srsLTE: An Open-Source Platform for LTE Evolution and Experimentation

Ismael Gomez-Miguelez
Software Radio Systems Ltd.
Ireland

Andres Garcia-Saavedra
NEC Laboratories Europe
Germany

Paul D. Sutton
Software Radio Systems Ltd.
Ireland

Pablo Serrano
University Carlos III of Madrid
Spain

Cristina Cano
Inria Lille-Nord Europe
France

Doug J. Leith
Trinity College Dublin
Ireland

ABSTRACT

Testbeds are essential for experimental evaluation as well as for product development. In the context of LTE networks, existing testbed platforms are limited either in functionality and/or extensibility or are too complex to modify and customise. In this work we present srsLTE, an open-source platform for LTE experimentation designed for maximum modularity and code reuse and fully compliant with LTE Release 8. We show the potential of the srsLTE library by extending the baseline code to allow LTE transmissions in the unlicensed bands and coexistence with WiFi. We also expand previous results on this emerging research area by showing how different vendor-specific mechanisms in WiFi cards might affect coexistence.

1. INTRODUCTION

Testbeds are today an essential platform for experimental research and prototype development. They enable researchers to test, validate and assess the performance of new technologies. In the context of LTE, a testbed typically includes one or several User Equipments (UEs), one base station (eNodeB) and an Evolved Packet Core (EPC), although the latter may be minimal. Each of these components is provided up front by commercial off-the-shelf (COTS) hardware solutions. Though usually expensive, the performance is excellent and the functionality is extensively validated.

Some research problems, however, require adding new features to the standard or modifying some of its parts. For instance, in the context of 5G research, several groups are exploring how new waveforms such as GFDM can fit in the current LTE resource grid and physical layer procedures [1]. Another example is IoT, where tight power and budget constraints may require simplified waveforms and protocols to be introduced in the standard and the performance assessed in a real scenario. Other research problems require instrumentation, often of the entire network stack (PHY to IP) to measure the impact these metrics might have on the user application. Such metrics include for example, the channel Doppler spread, the interference, the number of HARQ retransmissions, the number of turbo decoder iterations or the power headroom. These modifications, customizations or instrumentation in COTS hardware are extremely difficult, if not impossible, or prohibitively expensive.

Software-Defined Radio (SDR) is a popular concept for implementing radio equipment in software, using low-cost general purpose computers and radio frontends. In recent years, it's gaining popularity as a tool to build close-to-reality testbeds for experimental research. If the researcher has access to the SDR application code, as it is the case with open source, the testbed can be easily modified and instrumenting the stack becomes as simple as pulling out the required metrics from the code. Whilst in terms of performance or capabilities, open source SDR testbeds typically stand behind their commercial counterparts, this flexibility and openness can be much more valuable for many research problems.

The most popular open source LTE SDR software available for testbeds today are Eurecom's OpenAirInterface (OAI) [2] and openLTE [3]. OAI currently provides a standard-compliant implementation of a subset of Release 10 LTE, including key elements of the network such as UE, eNB, MME, HSS, SGw and PGw on standard Linux-based computing equipment (Intel x86 PC architectures). The software can be used in conjunction with standard RF laboratory equipment available in many labs (i.e. National Instruments/Ettus USRP and PXIe platforms). openLTE runs with the Ettus Research B2x0 USRP and provides eNB, MME and HSS functionalities. openLTE code is well organized, documented and easy to customize or modify. However, it is incomplete and many features are still unstable or under development. Furthermore, it does not provide an UE, limiting the testbed capabilities in terms of instrumentation and measurement. OAI on the other hand is comparatively very complete and provides good performance. However, the code structure is complex and difficult for a user external to the project to customize.

In this work, we present an open source LTE library (srsLTE) and a complete software radio LTE UE (srsUE).[1] We describe the architecture, evaluate the computational efficiency and discuss the suitability to research on future LTE enhancements. As a case study we also present a modification to srsLTE that implements a duty cycle-based access mechanism for unlicensed LTE [4] and evaluate the results obtained in a coexistence scenario with WiFi. Given the attention that the use of the unlicensed spectrum by LTE is gathering [5, 6], with concerns being raised by the Federal Communications Commission (FCC) [7] as well as by the WiFi alliance [8], experimentation of emerging coexistence mechanisms in real testbeds becomes very relevant in order to ensure that coexistence with WiFi is guaranteed. Since the Medium Access Control (MAC) layer of LTE needs to be modified to implement changes in the way LTE accesses the channel, the srsLTE platform is a perfect candidate for experimentation. The results presented in this work extend those reported in [9], that were obtained using a PHY-only implementation of LTE-A, by showing the impact of: *i)* LTE periodically leaving the channel idle for WiFi, *ii)* vendor-specific nuances of different WiFi cards, which may affect WiFi performance in initially unforeseen ways, and *iii)* the resulting throughput of an unlicensed LTE UE.

The rest of this article is organised as follows. In Section 2 we describe the srsLTE library and analyse its computational efficiency, while the UE implementation is described in Section 3. Then, in Section 4, we present the LTE/WiFi coexistence scenario we use for evaluation and discuss the results obtained. Finally, we conclude with some final remarks.

2. srsLTE: AN OPEN-SOURCE LTE LIBRARY FOR SDR

In this section we describe the srsLTE library and we analyse its computational efficiency proving its suitability for experimentation with future enhancements of the LTE standard.

2.1 Description

srsLTE is an open source library for the PHY layer of LTE Release 8. It is designed for maximum modularity and code reuse with minimal inter-module or external dependencies. The code is written in ANSI C and makes extensive use of Single Instruction Multiple Data (SIMD) operations, when available, for maximum performance. In terms of hardware, the library deals with buffers of samples in system memory thus being able to work with any RF front-end. It currently provides interfaces to the Universal Hardware Driver (UHD), giving support to the Ettus USRP family of devices. The aim of the library is providing the tools to build LTE-based applications such as a complete eNodeB or UE, an LTE sniffer or a network performance analyser.

[1]`https://github.com/srsLTE/`

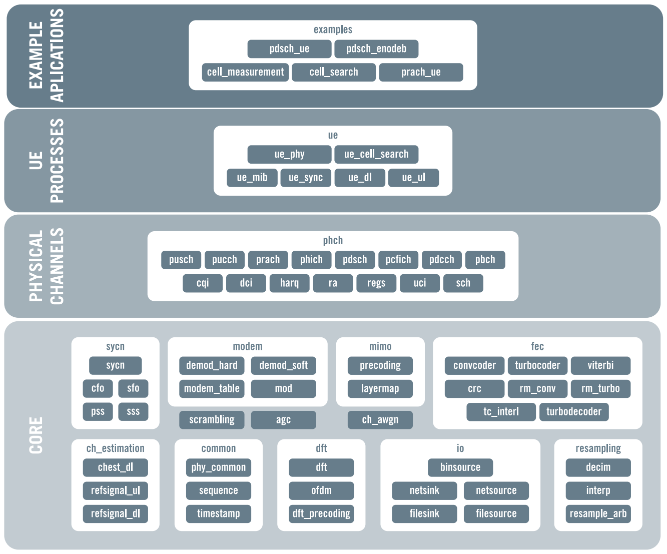

Figure 1: Module diagram for the srsLTE library.

The current features provided by the library are:

- LTE Release 8 compliant in FDD configuration;
- Supported bandwidths: 1.4, 3, 5, 10 and 20 MHz;
- Transmission mode 1 (single antenna) and 2 (transmit diversity);
- UE cell search and synchronization procedure;
- All DL channels/signals are supported for UE and eNodeB side: PSS, SSS, PBCH, PCFICH, PHICH, PDCCH, PDSCH;
- All UL channels/signals are supported for UE side: PRACH, PUSCH, PUCCH, SRS;
- Highly optimized turbo decoder in Intel SSE4.1/AVX (+100 Mbps) and standard C (+25 Mbps);
- MATLAB and OCTAVE MEX library generation for many components.

The modular library approach allows researchers to easily customize, improve or completely replace components without affecting the rest of the code. Modules are organized hierarchically in the following categories, as illustrated in Figure 1:

- **Core**: The core modules are the main building blocks within the PHY layer. Here we find the turbo and convolutional coders and decoders, modulator and demodulator, synchronization, channel estimation and reference signal generation, OFDM and SC-FDMA processing and so forth.

- **Physical Channels**: There is one module for each uplink and downlink channel (e.g. PDSCH, PUSCH, PDCCH, PUCCH, etc). Each module uses the core building blocks to implement the signal processing required to convert bits into samples ready for the digital converter and vice versa. Some physical channels share some functionality, which is implemented in common auxiliary modules, e.g., PUSCH and PDSCH share many functions which are defined in the SCH module.
- **UE Processes**: The UE processes implement the physical channel procedures for uplink and downlink making use of the physical channel modules.
- **Example Applications**: On top of the hierarchy we find a number of examples showing how to use the library through the UE modules. Among others, these examples include a PDSCH transmitter and receiver application and cell search examples.

2.2 Computational Efficiency

Computational efficiency is the most challenging aspect of an SDR application. As usual, the LTE receiver is much more complex than the transmitter. Though it can be ported to embedded processors, srsLTE initially targets general purpose processors (GPP), which is suitable for most research problems. Since memory is very cheap in GPPs, efficient code will make extensive use of look-up tables (LUT) and pre-generate as many signals as possible where memory access is more efficient than computation. For instance, all scrambling sequences, reference signals and some PUCCH signals can be generated for each subframe and input data combination. The CRC and encoder can use a LUT, and every interleaver can be pre-computed.

Clock speeds for individual processing cores in GPPs are beginning to reach their practical limits. For this reason, further performance improvements in modern CPUs must exploit data and instruction parallelism. These techniques are used for the most expensive parts of the receiver, namely the turbo decoder, the channel estimator and equalizer, the demodulator and the Viterbi decoder. In order to provide maximum portability, the VOLK library [10] has been used as much as possible. The VOLK library contains kernels of hand-written SIMD code for different architectures, including a general one written in C. At runtime, VOLK selects the correct kernel for maximum performance. When not possible, compiler intrinsics are used to provide two alternative versions of some modules (a generic and a SIMD one) and the best one is chosen at compile time.

We have measured the processing time of the UE PDSCH receiver for different bitrates (modulation and coding combinations) and analysed the per-module CPU usage in an Intel Core i7-3540M 3 GHz CPU. Table 1 shows the obtained results. The PDSCH includes all symbol and bit processing, including OFDM demodulation, channel estimation, symbol detection, rate recovery, resetting the soft buffer, turbo decoding and CRC check. Since the interval at which data arrives in LTE is 1 ms, 1 processing core is able to process the entire PDSCH chain in real-time. In practice, more cores are needed because the CPU needs to perform other tasks, such as synchronization, uplink signal generation, physical layer procedures, etc. The per-module usage results indicate that the turbo decoder is by far the most demanding part and its relative weight in the system increases as the rate increases. The SIMD turbo decoder implementation in srsLTE is based on the max-log-MAP algorithm for SIMD introduced in [11] with a minor modification to compute the horizontal maximum using the Intel's SSE4.1 `phminposuw` instruction.

Table 1: Total PDSCH receiver processing time and break-down of the CPU utilization for 20 Mhz bandwidth configuration.

Module Name	Percentage of CPU		
	75 Mbps 64QAM	30 Mbps 16QAM	3.62 Mbps QPSK
Turbo decoder (1 iteration)	78.14 %	64.21 %	20.89%
OFDM receive processing	6.08 %	11.70 %	33.33 %
RE de-mapping	4.92 %	9.31 %	25.26 %
Rate recovery	4.49 %	5.64 %	8.34 %
CRC checksum	2.92 %	2.23 %	0.72 %
Soft demodulation	1.76 %	2.11 %	3.38 %
Equalization	0.16 %	1.84 %	4.98 %
Others	1.53 %	2.96 %	55.12 %
Execution Time	954 μs	488 μs	170 μs

3. srsUE: AN UE SDR IMPLEMENTATION

srsUE is a software radio LTE UE covering all layers of the network stack from PHY to IP. It is written in C++ and builds upon the srsLTE library which provides the PHY layer processing. For some security functions and RRC/NAS message parsing, it uses some functions from the openLTE project. Running on an Intel Core i7-4790, srsUE achieves more than 60 Mbps downlink with a 20 MHz bandwidth SISO configuration, when tested against an Amarisoft LTE 100 eNodeB. Apart from the features listed above for the srsLTE library, srsUE provides the following additional features:

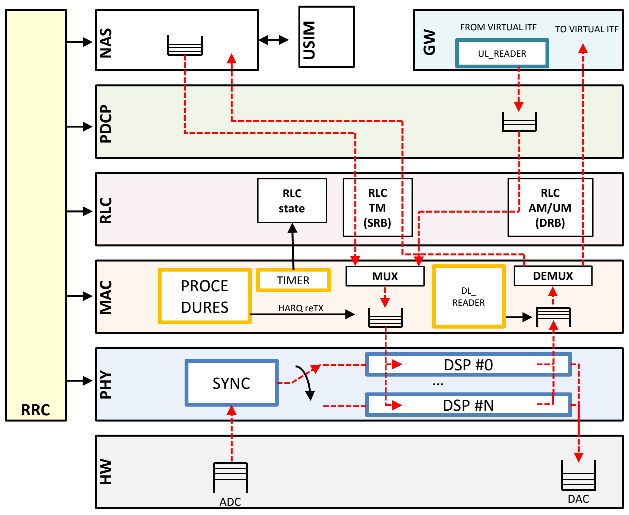

Figure 2: Threading architecture in srsUE. Boxes with coloured borders are threads.

- MAC, RLC, PDCP, RRC, NAS and GW layers;
- Soft USIM supporting Milenage and XOR authentication;
- Detailed log system with per-layer log levels and hex dumps;
- MAC layer `Wireshark` packet capture;
- Command-line trace metrics;
- Detailed input configuration file; *and*
- virtual network interface (i.e. *tun/tap* device) created upon network attach.

A configuration file is provided to set parameters such as the downlink carrier frequency and log or packet capture options, making the software easy to use. The UE starts setting the USRP sampling rate to 1.96 MHz, in order to capture the synchronization PBCH signals. Once synchronization and MIB decoding is successful, it reconfigures the sampling rate to the appropriate sampling rate for the LTE signal bandwidth. Next, the UE attempts an attachment by sending a PRACH sequence and if the correct response is received, it continues the connection setup procedure. Following successful network attachment, a new virtual network interface is created in the system and the user can then establish IP sessions with the eNodeB network.

This instrumentation and the `Wireshark` capture capabilities make srsUE an ideal tool for many applications, including cross-layer performance analysis, education and prototype development. srsUE opens the insights of the LTE stack to the user: the real-time traces, logs and packet captures can be used to correlate user experience in video streaming or web surfing with the signal quality or any other metric inside the stack, for example.

srsUE classes are organized by layers, one class per stack layer. Each class provides a separate clean interface to any other class that make use of it, which is used for message passing between layers of the stack.

A set of threads are created for performance and priority management reasons. Figure 2 illustrates the different layers and the threads within them. Red dashed arrows indicate data paths whereas dark arrows indicate interaction between threads or classes. Each thread performs the following tasks:

- *PHY DSP*: These threads perform all the processing associated with a subframe, including: OFDM demodulation, PDCCH search, PDSCH decoding, PUSCH/PUCCH encoding, uplink signal generation and transmission to the digital converter;
- *PHY SYNC*: Receives 1 subframe from the converter, performs time and frequency synchronization, copies the aligned frame into an idle PHY DSP thread and triggers its execution;
- *MAC PROCEDURES*: Manages MAC procedures, including random access, scheduling request and uplink/downlink HARQ;
- *MAC TIMER*: Provides timer services to upper layers. All timers in LTE have a resolution of 1 ms, thus are implemented with simple counters synchronised to the subframe reception instead of using system clocks.
- *MAC DL READER*: Reads downlink transport blocks from a buffer, processes the PDU and pushes the packet up the stack until the GW; *and*
- *GW UL READER*: Reads from the virtual network interface and stores it into the PDCP buffer.

The threading architecture in the PHY is motivated by the stringent latency requirements. In LTE, the required response time is 4 ms (4 subframes), which is imposed, in the UE, by (i) the transmission of an ACK/NACK HARQ indication after decoding a PDSCH transport block, (ii) the transmission of PUSCH after the reception of an uplink grant or (iii) the retransmission of PUSCH after the reception of a NACK HARQ indication. The worst case scenario happens by the superposition of case (i) and (ii) or (i) and (iii): in the same subframe the UE decodes the PDSCH and encodes a PUSCH. Considering the time needed to buffer 1 subframe, the UE has 3 ms to decode the PDSCH, generate the PUSCH with the ACK/NACK information in it and send the samples to the converter buffer.

Dividing the processing into uplink and downlink threads is ineffective, because the uplink processing needs to wait the downlink processing to finish. A more efficient approach is to parallelise all the processing associated with every subframe into a pipeline avoiding any delays (for thread synchronization or data transfer) within the tasks to be done in the 3 ms deadline. This architecture is efficient for dual- or quad-core CPUs. If more than 4 cores are available, increasing the number of PHY threads above 3 does not add any benefit, because the maximum tolerable latency is 3 ms (3 pipeline stages). The performance may then be improved by creating multiple threads associated to each subframe and split the multiple codeblocks in a transport block among them, a feature currently under development.

4. EXPERIMENTING WITH AN UNLICENSED LTE & WIFI COEXISTENCE SCENARIO

As mentioned earlier, a potential use of the srsLTE platform is to analyse the coexistence between unlicensed LTE and WiFi. The great advantage of using real hardware rather than simulations is that it allows us to evaluate the impact of the complex wireless propagation effects encountered indoors (where such small cells are most likely to be deployed) and also explore issues such as capture and carrier sense which are often difficult to model adequately.

4.1 LTE/WiFi Testbed

Our interest is in a WiFi link and an LTE link operating in the same band. The LTE stations consist of an USRP B210 board from Ettus connected via an USB 3.0 interface to a standard PC (Intel Core i7) running Linux Ubuntu Trusty, with the `uhd_driver` and version 1.0.0 of srsLTE. The distance from the LTE BS to the WiFi transmitter is 34 cm, and 35 cm to the WiFi receiver. The WiFi link is 94 cm long. We use a VERT2450 Dual Band (2.4 to 2.48 GHz and 4.9 to 5.9 GHz) omni-directional vertical antenna with 3dBi Gain and, unless otherwise noted, LTE is configured to use 100 physical resource block (PRBs), i.e., to use a 20 MHz channel bandwidth (the same as that used by WiFi).

The WiFi nodes are Soekris net6501-70 devices, which are low-power single-board computers equipped with a 1.6 GHz Intel Atom E6xx series CPU, 2 Mini-PCI sockets, 2048 Mbyte DDR2-SDRAM and 8 Gbyte usb-based storage. These run Linux Ubuntu (kernel 3.13). In most of our experiments, in order to detect vendor-specific performance issues that might be present in the 802.11 hardware (see e.g. [12]), we repeat all measurements using two different wireless NICs, namely, an Atheros AR9390-based 802.11a/b/g/n card and a Broadcom BCM4321 card. We configure the Atheros cards to operate in the 5 GHz band, which measurements using a spectrum analyser confirmed were free from other transmissions, and configure the Broadcom cards to operate in the more populated 2.4 GHz band (since they do not support 5 GHz operation).

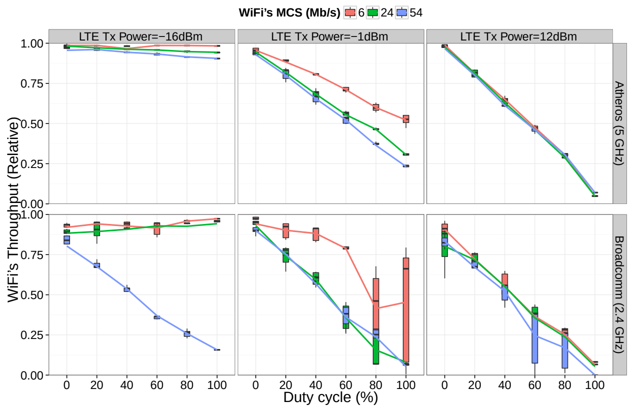

Figure 3: Impact of LTE duty cycle on performance.

4.2 Impact of duty cycle

We start our experiments by investigating the impact of LTE transmissions on a WiFi link. To this end, we configured a WiFi station to send 1500 Bytes UDP traffic. The WiFi station is constantly backlogged (always has a packet to send) and is configured to use a transmit power of 17 dBm. We vary the activity on the LTE link by modifying the `srsLTE` code (namely, the example program `src/examples/pdsch_enodeb.c`) that is executed in user space. In more detail, we implement a periodic duty-cycle channel access scheme similar to the proposed CSAT coexistence mechanism [13] where we fix an active interval during which the LTE base station transmits data to the UE followed by a randomized silent period.[2] The mean period (active plus silent) is equal to 150 ms. By changing the mean duration of the silent period we vary the "duty cycle" of the LTE link. The LTE base station does not perform carrier sensing and sends traffic at the start of the frame boundary. LTE control plane signals are transmitted during the on periods but the channel is completely freed during the silent periods. This implies that an UE loses synchronisation with the BS unless signaling is sent offloaded through other means (e.g. licensed band). We will extend our setup to maintain UE synchronisation and measure LTE throughput with a simple in-band signaling approach in §4.6.

We vary the LTE duty cycle between 0% (no LTE transmissions) and 100% (no LTE silent periods), and measure the resulting WiFi throughput for different settings of the LTE transmit power and of the Modulation and Coding Scheme (MCS) used by the WiFi link. The

[2]Randomising the LTE silent period alleviates potential quantisation effects [6].

results are shown in Figure 3, where for each configuration the average throughput is measured over a 10 s experiment, which is repeated 5 times, and we use box-and-whisker plots to represent the measured median, 25th and 75th percentiles, and the 1.5× interquartile range. Note that we plot the normalised WiFi throughput, i.e., the ratio of the measured throughput to the maximum achievable throughput (with 0% duty cycle), for ease of comparison across different choices of MCS.

In all of these experiments the WiFi throughput is inversely proportional to the LTE duty cycle, as might be expected as the LTE active and silent periods are relatively long compared to the WiFi transmissions (i.e. the LTE silent period leaves enough idle time to accommodate transmission of several WiFi packets). When the 5 GHz band is used the results show little dispersion and the use of a lower MCS increases the robustness of the WiFi transmissions. However, this effect is minor when the LTE transmission power is very low (so the LTE interference with WiFi is then negligible) and also when the LTE transmission power is high (when all choices of WiFi MCS are equally affected by the interference from LTE). When the 2.4 GHz band is used the results are qualitatively similar, although there is more dispersion, with the 54 Mbps MCS in particular being extremely sensitive to LTE interference.

To sum up, these measurements confirm that scheduling the LTE transmissions using a randomised duty cycle, with relative long on and off periods, has the expected qualitative impact for both of the WiFi wireless cards considered. We next analyse in more detail the impact on performance of the choice of MCS, transmission power and band used.

4.3 Impact of transmission power

The foregoing results confirm that adjusting the LTE duty cycle allows the impact of LTE interference on WiFi to be controlled. We now fix the LTE duty cycle to be 50% and vary the transmission power of both the WiFi and LTE transmitters, with the aim of understanding the sensitivity and robustness of WiFi performance to these parameters. Measurements are shown in Fig. 4, for the two WiFi wireless cards studied and for different choices of WiFi MCS.

It can be seen that the WiFi transmission power has little impact on performance, that is, results are very similar for the different WiFi transmission powers shown in Fig. 4. This confirms that, in our small testbed, the signal quality of the WiFi link is good for all the configurations of this parameter. The measurements also show that, as in the previous experiments, the use of a lower MCS results in an increased WiFi normalised throughput, which suggests that a large number of WiFi transmissions collide with LTE transmissions (i.e., the reduction in throughput performance is not only caused

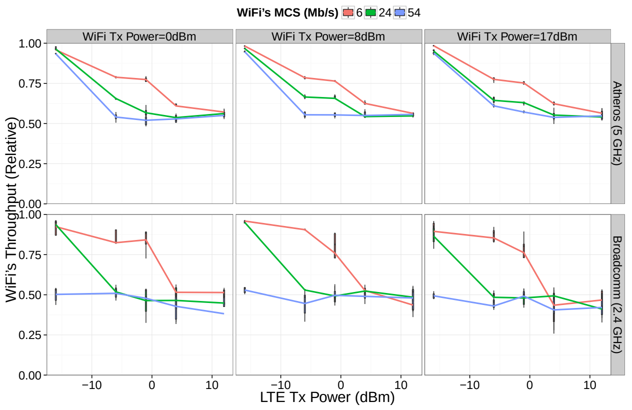

Figure 4: WiFi throughput vs. LTE transmission power

by channel deferral), and therefore the use of more robust transmission schemes favors the appearance of the *capture effect* [6]. In fact, for the case of the Atheros card, there is a small but consistent improvement in performance as the transmit power is increased from 8 dBm to 17 dBm when the MCS is 54 Mbps and the LTE transmission power is −6 dBm, which lends further support to this observation.

Given these results, we conclude that careful tuning of the transmission power could potentially be used to adjust the share of resources between WiFi and LTE. However, given the steepness of the variation in throughput vs. LTE transmission power and the only minor impact of the WiFi configuration, this might not be a practical approach.

4.4 Impact of bandwidth used

We next investigate the impact of varying the number of subcarriers used by LTE (i.e., its bandwidth). To this end, we perform a sweep on the number of PRBs used by LTE, and meausre the resulting WiFi throughput for different choices of the LTE transmission power and WiFi MCS. The measurements obtained are shown in Figure 5.

In contrast to the previous experiments, here there are significant differences in the WiFi throughput depending on the WiFi card used. While when the LTE transmit power is 12 dBm the number of PRBs does not affect WiFi throughput, when the LTE transmit power is -16 dBm the WiFi throughput varies considerably as the number of PRBs changes. It can also be seen that the Atheros and Broadcom WiFi NICs exhibit the *opposite* behaviour: for the latter, when the MCS is 54 Mbps the throughput reduces as interference increases (an arguably intuitive result, as LTE occupies more spectrum), while for the former throughput increases with the number of PRBs used. Furthermore,

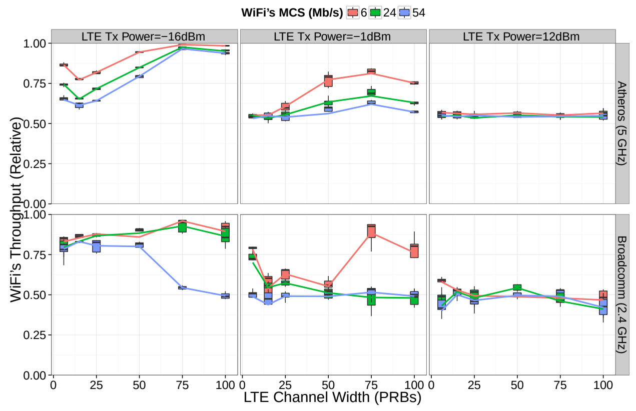

Figure 5: WiFi throughput vs. LTE bandwidth

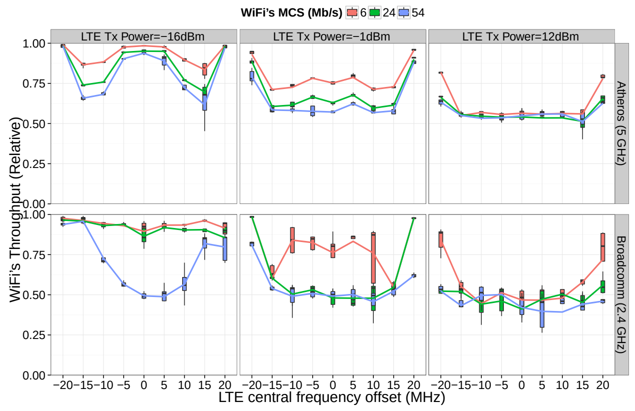

Figure 6: WiFi throughput vs. LTE central frequency

this behaviour is qualitatively consistent for the case of -1 dBm with the Atheros cards, while the Broadcom cards and an MCS of 6 Mbps now show a different pattern as compared to when the LTE transmit power is -16 dBm. Our hypothesis for this behaviour is that these WiFi cards make use of different (proprietary) clear channel assessment mechanisms for carrier sensing.

These results indicate that while changing the number of PRBs used by LTE might be useful to manage coexistence between LTE and WiFi, this may result in vendor-dependent performance. One potential consequence is that this may introduce fairness issues in a multi-vendor environment. Therefore, we argue that more experimental research, such as the enabled by our srsLTE-based platform, is required prior to the rollout of LTE in unlicensed bands.

4.5 Impact of the central frequency

Finally, we explore performance when the amount of spectrum overlap between LTE and WiFI is varied. We again configure LTE to use 100 PRBs (corresponding to a channel bandwidth of 20MHz) and now vary the central frequency used for LTE transmissions from -20 MHz to +20 MHz in steps of 5 MHz. Similarly to the previous experiments, we perform the tests for a range of LTE transmission powers and WiFi MCS. The resulting measurements are shown in Figure 6.

On the one hand, when the LTE transmission power is the highest considered (12 dBm) then the WiFi throughput performance is similar for both wireless cards: when there is large overlap in the spectrum used by LTE and WiFi the interference is large and only for the edge cases where the centre frequency is ± 20 MHz is it reduced (it is also worth noting that the interference pattern is practically symmetrical). On the other hand, when the LTE transmission power is the smallest considered (-16 dBm), the behaviour changes depending on the WiFi card used. For the case of the Broadcom card the results are as might be expected: the smaller the frequency offset the lower the WiFi throughput for an MCS of 54 Mbps, while for more robust MCS the performance does not change significantly with the frequency offset. For the case of the Atheros cards instead of a U-shaped figure we see that, for all choices of MCS, throughput is maximised either when the LTE transmissions are placed 20 MHz away from the center frequency or when LTE uses the exact same frequency as WiFi transmissions. For intermediate LTE transmission powers (-1 dBm), although in all cases WiFi throughput improves when using a more robust MCS, the results vary qualitatively depending on the hardware considered: for the case of Atheros, the offset has little impact on thoughput apart from when it is 20 MHz; in contrast, for the Broadcom cards and an MCS of 6 Mbps there is a remarkable drop in throughput for an offset of 15 MHz. These measurements confirm there is a strong dependency on the hardware of choice, which further strengthens the need of proper experimentation when studying LTE/WiFi coexistence.

These results further support the two conclusions in the previous experiments: firstly, an adequate tuning of the spectrum used by LTE may improve coexistence with WiFi; secondly, more real-life experimentation is required to better understand the reasons for the observed behaviour and to detect vendor-specific issues that might preclude fair coexistence.

4.6 Unlicensed LTE throughput

In the experiments presented so far, the LTE transmitter does not transmit any signal during the silent periods, which results in de-synchronisation between BS and UE. In order to measure LTE throughput, we implement an in-band signaling scheme, based on sending the LTE synchronisation signals during the inactive pe-

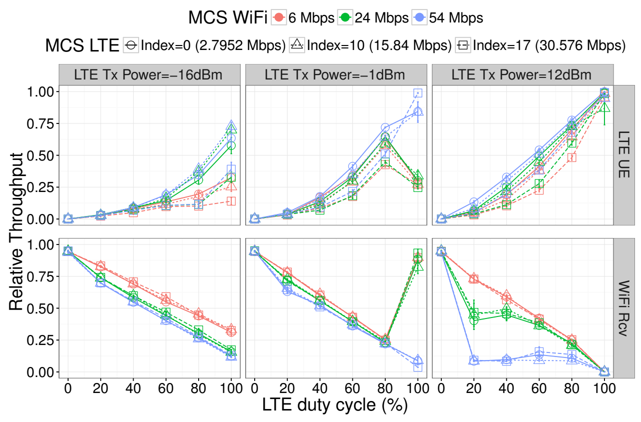

Figure 7: Throughput performance vs. duty cycle.

riods in order to keep the LTE UE synchronised. Note that this resembles the implementation of unlicensed LTE using the Almost Blank Subframes[3] as well as a standalone unlicensed LTE deployment. We depict in Fig. 7 the relative throughput of both technologies (we focus on the Atheros results due to space reasons) for different configurations of the LTE duty cycle and transmission power, and different MCS (the transmission power of WiFi is set to 17dBm, LTE is configured to 100 PRBs). From the figure, we can observe that now the throughput achieved by WiFi is lower than that of Fig. 3, the reason being that now the LTE BS sends ABSs during the silent periods.

5. CONCLUSIONS

In this work we have presented an open-source, modular and fully compliant with LTE Release 8 platform that allows for LTE extension and experimentation. We have described the architecture of the srsLTE library as well as the srsUE and evaluated its computational efficiency proving its suitability to current LTE testing. As a case study we have shown the potential of the library for extension of LTE to work in the unlicensed bands and coexist with WiFi networks. Our results in this regard motivate further experimental validation of emerging coexistence mechanisms as particularities of off-the-shelf wireless cards and synchronisation procedures between BS and UE might affect fair coexistence in ways difficult to predict via analysis or simulations.

6. ADDITIONAL AUTHORS

[3]Although ABSs were initially considered as a mechanism to implement CSAT, carrier aggregation features are now being considered instead. The srsLTE platform can be extended to include the out-of-band signaling used in carrier aggregation (using the licensed interface) and it is a feature we are currently working on.

7. REFERENCES

[1] G. Fettweis, M. Krondorf, and S. Bittner, "Gfdm - generalized frequency division multiplexing," in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, April 2009, pp. 1–4.

[2] "OpenAirInterface.org," 2015. [Online]. Available: http://www.openairinterface.org/

[3] "openLTE," 2015. [Online]. Available: http://openlte.sourceforge.net/

[4] D. Flore, "Chairman Summary," *3GPP workshop on LTE in unlicensed spectrum*, June, 2014.

[5] S. Sagari, S. Baysting, D. Saha, I. Seskar, W. Trappe, and D. Raychaudhuri, "Coordinated dynamic spectrum management of lte-u and wi-fi networks," in *Dynamic Spectrum Access Networks (DySPAN), 2015 IEEE International Symposium on*, Sept 2015, pp. 209–220.

[6] C. Cano, D. J. Leith, A. Garcia-Saavedra, and P. Serrano, "Fair Coexistence of Scheduled and Random Access Wireless Networks: Unlicensed LTE/WiFi," *arXiv:1605.00409*, 2016.

[7] FCC, "Office of engineering and technology and wireless telecommunications bureau seek information on current trends in LTE-U and LAA technology," May 2015.

[8] WiFi Alliance, "Wi-Fi Alliance statement on License-Assisted Access (LAA)," February 2015. [Online]. Available: http://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-statement-on-license-assisted-access-laa

[9] Y. Jian, C.-F. Shih, B. Krishnaswamy, and R. Sivakumar, "Coexistence of Wi-Fi and LAA-LTE: Experimental evaluation, analysis and insights," in *IEEE International Conference on Communication Workshop (ICCW), 2015*, 2015.

[10] T. O. T. Rondeau, N. McCarthy, "SIMD programming in GNURadio: Maintainable and user-friendly algorithm optimization with VOLK," in *Proc. Conference on Communications Technologies and Software Defined Radio (SDR12). Brussels, Belgium: Wireless Innovation Forum Europe*, 2012.

[11] K. Loo, T. Alukaidey, and S. Jimaa, "High performance parallelised 3GPP turbo decoder," in *Personal Mobile Communications Conference, 2003. 5th European (Conf. Publ. No. 492)*, April 2003, pp. 337–342.

[12] G. Bianchi *et al.*, "Experimental Assessment of the Backoff Behavior of Commercial IEEE 802.11b Network Cards," in *INFOCOM 2007*, May 2007, pp. 1181–1189.

[13] A. K. Sadek *et al.*, "Extending LTE to unlicensed band-Merit and coexistence," in *IEEE International Conference on Communication Workshop (ICCW), 2015*, 2015.