Design and implementation of an LTE system with multi-thread parallel processing on OpenAirInterface platform [Invited paper]

Hengyang Shen*, Xingguang Wei*, Haitao Liu*, Yang Liu+, and Kan Zheng*

*Wireless Signal Processing & Network Lab
Key laboratory of Universal Wireless Communication, Ministry of Education
Beijing University of Posts & Telecommunications
Beijing, China, 100876
+Technology Innovation Center of China Telecom Corporation Limited,
Beijing, China, 102209

***Abstract*—Software Defined Radio (SDR) system has great advantage in simulating and verifying the new communications technologies attributed to its striking advantage in flexibility and reconfigurability. Among all the GPP-based (general purpose processor) SDR systems, OpenAirInterface (OAI) is one of the most comprehensive and competitive open-source SDR system. Due to its signal-thread processing, the UE (user equipment) of OAI still cant process the vast amounts of data when the bandwidth is up to 20MHz in the real-time experimentation. This paper provides an overview of the system architecture of real-time experimentation of OAI and analyzes its baseband signal processing. Then, the authors of this paper propose and implement a method of multithread parallel processing on the UE of OAI. Over-the-air experiments are carried out with the proposed method for the purpose of performance evaluation in the real-time wireless environment. Results illustrate that the proposed method enhances the performance of the system and improves its potential for 5G technology development.**

***Keywords*—SDR, OAI, LTE, baseband signal processing, multithread.**

I. Introduction

With the rapid development of mobile communications technology in the past decades, various new communications technologies have been proposed, simulated, verified, and commercialized. A practical product must be tailored to verify the feasibility and efficiency of the communications technology with low efficiency and high cost before Software Defined Radio (SDR) was developed. The key idea of SDR is to construct a universal hardware platform, which is open, standardized, and modularized. All sorts of communications modules can be realized by software, i.e., frequency band selection, modulation & demodulation, encoding & decoding and communications protocol. To make the SDR system more flexible and more open, the A/D and D/A converters must be close to the antenna as far as possible to reduce the analog modules [1]. Thus, technology upgrading will only need to update the software rather than the hardware, which can shorten the development period and cut research and development spending.

The SDR systems can be developed on different hardware platform, i.e., the field programmable gate array (FPGA), digital signal processor (DSP), and general purpose processor (GPP) [2]. Compared to FPGA and DSP solutions, the GPP-based SDR system has advantage in flexibility and reconfigurability because it's easy to develop and upgrade on the GPP using high-level crossplatform programming languages, rather than hardware description languages. A typical GPP-based SDR system consists of a single piece of peripheral equipment connecting to the GPP. The peripheral equipment, e.g., universal software radio peripheral (USRP), is mainly responsible for frequency conversion and digitization and the GPP is responsible for baseband signal processing [3].

Recent years, many organizations have developed its own GPP-based SDR systems, e.g., Amari LTE 100, OpenLTE, and OpenAirInterface (OAI). To the best of the authors' knowledge, only the OAI program is a full open-source implementation of the entire LTE protocol stack that comprises all the components of the LTE system. OAI is a flexible platform created by the Mobile Communications Department at EURECOM, aiming to innovate in the area of wireless networking and communications [4]. As an excellent communications simulation and verification platform, OAI can verify various new communications technologies by changing some code but not the hardware. While up to now, the UE of OAI is still single-thread processing, and therefore the UE must receive and process the data from each subframe serially within 1 ms [10]. When the bandwidth is up to 20MHz, it's hard for the GPP to process vast amounts of data within 1ms, especially when the modulation mode is 64 quadrature amplitude modulation (QAM). This will limit the further development of OAI when facing more complex communications technology that needs greater bandwidth.

In order to realize the potential of OAI to simulate and verify more complex communications technologies, we improve the UE of OAI to multiple threads processing, thus the UE can parallelly process the data from each subframe. Consequently, OAI can be a competitive platform to verify the potential candidate techniques for the fifth generation (5G) mobile network.

This paper is organized as follows. We briefly discuss

the system architecture of OAI and analyze its deficiency in baseband signal processing in Section II. Then, Section III elaborates on the proposed multithread parallel computing method in UE. Real-time experimental results with the proposed processing method are presented in Section IV. Finally, conclusions are given in Section V.

II. System Architecture of OAI

OAI is an open-source platform for software/hardware development for the core network (EPC) and access-network (EUTRAN) of 3GPP cellular networks, which implements the entire LTE protocol stack and provides all elements of the 4G LTE system architecture [11]. There are two unique features of OAI platform, real-time experimentation and emulation [7]. In this paper, we focus on the real-time experimentation with OAI. In what follows, we will elaborate on the hardware and software features of OAI.

A. Hardware

Several software radio frontends are suitable for real-time experimentation and validation, e.g., USRP, ExpressMIMO2 PCI Express (PCIe) board, BladeRF board and so on [11]. The system architecture we use is composed of a USRP and a GPP. Considering the compatibility with the LTE system and the practicability, we select USRP B210 as the peripheral equipment. With continuous frequency coverage from 70 MHz to 6 GHz, all LTE frequency bands can be used. Besides, its 61.44 sampling rate is a multiplier of all the sampling rates defined in 3GPP LTE specifications, which benefits the reduction of complexity of baseband signal processing in the GPP. Moreover, it is fully supported by OAI. On the transmit path, the GPP generates the digital baseband signal, which follows the 3GPP LTE standards, and then transfers the data to the USRP via an USB 3.0 interface. The USRP completes the function of FPGA, digital to analog conversion (DAC) and then transmits the radio frequency (RF) signal. As for the receive path, the process operates reversely. The USRP receives the RF signal from the antenna, and after the procedures of the analog to digital conversion (ADC) and FPGA, the digital signal is converted to baseband, ready to be transferred to the GPP. The GPP uses OAI software to do baseband signal processing.

B. Software

All baseband signal processing of the OAI system is realized in software [3]. The operation system of the GPP is a GNU/Linux with low-latency patches, which helps attain a better real-time performance. Besides, the OAI system requires a great deal of open-source software libraries and toolkits. Among them, USRP hardware driver (UHD) [9] is the most important module. The UHD is a hardware driver library for all USRP series, which provides the consistent Application Program Interface (API) for developers to write UHD-based applications. The OAI program use the UHD with the API to control the USRP, including setting RF bandwidth, tuning frequency, sending/receiving samples and so on. Then with different configuration parameters, the OAI program can be compiled and executed as the eNB or the UE. Although OAI includes a full software implementation of 4th generation mobile cellular systems compliant with 3GPP LTE standards, only the PHY layers are realized in our real-time experimentation. In the following part, we mainly analyze the baseband signal processing procedure in the UE side and propose a multithread parallel computing method to do the baseband signal processing.

On the basis of 3GPP LTE standards [8], in the FDD system, each radio frame is 10 ms long and consists of 10 subframes. 14 OFDM symbols are contained in one subframe. As shown in Fig. 1, the yellow part is where the pilots are carried, which are used to do channel estimation [6]. Take the Fig.2 for example, after #0 OFDM symbol in subframe 4 is received, the estimation of the channel at the pilot frequencies is calculated based on LS. Then by the linear interpolation of the channel, the channel matrices of #12, #13 symbols in subframe 3 and #0 symbol in subframe 4 are derived. Similarly, we can get the channel matrices of #1 #13 symbols in subframe 4 after #0 symbol in subframe 5 is recevived. In other words, the OAI UE processes the data of subframe 4, including demodulation, unscrambling and decoding, during the duration of subframe 5.

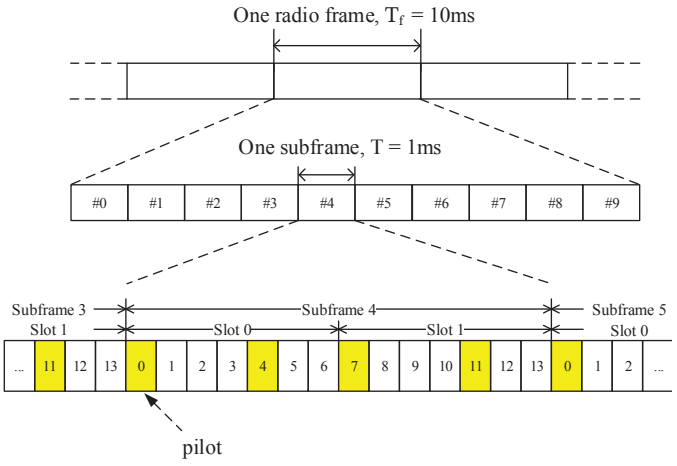


Fig. 1. Frame structure of FDD in time domain.

Currently in the UE software architecture of OAI, three process threads are set to handle all the procedures of the PHY layer [10]. One thread with the name of UE-RX-thread is used for the DL procedures, and another one, UE-TX-thread is responsible for UL procedures. The third process thread named UE-thread process the synchronization and data exchange with the USRP. The UE-RX-thread is scheduled every subframe, i.e., 1 ms, and the OAI uses the serial processing method. Therefore, the DL procedure must be done within 1 ms, or the next subframe cannot be processed, which will lead to the abort of the system. The procedure is illustrated in Fig. 2. However, when it comes to 20MHz bandwidth with 16-QAM, the growing amount of data leads to the execution time of the DL procedure longer than 1ms. The system cannot work properly in 20MHz bandwidth. This will cause a great limitation against promotion and reference for the OAI systems, and will be the obstacle of further development of OAI when facing more complex communications technologies. To solve this problem, we propose the multithread parallel processing method. In the following section, we will elaborate on the design and realization of the processing method.

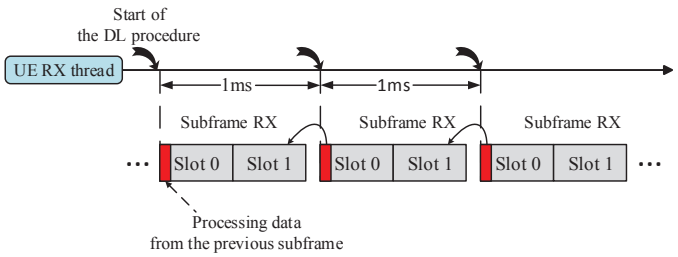


Fig. 2. The serial processing procedure of OAI.

III. Design and Implementation of the Multithread Parallel Processing Method

In this section, we propose a multithread parallel processing method. As shown in Fig. 3, 10 threads are established for the DL procedures instead of one, each of them is responsible for a subframe. Then the thread number is corresponding to the subframe number. With the help of decoding slot, data receiving and processing can be accomplished in one subframe. Further details will be presented as follows.

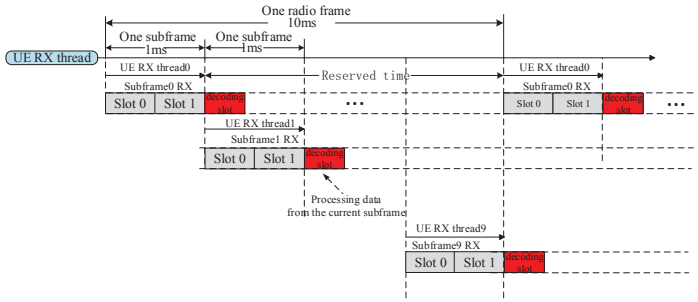


Fig. 3. The multithread parallel processing method.

A. Decoding slot

Considering of the channel estimation method in OAI, we propose the conception of the decoding slot to do demodulation, unscrambling and decoding of the current subframe. The odd and even slots in the subframe completes the function of acquiring data from the UE-thread and using the Fast Fourier Transform (FFT) algorithm to transfer time domain discrete signal into one of the frequency domain. The decoding slot receives the first OFDM symbol from the next subframe to get the pilot information for deriving the whole channel matrix of the current subframe. Thus, all the data in the current subframe can be processed in the corresponding thread. In this way, the processing procedure of one subframe can be independent from each other, which makes the thread of DL baseband signal processing of each subframe a separated procedure and also makes multithread parallel processing possible. Unlike the traditional slot introduced by 3GPP LTE standards [8], the decoding slot is just the concept using in data processing. It doesn't have a certain length in time domain. It differs according to different amount of the data.

B. Multithread parallel processing

With the constatnt decrease in feature size and the increasing conunt of transistors on chip area, the performance of the processors are improved greatly and multiple processor cores on the same integrated circuit die are built [12]. The multithreading technique can be used to take full advantage of the multicore processor. Thus, the design of multithread parallel processing of OAI is of great importance to improve the performance. The PHY layer of the OAI program is written in C language, which has many popular multithreading libraries for paralle programming [13] and makes it possible to change the baseband signal processing of OAI into multiple threads. The workflow of the multithread parallel processing is illustrated in Fig 4. The scheduling scheme remains the same, activating the UE-RX-thread every 1 ms. The difference is that 10 UE-RX-threads are initiated and the number of UE-RX-thread activated is due to the subframe number received. Therefore, when the processing of the current subframe cannot be completed within 1 ms, the next can still be processed parallelly. For instance, when subframe 0 is received, the UE-RX-thread0 is activated to do the processing preocedure. The execution time of subframe 0 can be longer than 1 ms, and when the next subframe arrives in 1 ms. As long as the execution time of decoding slot is shorter than 9ms, the UE can work properly in theory. In the next section, experiments will be carried out to verify the proposed processing method.

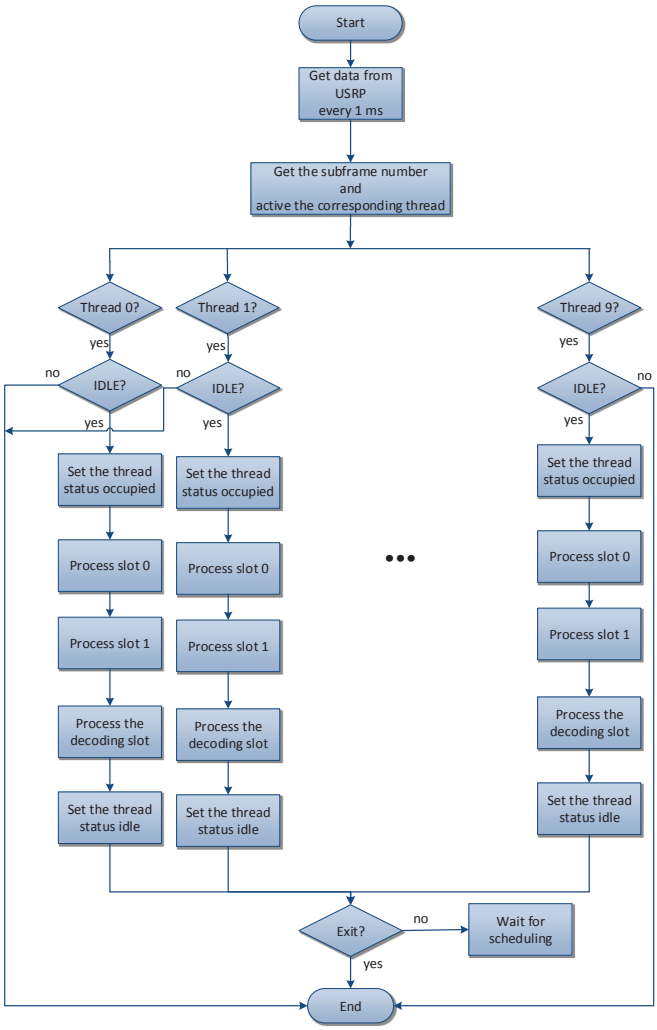


Fig. 4. The workflow of multithread parallel processing.

IV. Experimental Results and Analysis

A. system configuration

The real-time OAI system is implemented in a general purpose computer that connects to the Ettus Research USRP. With the features of four physical cores supporting hyper-threading and the 3.4 GHz maximum clock frequency, Intel Core i7-4770 CPU is selected as the CPU model of the general purpose computer, which has strong practicability and a good cost performance. As the OAI software suggests, the OS is the 64-bit Ubuntu 14.04 low-latency with the 3.17.0 kernel version, which reduces latency and provides preemptive scheduling [5].

In our experiments, the real-time OAI system is deployed in FDD SISO mode. The target frequency is 2.66 GHz(band7) in a controlled indoor radio environment. We use three types of system bandwidth to test the proposed processing method, i.e., 5MHz, 10MHz and 20MHz. The receiver employs quadrature phase-shift keying (QPSK) modulation, 16-QAM, and 64-QAM modulation schemes. The Table I is a summary of the parameters used in our experiment.

B. Experimental scenario

Our experimental scenario is depicted in Fig. 5, and consists of two pairs of computers and USRPs, which are used as the eNB and UE. Main system parameters are set according to the Table 1. We set the transmitting power of eNB USRP to about 10 dBm, and two USRPs are around two meters away with no obstruction in an indoor environment.

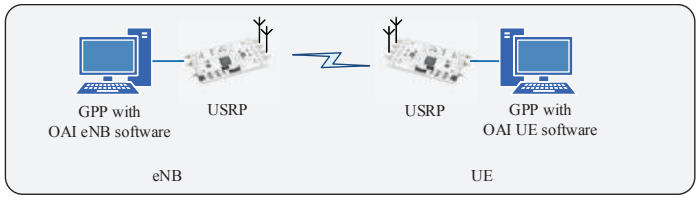


Fig. 5. The experimental scene.

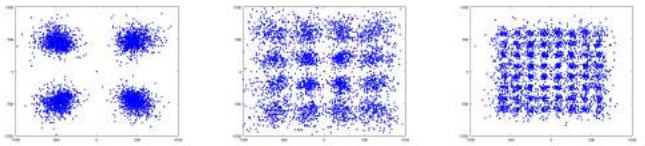
C. Experimental results

In out experiment, the proposed processing method is tested in real-time OAI system to evaluate the performance in the real wireless environment.

TABLE I. SYSTEM CONFIGURATION

	Item	Value
Hardware	USRP	Ettus Research USRP B210
	CPU	Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz
Software	OS	64-bit Ubuntu 14.04 LTS with low-latency kernel 3.17.0
	UHD	Version 3.8.0
	OAI	trunk_r7528
Parameters	Duplex mode	FDD
	Transmission mode	TM1(SISO)
	Carrier frequency	DL 2.6GHz (band7)
	CP type	Normal
	System bandwidth	5MHz,10MHz,20MHz
	Modulation scheme	QPSK,16-QAM,64-QAM

1) Execution time and constellation: In this experiment, the receiver uses the method of multithread parallel computing to do baseband signal processing and calculates the execution time of the downlink(DL) procedures of each subframe. The results are shown in Table II. In the meantime, symbols after equalization are written into files and Fig 6 shows the constellation of received signals with various modulation schemes.



(a) Signals with QPSK (b) Signals with 16QAM (c) Signals with 64QAM

Fig. 6. Signal constellation diagrams after channel equalization in 20MHz with different modulation schemes

As we can see, as the system bandwidth is increased and the higher order modulation scheme is used, the time required for processing the data is increased. The original OAI system cannot work properly when the execution time is more than 1ms because of its serial processing method. Using our proposed processing method, the UE can get the right decoded signal even in 20MHz bandwidth of 64-QAM modulation scheme with 2.688 ms execution time. Compared to 10 ms maximum execution time, the system still has great potential to add more modules and validation new technologies.

2) Thoughput comparison: With the same system configurations in Table I, the real-time experiments are carried out with the OAI original scheme and the proposed scheme. The throughputs of the systems are shown in Fig 7.

From the results, we can conclude that the original OAI system can perform well in the 5MHz bandwidth, but due to its single-thread processing method, the system will crash in the situation of 10MHz bandwidth with the index of modulation coding scheme (MCS) higher than 20 and the 20MHz bandwidth with the MCS index higher than 10. With our proposed processing method, the throughput of the system remains the same with the original OAI when the receiver is in the working state and increases as the MCS index grows higher with wider bandwidth. Limited by the equalizer's performance of OAI, the maximum throughput of the system is reached when the mcs index equals 22, at 34905.5 kbps in 20MHz.

TABLE II. AVERAGE EXECUTION TIME OF EACH SUBFRAME

System bandwidth	Modulation scheme	Time consumption(ms)
5MHz	QPSK	1.180
	16-QAM	1.382
	64-QAM	1.680
10MHz	QPSK	1.235
	16-QAM	1.570
	64-QAM	1.942
20MHz	QPSK	1.435
	16-QAM	2.092
	64-QAM	2.688

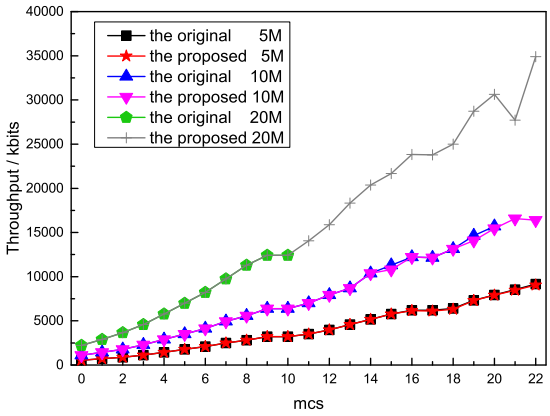


Fig. 7. The throughput of the real-time systems with different processing method.

V. Conclusion

In this paper, we present the OAI platform as a better approaching to the implementation of open-source SDR-based LTE systems and the research for 5G. The features of flexibility and reusability of open-source SDR make OAI the ideal platform to verify the candidate techniques for 5G. We first introduce the structure of the platform and the system architecture of real-time experimentation. Then we analyze the baseband signal processing of OAI, and due to the limit of original OAI processing method, we propose the multithread parallel computing way to do baseband signal processing. Experiments were carried out with the proposed method to evaluate the performance in the real-time wireless environment. Results shown in this paper illustrate that the improvement of baseband signal processing helps the UE be able to work properly in 20MHz bandwidth with 64-QAM modulation scheme. It is clear from the results that there is still a large potential to the extension of the current system with its open and flexible feature. The platform can be useful and helpful for learning and researching LTE systems, validating the key technology of 5G, making demonstrations and prototypes, etc. In future, we will continue to optimize the system and enhance its stability, to extend this platform for flexible deployment of 5G experimentation.

References

[1] A. A. Abidi, "The path to the software-defined radio receiver," *IEEE J. Solid-State Circuits*, Vol. 42, No. 5, pp.954-966, May 2007.

[2] T. Ulversoy, "Software Defined Radio: Challenges and Opportunities," *IEEE Communications Surveys & Tutorials*, Vol. 12, No. 4, pp.531-550, Fourth Quarter 2010.

[3] X. Xiong, W. Xiang, K. Zheng, H. Shen and X. Wei, "An open source SDR-based NOMA system for 5G networks," *IEEE Wireless Communications*, Vol. 22, No. 6, pp.24-32, December 2015.

[4] N. Nikaein, MK. Marina, et al. "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, Vol. 44, No. 5, October 2014, pp.33-38.

[5] OpenAirInterface, *http://www.openairinterface.org/*

[6] S. Coleri, M. Ergen, A. Puri and A. Bahai, "Channel estimation techniques based on pilot arrangement in OFDM systems," in *IEEE Transactions on Broadcasting*, Vol. 48, No. 3, pp. 223-229, September 2002.

[7] R. Wang, Y. Peng, et al. "OpenAirInterface-an effective emulation platform for LTE and LTE-Advanced," in *2014 6th International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2014, pp.127-132.

[8] *3GPP TS 36.211 version 12.3.0*, "Evolved Universal Terrestrial Radio Access (E-UTRA);Physical channels and modulation," pp.10-12, October 2014.

[9] USRP Hardware Driver, *http://ettus-apps.sourcerepo.com/redmine/ettus/projects/uhd/wiki*

[10] I. Alyafawi, E. Schiller, et al. "Critical issues of centralized and cloud-ified LTE-FDD Radio Access Networks," in *2015 IEEE International Conference on Communications(ICC)*, June 2015, pp.5523-5528.

[11] N. Nikaein, R. Knopp, et al. "OpenAirInterface 4G: an open LTE network in a PC," in *the 20th Annual International Conference on Mobile Computing and Networking*, August 2014, pp.305-308.

[12] Y. Yaseen, "Multi-core Processing-Advantages and Challenges," *Student thes., University of Jordan Computer Engineering Department*, 2009.

[13] E. Yip, P. S. Roop, M. Biglari-Abhari and A. Girault, "Programming and Timing Analysis of Parallel Programs on Multicores," in *2013 13th International Conference on Application of Concurrency to System Design (ACSD)*, July 2013, pp.160-169.