channel characterization environment via hardware in the loop (HALO) using real-time radio control will be made available within ORCA and is described in more detail in the subsequent subsections.

2.1 Implementation results

The year 1 implementation results are described in the following subsections for starting each with an overview and highlighting the improvements as implementation results.

2.1.1 PHY improvements

2.1.1.1 Hardware in the loop (HALO) setup

The TUD hardware-in-the-loop (HALO) setup shown in Figure 3 consists of two NI PXI units and one PC with LabVIEW. One PXI unit represents the transmitter and the other the receiver, and the PC is responsible for the deployment of the code and control of the PXI units. Each PXI unit has many functional elements depending whether it is the transmitter or receiver. The system synchronization is performed by trigger signals generated by the transmitter PXI unit, which works as the master.

The transmitter unit has three main functional blocks. The first is the TX HOST CTRL, which provides the main coordination and control of the measurement. The second is the DAC MODULE, which serves as the master unit generating trigger signals for synchronization and transmitting the test signal. And the last one is the RF CONTROL, which makes the interface of the RF transceiver through the digital I/O module NI-6583. The main components of the receiver unit are similar to the transmitter one, however it has the ADC MODULE instead of DAC MODULE, which is responsible for receiving the trigger signals from the transmitter unit and distributing it to other modules.

The transmitter and receiver antennas have 25 possible beams each, so the idea is for the transmitter to select one beam and sweeps the 25 receiver beams where the received power is calculated for each beam combination, then the transmitter select another beam and the process is repeated. In the end, there are 25x25 = 625 beam combinations that is possible to map.

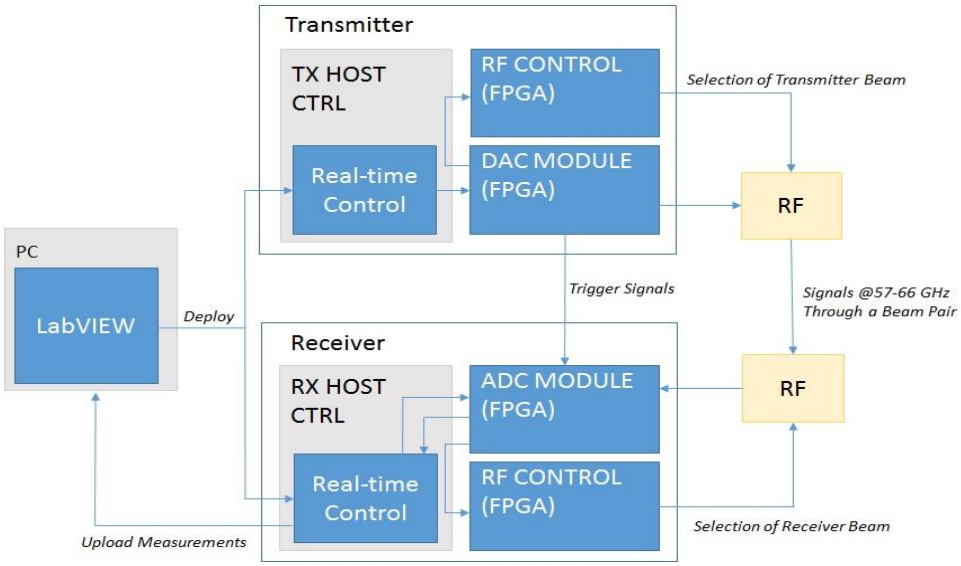

Figure 3: Diagram of HALO setup.

Improvements:

One of the advantages of the HALO setup is the possibility of providing data such that it can be further processed offline. To this end, TUD included the feature of saving the channel characterization and then upload it to the PC. The data is saved in the TDMS [4] format, that can be further saved as a xlsx excel file, which is more convenient for reading the data. Figure 4 shows the data imported by Matlab and plotted in a 3-dimentional graph. The User Device (UD) Beam index is related to the beam selection at the receiver antenna, while the Access Point (AP) Beam index indicates the beam selection of transmitter. In this channel characterization, we observe that the transmitter and receiver were aligned, thus we observe that the region in the center of the graph is where the channel gain is more concentrated.

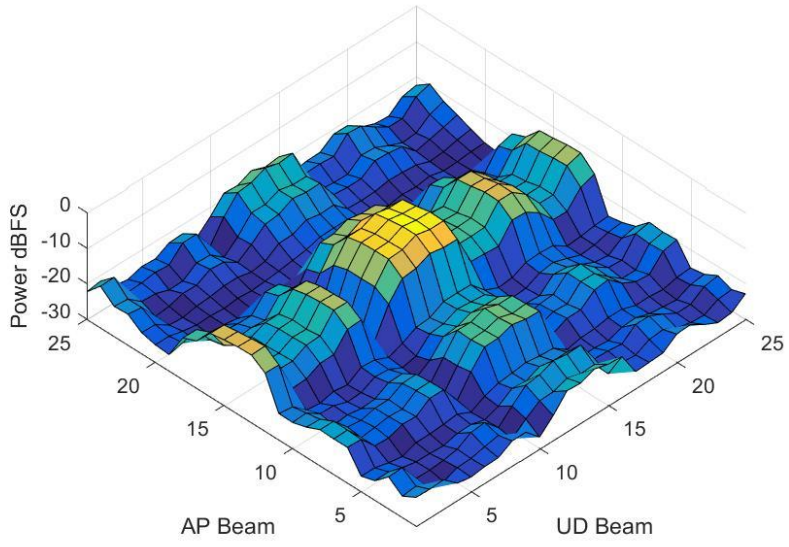

Figure 4: Channel characterization in Matlab.

2.1.1.2 Physical Layer overview

This section provides background information of the NI physical layer running on bidirectional, closed loop mmWave system using V-Band directive antennas suitable for small cell access see Figure 1. It's an outcome of the MiWaveS EU project [2] and represents the baseline for further development within ORCA. After establishment of the HALO setup, this setup will become available in a second step, see section Implementation plan 2.4.

The physical layer implements real time inner and outer transceiver signal processing distributed over multiple FPGAs. Figure 5 summarizes the main functionality.

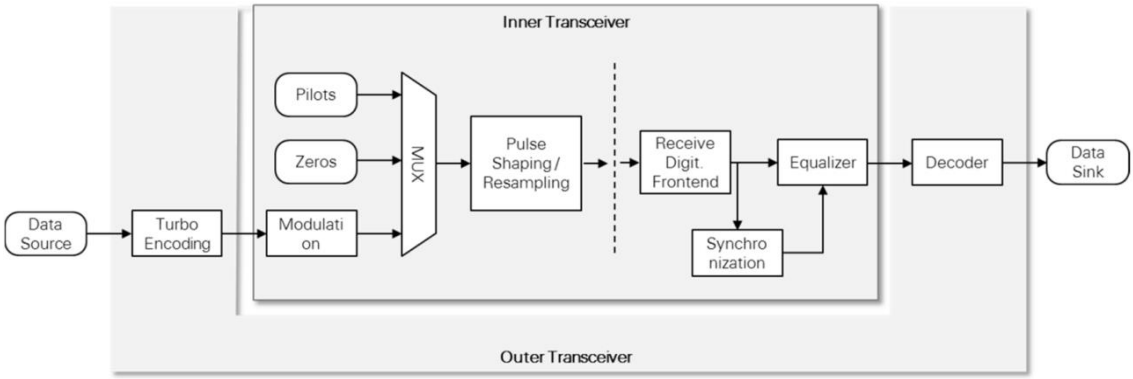

Figure 5: Structuring the physical layer processing into inner and outer transceiver. [3]

The mapping of signal processing functionality to FPGAs is shown in Figure 6. Massive parallelism has

been employed at many parts of the implementation. The turbo decoding procedure, for instance, is implemented using 12 parallel decoder cores distributed over two FPGA modules. The inner receiver signal processing processes 8 samples per FPGA clock cycle in parallel to support the wide bandwidth. Functionality such as filters or FFT are purpose-built to support this parallelism.

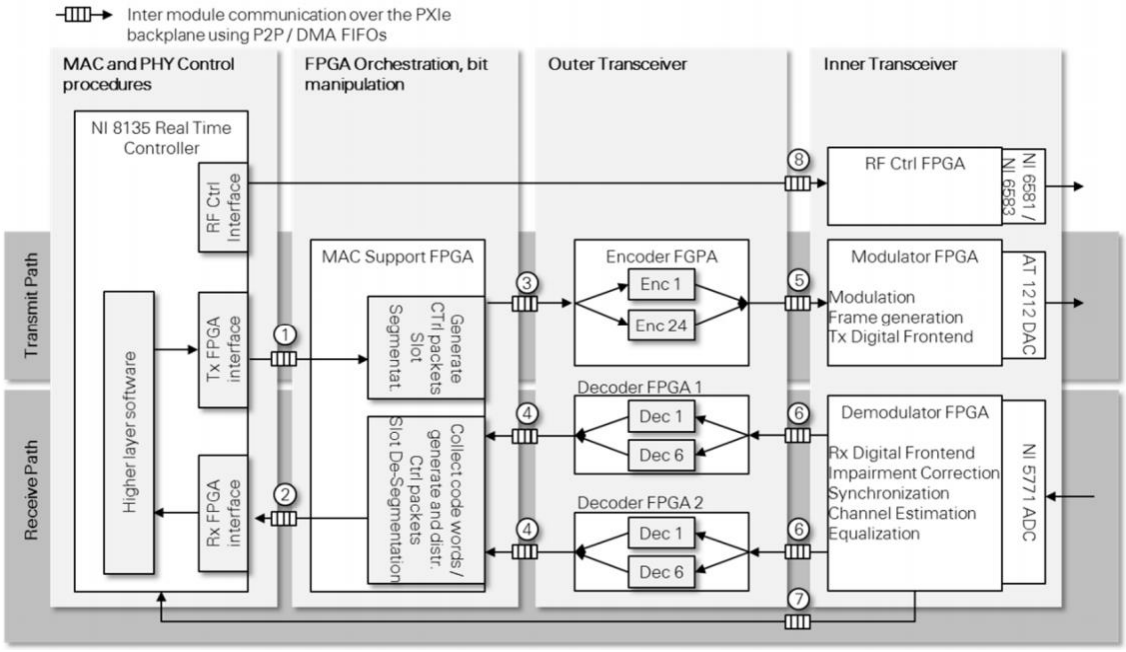

Figure 6: Mapping of Algorithms to base band hardware modules. [3]

A more detailed description can be found in MiWaveS D6.5 [3].

2.1.2 Higher layer protocol integration

As an introduction this section outlines the NI physical layer control and medium access layer software parts which are executed on a real-time controller. The communication with the FPGA based physical layer is handled through FIFOs and registers. Figure 7 shows the layered software architecture implemented on the real-time controller.

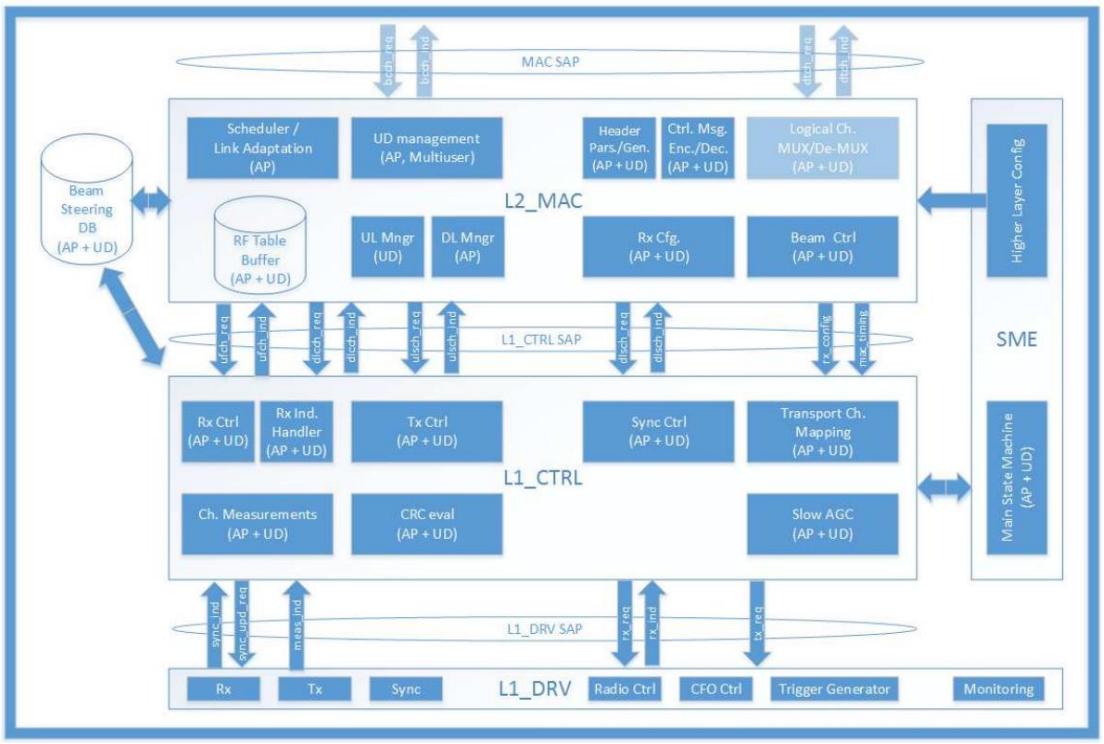

Figure 7: Physical layer driver and control layer, and MAC software architecture. [3]

The most important functions implemented on the real-time controller are: Management of system states, link setup, beam steering, multi-user scheduling, computation of radio control parameters. A more detailed description can be found in MiWaveS D6.5 [3].

Improvements:

To integrate data plane SDR functionality into the ORCA testbeds and demonstrate the capabilities through implementation of showcase 1, the mmWave system was extended by NI with a data path in order to support end-to-end data transmission. The User Datagram Protocol (UDP) was chosen to minimise protocol overhead and further to enable also unidirectional data streaming. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for packets delayed due to retransmission, which may not be an option in a real-time system [5]. Further the mmWave protocol stack was extended by basic RLC functionality for supporting sequence numbering to detect packet losses. The extended mmWave protocol stack is shown in Figure 8.

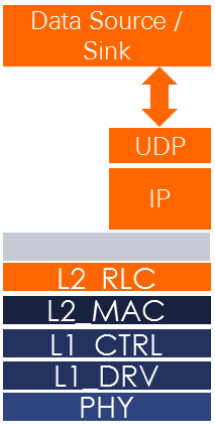

Figure 8: extended mmWave protocol stack.

Key features of this implementation are

- Bi-directional data streaming
- Dynamic packet size (up to 1500 bytes = maximum MTU size)
- 1 UDP packet per slot (1 slot = 100us)
- Status output and logging

With this feature set a complete end-to-end solution for data transmission over a 60GHz (V-Band) mmWave link was achieved, see Figure 9. As a possible first application within the testbeds over-the-air (OTA) video transmission was tested and verified.

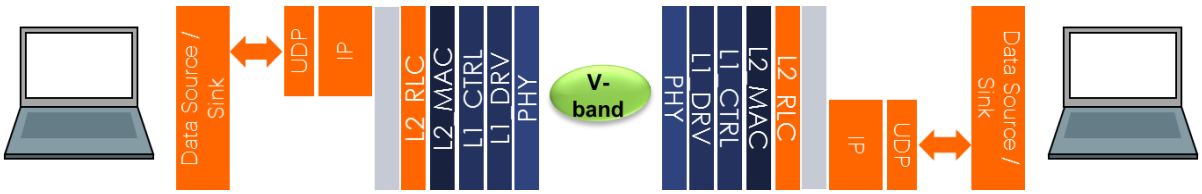

Figure 9: End-to-end data transmission over mmWave link.

To drive more advanced end-to-end applications the goal was to further extend the mmWave demonstrator in order to enable an internet connection using the provided 60 GHz mmWave link as shown in Figure 10.

Figure 10: Internet connection using 60 GHz mmWave link.

Since the UDP interface was made available as described above, the mmWave system was extended by integrating an VPN connection on top of UDP using OpenVPN [6]. With this all higher layer internet traffic based on TCP/IP could be tunnelled over a single UDP port and forwarded to the UDP interface offered by the mmWave system, see Figure 11. Likewise, the encapsulated IP packets are extracted from UDP packets received from the mmWave system and forwarded to their correct destination. All in all, this provides a very flexible solution to offer development and evaluation of new applications for the mmWave PHY as described in ORCA deliverable D2.2 [7].

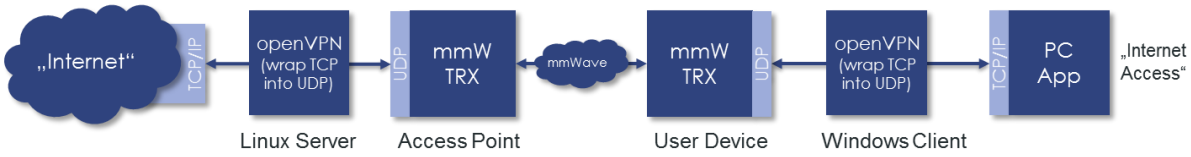

Figure 11: Application example for internet connection over mmWave link using OpenVPN.

2.1.3 Testbed integration

The described mmWave system will become available at the TUD macro scale testbed

1. For the first open call for experimentation starting in ORCA Year 2 the focus is on channel characterization environment via HALO (Hardware in the loop) with real time radio control, allowing to capture datasets for offline evaluation.
2. In a second step the bidirectional, closed loop mmWave system will be established

These features will be integrated to the testbed allowing access to external users.

2.2 Relation to showcase

As described in deliverable D2.1 [8], the showcase for the high throughput mmWave system is showcase 1: “High throughput - mmWave”.

2.3 Risk analysis

There is no specific risk observed at this phase of the work package for the data path implementation based on UDP and OpenVPN. On the other hand, this showcase makes use of specialized hardware such as V-Band antennas integrated into a complicated setup consisting of control PC, real-time Linux controller and 6 FPGAs. Thus, it requires a lot of testing to support full testbed integration.

2.4 Implementation plan

This section describes the plan for functionality to be implemented in the upcoming year, focusing on

- mmWave operation - use real time steerable antennas
- bi-directional TDD protocol enabling beam tracking
- 28 GHz RF front end for USRP

2.4.1 PHY improvements

Regarding physical layer improvements, investigations and improvements on phase noise compensation is planned by NI. Since currently the mmWave V-Band transceiver supports only codebooks in azimuth direction, NI plan to extend the antenna control interface to two-dimensional codebooks and characterize these codebooks.

In addition, TUD plan to implement a real-time 28 GHz mmWave setup for USRP, which allows the use of flexible waveforms, e.g., GFDM.

2.4.2 Higher layer protocol integration

For the ORCA Year 2 the support of automatic and manual beam steering functionality at MAC layer is planned, see offer in D2.2 [7]. For the automatic beam steering functionality, the real-time mmWave baseband system (Figure 1) supporting bi-directional TDD MAC protocol will be used and it is planned to integrate at least two different beam steering algorithms. The main task of the beam steering algorithm is to search the best beam pair in respect to channel quality.

Possible algorithms are

- Exhaustive Search (without beam tracking functionality)
- Parameterized Exhaustive Search (with beam tracking functionality)

For the manual beam steering functionality, the MAC layer needs to be modified in order to support manual asynchronous beam switching by selecting a Beam ID from the LabVIEW front panel (GUI). Further it might be considered to use a separate control application (PHY control) on top of the physical layer, which will be used for the manual beamsteering purposes.

With those features a runtime reconfiguration of beams and beam steering algorithms will be possible. This is described in more detail in D4.1 [1]. If needed the MAC scheduler will be extended in order to support more advanced radio slicing.

3 SDRS WITH IN-BAND FULL DUPLEX CAPABILITIES AND COLLISION DETECTION

3.1 Implementation results

To provide a Full-Duplex capable SDR, we integrated an Electrical Balance Duplexer (EBD) able to achieve analog self-interference cancellation up to 70dB to the NI USRP 2952R SDR platforms. The EBD has to be tuned in order to provide sufficient Tx-Rx isolation. The tuning procedure is implemented on a MicroBlaze softcore which is deployed on the Xilinx Kintex 7 FPGA. The MicroBlaze facilitates implementation of iterative algorithms such as a Particle Swarm (PS) optimizer which finds the optimal settings to minimize the residual self-interference. Cancelling the self-interference allows simultaneous data transmission and reception on the same channel. This bi-directional capability of IBFD doubles the throughput of the physical layer. In addition, the IBFD enhances the reliability of the network as it enables the wireless node to listen to the channel during transmission. Hence, the node can perform collision detection in transmission time and abort in case of interference. The collision detector is fully implemented in FPGA to gain the reliability. We also established another softcore to run CSMA/CD MAC protocol to employ the collision detector.

3.1.1 PHY-MAC implementation

As shown in Figure 12, the Full-Duplex (FD) capable node in ORCA testbed is equipped with an Electrical Balance Duplexer (EBD) providing 50-70 dB Tx-Rx isolation.

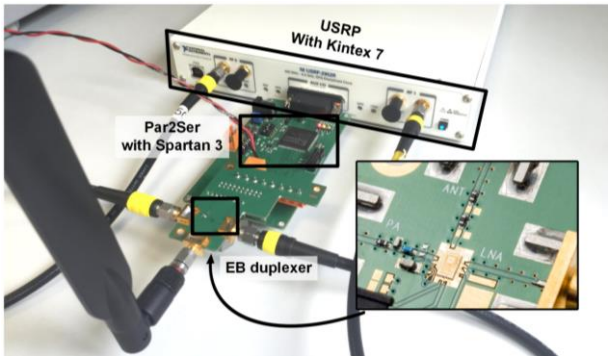

Figure 12: Full-Duplex capable SDR.

The balance network in the duplexer has four 8-bit tuneable capacitors illustrated in Figure 12, creating a four-dimensional optimization space with over that 4 billion settings.

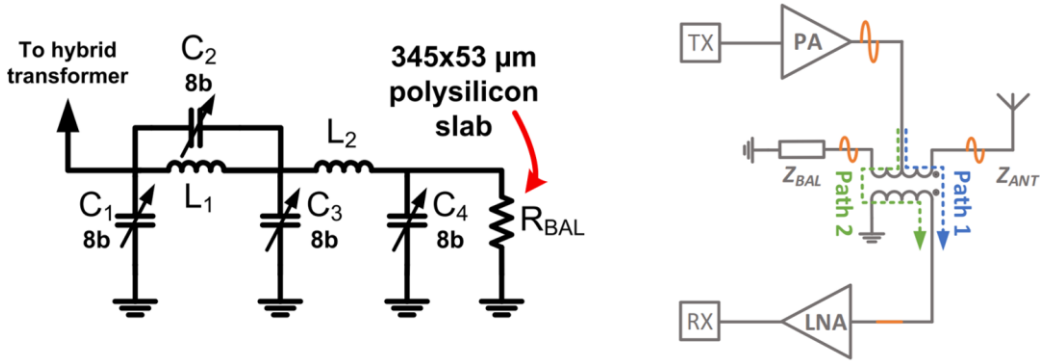

Figure 13: Electrical Balance Duplexer (right), Balance impedance (left).

The FD PHY uses a MicroBlaze softcore running at 150MHz deployed on the FPGA to tune the EBD. It runs an optimization algorithm which controls the tuneable capacitors of the EBD to achieve adequate self-interference cancelation. To this end, a constant sine wave is transmitted and the power of the

residual self-interference is measured with the FFT technique. As shown in Figure 14, the MicroBlaze then runs Particle Swarm optimizer to find the optimum solution aiming to minimize the self-interference. The algorithm is able to find this solution in less than 1 ms and is fully implemented in C.

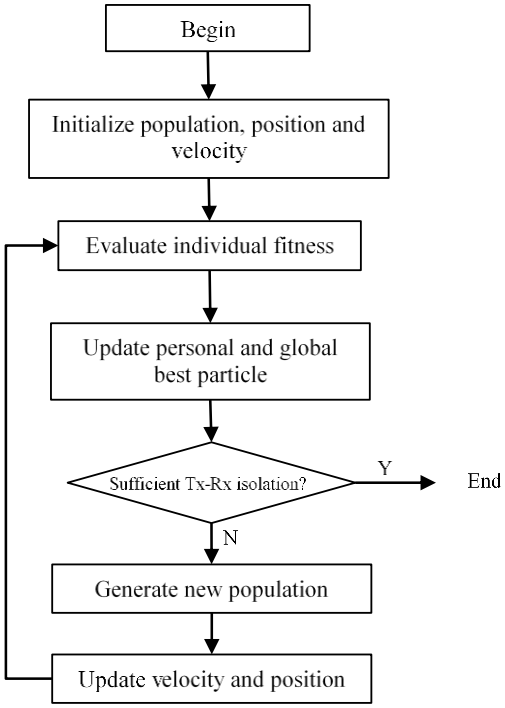

Figure 14: Particle Swarm optimizer for EBD tuning.

As shown in Figure 15, the hardware is based on a cross-layer architecture including IEEE 802.15.4 compliant PHY and unslotted CSMA MAC protocol. To satisfy heavy data processing tasks in the physical layer, it is implemented in a LabVIEW FPGA module using LabVIEW Communication software. Besides, to reach the optimal optimum performance, the MAC is divided into low and high-level MAC blocks. The low-level part is implemented closely to the PHY and performs a number of basic functionality of the MAC such as generating and checking Frame Control Sequence (FCS).

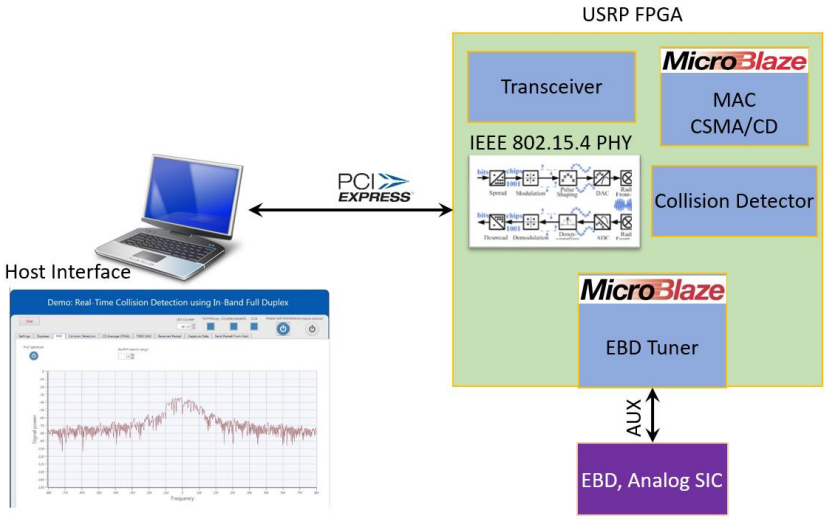

Figure 15: FD SDR architecture.

The high-level MAC is also deployed on a MicroBlaze softcore to allow runtime programmability. The

softcore establishes the CSMA scheme with collision detection. As it is illustrated in Figure 16, the MAC assesses signal interference at the transmitter side. In this scheme, the sender listens to the channel and stops transmission in case of collision, saving energy and improving the reliability and throughput of the network.

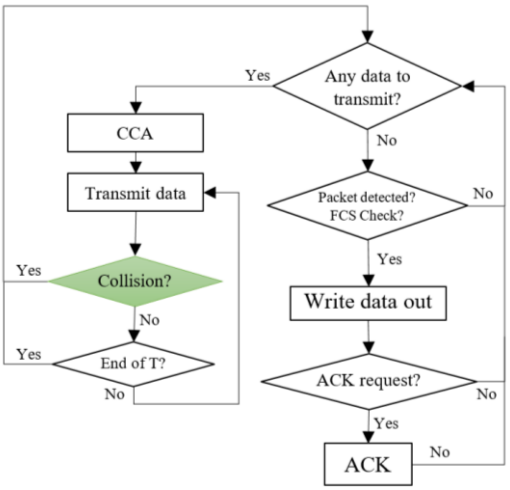

Figure 16: CSMA/CD MAC protocol

In our design, the collision detector is fully implemented on FPGA enabling rapid interference detection within 250 µs. The goodness-of-fit Kuiper test is applied to detect signal collision. As shown in the Figure 17, this statistical approach can detect collisions from interference stronger that -75 dBm with 95% accuracy. This figure also reveals that the sensitivity of Kuiper test is 15 dBm better than the energy detector.

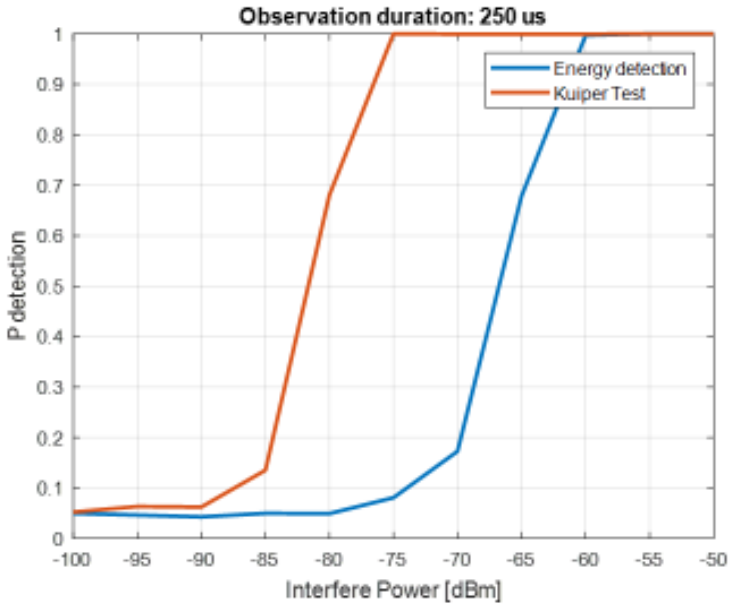

Figure 17: Implementation results for collision detection using 2000 samples (250 µs).

3.1.2 Higher layer protocol integration

The ORCA FD testbed for collision detection is integrated with WiSHFUL [9] allowing the experimenters to design MAC schemes and program them at run-time. Therefore, WiSHFUL compliant Unified Programming Interfaces (UPIs) are developed enabling the experimenters to control the testbed and run measurements. The control plane has full control over the data plane using a PC interface. It can

route different data sources to the PHY which enables various test scenarios.

3.1.3 Testbed integration

The FD PHY and CSMA/CD are integrated to the ORCA FD testbed allowing a wide range of test scenarios. It is fully available for the upcoming calls for extension and experiments.

3.2 Relation to showcase

The developed PHY and MAC in the first year of ORCA project provide the essential link for ORCA showcase 2, low-latency industrial communication. In this showcase, the FPGA-based PHY connects the control unit to a mesh network including a number of robots. The robots transmit sensory data to the control unit and execute the control commands received from it.

3.3 Risk analysis

The EBD functionality can be affected by changes in the channel. It should be considered whether variation in Tx-Rx isolation affects the performance of the collision detector. One solution is to develop a faster optimizer for EBD tuning.

3.4 Implementation plan

In this section, we describe our plan for the coming year to extend and improve the functionality of ORCA FD testbed.

3.4.1 PHY improvements

In general, our aim for ORCA's second year is to employ FD to increase network reliability as well as to enhance the overall network throughput. In Year 2, we would like to apply context awareness to establish an environment-aware PHY. This capability can be applied to retune EBD faster and make it an appropriate solution for more dynamic scenarios, for example car to car communication. The next planned extension is to start developments to finally enable the support of the IEEE 802.11p standard which is mostly targeted by vehicular applications, although it is too ambitious to target a full IEEE 802.11p implementation within ORCA.

In addition, we are developing a faster and more accurate collision detector and differentiating between collision and high-priority message. This is possible using elaborate machine learning approaches and adds a merit design to the MAC protocol.

3.4.2 Higher layer protocol integration

The higher layer protocol is already integrated, and we continue the integration according to the upcoming updates in the lower layers. The developed control plane can configure a four-node IEEE 802.15.4 network and for next year we would like to integrate the new functionalities of the lower layers planned in the Year 2.

3.4.3 Testbed integration

The new developments on PHY and Collision detection will be integrated into the testbed and made available to experimenters. In other words, they can select among the PHYs and collision detectors according to their test scenario regardless of higher layer functionality. For example, they could run a test either on IEEE 802.15.4 or IEEE 802.11p according to the requirements of their experiment.

4 MASSIVE MIMO DEPLOYMENT FOR UPLINK THROUGHPUT IMPROVEMENT AND CHANNEL DECOUPLING

4.1 Implementation results

The performance of Massive MIMO (MaMIMO) depends critically on the quality of the channel estimation. Different factors could influence the channel estimation quality, even in static scenarios as showcase 1. We focus our PHY layer improvements and experiments on distributed Massive MIMO with TDD mode. Two main activities were carried out through the year. On the one hand a new method for reciprocity calibration (to enable downlink Massive MIMO from uplink channel estimates) was developed, implemented and tested. On the other hand, collection of channel data for a distributed Non-LoS (NLoS) scenario was carried out to evaluate channel orthogonality in a distributed setting and hardening, and how the number of antennas influenced in the overall throughput performance of the system. To enable distribution of our testbed, a testbed extension was needed to enable distribution of the FPGA code on two half testbeds. Below, we detail both PHY layer improvements.

4.1.1 PHY improvements

For our first activity a TDD reciprocity calibration was developed and applied in a Distributed Massive MIMO, it uses an inter-cluster combining (ICC) method, improving the signal-to-quantization-noise ratio (SQNR), and hence achieving a more robust calibration accuracy in comparison with Least Square (LS) method and a traditional hierarchal (TS) method. EVM and throughput at the receiver is shown in Figure 18 and Figure 19.

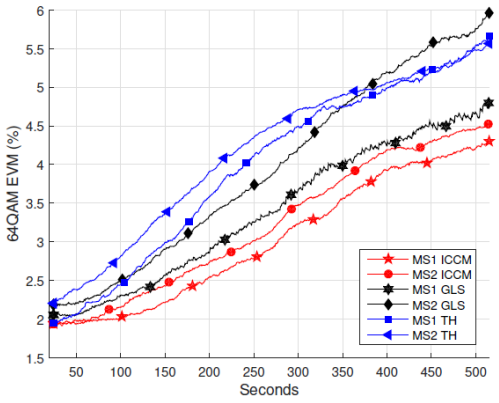

Figure 18: EVM after finishing reciprocity calibration.

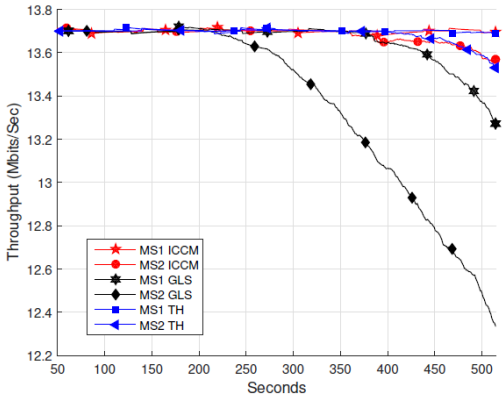

Figure 19: DL Throughput comparison of two MSs receive.

In the second PHY extension and experiment the UL throughput of 12 UE as function of the number of

antennas in combination with collocated and distributed topology of the BS antenna array was collected. In Figure 20, the system achieved almost peak throughput for both 64 antenna cases. For 32 antennas, we see a larger imbalance between users for the collocated BS, while the distributed deployment achieves a more homogeneous performance across UEs.

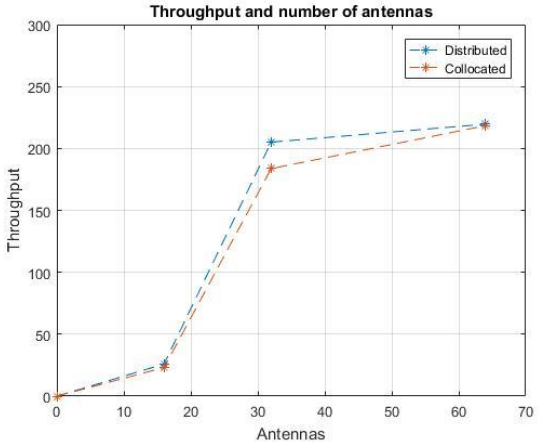

Figure 20: Overall throughput per scenario according their antennas number.

Figure 21 and Figure 22 representing channel orthogonality as a cumulative distribution function (CDF) of singular values spread (SVS) for all channels measured for collocated and distributed topology, and compare with i.d.d. Rayleigh channel. Noticing a trade-off between channel orthogonality and hardening, dependent on the sub-array location in our experiment.

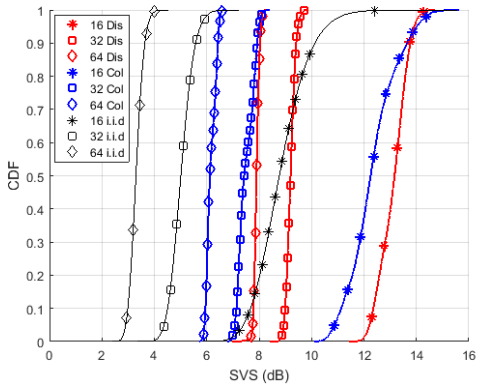

Figure 21: CDF of SVS.

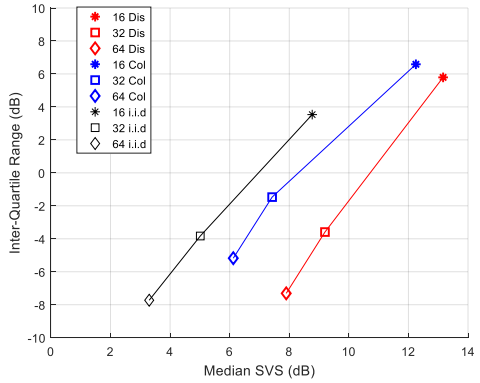

Figure 22: Median and IQR of SVS values.

We believe that our measurements results allow for some careful early conclusions. First, when the number of antenna elements is large, in a small cell scenario, the topology or location of the antenna

elements is not that important. When the channel is good and orthogonal enough, nothing is to be gained by placing the arrays optimally. Second, when the number of UE is large compared to the number of antenna elements (12 versus 32), the array topology matters more, and the distribution of the antenna's helps to improve the spread of the performance across users). All in all, we believe it is fair to say that the distribution of antennas improves performance. We can however not yet conclude that it is an optimal topology, especially if the extra deployment cost cannot be neglected.

4.1.2 Higher layer protocol integration

The Massive MIMO framework is real-time and enables both uplink and downlink communication. As such, it can be integrated with any end-to-end protocol stack.

4.1.3 Testbed integration

Both activities were carried out in KU Leuven MaMIMO testbed. The distributed MaMIMO is deployed on two antenna arrays of 32 patch elements each. Those are connected to two semi-mobile clusters with 32 National Instrument universal software radio peripherals (USRPs) which run LabVIEW Communications MIMO Application Framework 1.1. The UEs in each experiment were implemented with USRPs, each of them with two dipole antennas. We conducted the experiments in a stationary environment, meaning no people were walking around during the measurement.

All data collection and processing scripts will be made available in repository online as part of Deliverable D3.2 [10]. Remote access to the testbed is also possible, although it is not recommended to run experiments remotely on the bulky MaMIMO testbed setup.

4.2 Relation to showcase

The previous experiments are oriented to SC1: High throughput, in where each UE could be considering as a static robot, being MaMIMO a technology which ensures to deliver a high throughput.

4.3 Risk analysis

The results discussed are scenario-dependent due to channel propagation; therefore, more experiments must be carried out in different scenarios.

More antennas will be needed to achieve better conclusions as in the distribute case of MaMIMO 32 antennas in each array is a limited number.

4.4 Implementation plan

The plans to extend the functionalities of the MaMIMO testbed in the second year are discussed below.

4.4.1 PHY improvements

In the upcoming year we would like to include analysis for the physical layer to implement a periodical calibration to improve stability for the DL throughput. In addition, more experiments will be carried out for the indoor setup with mobile USRPs to cover more UE locations, emulating one and two cells to reduce intra-cell interference.

4.4.2 Higher layer protocol integration

The Massive MIMO testbed is as such fully bidirectional and closed-loop, i.e., it is possible to transmit and acknowledge in real-time video streams over the setup. Further integration with higher layer protocols is not in the scope of the ORCA project.

4.4.3 Testbed integration

Remote access of the testbed is currently enabled, but this needs to be further verified and tested (e.g., during open call for experiments). It needs to be seen to what extent remote access of such a complex distributed Massive MIMO testbed is meaningful.

5 FLEXIBLE PHYSICAL LAYER BASED ON GFDM

Generalized Frequency Division Multiplexing (GFDM) is a flexible PHY waveform developed to address the future communication systems requirements. For example, GFDM is suitable for cognitive radio applications, since the out-of-band emission can be significantly suppressed, and it can be configured to fulfil certain latency requirements [11]. In addition, GFDM can also be seen as a waveform generation framework, which can cope most modern waveforms. Basic principle behind is that usually waveforms can be described with the help of the modulation matrix $\boldsymbol{A}$ which is applied on the vector containing the data symbols $\vec{d}$

$$x = \boldsymbol{A} \cdot \vec{d}$$

The FPGA implementation of GFDM is using a time-domain approach described in 12 to realize the equation above. The development has been initiated in the EU-Project eWINE and is extended here to multiple waveforms instead of limiting it to the GFDM case only. A clear and simplified FPGA programming is used based on LabVIEW Communications Design Suite 2.0. The transceiver is designed for the NI USRP-RIO 2953R platform.

A general overview of the architecture is depicted in Figure 23. It consist of two different parts. All the timing critical signal processing is placed on the FPGA. This includes all the PHY functionalities as well as lower MAC functions such as Listen Before Talk. Higher layer functionality such as MAC, including creation of the headers are executed on a host computer to allow a fully flexible implementation. The blocks marked in blue in Figure 23 are taken from NI code libraries, whereas all white marked blocks are self developed and further documented in [13].

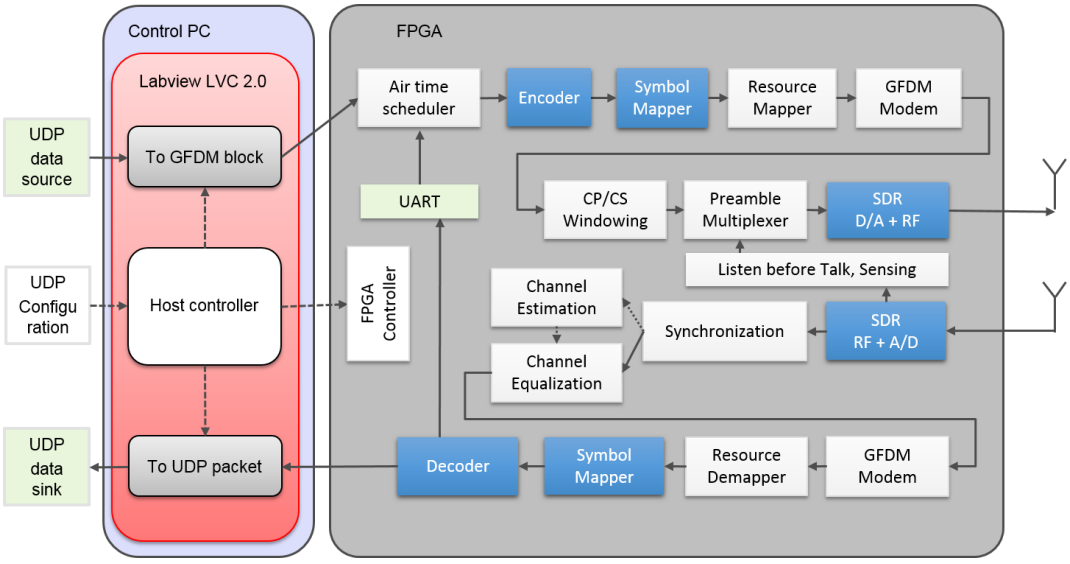

Figure 23: GFDM FPGA architecture.

5.1.1 PHY Improvements

The FPGA transceiver design differs from other flexible designs like RFNOC, such that the routing of the data path is fixed and cannot be changed. Typically, this is not needed because the processing steps have to happen after each other and cannot be executed arbitrary. However, the parameters of the blocks can be changed or a respective block can be bypassed, if a certain waveform does not require it.

Waveform	Resource Mapper	Modem FFT	Modem Pulse Shaping Filter	CP/CS unit	Windowing	Preamble Insertion
OFDM	Depends on frame structure	Active	Deactivated	Active	Deactivated	Depends on frame structure
LTE	Insertion of Pilots and control sequences	Active	Deactivated	Active	Deactivated	Deactivated
WIFI	1. header 2. payload	Active	Deactivated	Active	Deactivated	Active
5G NR	Insertion of Pilots and control sequences	Active	Deactivated	Active with overlapping	Active	Deactivated
GFDM	Depends on frame structure	Active	GFDM pulse shape loaded	Active	Active	Depends on frame structure
FBMC	Insertion of Pilots and control sequences	Active	FBMC pulse shape loaded	Active with overlapping	Deactivated	Deactivated
Single Carrier	Depends on frame structure	Deactivated	Deactivated	Active	Deactivated	Depends on frame structure
Spread Spectrum	Depends on frame structure	Deactivated	Spreading function loaded	Deactivated	Deactivated	Depends on frame structure
Chirp based	Depends on frame structure	Active	Chirp function loaded	Active	Alternative to apply chirp function	Depends on frame structure

Table 1: Configuration of the transmitter depending on the desired waveform

Table 1 gives an overview which processing units are needed to create a waveform. Some functionality needs preparation on a host computer, like the creation of the pulse shaping filter, windowing function and the resource map including the training sequences. During run-time these vectors will be downloaded to memory blocks on the FPGA. Unlike the filters or windowing functions, the number of subcarriers is restricted to power of two because of the used XILINX FFT block. In addition, the number of subsymbols M for the GFDM/FBMC mode is limited by 16. The cyclic prefix and suffix can be parametrized up to 257 elements.

Currently, the transmitter can process one configuration at each time. Figure 24 depicts the latency of the transmitter if all processing units or only a minimum set of units are used. The latency is depending on the number of samples to be generated for one block. Please note, that the scheduling unit and the RF sampling rate are ignored. The latency is depicted for the first sample received on the FPGA and the last sample is given to the RF fronted of the USRP. The left axis shows the number of clock cycles, whereas the right axis shows the time when the processing is running at 200 MHz clock speed. For example, a WLAN signal would have 64 subcarriers in one OFDM symbol, which relates to 64 samples

to be created. In case multiple OFDM symbols are needed, then the overall latency increases by adding the amount of samples as one sample is created with one clock cycle.

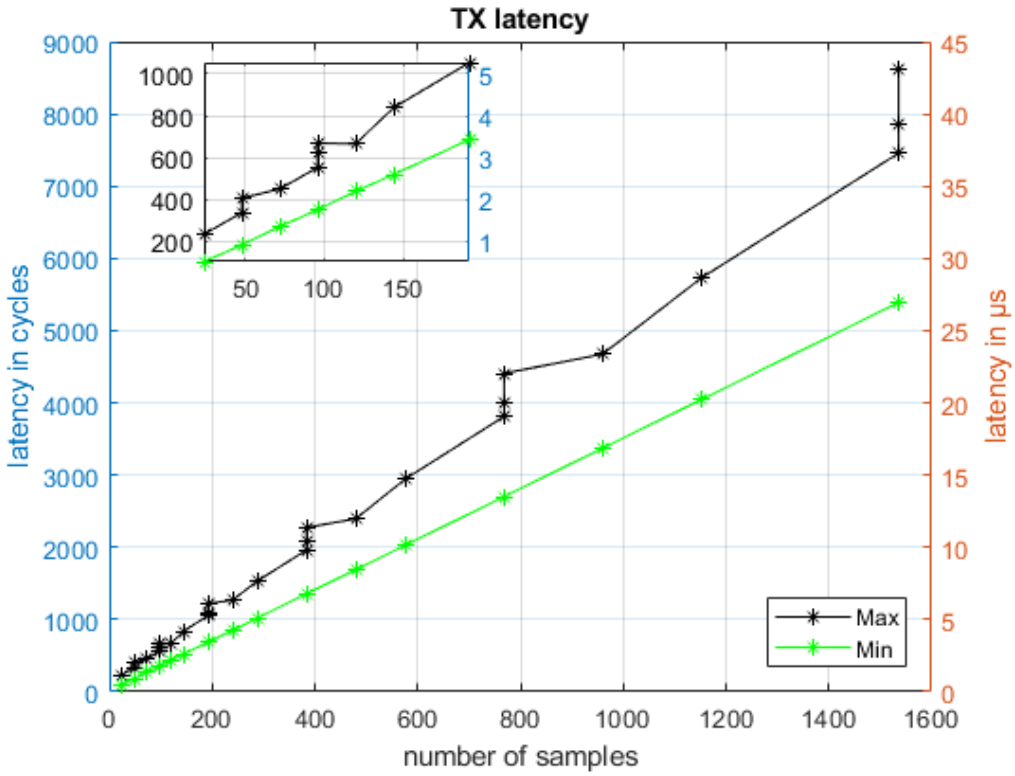

Figure 24: Latency of the TX signal-processing

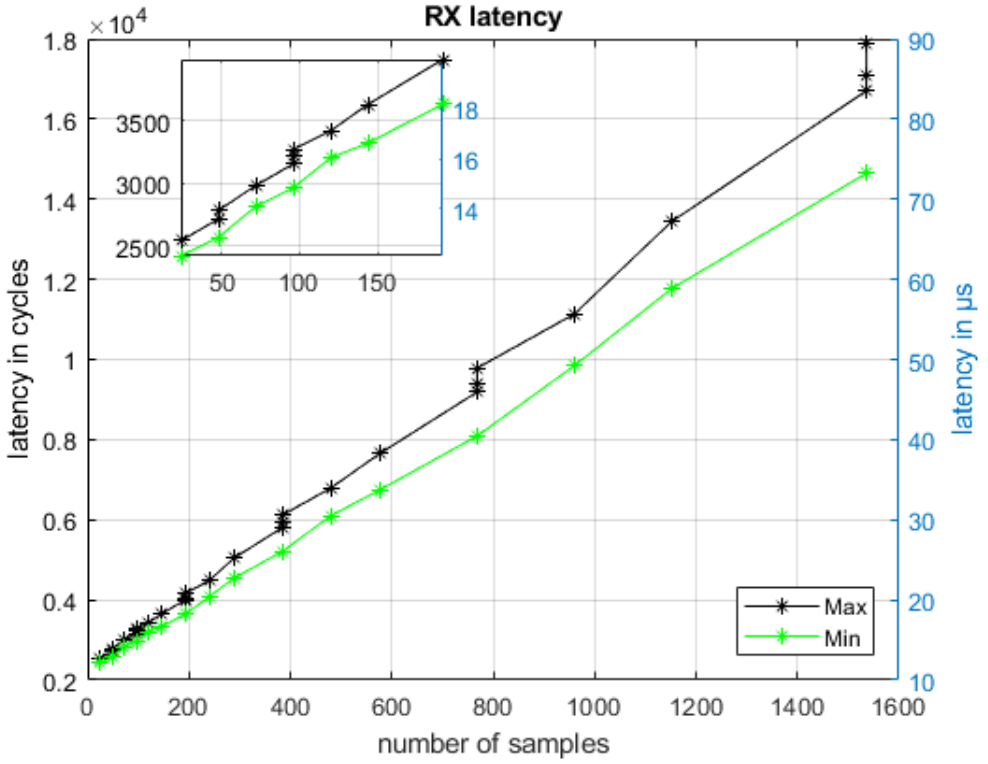

Figure 25: Latency of the RX signal-processing

The latency of the receiver, shown in Figure 25 is higher in comparison to the transmitter, because the synchronization module is not yet fully optimized. A future revision will improve the speed of the synchronization such that the receiver is similar to the transmitter. The jumps in the both graphs can be explained by the FFT latency in GFDM mode, which increases with the number of subcarriers. Similar to the transmitter graph, the latency is depicted for the first sample received on the FPGA by the RF frontend and the last sample is written into the FIFO to the host computer.

Validation measurements depicted in Figure 26 show that the implementation meets the theoretical symbol error rate. The experiment were made with $K = 512$ and $K = 1024$ for both types of receivers, i.e., matched filter (MF) and zero-forcing (ZF). These results indicate that the implementation is correct.

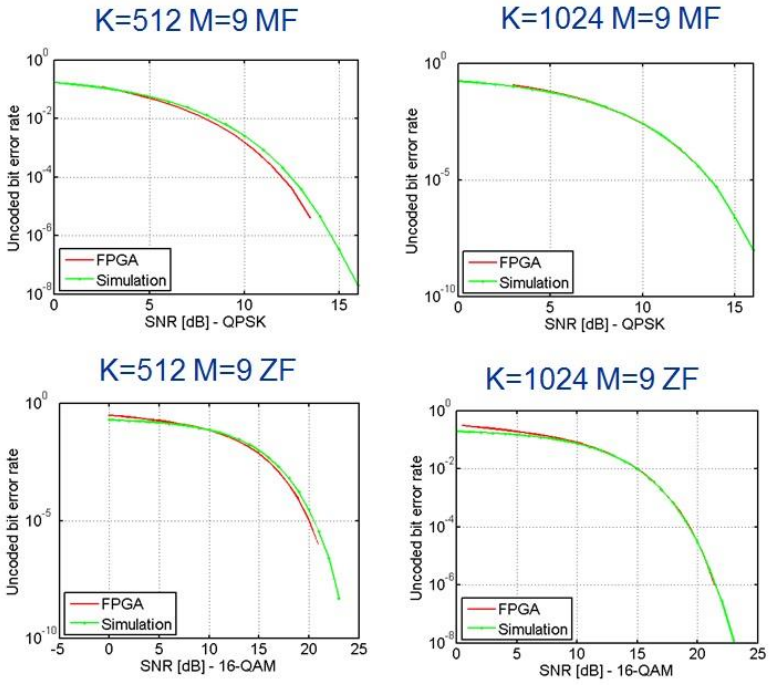

Figure 26: Symbol error rate

Finally, the parametrization of the PHY including a complete change of the waveform can be achieved within 100 ns or 20 clock cycles. Figure 27 shows two different PHY configurations multiplexed in time, captured by another SDR device. The first, longer signal has a WLAN-like configuration with 64 subcarriers and 9 subsymbols, whereas the second signal only uses 32 subcarrier and three subsymbols for very fast signal processing as favoured by low latency applications. In both cases, different preambles are prepended before the data block to distinguish between the different services. Please note that the signal shape of the preambles are modified for simplified identification and will not be used in practical systems.

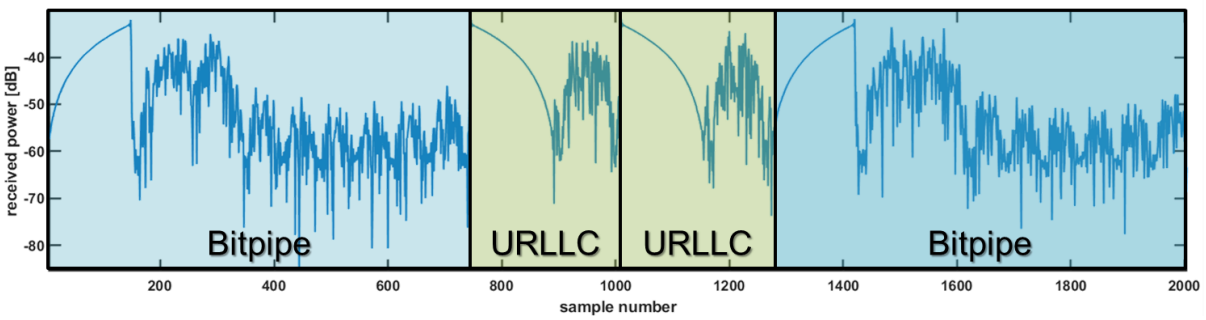

Figure 27: Captured RF signal of two different numerologies created by the flexible transmitter multiplexed in time

5.1.2 Higher layer protocol integration

Currently, the transceiver implementation supports the timing critical lower MAC-functionalities on the FPGA, mainly to support Listen before Talk (LBT) for unlicensed bands operation. This includes to set the parameters for the energy threshold for the carrier sensing unit as well as timing relevant settings for the medium access unit such as backoff-times, waiting times etc. However, these functionalities can be deactivated in case licensed spectrum is going to be used. For end-to-end networking use cases, non-timing critical parts of the MAC will be running on a Real-time Linux system attached to the FPGA. This allows supporting various MAC types and will be part of the development in the second year.

5.2 Relation to showcase

The second showcase is related to industrial control applications, where a tight control of machines is envisioned. To demonstrate such capabilities, the GFDM PHY is parametrized to achieve a less than 100 µs round trip latency (without higher layer and cloud processing), facilitating a time-critical control loop application. Figure 28 depicts the evaluation setup, where the sensor data is send from the robot to the cloud. The cloud will process the data and send specific motor control commands back to the robot, such that it can keep the balance.

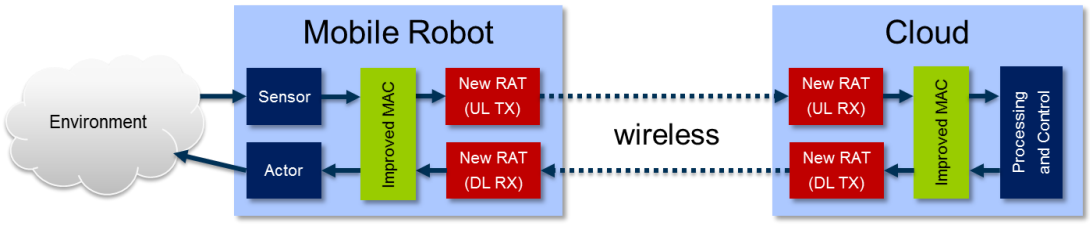

Figure 28: Setup of the real-time loop application

5.3 Risk analysis

There is no specific risk observed at this phase of the work package.

5.4 Implementation plan

This section describes the plan for functionality to be implemented in the upcoming year.

5.4.1 PHY improvements

Firstly, the current implementation will be optimised to reduce the consumed area. Second, a messaging system similar to the Femto-Forum-API will be introduced. This will allow to pass messages from higher layers to FPGA processing units itself. Thus, the FPGA hardware accelerators can be programmed and

parametrized in real-time, without additional configuration delays using the TestMan techniques introduced in the CREW project [14]. Further, also control information, events as well as complete data samples/frames, such as the channel coefficients can be passed back to the higher layer application. This enables a tight control of the signal processing. For example, based on the high-resolution channel measurements an intelligent MAC layer can adjust not only the MCS but the whole waveform according to the current wireless situation. In addition, the FPGA signal processing will be extended to pipelining. So, each block in Figure 23 can apply a different waveform configuration on the data stream. Using this feature, the overall throughput of the transmitter reaches up to 200 MS/s, which will be sufficient to facilitate the 160 MHz bandwidth RF frontend of the USRP. Further, the signal processing will be extended to support concurrent operations of multiple MACs.

5.4.2 Higher layer protocol integration

The higher layer protocol integration will be realized through the Femto-Forum-similar API using the TestMan principle and be part of the development carried out in work package 4.

5.4.3 Testbed integration

The described systems will be made available as part of the TUD testbed. At least one base station will be equipped with a HPE edgeline server and an attached USRP-RIO SDR platform. The user terminals are based on a USRP-RIO SDR with a control PC each and an additional access point will have a NI PXI based chassis. All platforms support the PHY and higher layer capabilities with a varying processing speed.

6 HYBRID FPGA PLATFORM INCL. FLEXIBLE MAC

6.1 Implementation results

The Hybrid FPGA platform (i.e. Xilinx ZYNQ) contains a dual-core ARM processor, referred to as PS (processing system) and FPGA fabrics, referred to as PL (programmable logic). A high-level architecture view of the radio design running on ZYNQ is presented below (Figure 29). In the first year of ORCA, the focus is on the "PHY accelerator pool", two types of PHY accelerators are made available, namely the OFDM waveform transceiver (similar to Wi-Fi), and direct-sequence spread spectrum (DSSS) transceiver (similar to ZigBee). Regarding to integration towards higher layer protocol stack, this is closely related to control interface of individual PHY accelerator interfaces.

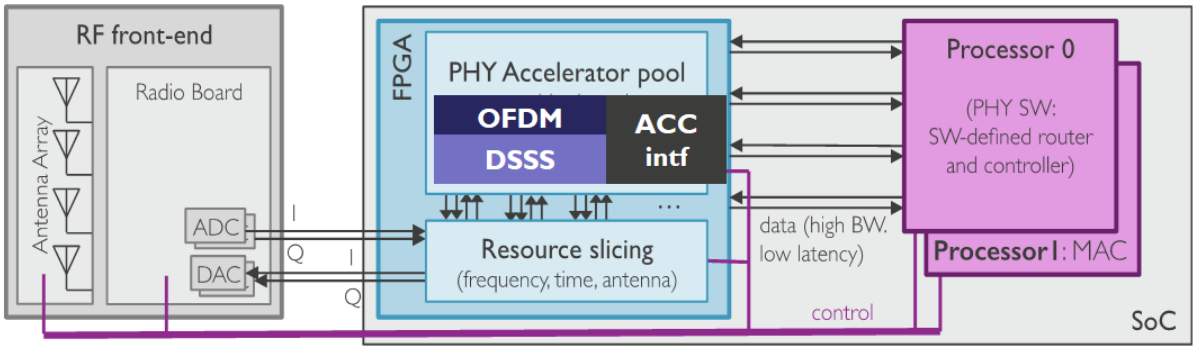

Figure 29: General architecture of the hybrid SDR on ZYNQ.

6.1.1 PHY improvements

There are various approaches to design a PHY accelerator. Conventionally, FPGA designer works directly on HDL coding. In recent years, various tools are available on the market, such as LabVIEW FPGA, Matlab system generator. Each approach has its pros and cons, the final choice in fact depends on the implementer's personal preference. For instance, the DSSS accelerator is implemented directly with HDL, which has a compact FPGA footprint, while the OFDM accelerator is implemented in Matlab system generator. Regardless the design tools, all accelerator expose similar interface towards the ARM, which will be addressed further in the next section.

The design of DSSS accelerator is mostly achieved in WiSHFUL project [9], the work of ORCA in year 1 focuses on integration of this accelerator with the rest of the design.

The implementation of OFDM accelerator is a new development in year 1 of ORCA. The core contains a classical signal processing chain, taking example of the receiver, the following high-level blocks exist: (i) the interface to receive samples, (ii) the preamble detector, based on auto and cross correlation, (iii) the synchronization and frequency offset correction, (iv) the FFT, (v) the channel estimation and equalization, and (vi) decoding process. Many configurable parameters exist for each of the processing stage, which are all accessible via ARM for run time reconfiguration. The details of these parameters are provided in D4.1 [1].

The operation speed of the OFDM accelerator is run time configured controlled by the ARM processor, via another hardware module, referred to as 'ACC intf' in Figure 29. This is similar to the concept of clock gating, i.e. the system clock is running at a fixed rate, however the clock enable (CE) signal is run time configurable. The CE may run as fast as needed, however it must allow at least 4 system clock cycles in between adjacent CE pulses. The 4 system clocks are necessary for the transition of internal finite state machines to complete.

The support of flexible operation speed is important. For instance, when the speed is set to 20 MHz, the waveform matches the bandwidth of IEEE802.11a/g, when operating at 2 MHz, it matches the waveform specification of IEEE802.11ah. In addition to the global operational speed (e.g. 20 MHz vs 2 MHz), the

'ACC intf' must fine tune the clock enable speed of the accelerator to be in line with the speed of the RF frontend (i.e., ADC and DAC). This is necessary because the FMCOMM board and the ZYNQ board are driven by different clocks. Essentially the operation speed of the accelerator should be greater or equal to the sample rate of the RF frontend. When the accelerator runs at the multiples of ADC or DAC, the hardware virtualization is achieved.

The latency profile of the OFDM Rx accelerator is measured through the scope in system generator. A Wi-Fi packet is captured in the air, and the IQ samples are streamed to the accelerator. In total, there are 595 system clock cycles after the last IQ sample is received, and before the valid signal of Frame Control Sequence (FCS) rises, which determines the latency of the accelerator. This is shown in Figure 30. The complete design is running at 200 MHz system clock, hence 595 clock cycles maps to 2.9us, which is well within the latency requirement. Note that the latency can further decrease when increasing the CE rate.

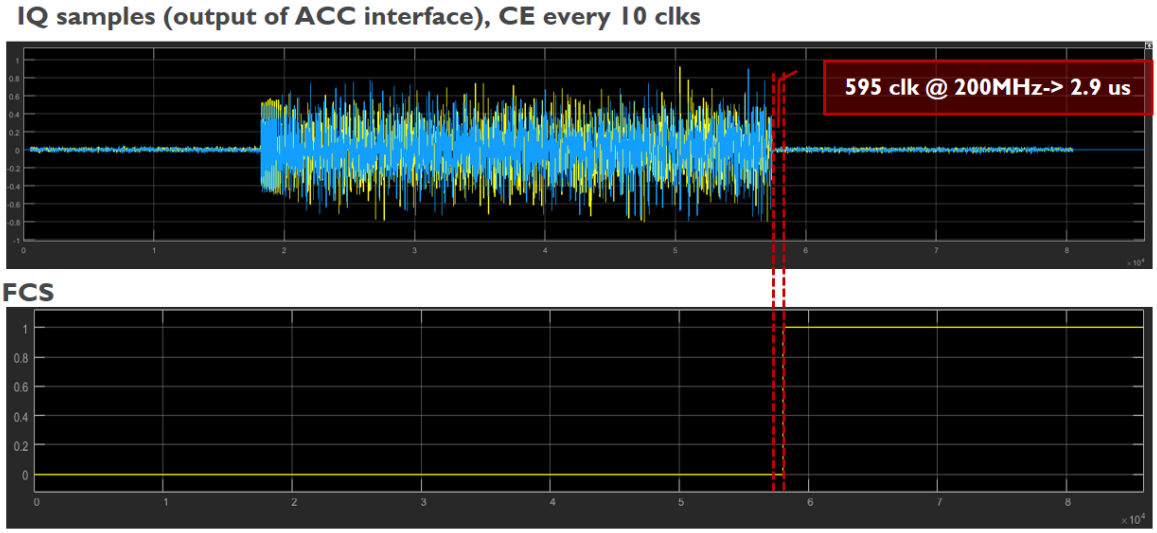

Figure 30: Latency measurement of OFDM Rx Accelerator.

6.1.2 Higher layer protocol integration

The integration towards higher layer protocol is a continuation of the work done in WiSHFUL for porting the TAISC (Time Annotated Instruction Computer Set) radio architecture and Contiki OS to the ARM processor of the ZYNQ board. The Contiki OS includes a complete 6LowPan IPv6 network stack on top of TAISC and incorporates the WiSHFUL extensions, enabling full control of the network stack (i.e. PHY to APP). Moreover, the MAC schemes developed for the TAISC radio architecture are also available on the ZYNQ board when the required functionalities are supported by the SDR radio boards.

In short, TAISC is an architecture for designing, implementing and runtime configuration of flexible MAC schemes. A MAC scheme can be designed by composing a TAISC chain. A TAISC chain is defined as a sequence of instructions with one optional time reference instruction. The TAISC architecture annotates each instruction of a TAISC chain (stored in the program memory) with timing information (both execution time and offset with respect to the time reference). To optimize time synchronisation, all instructions are packed before and after the reference instruction.

TAISC was closely coupled with the hardware modules of the MSP430 processor that is used in the RM090 mote, the first sensor node platform TAISC was supporting, so a redesign of TAISC including the definition of a generic southbound interface to the HAL (Hardware Abstraction Layer) of any specific platform was performed in the WiSHFUL project, for details please refer to WiSHFUL D3.4 [15]. This deliverable focuses on the realization of TAISC on ZYNQ SDR platform.

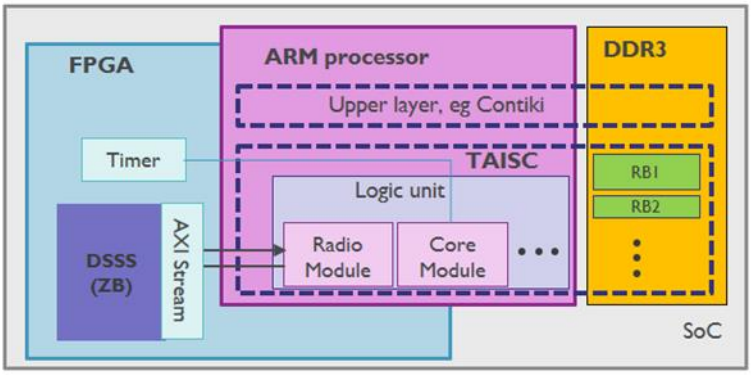

Figure 31: TAISC on ZYNQ - coupled with DSSS accelerator.

A very high-level diagram of TAISC on ZYNQ is sketched in Figure 31. The 'Logic unit' of TAISC is running on the ARM processor, while the TAISC memories are allocated on the DDR3. The TAISC memory is split into ROM and RAM sections. ROM is mostly used to store pre-compiled binaries (i.e. radio binaries) for specific MAC program, while RAM is used to store dynamic content, such as context storage for management purposes. Upper layer protocol can be constructed directly above TAISC, or with the aid of framework such as Contiki.

Basic driver to support configuration of the hardware accelerator in FPGA (i.e., DSSS or Zigbee PHY) is made available through the Radio Module of TAISC. A dedicated timer is constructed in the FPGA to support the 'time annotated' execution engine of TAISC. This comes down to following steps: (i) TAISC configures the timer according to the binary instruction for specific MAC chain, (ii) the timer fires an interrupt request towards ARM, triggers the interrupt service routine and reaches the TAISC Core, (iii) the planed instruction can then be served on time.

In order to proof TAISC operation on SDR with proper MAC configuration, we establish a TDMA link between a ZYNQ SDR and commercial ZigBee sensor node. The sensor node sends out beacon frame every 90 ms, to maintain the synchronization. Within the 90 ms super frame, there are 9 sub frames allocated to specific nodes for transmission, each sub frame lasts for 10 ms. In this particular setup, the sensor node has node ID 1, while the SDR node has ID 5.

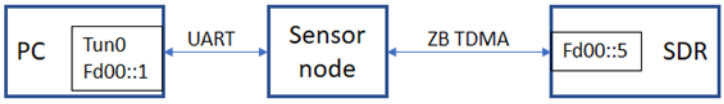

Figure 32: TDMA connection between SDR and commercial sensor node.

From the application layer point of view, the sensor node is acting as a boarder router, which is connected to a PC via UART. The PC regards the link as a tunnel device with certain IPv6 address. The SDR node also has an IPv6 address, determined by its node ID. Hence a communication can be initiated from the PC over the sensor nodes to the SDR node. The setup described above is also shown in Figure 32.

Figure 33 shows the 'over-the-air' activity of the TDMA link. The figure is generated using IQ samples, captured by a third USRP. The PC is pinging the SDR every 200 ms. The shortest frames are the beacons, whereas the slightly longer packets are the ping (send out by commercial sensor node) and the reply packets sent out by ZYNQ SDR. It is observed that the ping packets are occurring at the correct slots within the TDMA super frame.

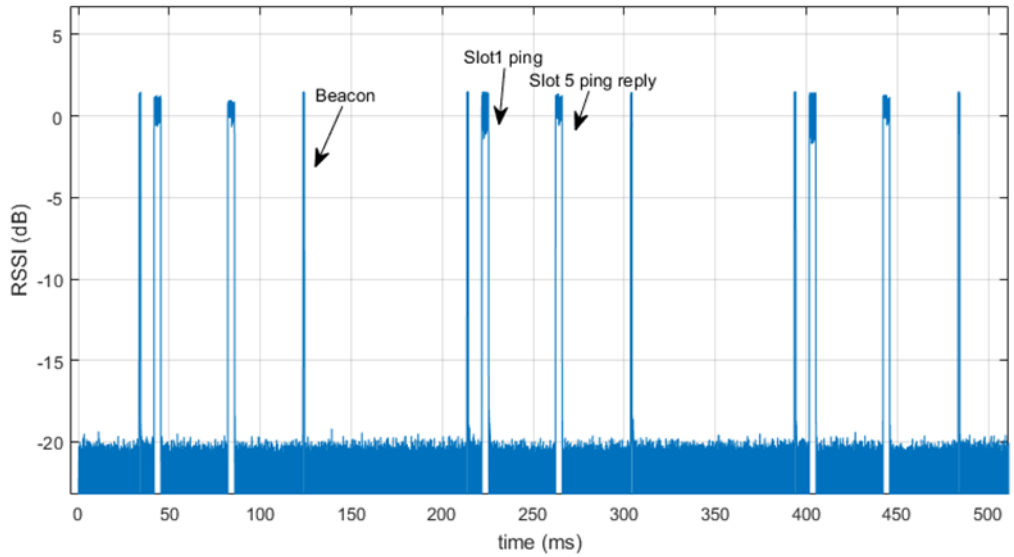

Figure 33: Over-the-air activity of TAISC TDMA running on ZYNQ SDR.

In addition, the TAISC operation on SDR is monitored via a logical analyser, as illustrated by the screenshot below (Figure 34). Each time when a TAISC command is executed, the ID of the command is 'print out' via a customized software SPI bus, this bus is connected to GPIO pins on the SDR board and captured by the analyser. More specifically, the first three rows in the figure below are the data, clock, and enable signals of the SPI bus. The short pulses in the last row corresponds to the event that a packet has been received, where the long pulse signifies that a packet is being transmitted. Hence the two short strokes are caused by the reception of beacon frame and the ping packet, while the longer pulse corresponds to the ping reply towards the commercial sensor node. It is also worth mentioning that one chain contains several commands of TAISC, this can be observed via the 4th and 5th of row in the logical analyser. The TDMA configuration is simply one type of 'chain' in TAISC.

Figure 34: TAISC activity on SDR observed by logical analyser.

6.1.3 Testbed integration

The ZYNQ based SDR platform is foreseen in w-iLab.t testbed, an image with appropriate Xilinx tools is prepared for users to experiment with both OFDM waveform and the DSSS SDR communication links. For more details please refer to D6.1 [16].

6.2 Relation to showcase

For both showcase 2 and showcase 3, the hybrid architecture (i.e. ARM processor + hardware accelerator) is essential to support radio hardware virtualization, which is needed for flexibility.

The DSSS accelerator, and the integration to higher layer protocols (MAC and above) are used in showcase 2, where a low latency link is established between SDR nodes for controlling the motion of a robot.

The OFDM waveform accelerator is a major building block for showcase 3, its configurability regarding bandwidth (or subcarrier spacing), and various lower level parameters makes it possible to configure a slice to fit into radio environment and upper layer traffic demands.

6.3 Risk analysis

There is no specific risk observed at this phase of the work package.

6.4 Implementation plan

This section introduces functionality to be implemented in the upcoming year.

6.4.1 PHY improvements

In the second year of ORCA, we would like to continue build on the existing accelerators, enable tight couple of channel assessment, which is necessary to support CSMA like MAC protocol running on either OFDM or DSSS transceiver.

Another goal is to support fast switching of transmit or receiving frequency. This would help to enable the support for MAC protocol like TSCH [17], since it requires frequency hopping. A more optimized approach could be used to improve the frequency switching performance, for instance using mixer in digital domain instead of tuning the frequency at the analog front-end.

6.4.2 Higher layer protocol integration

The base of higher layer protocol integration has already been prepared in year 1. We would like to continuously update the support for higher layer protocol, mainly via TAISC platform. The PHY updates will naturally trigger the support for more MAC or improvement of existing MAC protocols.

6.4.3 Testbed integration

The code repository will be updated on regular bases, which will be made available on ORCA portal and on w-iLab.t testbed tutorials. In this way, experimenters will be able to follow the newest developments.

7 RADIO ENVIRONMENT MONITORING

This section concerns TCD's signal classification framework implemented in Year 1 and the main results obtained. In this deliverable, we describe the general PHY functionality, its integration with upper layers, and future steps. For further details on the control plane and slicing aspects of this implementation, please refer to Section 6 of D4.1 [1].

We provide below the general high-level diagram of our framework. Throughout this section, our description of its capabilities will be mainly centered on its use for identification of RF slices/waveforms. This is in line with its intended use in the Year 1 Showcase 3, where TCD is involved. Additionally, we expect that a considerable percentage of the functionality provided by this framework can also be repurposed for other types of classification problems involving RF signals.

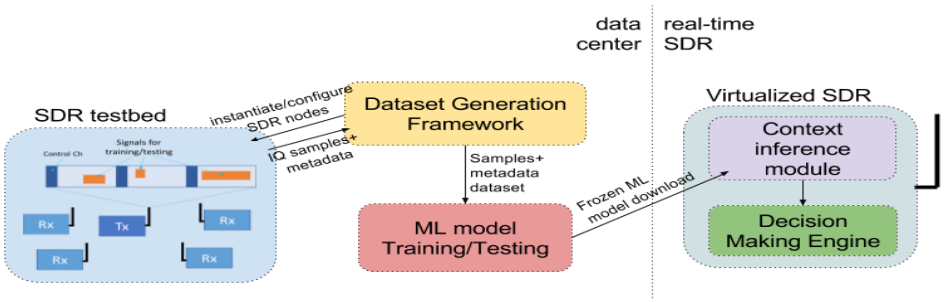

Figure 35: High level diagram of the several components involved in the ML-based signal classifier.

Our framework primarily relies on machine learning (ML) for the design of RF signal classifiers. The design flow of a classifier can be divided into the three main stages: (i) dataset collection, (ii) training and validation, and (iii) inference. During the first stage, the experimenter characterizes the set of possible RF waveforms and respective RF parameters (e.g. frequencies, duty cycle) that its SDR may encounter in the RF environment during its normal operation. For the specified waveforms and parameters, the framework will then generate a dataset composed by several IQ sample files, and associated metadata/labels. This data collection process is managed automatically by the provided framework, which will remotely access, instantiate and configure SDR nodes of the IRIS testbed to perform RF signal transmission/reception and data file storage. The resulting dataset is then used in the following stage to train and test ML-based signal classifiers. Classifiers can then be "frozen", transferred and deployed in real SDR nodes for context awareness.

7.1 Implementation results

In this section, we describe the PHY layer design decisions made by TCD for its dataset collection framework, and the real-time capabilities of its machine learning (ML) classifier models.

7.1.1 PHY improvements

In Section 6 of D4.1 [1], we describe the general architecture of TCD's dataset collection framework, the main motivations behind each design decision, and its reconfiguration APIs. In particular, we highlight the importance of highly detailed labels of RF signal features for the purpose of training and validation of ML classification models. We then describe how such data collection and labelling can be done more effectively with the proposed framework.

7.1.1.1 Dataset collection framework

In this subsection, we focus on the performance capabilities related to the implemented data collection framework's individual PHY components. Consider the SDR testbed setup in Figure 36 below, involving two USRP boards, one as a transmitter (Tx) and the other as receiver (Rx). An experimenter

interested in collecting over-the-air IQ samples and associated labels for a specific set of waveforms and RF parameters (e.g. frequency, hardware gains), writes a configuration file (cfg). The data collection framework will use this configuration file to setup the USRPs, and record the USRP-Rx received samples into files stored in a database. Each file will index a certain waveform and parameter configuration defined by the original experimenter's configuration file. For further details, we encourage the reader to consult Section 6 of D4.1 [1].

While the experimenter knows beforehand the types of signal that are going to be stored into the database by consulting his configuration file, certain RF parameters also relevant to the training of ML models, such as SNR, Carrier Frequency Offset (CFO), and Direct Current (DC) offset, still remain to be known. Furthermore, there is also a general need to keep the USRP-Tx and USRP-Rx synchronized, so that the labels of their transmitted and received samples remain consistent, respectively. We accomplish this through the periodic transmission of preambles.

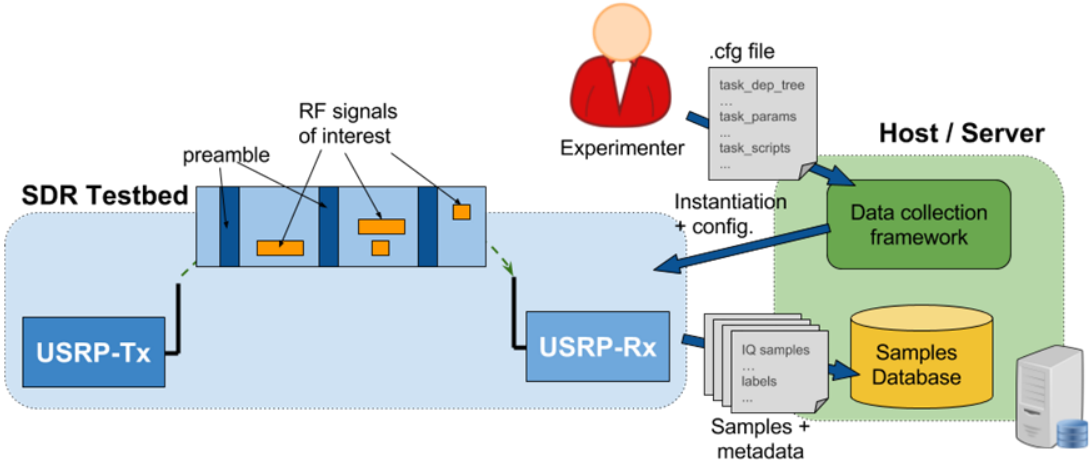

Figure 36: Illustration of a RF signal collection procedure.

RF signal/waveform classification can generally be performed at very low SNRs (below 0 dB). For the estimated RF channel parameters by TCD's framework (e.g. time synchronization and SNR) to remain valid at such low SNRs, the preamble structure used has to be very robust to noise. TCD chose a structure composed by several short Zadoff-chu sequences with absolute phase shifted by an m-sequence for coarse frequency and time offset estimation and disambiguation, followed by a long Zadoff-chu sequence for precise offset estimation. For a preamble size set to 1031 samples, we show below the USRP detection performance, and the root mean squared error (RMSE) and bias of several RF parameters' estimators. As can be seen, this preamble format enables the collection of samples at very low SNRs.

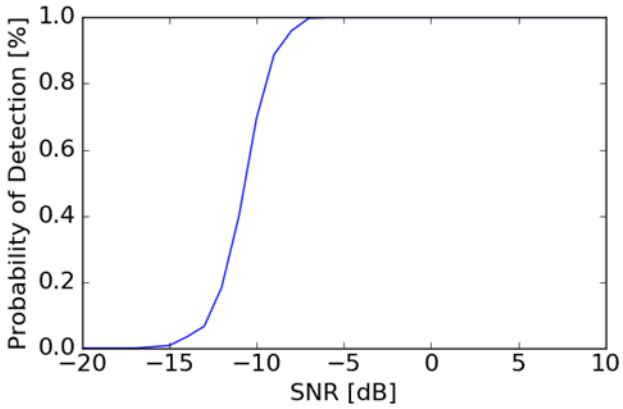

Figure 37: Probability of detection of a preamble of length 1031.

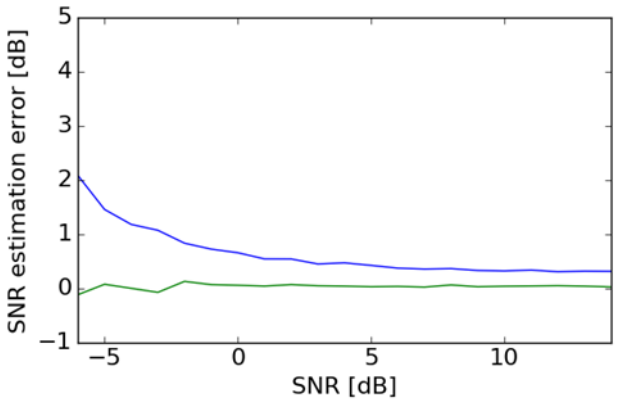

Figure 38: RMSE of the SNR estimator for a preamble of length 1031.

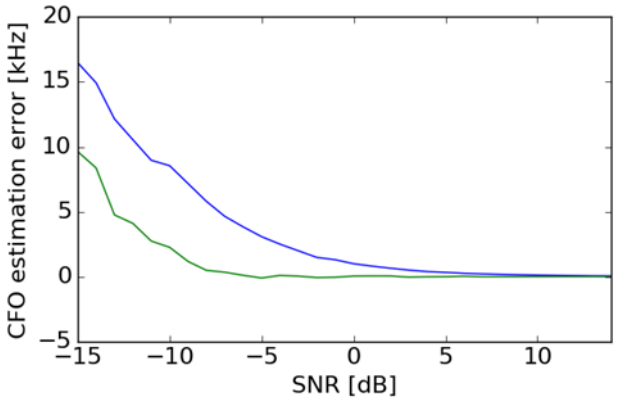

Figure 39: RMSE of the CFO estimator for a preamble of length 1031.

To enable synchronization in real time, several optimizations and considerations were taken during the design and implementation of the proposed preamble structure. First, all operations associated to autocorrelation, cross-correlation, averaging, and magnitude computation were heavy optimized, leveraging single instruction multiple data (SIMD) instructions. Furthermore, the proposed structure avoids cross-correlating the received signal with long pseudo-random sequences during the initial steps of preamble synchronization, relying primarily on inexpensive single-delay autocorrelation computations. Such decisions allowed our preamble detector to cope with sampling rates of 20 MS/s in a modern CPU (Intel Core i7-6820HK Processor).

7.1.1.2 Machine Learning-based Classification

In this subsection, we focus on the real-time capabilities of some of the ML models designed by TCD for RF signal classification during Year 1. For information related to classification performance of these models, the reader may consult Section 7 of D4.1.

In [18], TCD utilizes a Convolutional Neural Network (CNN) to detect and distinguish between different frequency hopping patterns and duty cycles of another RF device's transmissions. An USRP N210 is used to collect IQ samples, which are converted in the host computer into spectrograms before entering the CNN. Being this work part of the live Dyspan Challenge 2017 [19], there were strict requirements for the CNN to operate in real-time for input signals sampled at a rate of 10 MS/s.

To build the CNN model, TCD started by training a CNN architecture similar to [20] with input resolution 227x227. The main drawback of this model was that it took 0.2 seconds to classify one spectrogram on the used machine (Intel Core i7-6820HK Processor), which is not feasible to run in real time. For that reason, TCD experimented with other simpler configurations, using lower input dimensions and fewer convolutional layer filters. For each configuration, a CNN model is trained, its

classification accuracy was tested, and the CPU-feedforward time was calculated. After this process, TCD reached the proposed architecture shown in the Figure below, using gray-scale spectrograms with input size of 64x64, which gives high classification accuracy and CPU-feedforward time of 6 msec per spectrogram.

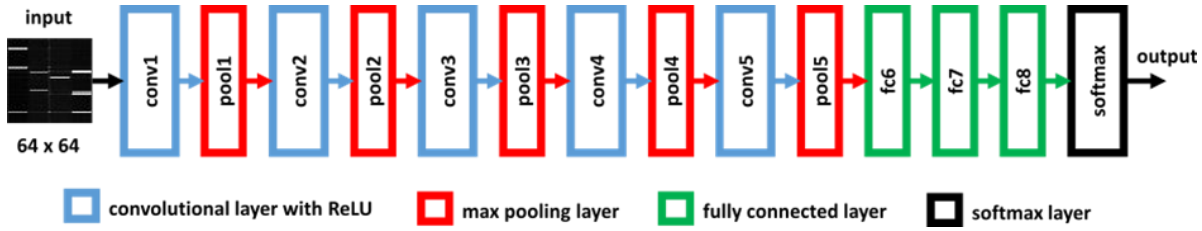

Figure 40: Convolutional Neural Network (CNN) architecture used by TCD used in [TCCN].

Each horizontal line of the spectrogram image represents the average power of 120 successive FFT bins (0.768 ms), and each channel (2.5 MHz) is represented with 16 bins (i.e. one spectrogram covers a duration of 0.768x64= 49.2 ms and a bandwidth of 2.5x4 = 10 MHz). TCD concluded that a window of 49.2 ms is enough to create distinctive sets of spectrograms for each scenario. The CNN network consists of 5 convolutional layers equipped with 48, 128, 192, 192, and 128 filters, respectively. Each convolutional layer is accompanied by a rectified linear unit (ReLU) and followed by a max-pooling layer. The output of these convolutional layers is then passed through 3 fully connected layers with 1024, 1024, and 10 neurons, respectively. The last 10 neurons are fed to a softmax classifier to compute the probability of each scenario.

While deep learning is well known for requiring intensive computational resource usage, this usage usually refers to the training and testing procedures. The previous example has served as proof that deep layered CNNs can be effectively used for inference in real-time RF signal classification applications, even without resorting to multi-threading or GPUs.

7.1.2 Higher Layer Protocol Integration

For the training, testing and deployment of the ML-based classification models implemented in Year 1, TCD used the widespread TensorFlow and Caffe deep learning frameworks. The maturity and extensive documentation of these frameworks facilitates the reuse of these models by other experimenters, as starting points for solving new classification problems, or for re-training and testing new deep learning networks. TensorFlow, for instance, provides the facility to save “checkpoint” files during training. Training can then be resumed starting from any of these files, even if some changes have been made to the neural network architecture. This facility represents one of the possible methods of “transfer learning”, enabling not only the reduction of training times, but also the demand for very large datasets. The resulting models can then be “frozen”, and used for inference in applications.

While the training and validation facilities provided by TensorFlow and Caffe are mature and easy to use, TCD found that their integration into SDRs for real-time RF waveform/context inference required further development steps.

Like in computer vision and speech recognition, input data representation is essential to determine the accuracy of an ML model when applied to RF classification. We show in Section 6 of D4.1 and [21] the different classification performances that TCD achieved with a CNN, when its input was in spectrogram or amplitude+phase shift formats. In the light of these findings, TCD decided to make available the off-the-shelf GNURadio blocks and python scripts to convert IQ samples to different data input representations that are understood by their trained ML models.

The ML model inference speed is determinant in real-time applications. For this reason, TCD leverages the TensorFlow and Caffe C++ APIs, when deploying “frozen” ML models in SDRs. As these APIs mostly assume fixed-point byte or image representations of their input data, TCD had to design the appropriate data memory format conversion functionality that allowed the integration of these deep learning frameworks with existing floating point-based SDR tools like UHD and GNURadio. All the

ML models TCD used output their results as a list of (label, probability) tuples. Further postprocessing, like thresholding to remove outputs with low probability, can then be carried out. As the ML model output has considerably lower data rate than its input, it can be conveniently converted to uncompressed text format (e.g. JSON) and sent to upper layers or devices via TCP sockets.

7.1.3 Testbed Integration

TCD makes available through a git repository the following functionality concerning ML-based RF classifier models' PHY layer and integration with upper layers:

1. Scripts and GNURadio blocks for all the preprocessing and conversion of IQ samples to the input data formats of the ML models TCD used during Year 1.
2. The integration of TensorFlow and Caffe with GNURadio for real-time inference and control through simple APIs.

Assuming that the input data conversion and SDR integration tools provided by TCD are used, to run their ML models in an SDR in real-time, experimenters just have to specify the following arguments/parameters:

- "Frozen" ML model filename and path from where GNURadio can find and load the intended classifier
- The mapping between label indices (integer value) and names (string), which will convert the ML model output code to a human-readable name
- The appropriate data pre-processing blocks that receive raw IQ samples and converts them the data format understood and expected by the specified ML model
- The channel to which the ML model outputs will be sent (e.g. TCP socket), and expected data format (e.g. JSON).

7.2 Relation to Showcase

The showcase 3 will be divided into two separate stages: (i) ML model generation, and (ii) over-the-air radio slicing coexistence.

During the ML model generation, TCD demonstrates the capabilities of the dataset collection framework to simultaneously collect IQ samples for different waveforms and RF parameters' combinations and store the associated labels/metadata. With regard to the real-time and PHY aspects of this framework, TCD will show how the designed preamble ensures the correct synchronization of the USRPs involved in the data collection process. For further details regarding the control plane aspects of this stage of the showcase, please refer to Section 6 of D4.1 [1].

During the radio slicing coexistence experiment, TCD integrates one of its "frozen" ML models with the IMEC's virtualized radio. The ML model will not only classify the types of signals present in the spectrum, but also estimate other RF specific parameters, such as frequency, bandwidth and time of arrival. Through this high-level information, it will be possible to discern which channels are occupied by radios with a MAC scheme compatible with the IMEC's virtualized radio. TCD will show the capability of their ML model to make these decisions in real time, and how such information can be efficiently remotely transferred to the IMEC's radio's decision-making module.

7.3 Risk analysis

There is not any identified risk at this stage, with regard to the PHY and higher layer integration of the proposed solutions.

7.4 Implementation plan

This section describes TCD's plans for implementing low latency wireless communication on SDRs for Year 2. We want to assess what are the most effective techniques for reducing latency on both LTE and 5G New Radio (NR). We plan to implement latency reduction techniques by extending an open-source implementation of LTE, e.g., srsLTE. Our implementation will allow an experimental evaluation between latency reduction techniques, in terms of the actual latency reduction and their drawbacks.

7.4.1 PHY Improvements

We will extend current the srsLTE platform to support a set of latency reduction techniques. This set includes employing different scheduling mechanisms to preallocate resources to subscribers, using new frame structures with reduced Transmission Time Interval (TTI), or completely scaling the LTE PHY numerology to decrease the TTI without modifying the frame structure or the MAC layer.

7.4.2 Higher Layer Protocol Integration

The proposed implementation will extend the PHY and MAC functionalities of an LTE experimentation platform to support low latency communication. Higher layers should operate in a seamless fashion but will be able to leverage the reduced PHY and MAC latency.

7.4.3 Testbed integration

We plan to provide more flexible PHY and MAC configurations for experimentation on both LTE and NR. Future experimenters will be able to create experiments with the following functionalities:

- Specify a type of frame structure depending on the type of service or traffic class.
- Change the scheduling mechanism to pre-allocate resources to subscribers.
- Support variations of the LTE PHY numerology.

8 NS-3 BASED PROTOTYPING PLATFORM FOR RAT INTERWORKING

Figure 41 below shows the anticipated Multi Radio Access Technology (Multi-RAT) system described in more detail in deliverable D2.1 [8]. The basic underlying idea is to enable researchers to do experiments on a Multi-RAT system including Wi-Fi, LTE as well as 5G. This will lead to a better understanding of the practical trade-offs when dealing with existing and new technologies.

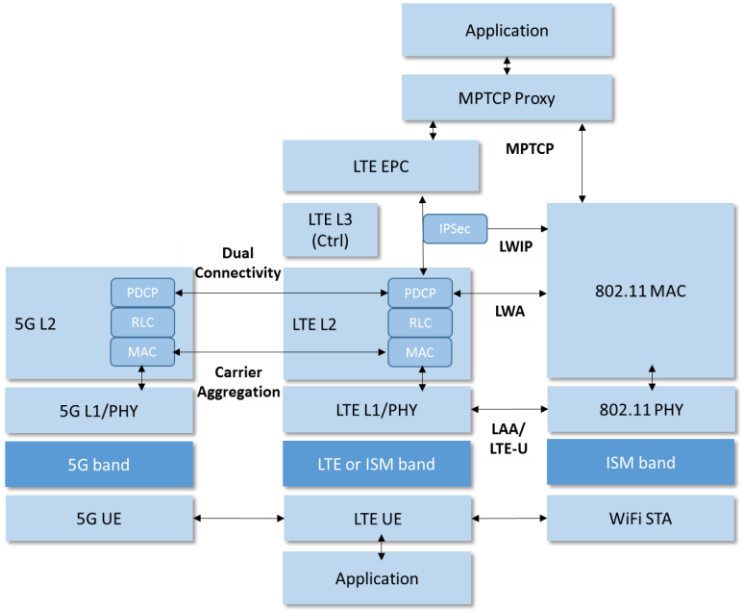

Figure 41: Anticipated Multi-RAT Platform.

For Year-1 the system depicted in Figure 42 was targeted focusing mainly on a combined Wi-Fi and LTE platform allowing for RAT interworking experiments as well as creating parallel 5G link that will be integrated in Year-2. This system will also be used in the first open calls for experimentation as described in deliverable D2.2 [7].

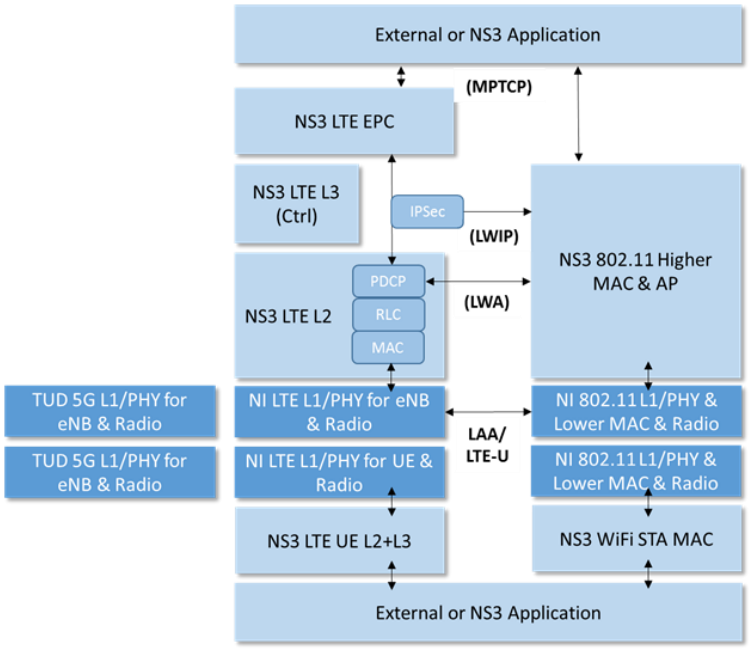

Figure 42: Targeted Year-1 Platform

In the following the status of the system implementation as well as the plans for Year-2 are described.

8.1 Implementation results

8.1.1 PHY improvements

As for improving the physical layer implementation, TUD focused on extending their 5G flexible numerology transceiver. The details of the implementation are provided in Section 5. The new transceiver can support dynamic configuration of:

- Number of subcarriers and number OFDM symbols per subframe
- Cyclic prefix length
- Modulation pulse and window length

Furthermore, NI extended its LTE Application Framework physical layer implementation with the support of multiple UEs per TTI. With this feature, multiple UEs can share the spectrum at the same time in a frequency multiplex. This is needed for flexible allocation of the available resources in multi-user transmission scenarios as intended by the ORCA project.

8.1.2 Higher layer protocol integration

As for the integration of higher layer protocols with the physical layer, NI focused mainly on connecting their 802.11 Application Framework that consists of a physical layer and the lower MAC with the NS-3 based Wi-Fi higher MAC module [22]. The basic block diagram of the NS-3 Wi-Fi module is shown in the figure below:

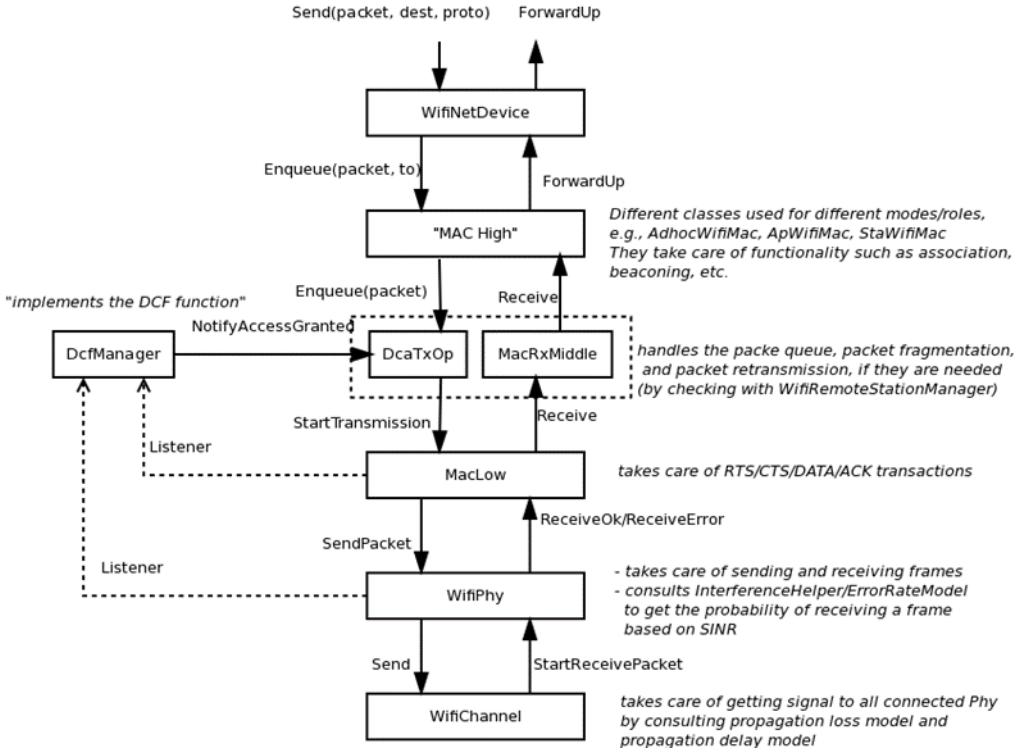

Figure 43: NS-3 Wi-Fi Module Block Diagram. [22]

Furthermore, a system sketch of the NI 802.11 Application Framework is shown in Figure [23].

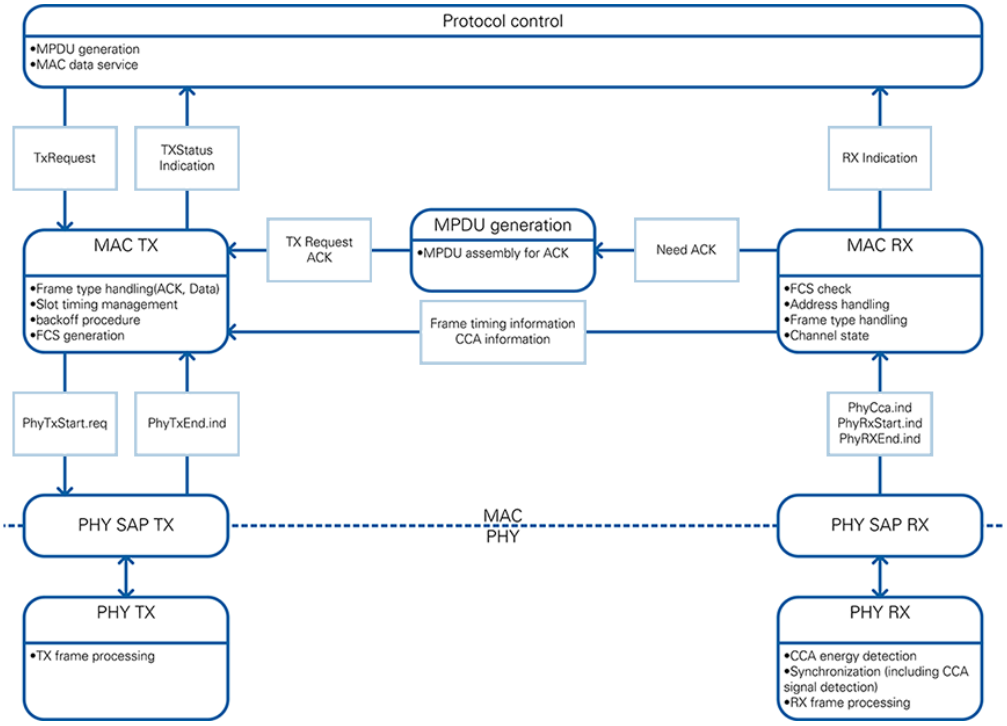

Figure 44: NI 802.11 Application Framework Module Block Diagram. [23]

In order to connect the NS-3 higher MAC with the NI 802.11 Application Framework lower MAC the following changes in the NS-3 protocol stack where required:

- Analysis of options for the interface spit between the NI Application Framework lower MAC and NS-3 Wi-Fi Higher MAC
- Implementation of the API communication protocol and mapping of the NS-3 Higher MAC parameters
- Split of the NS-3 Wi-Fi module in access point in station

Figure 45shows the block diagram of the connection that has been implemented as part of the NI work.

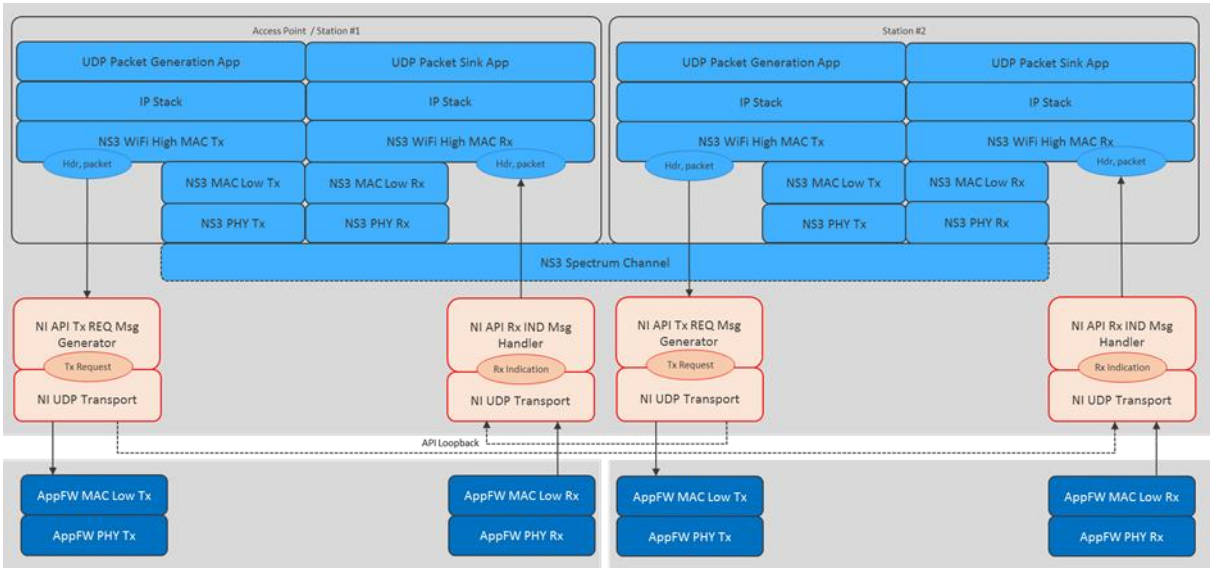

Figure 45: Block Diagram of new API Implementation.

As shown in the figure, at the transmitter side in the NS-3 Wi-Fi High MAC module a switch has been implemented to either use the legacy NS-3 simulation mode or to use the connection to the real-time capable NI 802.11 Application Framework. For the second option the NS-3 internal variables containing the specific information for the lower MAC need to be translated to the used NI API that has been specified for the general communication between MAC and lower layers. The concept builds upon the nFAPI specification [24] which is basically a proposed set of messages and procedures allowing for a reliable and efficient communication between a LTE MAC and PHY. For the modifications described here the NS-3 Wi-Fi Tx MAC configuration parameters as well as the payload data are included into a set of NI API Tx request messages. After a Tx request message is generated it will be transferred to the NI Application Framework lower MAC through an UDP connection. The NI Application Framework processes these messages according to its definition and will send signals over the antenna. At the NI 802.11 Application Framework receiver an NI API Rx indication message is created that triggers the NS-3 Wi-Fi High MAC module to process the incoming data. With this system a real-time capable end-to-end connection is possible using the application, network and higher MAC layer of the NS-3 system as well the lower MAC and PHY from the NI Application Framework.

8.1.3 Testbed integration

The described combination of the NS-3 Wi-Fi platform with the NI 802.11 Application Framework is available for the upcoming open calls for extensions and experimentation. As one core element of the existing Multi-RAT system based on NS-3 also the extension can be seamlessly used as part of the overall experimentation platform.

8.2 Relation to showcase

As described in deliverable D2.1 [8], the showcase for the NS-3 based prototyping platform is showcase 4: “Interworking and Aggregation of Multiple Radio Access Technologies”.

8.3 Risk analysis

One risk is to separate the access point and station of the NS-3 Wi-Fi module and connect it to the NI 802.11 application framework due to several modifications required in the NS-3 module. A fall-back mode to the full bidirectional infrastructure mode is to use the simpler unidirectional ad-hoc mode. A second risk is that there may be issues with when implementing certain LTE-Wi-Fi interworking options. If such an issue occurs, simpler techniques for LTE-Wi-Fi interworking could be used such as MPTCP.

8.4 Implementation plan

In Figure 41 above the targeted Multi-RAT platform is depicted. While in Year-1 the focus was to create a LTE-Wi-Fi interworking system as well as to improve the 5G PHY, the goal of Year-2 is the creation and integration of the 5G layer 1 and layer 2 as part of the Multi-RAT platform.

8.4.1 PHY improvements

The flexible 5G numerology transceiver developed by TUD will be extended in Year-2 as described in section 5.4.

8.4.2 Higher layer protocol integration

After finishing the connection of the NS-3 Wi-Fi Higher MAC with the NI 802.11 Application Framework the next step is to create an integrated LTE-Wi-Fi system that allows for interworking experiments. This is needed for open call of extension 1 as well as planned for the open call for extension

2.

Furthermore, an NI focus in Year-2 will be on developing a 5G MAC capable of interworking with a legacy LTE system. The outcome shall be a system where selected options for the interworking between these RATs can be practically investigated. This is in particular required for the open calls for experimentation.

8.4.3 Testbed integration

The NS-3 based changes are an inherent part of the Multi-RAT platform. The targeted implementation for the 5G MAC shall include options for interworking with the LTE stack using e.g. dual connectivity or carrier aggregation.

For the TUD 5G flexible numerology PHY the integration into the NI outer transceiver is planned. As the NI outer transceiver includes an L1-L2 API implementation the goal is that the TUD 5G PHY can be connected to the NI 5G MAC via this API.

9 CONCLUSIONS

In conclusion, this deliverable reports on essential data plane SDR functionality to achieve low latency and high throughput operation by allowing real-time operation. Therefore, ORCA extends the current state-of-the-art SDR capabilities by designing data-plane SDR solutions on heterogeneous hardware platforms that can combine high data rates or low latencies with fast design cycles and high versatility. Therefore, each of the partner has brought in new functionalities which can be summarized as follows:

TUD/NI summarized the physical layer capabilities of the high throughput mmWave system and described the improvements on the hardware in the loop (HALO) setup and on the higher layer data plane.

KUL described achievements on analog self-interference cancellation and the plan to employ full duplex and advanced collision detectors. Further KUL introduced physical layer improvements and experiments on distributed Massive MIMO.

TUD illustrated improvements on Generalized Frequency Division Multiplexing (GFDM) physical layer which ensures full flexibility and runtime configuration.

IMEC's described SDR improvements with focus on "PHY accelerator pool" and its control interfaces on their hybrid FPGA platform incl. flexible MAC.

TCD's introduced their signal classification framework, the general PHY functionality, its integration with upper layers for radio environment monitoring.

NI described improvements on the ns-3 based prototyping platform for RAT interworking that enables experimentation with LTE, Wi-Fi and 5G.

REFERENCES

[1] ORCA Deliverable D4.1, “First operational SDR platforms with end-to-end capabilities”, December 2017

[2] MiWaveS, Beyond 2020 Heterogeneous Wireless Networks with Millimeter-Wave Small Cell Access and Backhauling, Website, http://www.miwaves.eu, December 2017

[3] MiWaveS, JOINT DELIVERABLE D6.3-D6.4-D6.5, “mmWave Access and Backhaul Link Tests and Presentation of Final Demonstrator”, http://www.miwaves.eu/deliveraables/D6.3_6.4_6.5.html, June 2017

[4] National Instruments, “The NI TDMS File Format”, White paper, http://www.ni.com/white-paper/3727/en/, November 2017

[5] Kurose, J. F.; Ross, K. W. (2010). Computer Networking: A Top-Down Approach (5th ed.). Boston, MA: Pearson Education, 2010

[6] OpenVPN Inc., Website, https://openvpn.net, December 2017

[7] ORCA Deliverable D2.2, “Technical requirements of the ORCA test facility”, https://static.martel-innovate.com/wp-content/uploads/sites/4/2017/01/ORCA_D2.2_Final_v1.1.pdf , June 2017

[8] ORCA Deliverable D2.1, “Definition of ORCA showcases”, https://static.martel-innovate.com/wp-content/uploads/sites/4/2017/01/ORCA_D2.1_Final.pdf, March 2017

[9] WiSHFUL project, Website, http://www.wishful-project.eu, December 2017

[10] ORCA Deliverable D3.2, “First toolset for real-time SDR design and operation”, December 2017

[11] N. Michailow et al., "Generalized Frequency Division Multiplexing for 5th Generation Cellular Networks," in IEEE Transactions on Communications, vol. 62, no. 9, pp. 3045-3061, Sept. 2014.

[12] M. Danneberg et al., “Short Paper: Flexible GFDM implementation in FPGA with support to run-time reconfiguration” in Proceedings of VTC Fall'15

[13] eWINE Deliverable D4.1, “Optimization and negociation algorithms design and implementation” https://ewine-project.eu/wp-content/uploads/eWINE_D4.1_Final_v1.0.pdf

[14] CREW project, Website, http://www.crew-project.eu, December 2017

[15] WiSHFUL Deliverable D3.4, “Second operational radio control software platform”, http://www.wishful-project.eu/sites/default/files/images/review/WiSHFUL_D3.4_CNIT_R_PU_2016-12-23_v1.0.pdf, December 2016

[16] ORCA Deliverable D6.1, “FED4FIRE compliance of ORCA testbeds and initial integration of SDR platforms”, December 2017

[17] IETF, RFC 7554, “Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT)”, https://tools.ietf.org/html/rfc7554, May 2015

[18] Felix Wunsch, Francisco Paisana, Sreeraj Rajendran, Ahmed Selim, Pedro Alvarez, Sebastian Muller, Sebastian Koslowski, Bertold Van den Bergh and Sofie Pollin, “DySPAN Spectrum Challenge: Situational Awareness and Opportunistic Spectrum Access Benchmarked”, IEEE Transactions on Cognitive Communications and Networking (TCCN), 2017.

[19] Francisco Paisana, Ahmed Selim, Maicon Kist, Pedro Alvarez, Justin Tallon, Christian Bluemm, Andre Puschmann and Luiz DaSilva, “Context-aware cognitive radio using deep learning”, IEEE Dynamic Spectrum Access Networks (DySPAN), 2017.

[20] Alex Krizhevsky, Ilya Sutskever, George E. Hinton, “Imagenet classification with deep convolutional neural networks”, Advances in neural information processing systems, 2012.

[21] Ahmed A. S. Selim, Francisco Paisana, Jerome A. Arokkiam, Yi Zhang, Linda Doyle, Luiz A. DaSilva, “Spectrum Monitoring for Radar Bands using Deep Convolutional Neural Networks”, IEEE Global Communications Conference (GLOBECOM), 2017.

[22] ns-3, The ns-3 Wi-Fi Module Documentation, Design Documentation, https://www.nsnam.org/docs/models/html/wifi-design.html, January 2016

[23] LabVIEW Communications 802.11 Application Framework 1.1 White Paper, online available: http://www.ni.com/product-documentation/52533/en/, September 2016

[24] Small Cell Forum, FAPI and nFAPI specifications, http://www.scf.io/en/documents/082_-_nFAPI_and_FAPI_specifications.php, May 2017