Real-time Prototyping of 5G Software Defined Networks

Version 4

December 2, 2016

Nicola Michailow, PhD
Staff Development Engineer
National Instruments
Dresden, Germany

Vincent Kotzsch, PhD
Sr. Development Engineer
National Instruments
Dresden, Germany

Douglas Kim, PhD
Sr. Product Marketing Manager
National Instruments
Texas, United States
## Table of Contents

Introduction ................................................................................................................................. 3  
   The Future of 5G and Wireless Network Prototyping .............................................................. 3  
   Prototyping Requirements & Application Use Cases ............................................................... 5  

The NI MAC/PHY Prototyping Platform ..................................................................................... 7  
   Host Controller ....................................................................................................................... 7  
   FPGA & RF Tranceivers ......................................................................................................... 7  

System Architecture .................................................................................................................. 9  
   System Hardware Configuration ............................................................................................ 9  
   System Software Architecture ............................................................................................. 10  

The LTE Protocol Stack ............................................................................................................ 13  

Network Simulator 3 .................................................................................................................. 17  

The LTE Application Framework ............................................................................................. 19  

NI L1-L2 UDP Interface .......................................................................................................... 21  

Conclusion .................................................................................................................................. 29  

Ordering & Purchasing Information .......................................................................................... 30  

References ................................................................................................................................. 30
Introduction

The Future of 5G and Wireless Network Prototyping

Consumer demand for greater bandwidth continues to outpace the capacity available in present-day networks. Even as wireless service providers upgrade their networks to 4G LTE and LTE-A, data demands are projected to still go largely unmet. Therefore, next generation wireless communication networks will need to evolve to overcome the limits of current systems by serving a number of novel use cases with very diverse requirements including the following:

- Enhanced multimedia broadband (eMBB) for massive mobile communications
- Massive machine type communication (mMTC) with focus on scalability and connectivity
- Ultra-reliable and low latency communication (URLLC) for real-time and safety-critical applications

New technologies have been proposed to address these capacity and performance demands [1, 2, 3, 4, 5] including the following:

- Massive MIMO
- mmWave
- Virtualization of radio access functionality in the Cloud (CRAN)
- Flexible radio access network
- New radio access technologies (RAT), e.g., new waveforms and flexible frame structures

Shown in Figure 1 is an illustration of the evolution 4G heterogeneous networks will undergo in order to serve the diverse 5G applications of the future. In designing the networks of tomorrow, a number of challenges will need to be addressed. For example, in order to meet the strict low latency performance requirements of network virtualization and ultra dense deployments, all layers of the communication system including the core network must be optimized in a joint manner. Furthermore, as cell sizes decrease and the number of deployments increase, interference management becomes an even greater challenge. As such, interference coordination techniques such as eICIC, FeICIC and CoMP must evolve further to support the far denser networks of tomorrow. Lastly, for SDN wherein network resources are dynamically reconfigured as traffic conditions change, control and data planes must be properly abstracted in order to conceal the complexities of network management associated with such diverse applications.

It is widely accepted that PC based simulations alone are not sufficient to fully realize the 5G vision due to the challenges described above in addition to many others. Rather, working prototypes of the various network elements, from the physical to application layers, must be created to determine what can actually meet the demanding implementation requirements not considered in simulations. National Instruments has embraced this reality and offers a number
of hardware and software solutions for wireless communication systems prototyping that reduce the time and cost needed to go from concept to reality.

NI and some of its industry and academic partners have successfully demonstrated a number of 5G prototyping systems including the world’s first 128-antenna Massive MIMO communication system that was used by the universities of Bristol and Lund to break records in spectral efficiency [6]. NI also partnered with Nokia in demonstrating a dual channel, 10 Gbps wireless link at 73 GHz with the world’s first mmWave software defined radio (SDR) [7]. In addition to contributions to massive mobile communications, NI also offers hardware and software solutions that are capable of performing network level research for 5G applications such as CRAN and SDN. Unlike simulations that inaccurately model the many complexities of real-world networks and may take a very long time to complete, the NI platform provides a complete, real-time physical layer that enables researchers to conduct real-world, over-the-air experiments.

The NI LTE Application Framework [8] is a real-time LTE physical layer reference design that can be combined with the widely used LENA stack and Network Simulator 3 (NS-3) [9] to provide a rich set of PHY, MAC, and network capabilities with which researchers can rapidly begin experimenting and innovating faster. Furthermore, as the source code for every layer of the overall system is available, researchers can customize every aspect of the system to suit their needs. And as the various system layers are tightly integrated with the NI SDR baseband and RF hardware platform, reliable and real-time execution is guaranteed. From the high-speed signal processing of RF samples at the PHY layer to the routing of packets at the network layer, NI offers the tools necessary to implement fully functional wireless communications networks.

The NI prototyping system has been used in the European FP7 project CROWD to successfully demonstrate an SDN algorithm for interference management within dense heterogeneous

Fig. 1: Heterogeneous networks today and an outlook for the future (from [15])
deployments of cellular wireless networks [10]. Using Distributed Mobility Management (DMM), the prototyping system employs MIH IEEE 802.11u to demonstrate the successful handover of a mobile UE as it traverses across multiple LTE and WiFi networks under varying interference and traffic conditions. The CROWD project has also successfully demonstrated eICIC almost blank subframe transmissions with multiple base stations and terminals [11]. Device-to-device (D2D) communication was also successfully demonstrated using LTE as a primary cellular link and WiFi for inter-UE communication [12]. Furthermore, as the source code for the entire stack from the PHY layer to the network layer is available and modifiable, the prototyping system can also be used in the following research areas:

- Latency reduction optimizations of the full system stack
- Interference coordination and cancellation algorithms (CoMP, eICC, etc.)
- Flexible numerologies and protocols for new waveforms
- Narrowband IoT protocols
- New SDN and network slicing
- CRAN and functional splits

Though the future of 5G holds a great deal of promise, the path to commercial deployment is paved with unprecedented challenges. Proposed technologies and concepts, while promising in theory, must go beyond simulations to real-world, working prototypes in order to determine what performance gains can be realized. NI is committed to the pursuit of this goal by offering a diverse and capable line of SDR platforms that enable researchers to rapidly prototype at all levels of the communication stack. With a rich set of hardware, software, and IP, researchers are equipped with the tools necessary to turn their visions into reality.

To aid researchers in this journey towards 5G, NI offers flexible SDR prototyping solutions that enable scientists and engineers to make rapid progress with the development of new algorithms. The LTE Application Framework [8] has proven a useful starting point in the exploration of 4G enhancements [13] and proposed 5G PHY techniques [14]. In this document, we present the NI MAC-PHY research system as an example that illustrates how a third-party upper layer protocol stack or MAC can be incorporated with the LTE physical layer implementation to facilitate real-time, over-the-air transmission.

**Prototyping Requirements & Application Use Cases**

Computer-based simulations, while useful in generating nominal performance benchmarks of wireless communication systems, often make inaccurate assumptions of various system model components that largely limit the ability to predict how an actual system will behave in practice. Therefore, functional prototypes that operate over real-world wireless channel conditions in real-time are essential in order to determine the feasibility of new technologies and the extent to which their promised gains in performance can be achieved. Such mandatory prototypes and field trials are necessary in order to gain broader acceptance of next generation technologies within the wireless industry.
Although necessary, prototyping has traditionally presented challenges that stem from the many complexities associated with the various layers in the network communication stack including the PHY, MAC, and Network layers. Furthermore, each layer traditionally has required the use of highly disparate development tools and the expertise of skilled researchers and engineers to perform tasks such as programming FPGAs and designing RF circuits. Consequently, such prototyping design cycles are oftentimes overly lengthy and costly.

In addressing the challenges in prototyping real-time wireless communication systems, NI offers a number of SDR prototyping platforms with capabilities that satisfy a variety of hardware and software requirements for the following use cases and specifications:

- Prototyping of base stations and terminals
- Support of RF frequencies both above and below 6 GHz
- Support of bandwidths ranging from 40 MHz to 2 GHz
- Scalable CPU and FPGA architectures for additional processing resources
- Rich collection of customizable software reference designs for LTE, 802.11, and other 5G technologies
- Unified and comprehensive software design tools supporting both CPUs and FPGAs

With such tools and capabilities, the NI SDR prototyping platform can be used for conducting real-time, over-the-air experiments of heterogeneous networks. As an example, this paper describes a prototyping platform that combines the NS-3 open source upper layer protocol stack with the NI SDR LTE MAC and PHY layers. In general, this approach of interfacing open source tools with the NI SDR prototyping platform can be extended to access technologies other than LTE such as 802.11 and other 5G eMBB technologies. Examples of other open source stacks include Open Air Interface (OAI), srsLTE, OPNET, and Amarisoft.

For the system described in this paper, the software architecture is divided across CPU and FPGA. Using LabVIEW Communications System Design Suite, the NI Linux Real-Time OS can be deployed on the CPU of a PXI controller. NI Linux RT is based on the widely used Linux OS, so many open source libraries and tools used in programming embedded systems are also supported on NI CPU targets. That provides greater flexibility and enables the reuse of existing code that customers may have, thereby accelerating the development of wireless communication system prototypes. It also gives access to the vast amount of open source that is available on the Internet.

NI Linux RT makes possible the real-time execution of the upper layer stack provided by NS-3 on the CPU while the more intensive signal processing associated with the PHY layer executes on the FPGA. By combining the widely used LTE EPC Network Simulator (LENA) stack of NS-3 and the NI LTE Application Framework, researchers and engineers are able to reuse their existing network layer algorithms without the need for developing their own LTE PHY layer, thereby greatly reducing the overall time required to create a fully functional network prototyping
platform. Therefore, the benefit for researchers and engineers who do not possess extensive experience in developing real-time wireless communication systems, particularly with regard to FPGA programming, is that they can use the NI SDR prototyping platform to reduce both development time and cost in order to create a complete end-to-end system more rapidly, which increases productivity and shortens time to results.

The NI MAC/PHY Prototyping Platform

A system level diagram of the NI MAC/PHY prototyping system is shown below in Figures 2a and 2b. Shown in Figure 2a is a high-level diagram depicting the functional components of the LTE eNB and UE consisting of the NS-3 upper layer stack and the NI LTE Application Framework real-time PHY layer, which are joined together with high-speed UDP interfaces. Figure 2b depicts the hardware architecture of the prototyping system and a mapping of the various PHY layer and upper layer software components to the individual hardware devices. The following is a description of the various hardware devices within the system and their respective software components.

Host Controller

The PXie-8135 controller is equipped with a 2.3 GHz Intel quad-core i7 processor and up to 16 GB of RAM. The controller is also equipped with a high-speed PXI Express bus interface for high-throughput data transfers with other devices on the PXI Express backplane such as FlexRIO FPGA modules. The CPU runs the NI Linux RT OS that provides greater determinism for time critical operations such as those associated with NS-3 and the Host interface of the LTE Application Framework. This ensures that NS-3 executes in real-time and is able to complete operations on a 1 ms subframe basis, as dictated by the LTE standard.

FPGA & RF Transceiver

The NI USRP RIO is a complete SDR solution equipped with a Xilinx Kintex 7 FPGA, dual high speed ADC/DAC’s, and two RF transceivers. The USRP RIO comes in a number of different models which support varying carrier frequencies and bandwidths options that can be used for different applications. For the NI MAC/PHY prototyping platform, the LTE PHY layer signal processing is implemented on FPGA to ensure real-time execution of the system. Moreover, control of the RF transceiver hardware is also implemented on the FPGA.
Additional information on the implementation of the various system layers is listed below in Table I. For both the PHY and MAC layers, listed are the required functionalities describing the behavior of each respective layer and their corresponding software implementation and hardware targets. A similar set of information is also provided for the interface between the MAC and PHY layers, which in this case joins the NS-3 upper layer stack and the LTE Application Framework real-time PHY layer.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Required Functionality</th>
<th>Software Implementation</th>
<th>Hardware Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Upper layer protocol stack</td>
<td>NS-3, 3rd party protocol stack</td>
<td>X86 general purpose processor based platform</td>
</tr>
<tr>
<td></td>
<td>• Flow control via state machines</td>
<td>• Includes MAC and upper layers</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Add redundancy to increase reliability</td>
<td>• Runs on real-time operating system</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Memory operations</td>
<td>• Flexible network configurations</td>
<td></td>
</tr>
<tr>
<td>MAC/PHY Interface</td>
<td>High speed exchange of control &amp; data information</td>
<td>Lean NI L1-L2 interface</td>
<td>X86 general purpose processor based platform</td>
</tr>
</tbody>
</table>
PHY | Over-the-air wireless link  
| Sample-by-sample streaming operations  
| Computationally heavy operations (FFT, decoding, etc.) | NI LTE Application Framework | FPGA-based SDR platform PXI or USRP-RIO

Table I: Implementation information on the various layers of the NI MAC/PHY prototyping system

System Architecture

In this section, descriptions of two common hardware configurations of the LTE MAC/PHY prototyping system are provided to illustrate how such systems are configured and used in practice. A description of the software architecture of the single eNB and UE use case is also presented to further illustrate how the various software components are distributed across the various hardware targets.

System Hardware Configuration

The following hardware components are required for the LTE MAC/PHY prototyping system:

- PXIe-1082 chassis
- PXIe-8135 controller
- USRP RIO (e.g., USRP-2943R) with the PXIe-MXI Express Interface Kit

The primary hardware setup is shown below in Figure 13. Please note that all hardware configurations require a Microsoft Windows PC to control the system. It is recommended that the NS-3 MAC execute on the same PXI controller that runs the NI Linux RT OS to guarantee real-time performance. The LTE PHY executes on an FPGA target on a cabled USRP RIO device.
**System Software Architecture**

The following software is required to use the LTE MACPHY prototyping system:

- [LabVIEW Communications System Design Suite 2.0 with NI Linux RTOS](#)
- [LTE Application Framework 2.0](#)
- [Open source network simulator NS-3](#)

Shown below in Figure 14 is a system diagram that depicts each of the hardware targets for a single eNB and single UE configuration and the corresponding software components executing on them. The two PXI systems shown to the far left and far right of the diagram represent the eNB and UE, respectively. Within each of the devices are block diagrams representing the various hardware elements within the PXI chassis, such as the controller and FPGA module, and the various software components executing within each target. The NI Linux RT PXI controllers run NS-3, the Host interface of the LTE PHY, and the L1-L2 API linking the two together, and the FPGA module runs the real-time LTE PHY.

At the center of Figure 14 is a Windows machine used to control both the eNB and UE. The Windows machine is connected to each target via Ethernet and performs tasks such as deploying and running project executables on each PXI target, configuring the overall network of devices, and monitoring the behavior of each PXI system via plots and terminal outputs.

Although not shown, the system can be scaled to support additional eNBs and UEs to form larger LTE wireless networks where each device possesses similar hardware and software architectures like that shown above. And because the source code for every device is open and modifiable, each device can be customized with unique upper layer stack protocols to support capabilities such as the rapid handoff of UEs across neighboring cells or novel interference...
cancellation algorithms. The flexibility and hardware capabilities of the LTE MAC/PHY prototyping platform enable it to be used to conduct real-time over-the-air trials for the evaluation of 5G wireless networks.

Shown in Figure 15(a) is the user interface of the MAC/PHY prototyping system for the eNB. Note that the highlighted MCS parameter and RB allocation can be configured by the NS-3 MAC during runtime. Further, the execution of the NS-3 binary can be started and stopped from within the LabVIEW GUI and its output is also piped out to the GUI for additional convenience.

The GUI shown in Figure 15(b) is used to configure the UDP network parameters of the system, i.e. the IP address of the physical machines and the ports for transmission and reception. Also, there is the option to set loopback flags to bypass the PHY in the DL and/or UL directions, which under the hood also reflects the IP configuration. One important feature in this GUI is that the required configuration scripts to start the NS-3 application on the Linux machines properly are generated automatically and distributed to the Linux systems. Hence, there is no need to edit config files manually in a text editor.
The LTE Protocol Stack

Shown below in Figure 3 is a diagram depicting the various system components of a typical 3GPP LTE network. It is composed of the radio components of the E-UTRAN access network and the core network, referred to as the System Architecture Evolution (SAE) or evolved packet core (EPC) network [16]. The combination of both network elements is referred to as evolved packet system (EPS). The elements of an EPS network and their main functions are summarized in Table II.

Fig. 15: The LTE MACPHY prototyping system graphical user interface
As depicted in Figure 3, the LTE network architecture consists of the control and user planes. The former conveys control information that is used to maintain the communication link and the latter transmits varying application-specific payload data to and from the user equipment. LTE is a packet oriented all-IP network protocol with connections between the UE and the eNB referred to as access stratum protocols. Additional protocols that are transparent to the eNB (e.g., between a UE and a MME) are referred to as non-access stratum (NAS) protocols. The routing of data packets between the public data network (PDN) and a UE is accomplished via bearers, which can be configured with different quality of services (QoS) parameters depending on traffic requirements. The interfaces between the various network components in Figure 3 can be treated as the following reference points:

- LTE-Uu: Radio access link
- S1-MME: Control plane interface between the eNB and the MME
- S1-U: User plane interface between the eNB and the S-GW
- S11: Bearer control interface
- X2: Connection link among different eNBs
The LTE protocol stack consists of multiple communication layers needed to establish a link between the EPC and the E-UTRAN. Table III lists the various LTE protocol layers and a brief description of each layer’s functions. Shown in Figure 4, are the various channel mappings between the different LTE protocol layers.
<table>
<thead>
<tr>
<th>LTE Protocol Stack Layers</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NAS</strong> (Non-access stratum, only control plane)</td>
<td>• EPS bearer management</td>
</tr>
<tr>
<td></td>
<td>• Signaling between core network and UE</td>
</tr>
<tr>
<td></td>
<td>• Authentication and UE attachment</td>
</tr>
<tr>
<td></td>
<td>• Mobility and session management</td>
</tr>
<tr>
<td><strong>RRC</strong> (Radio Resource Control, only control plane)</td>
<td>• RRC connection management</td>
</tr>
<tr>
<td></td>
<td>• Broadcast of system information (MIB, SIB, etc.)</td>
</tr>
<tr>
<td></td>
<td>• Configuration, maintenance and release of Signaling and Data Radio Bearers (SRBs and DRBs)</td>
</tr>
<tr>
<td><strong>PDCP</strong> (Packet Data Convergence Protocol)</td>
<td>• Ciphering</td>
</tr>
<tr>
<td></td>
<td>• Header compression</td>
</tr>
<tr>
<td></td>
<td>• Integrity protection</td>
</tr>
<tr>
<td></td>
<td>• Handover support</td>
</tr>
<tr>
<td><strong>RLC</strong> (Radio Link Control)</td>
<td>• Segmentation and concatenation of SDUs</td>
</tr>
<tr>
<td></td>
<td>• Packet reordering of PDUs</td>
</tr>
<tr>
<td></td>
<td>• Transmission modes: Transparent Mode (TM), Unacknowledged Mode (UM), Acknowledge Mode (AM)</td>
</tr>
<tr>
<td></td>
<td>• Error correction through HARQ</td>
</tr>
<tr>
<td><strong>MAC</strong> (Medium Access Control)</td>
<td>• Error correction through HARQ</td>
</tr>
<tr>
<td></td>
<td>• Multiplexing and mapping from logical to transport channels</td>
</tr>
<tr>
<td></td>
<td>• Channel prioritization</td>
</tr>
<tr>
<td></td>
<td>• Dynamic resource scheduling and multiplexing of MAC SDUs to transport blocks</td>
</tr>
<tr>
<td></td>
<td>• Random access control</td>
</tr>
<tr>
<td><strong>PHY</strong> (Physical Layer)</td>
<td>• Over-the-air link transmission</td>
</tr>
<tr>
<td></td>
<td>• Data and pilot frame structure</td>
</tr>
<tr>
<td></td>
<td>• Modulation and coding</td>
</tr>
<tr>
<td></td>
<td>• Time and frequency synchronization</td>
</tr>
<tr>
<td></td>
<td>• Channel estimation and data equalization</td>
</tr>
</tbody>
</table>

**Table III:** Layers of the LTE protocol stack
Network Simulator 3 or NS-3 is an open source simulation tool created by the NS-3 Project that is used to simulate large IP networks including LTE [17]. The software is written in C++ and is free under the GNU GPLv2 license. The NS-3 source code is also fully open and can be modified to support new protocols that are unavailable in the current distribution. Among the various building blocks needed to simulate fixed IP networks, the NS-3 Project also includes a model of the 3GPP LTE EPS consisting of an eNB and UE protocol stack, and an EPC like that shown in Figure 3. The NS-3 3GPP LTE models support complete end-to-end IP connectivity, which includes reference applications for generating traffic and exposes interfaces to communicate with external applications, thereby, enabling the use of live traffic with the simulator.

The NS-3 LTE module includes a C++ implementation of LTE layers 1-3 (RRC, PDCP, RLC, MAC, PHY emulation) for the eNB and UE and supports the use of multiple eNBs and UEs connected to the core network (MME, SGW, PGW). Furthermore, NS-3 supports X2 links between multiple eNBs that enable handover procedures and interference coordination techniques. Other network functionalities supported by NS-3 that are essential in the simulation of 5G wireless networks include the following [18]:

---

Fig. 4: Channel structure in the LTE downlink
• Radio resource management
• QoS-aware packet scheduling
• Inter-cell interference coordination
• Dynamic spectrum access

NS-3 also includes an implementation of the FemtoForum MAC Scheduler API, which simulates scheduling decisions on the use of radio resources for UEs requesting access to the core network. Moreover, the FemtoForum MAC Scheduler API supports an interface to external applications and includes a basic HARQ model with corresponding error models.

NS-3 is designed as an event-driven system that operates based on queues with scheduled events that are executed at discrete time points. One main feature of the NS-3 simulator is the option to schedule events in a real-time mode. This is critical for error-free, real-time stack operations as is the case with the NI LTE MAC/PHY prototyping system described in this document. Furthermore, because LTE operates on a 1 ms subframe time boundary, NS-3 must operate under this time constraint and does so successfully for the given implementation of the prototyping system.

Because NS-3 is designed to simulate large networks at a system level, the PHY models included in the simulation tool are highly simplified and largely limited in their representation of actual RF devices and real-world over-the-air channels. For this reason, NI has created the LTE MAC/PHY prototyping system, which combines the NS-3 LTE protocol stack model with an LTE PHY layer deployed on an SDR that enables the real-time over-the-air transmission of actual RF signals over real-world wireless channels. Thus, actual real-time over-the-air experiments can be conducted to more closely represent the behavior of real-world wireless networks and better understand the feasibility and promise of next generation 5G wireless networks. More information on the NI LTE PHY layer reference design, known as the LTE Application Framework, is available in the following section.

Table IV lists the key features for each of the NS-3 LTE protocol layers. All protocol layers are available for both the eNB and UE. A detailed description of the various NS-3 protocol layers is also available in [18, Chapter 20].
### Table IV: Features supported by NS-3

**The LTE Application Framework**

The NI MACPHY prototyping system uses the LabVIEW Communications LTE Application Framework, which serves as a fully real-time, advanced FPGA-based LTE PHY layer reference design that supports over-the-air communications [8]. The LTE Application Framework Host and FPGA source code is fully accessible and can be modified to support new algorithms or protocols that are unavailable in the current implementation. Built in [LabVIEW Communications](http://www.ni.com/labviewcommunications), the LTE Application Framework includes the following features from Release 10 of the 3GPP LTE standard:
• Uplink and downlink transmission with 20 MHz bandwidth in TDD and FDD modes
• LTE-compliant channel encoding and decoding
• Data channels (PDSCH, PUSCH) and a simplified control channel (PDCCH)
• Achievable data rates up to 75 Mbps
• Reference symbols: CRS, UERS, PSS, SRS

With respect to the receiver, the LTE Application Framework includes the following features:

• RF impairment correction
• Automatic gain control (AGC)
• Time and frequency synchronization
• Cell-specific and UE-specific channel estimation
• Channel equalization of data symbols

With such features, the LTE Application Framework enables researchers to rapidly prototype new algorithms that can be used to enhance modern-day 4G systems or future 5G systems. For example, using the LTE Application Framework and USRP RIO devices, Samsung created UE emulators that interoperated with its own custom FD-MIMO base station to successfully demonstrate the company’s new 3D beamforming technology that has since been introduced into the LTE standard [26].

Shown below in Figure 5 is an overview of the functional split between the host and FPGA targets and the communication interfaces between the various components of the NI MAC/PHY prototyping system. For the host or CPU, the various PHY layer-related functions are shown in red and functions related to NS-3 are shown in green. Note the L1-L2 API highlighted in blue that links control and data information between the MAC and PHY layers. More information on the MAC/PHY interface that enables communication between NS-3 and the LTE Application Framework is available in the following section. For additional information on the software architecture of the LTE Application Framework host and FPGA designs, please refer to the detailed technical white paper in [8].
Fig. 5: Conceptual overview of the MAC/PHY system: The PHY layer-related blocks are marked with red, while green denotes MAC functionality.

**NI L1-L2 UDP Interface**

With the MAC and PHY layers of the NI LTE MAC/PHY prototyping system introduced in the sections above, we now describe the L1-L2 message interface that enables the rapid transfer of control and user plane data between the protocol stack and the PHY layer. This interface is implemented as a flexible UDP based API that consists of two components, one associated with upper layer processes related to NS-3 and the other associated with the LTE PHY layer implemented in LabVIEW Communications.

A high-level overview of the system is given in Figure 2(a). In general, the MAC and PHY are separate entities that communicate via UDP, wherein communication between the eNB and UE PHY layers occur in real-time and over the air. New real-time capabilities in LabVIEW Communications 2.0, in combination with the NI Linux RT OS, ensure that the strict 1 ms LTE timing requirement for the combined MAC/PHY operations is met, including the exchange of control and user plane data between the two layers.
Figure 6 illustrates the most significant operational change that has been made within the NS-3 system. By default, the NS-3 stack does not include a PHY implementation but relies on PHY abstraction to circumvent computationally complex operations like FFT and equalization procedures. This means various physical channels, e.g., PDSCH and PDCCH, are emulated. The fundamentally new feature of the LTE MAC/PHY prototyping system is the connection of the NS-3 upper layer protocol stack with the NI LTE Application Framework via a dedicated L1-L2 API. With this system, a fully real-time, end-to-end, over-the-air communications link can be utilized rather than just an emulated wireless channel that inaccurately models the behavior of a real-world communication system.

From a source code perspective, the main modifications to NS-3 include the following:

- Modification of the stack to operate in real-time (1 ms TTI timing in LTE)
- Creation of an API to link NS-3 to the PHY layer
- NS-3 configuration to start in either eNB or UE mode
A system level diagram of the default LTE stack in NS-3 and the aforementioned NI API changes are shown in Figure 7. The majority of the modifications to the NS-3 source code begin at the PHY level with modules such as LteEnbPhy, LteUePhy and LteSpectrumPhy which are described below:

- **LteEnbPhy StartSubFrame**
  - In order to account for the MAC to PHY delay the MIB transmission is sent \( n \) TTIs earlier whereas \( n \) is configured through `enbMacToChannelDelay`
  - In order to process received messages from the uplink the function `NIL1StartRxDataCtrlFrameStandalone` is called in eNB mode
  - In order to create transmit messages for the downlink the function `NIL1StartTxDataCtrlFrameStandalone` is called in eNB mode
  - In order to enable a better synchronization between the NS-3 stack and the PHY implementation transport blocks for every TTI are generated even if no payload data is available

- **LteUePhy SubframeIndication**
  - In order to account for a better synchronization of the SFN at eNB and UE the SFN at the UE is synchronized to the eNB SFN included in the MIB
  - In order to process received messages from the downlink the corresponding function `NIL1StartRxDataCtrlFrameStandalone` is called in UE mode
  - In order to create transmit messages for the uplink the corresponding function `NIL1StartTxDataCtrlFrameStandalone` is called in UE mode

- **LteSpectrumPhy NIL1StartTxDataCtrlFrameStandalone**
  - Distinguish between the eNB downlink and UE uplink cases
  - Handle control and payload messages separately, e.g. in `NIL1StartTxDLCtrlFrame`, `NIL1StartTxULCtrlFrame` and `NIL1StartTxDataFrame`
  - For control messages handle specific message types, e.g. RAR and MIB messages in downlink
  - Create an NI API Tx request message through the function classes defined in `ni-l1-api`
  - The NI API message is sent to the PHY via an UDP socket

- **LteSpectrumPhy NIL1StartRxDataCtrlFrameStandalone**
  - Receive the NI API message sent by PHY from the UDP socket
  - Unpack the NI API Rx indication message through the function classes defined in `ni-l1-api`
  - Distinguish between the eNB uplink and UE downlink cases
  - Handle control and payload messages separately, e.g. in `NIL1StartRxDLCtrlFrame`, `NIL1StartRxULCtrlFrame` and `NIL1StartRxDataFrame`
  - For control messages handle specific message types, e.g. RACH Preamble or DL CQI in uplink
  - Put control and payload data messages into the Rx queues for processing in next TTI
The following changes have been introduced to the NS-3 upper layers:

- **LteHelper Attach**: Point to NI API functions
- **LteRrcProtocolReal SetUeRrcSapProvider**: Add virtual UE if the NS-3 is started as eNB
- **LteUeMAC SendRaPreamble**: Include MAC to Channel delay for the RA RNTI calculation for RA preamble
- **LteEnbMAC DoSchedDlConfigInd**: Include MAC to Channel delay for the RA RNTI calculation for RA response
- **RrFfMacSchedule**: Changes for the external scheduler interface

Furthermore, additional real-time logging print outs have been enabled in dedicated log files for the eNB and UE which can be used for offline debugging purposes. There are also several test and debug modes available. First, NS-3 can be executed in a mode that does not include the NI API modifications. This is useful to verify that the source code compiles correctly on the target system. Additionally, a ‘loopback mode’ mode has been implemented for the NI API, which connects the NS-3 eNB and the NS-3 UE directly via the UDP interface. This is particularly useful if an over-the-air transmission including the PHY is not desired or possible, e.g. during initial tests of new NS-3 modifications and features. The loopback mode can be enabled for either uplink, downlink or both. When using the NI API loopback, the **NIL1StartTxDataCtrlFrameStandalone** function directly generates Rx Indication packets instead of Tx Request packets. These packets will then be received by the **NIL1StartRxDataCtrlFrameStandalone** directly without going through the NI PHY.

Shown in Figure 8(a) is an illustration of the MAC-PHY communication procedures. NS-3 uses the Tx Request (TX REQ) message to create messages that contain information from the MAC that are required by the PHY to operate in a scheduled manner. With the Rx Indication (RX IND) message, the PHY provides the MAC with the successfully received data and necessary control information. These and other NI API function prototypes that represent the messages linking the MAC and PHY layers are included in the NS-3 C++ ni-I1-api class and are described as follows:

- **Tx Request EncodeMessage**: Encode Tx REQ message for uplink or downlink
- **Rx Indication EncodeMessage**: Encode Rx IND message (only used in L1-L2 loopback mode)
- **Rx Indication DecodeMessage**: Decode Rx IND message for uplink or downlink

A UDP message-based interface is used to connect the NS-3 MAC with the PHY. In general, the interface consists of two basic message types with specific content that enable a flexible communication link between the MAC and the PHY. The detailed structure of the messages is depicted in Figure 8. The TX REQ consists of a message header, the number of packets in the
message, and the respective DCI and payload for each packet. Up to N packets can be included in one message. The payload itself has an individual header and a body, where the length of the body of the Nth payload is $M_n$. In practice, the total length of an API message is limited by the UDP maximum safe size. All elements in the message are specified as unsigned 32-bit integers with the exception of the payload being unsigned 8-bit integers. Likewise, the RX IND consists of a message header, the number of packets in the message, the CQI status, and the payload.

The LabVIEW component of the L1-L2 interface is an extension that has been incorporated in the eNB and UE PHY layers. It runs in LabVIEW Communications and executes on the host. As such, modifications of the FPGA code are not necessary. Based on the methods described in the previous section, Figure 9 illustrates how the downlink control path, i.e., the DCI elements, is exposed in the LabVIEW Communications host application. Note that the data path is handled
separately and hence included in the picture. Users can easily modify and pass these parameters from the MAC to the PHY by connecting the respective wires. Additional parameters such as transmit power and center frequency can be included in the payload of the API. These modifications do not require changes to the FPGA design.

Fig. 9: Structure of the L1-L2 interface control messages exposed in LabVIEW Communications

A conceptual overview of the UE host processes is shown in Figure 9. Note that although not explained in detail in this white paper, the same structure applies for the eNB.

Fig. 10: Conceptual overview of the LabVIEW host side of the L1-L2 API (for UE)

The L1-L2 interface is initialized with the rest of the LTE code in a central module at the beginning of program execution, and a common cleanup takes place before program termination. Processes 1 to 8 are part of the LTE Application Framework that correspond to PHY layer functions. The LabVIEW Communications component of the L1-L2 API includes processes 9
Processes 9 and 10 handle the PHY-to-MAC part of the interface including downlink reception. Correspondingly, processes 11 and 12 handle the MAC-to-PHY component of the interface such as uplink transmission. Lastly, processes 13-15 enable the start, stop, and monitoring of the third-party MAC, i.e., the NS-3 binaries, from within the LabVIEW GUI.

<table>
<thead>
<tr>
<th>Process ID</th>
<th>Functionality</th>
</tr>
</thead>
</table>
| 9          | • Process in real-time  
              • Read payload from FPGA  
              • Drop empty packets  
              • Translate payload to the format that is expected by the MAC  
              • Pass payload to process 10 |
| 10         | Send packet to third-party MAC |
| 11         | Receive packet to third-party MAC |
| 12         | • Process in real-time  
              • Obtain packet from process 11  
              • Translate it to the format expected by the PHY  
              • Configure dynamic PHY parameters via DMA FIFO and host FPGA interface  
              • Transfer payload to the FPGA via DMA FIFO |
| 13-15      | • Start and stop the NS-3 binary  
              • Print NS-3 console output in LabVIEW GUI |

**TABLE V:** LabVIEW host extension for the L1-L2 API

In order to ensure real-time performance, the system has been designed to guarantee that time-critical operations occur on a 1 ms time boundary. Based on the LTE frame structure, the FPGA expects a new payload every 1 ms. Other parameters such as MCS and resource block allocation are also configurable on a per TTI basis. As such, a real-time capable architecture has been designed that distributes the various processing loads between the FPGA and real-time enabled Linux host processor. The concept is illustrated in Figure 11. The data flow is driven by the FPGA, which provides a trigger signal at the beginning of each new TTI. This trigger signal has two main functions:

- On the FPGA, initiate the processing and transmission of the current TTI.
- On the host, initiate the preparation of the subsequent TTI.
The MAC to PHY host processing is subdivided into the following steps:

1. Receive a new packet from MAC.
2. Translate the new packet using an L1-L2 interface.
3. Send dynamic PHY configuration parameters to the FPGA.
4. Send the payload to the FPGA

Thus, the rapid and timely transfer of packets between the MAC and PHY layers or between NS-3 and the LTE Application Framework can be performed reliably with sufficient determinism enabled by the NI Linux RT OS.

The PHY to MAC communication at the receiver is implemented essentially with the same blocks wired in reverse order. The portion of the PHY that is deployed to the FPGA executes deterministically in real-time. On the host counterpart, determinism is achieved by using a real-time enabled Linux OS, Linux RT [19]. As an example, the LabVIEW Communications source code of the real-time process 9 (UE side), which executes on Linux RT is shown in Figure 12. The diagram depicts a timed loop that is configured to execute once every 1 ms. The individual processing steps depicted in Figure 11 can be identified as sub-VIs within the diagram.
Conclusion

The promises of 5G are great and many, but in order to deliver the envisioned capabilities in real-world commercial deployments, further experimental studies must be performed to better understand the challenges and limitations associated with the development of such systems at all levels of the communication stack. With a deeper understanding of the implementation challenges and problems, researchers can then begin to create novel solutions that go beyond just computer-based simulations to fully functional real-time prototypes that operate under real-world wireless channel conditions.

For such experimental studies, the NI LTE MAC/PHY prototyping system offers a flexible hardware and software reference architecture complete with a real-time upper layer stack and PHY layer that enables wireless researchers to rapidly prototype networks of LTE devices that communicate over real-world wireless channels. Furthermore, because the source code for all layers of the communication stack is available for users to modify and customize, the LTE MAC/PHY prototyping system’s set of capabilities can be extended to support other novel algorithms and protocols to explore the feasibility of 5G technologies such as software defined networks.

The LTE MAC/PHY prototyping system offers a rich set of baseband hardware and RF capabilities matched with open and modifiable software IP for all layers of the communication stack. And with the ability to use Linux-based open source tools and libraries, the LTE MAC/PHY prototyping system is the world’s first wireless testbed that offers all the benefits of NS-3 combined with the real-time over-the-air capabilities of the LTE Application Framework PHY layer. With such capabilities, wireless researchers can spend less time and fewer resources developing similar hardware and software of their own, thereby accelerating the prototyping process and allowing them to focus on that which is most important: results.
Ordering & Purchasing Information

The project files for the LTE MAC/PHY prototyping system, including all of the source code related to both NS-3 and the L1-L2 interface, are available for free with the purchase of the required NI hardware and software products: NI SDR hardware, LabVIEW Communications 2.0, and the LTE Application Framework 2.0.

To receive a quote for the LTE MAC/PHY prototyping system hardware and software and a copy of the project files, please contact your local NI sales representative or email labview.communications@ni.com.

References


The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Permission to use and/or modify the penguin image is granted by Larry Ewing lewing@isc.tamu.edu and The GIMP.