Design Issues and Performance Evaluation of a SDR-based Reconfigurable Framework for Adaptive OFDM Transmission *

Milan Zivkovic, Rudolf Mathar
Institute for Theoretical Information Technology
RWTH Aachen University, Germany
D-52056 Aachen, Germany
{zivkovic, mathar}@ti.rwth-aachen.de

ABSTRACT

The investigation and assessment of information theoretic concepts for wireless resource management in real-world scenarios requires flexible testbeds with wide range of reconfigurable parameters. These functionalities are currently offered only by Software Defined Radio (SDR) technology based on general purpose hardware. In this paper a reconfigurable framework for adaptive OFDM transmission, named TIGR, is presented. Based on combination of GNU Radio platform and CORBA communication model, TIGR allows for capacity achieving OFDM-based data transmission between two or more network nodes with optimally configured transmission parameters for given system constraints. Furthermore, a highly reconfigurable framework with large set of adaptable parameters allows for implementation and experimental evaluation of various transmission strategies for different classes of given requirements.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication; C.4 [**Performance of Systems**]: Designu studies

General Terms

Algorithms, Design, Experimentation, Performance, Measurement

Keywords

OFDM, Rate and power allocation, Dynamic Spectrum Access, GNU Radio, CORBA

*This work has been supported by the UMIC Research Center, RWTH Aachen University.

WiNTECH'11, September 19, 2011, Las Vegas, Nevada, USA.

1. INTRODUCTION

Broadband wireless standards, such as WiFi, WiMAX, and LTE, are based on Orthogonal Frequency Division Multiplexing (OFDM) [1], a multicarrier modulation scheme which provides strong robustness against intersymbol interference (ISI) by dividing the broadband channel into many orthogonal narrowband subchannels in such a way that attenuation across each subchannel stays flat. An important task in the design of future OFDM based system is to exploit frequency diversity offered by broadband channel by adaptable transmission parameters (bandwidth, coding/data rate, power) in order to preserve power and bandwidth efficiency according to subchannel conditions at the receiver.

The purpose of this paper is to present the design efforts and basic concepts of TIGR, a modular, SDR based reconfigurable framework which allows for adaptive OFDM transmission with large set of adaptable parameters for different radio scenarios. Developed at the Institute for Theoretical Information Technology (TI) at RWTH University in Germany, TIGR is based on GNU Radio framework [2], a free and open source software toolkit based on hybrid C++/Python programming model that provides library of signal processing blocks for developing communications systems and conducting experiments in different radio scenarios. GNU Radio runs in real time and can be interfaced with RF hardware, thus allowing for transition from experimentation to deployment within the same framework. Additional flexibility and reconfigurability which enables parameter exchange among different components within the system are introduced by adding Common Object Request Broker Architecture (CORBA) [3], an event-based distributed communication model that allows interaction between software elements located in different logical and/or physical entities. In SDR context CORBA can be also concerned as middleware which provides interfaces between the different software components within the system.

The control and feedback mechanisms provided by TIGR are based on CORBA allowing for optimal assignment of predefined transmission parameters at the input and estimation of link quality at the output. High flexibility, provided by large set of reconfigurable parameters, which are normally static in real systems, enables implementation and assessment of different resource allocation strategies for various classes of system requirements.

In this paper we will present the current state of TIGR framework and supported functionalities while discussing

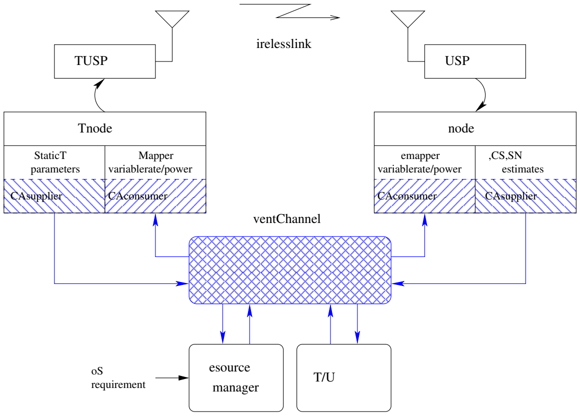

Figure 1: The system overview

the ongoing development issues. In the next section, system model will be introduced explaining basic TIGR functionalities, while different components of the system will be separately introduced. Afterwards, two experimental scenarios and corresponding implementations will be presented in Section 3. Finally, some concluding remarks and outline are drawn in Section 4.

2. SYSTEM MODEL

The system diagram of TIGR framework is shown in Fig. 1. Transmitter and receiver node are composed of a host commodity computer and general purpose RF hardware, Universal Software Radio Peripheral (USRP) [4]. Baseband signal processing at host computers is implemented in GNU Radio framework, while the USRP performs computationally intensive operations as filtering, up- and downconversion controlled through a robust application programming interface (API) provided by GNU Radio.

The communication between transmit and receive node is organized as reconfigurable continuous one-way transmission of OFDM symbol frames. As shown in Table 1, the set of input configuration parameters can be divided into two classes. The set of static parameters containing FFT size, number of subchannels, frame size, etc., is initialized at transmission start and is known to both nodes. The set of dynamic parameters which are reconfigurable at run-time includes total transmit power, carrier frequency and allocated rate and power over subchannels. TIGR OFDM frame structure is shown in Fig. 2. The sequence of 10 data symbols is preceded with 2 preambles (one synchronization preamble and one used for channel estimation) and one ID symbol which is used for CORBA synchronization.

2.1 CORBA integration

The backbone of the system is realized over local Ethernet network by CORBA event service, a distributed communication model that allows an application to send an event that will be received by any number of objects located in different logical and/or physical entities. The typical communication model through an *event channel* is shown in Fig. 3. Estimated parameters that indicate link quality (average S(I)NR, CSI, and BER) and current static transmitter's parameters are supplied as CORBA events to *event channel* which allows other components (consumers) within the system to register their interests in events.

From a supplier's perspective, the *event channel* appears as a single consumer, while from a consumer's perspective, the *event channel* appears as a single supplier. In this way, suppliers and consumers are decoupled where any number of suppliers can issue events to any number of consumers while using a single *event channel*.

Table 1: TIGR OFDM symbol parameters

Bandwidth (*static*)	Variable, up to $8MHz$
FFT length (*static*)	$64 - 1024$
Frame length (*static*)	Variable
Carrier frequency (*dynamic*)	$2400 - 2483MHz$
Modulations (*dynamic*)	BPSK, QPSK, 8-PSK, 16-QAM, 32-QAM, 64-QAM, 128-QAM, 256-QAM
Power (*dynamic*)	Up to 20 mW

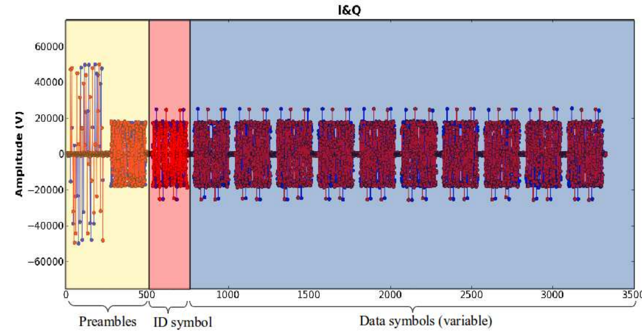

Figure 2: Frame structure

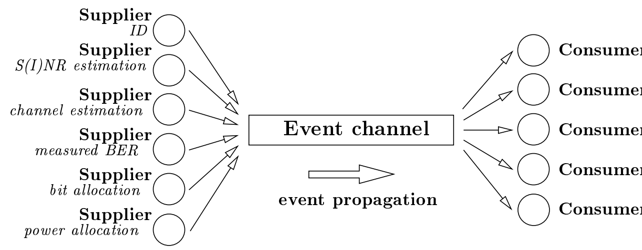

Figure 3: Communication model through an Event channel

2.2 Resource manager

The central control unit that determines optimal input transmission parameters for given requirements is *resource manager*, whose basic functionalities are depicted in Fig. 4. Controlled by interactive GUI it consumes supplied events

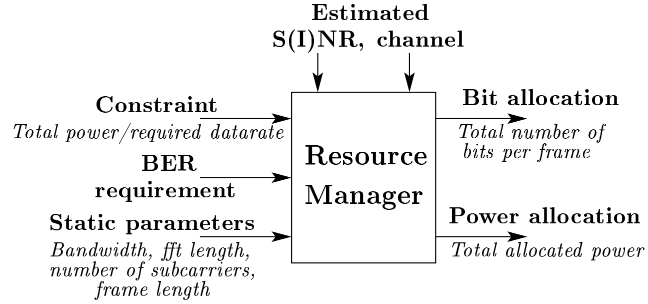

Figure 4: Resource manager

Figure 5: The transmitter's GUI

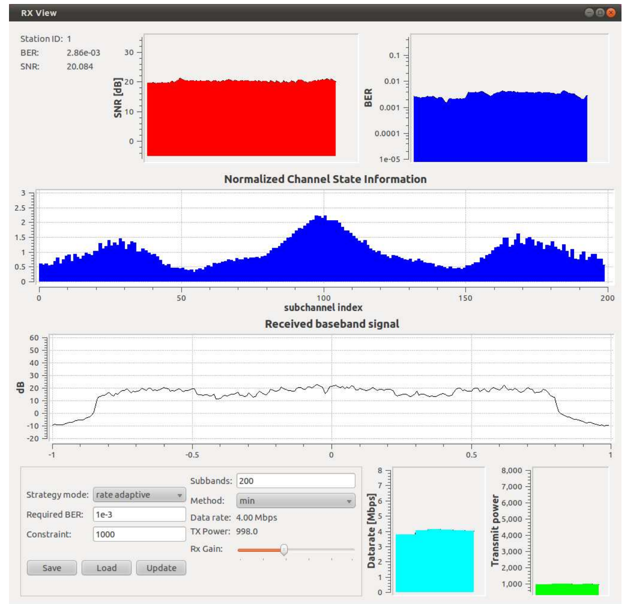

Figure 6: The receiver's GUI with interactive control interface

forwarded from event channel, performs allocation in an optimal manner, and supplies new transmission parameters, i.e. total transmit power and power/rate per subchannel (rate and power allocation), which are finally consumed by other components in the system. In the current stage of TIGR framework, *resource manager* is implemented as Python class, thus allowing for flexible design of different scenarios, such as grouping of subcarriers in subbands in the presence of weak frequency selective channels or enabling an opportunistic use of available spectrum through Dynamic Spectrum Access (DSA). The two of them are more detailed described in Section 3.

2.3 GUI

The Graphical User Interface (GUI), facilitating the demonstration, is developed in Qt C++ framework. The transmitter's GUI contains static transmission parameters and current allocation of rate and power over subchannels, as shown in Fig. 5. Furthermore, the receiver's GUI, given in Fig. 6, dynamically shows estimated channel parameters (average S(I)NR, CSI, BER) and contains interactive interface for controlling allocation strategy in *resource manager*, thus allowing for on the fly configuration of the number of allocation subbands in resource allocation, given certain constraints and requirements. For example, framework can reconfigure from per subcarrier allocation unit for data rate maximization based on SNR measurements to per subband allocation unit in DSA environment based on SINR measurement.

2.4 Transmitter and receiver implementation

Signal processing block are implemented as C++ classes and connected in directed acyclic graph forming a *flowgraph* in Python. *Flowgraphs* are implemented as Python scripts which take care of communication among signal processing blocks. The most of the used signal processing blocks are developed at TI and are further referred with the prefix *ofdm_*, while CORBA blocks have prefix *corba_*. The blocks from standard GNU Radio library are usually prefixed as *gr_*.

There are two basic scenarios which are concerned. The first assumes the RF transmission where transmitter and receiver are interfaced with the USRPs as separated Python top blocks. The second is the simulation mode where transmitter and receiver "communicate" over an artificial channel, without the RF interface (USRPs), thus forming one Python top block. That allows for excluding of unknown distortions caused by hardware, but also for system evaluation in the presence of various controllable channel and hardware impairments such as carrier frequency offset (CFO), sampling frequency offset (SFO), additive noise, interference, and multipath propagation. An example of simulation channel is shown in Fig. 7.

2.4.1 Transmitter

The *flowgraph* of the TIGR OFDM transmitter is shown in Fig. 8. The transmitter is controlled by *corba_tx_control*, a Python hierarchical block containing *corba_* C++ blocks

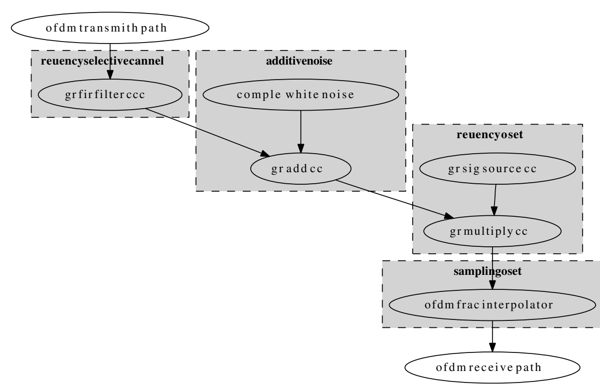

Figure 7: Simulation channel between the transmit and receive chain

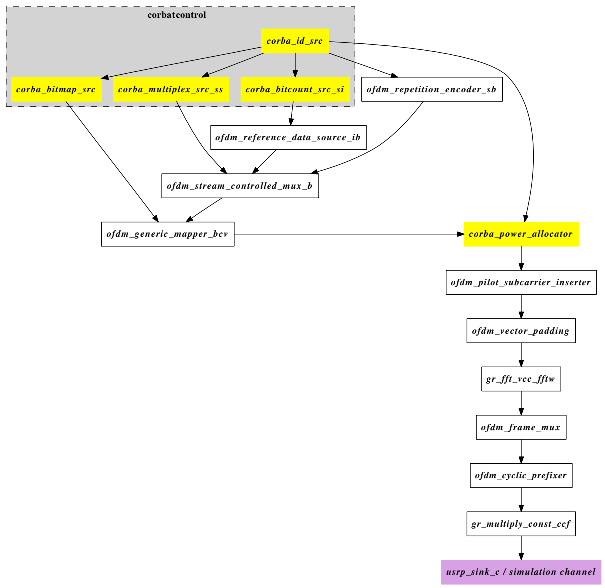

Figure 8: The TIGR OFDM transmitter

which represent wrappers for CORBA consumers. The control of the whole adaptation procedure starts from *corba_id_src* which consumes ID (realized as a simple counter) from the *event channel*. Every time when ID is changed it "triggers" other CORBA blocks to consume events from the *event channel*.

The ID is also encoded with repetition code and is being included in the frame after preambles as shown if Fig. 2. The functionality of *corba_bitmap_src* is, whenever ID at its input is changed to connect to event channel, to consume the event which is the bit allocation vector and finally, to forward this vector to the *ofdm_generic_mapper*. Similarly, *corba_bitcount_source_src_si* consumes the bit allocation and, given the given frame size, sum the number of bits which will be transmitted within one frame.

Block *ofdm_reference_data_source_id* reads that number of bits from the stored random data file and supplies it to *ofdm_stream_controlled_mux_b* which multiplexes it with the encoded ID, thus forming a data part of the frame. The output of this block is forwarded to *ofdm_generic mapper_bcv* which actually performs mapping of incoming bits to complex signals. The block is triggered for each frame and performs mapping according to *corba_bitmap_src*.

The ID OFDM symbol is always transmitted with BPSK modulation in order to provide safe communication to the receiver. Following the mapper, the signal is derived to *corba_power_allocator* which loads on individual or grouped subcarriers the power allocation vector consumed from *event channel*. The data stream is further forwarded to standard OFDM transmit chain. Firstly,*ofdm_pilot_subcarrier_inserter* inserts stored vector of pilot subcarriers into the each data OFDM symbol. Currently, static configuration of 8 pilot subcarriers is supported, but in general TIGR allows for inserting arbitrary number of pilots depending of FFT length and the number of available data subcarriers. Then, before performing IFFT, each data symbol is padded with the zeros at the edges forming the guard bands in order to avoid out-of band transmission due to high sidelobes while also nulling out 2 subcarriers in the middle to avoid unwanted DC and

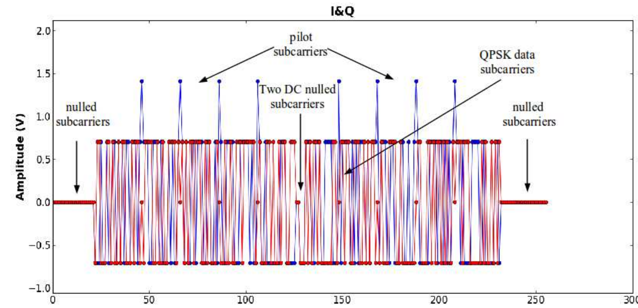

Figure 9: Frequency domain representation of QPSK modulated OFDM data symbol

low-frequency components generated by the receiver's front-end.

The number of nulled DC and side subcarriers is also reconfigurable and can be defined at the initialization of transmitter's and receiver's script. The zero padded OFDM data symbol, shown in Fig. 9, with FFT length of 256, having 200 data subcarriers, 2 nulled RF subcarriers, 8 pilot subcarriers and 23 nulled subcarriers at the both edges, is derived to IFFT block taken from standard GR library. After IFFT, the time domain signal is forwarded to *ofdm_frame_mux* which prepends two predefined and prestored preambles. Afterwards, to protect against multipath propagation causing intersymbol interference (ISI), cyclic prefix (CP) is prepended to each symbol in the frame. The complex OFDM baseband signal is multiplied with complex amplitude and such signal is then derived to simulated channel or USRP where RFX2400 daughter boards are used to transmit the signal on 2.45GHz.

2.4.2 Receiver

The *flowgraph* of the TIGR receiver is shown in Fig. 10. with incoming complex (baseband) data stream from simulated channel or USRP (downconverted signal after ADC). It is clear that the receive chain is more complex while processing requires greater computational efforts compared to the transmitter, due to synchronization and data acquisition that is required for decoding of the transmitted signal. In general, besides receiver chain, TIGR also allows for simultaneous observation of the spectral content by putting incoming complex data stream to Power Spectral Density (PSD) analyzer whose output is also supplied to *event channel* and consumed at receiver's GUI as shown in Fig. 6.

The first synchronization stage is symbol timing synchronization which outputs the timing trigger (start of the each OFDM frame). We implemented slightly modified version of Schmidl & Cox timing estimator [5] which is based on correlation of time periodic preamble structure. The resulted timing metric for received SNR = 10 dB is shown in Fig. 11 which is actually taken as a screenshot from TIGR GUI. The exact timing trigger is located within plateau that is induced by CP insertion. For further information on experimentation with different timing synchronization algorithms in GNU Radio, readers are referred to [6]. Since first preamble is constructed to have eight identical parts by nulling the subcarriers between each eighth subcarriers, *ofdm_time_sync* has two outputs, denoted 0 and 1, where 0 is the time synchronized complex data stream while the 1 is the frame (timing) trigger.

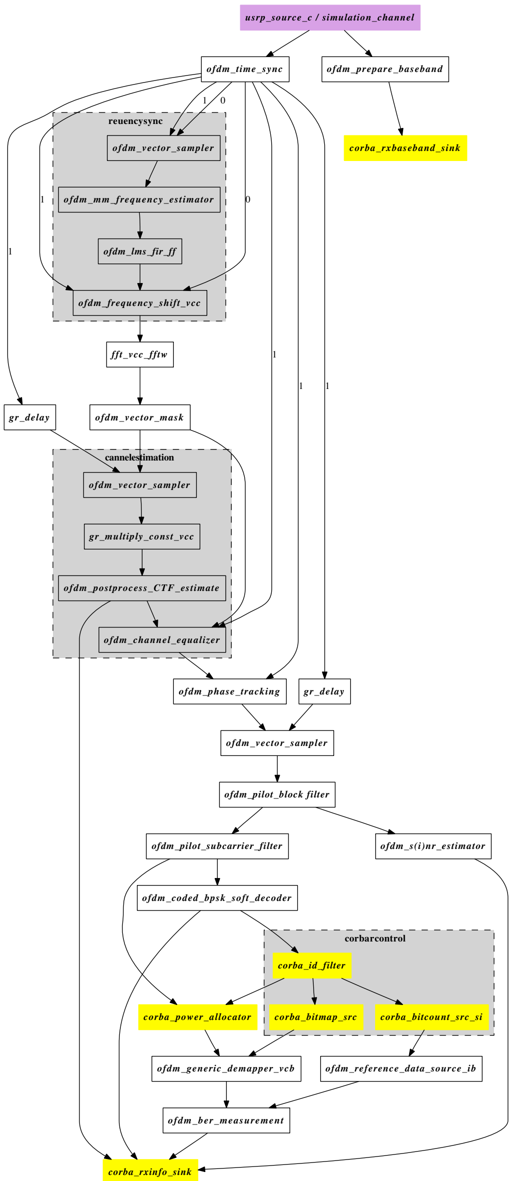

Figure 10: The TIGR OFDM receiver

After initial timing estimation, the receiver has to estimate carrier frequency offset (CFO) which arises due to the mismatch between transmitter's and receiver's oscillator. It is customary to divide the CFO into an integer part (ICFO), a multiple of subcarrier spacing, which produces a shift of subcarrier indices, plus a fractional part (FCFO) which results into interchannel interference (ICI) due to loss of orthogonality among subcarriers. All standard designs of receiver first assumed the stage of the fine frequency synchronization (FCFO estimation based on the correlation of preamble structure in the time domain) while the stage of ICFO estimation is performed after FFT and is based on the correlation of frequency domain structure of used preamble(s). However, in order to avoid already increased receiver complexity, by leaving ICFO estimation out, we implemented the method of Morelli and Mengali [7] which suggests the increase of the number of time periodic

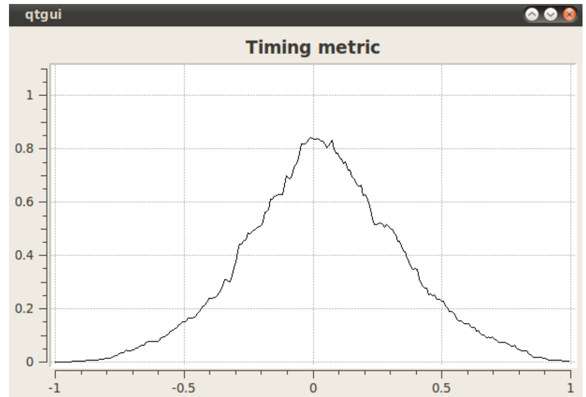

Figure 11: The Schmidl & Cox timing metric at SNR $= 10$ dB

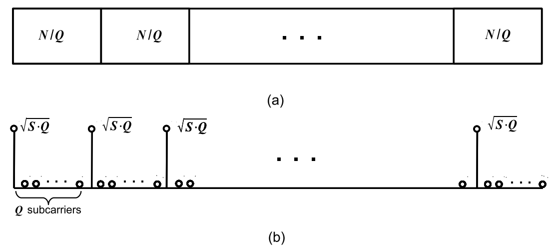

Figure 12: Preamble structure in (a) time and (b) frequency domain

parts in order to extend acquisition range of the CFO. In TIGR implementation, after some experimental investigation, we fix number of periodic parts to 8, thus allowing CFO acquisition in range of $(-4, +4]$ subcarrier's spacings Δf which was sufficient for the signal bandwidth $(2-8$ MHz) we transmit using USRPs. The structure of synchronization preamble is show on in Fig. 12. Estimated CFO is then filtered and phase of the data is corrected. Nevertheless, there is still residual CFO and SFO that needs to be corrected.

Following the CFO estimation, FFT is performed bringing the signal to frequency domain. In order to compensate any amplitude or phase noise introduced by the wireless channel, equalization needs to be performed by means of LS estimation based on multiplication of the second preamble with the stored conjugated replica of transmitted version. Channel estimates from *ofdm_postprocess_CTF_estimate* are then derived to channel equalizer and also supplied to event channel in order to be consumed at resource manager as RA inputs and shown at GUI as depicted in Fig. 6. Due to residual CFO and SFO additional phase correction is performed in *ofdm_phase_tracking* based on the phase tracking of pilot subcarriers.

Then, in order to obtain only the data symbols (including ID symbol), preambles and pilot subcarriers are removed in *ofdm_pilot_block_filter* and *pilot_subcarrier_filter*, respectively. The TIGR also performs SNR or SINR estimation based on the synchronization preamble in order to provide an information about signal quality to the *resource manager*. The implemented algorithm can be found in authors' previous work [8] [9].

After realocating the power on data subcarriers in CORBA contolled *corba_power_allocator*, the demapping of complex

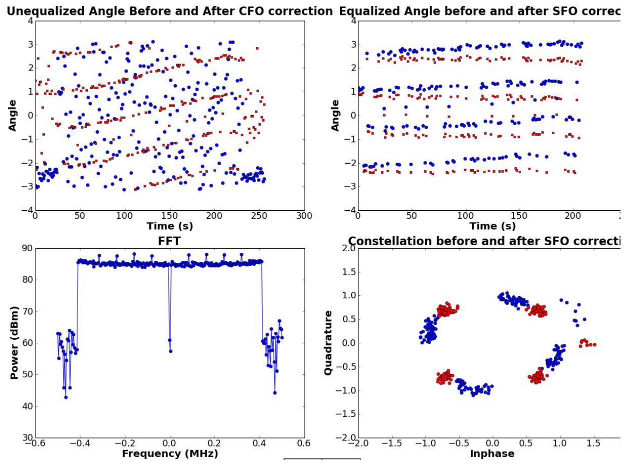

Figure 13: The TIGR receiver performance for QPSK signal in the simulated channel at SNR $= 25$ dB, CFO $= 0.5\Delta f$, and SFO $= 20$ ppm

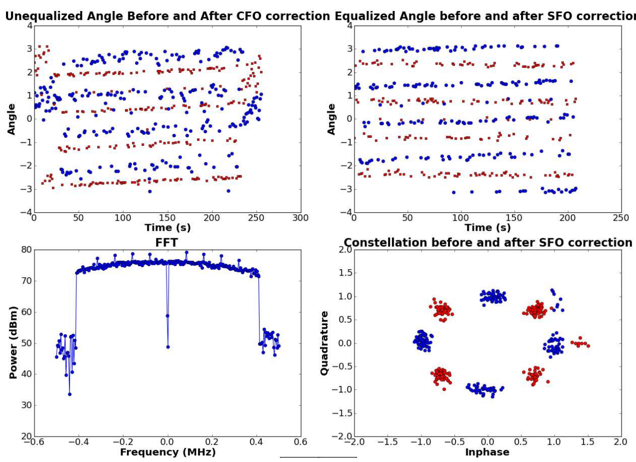

Figure 14: TIGR receiver performance for QPSK signal in the RF channel at 2.45 GHz carrier frequency and 1 MHz bandwidth estimated SNR $= 25$ dB

samples to bitstream is performed in *ofdm_generic_demapper_vcb* according to bitmap allocation taken from CORBA event channel in *corba_bitmap_src*, similarly as it was done at the transmitter. In order to measure BER, data from the same random file as the one which is stored at the transmitter is derived to *ofdm_ber_measurement* block which performs BER measurement and supplies it to the event channel. The S(I)NR estimation, channel estimation, BER measurements results and current ID block are derived to *corba_rx_info_sink* containing wrappers for CORBA suppliers which enable for pushing them into the *event channel*.

The TIGR receiver performance in simulated AWGN channel for QPSK modulated subcarriers at SNR $= 25$ dB, CFO $= 2.5 \cdot \Delta f$, and SFO $= 20$ ppm is shown in Fig. 13 using modified plotting tool available in standard GNU Radio distribution. The spectrum (PSD) of received signal is depicted in the bottom left corner. The upper left plot shows the phase of received data samples before (blue) and after (red) frequency synchronization performed on time domain signals. It can be noticed that the effect of residual CFO and

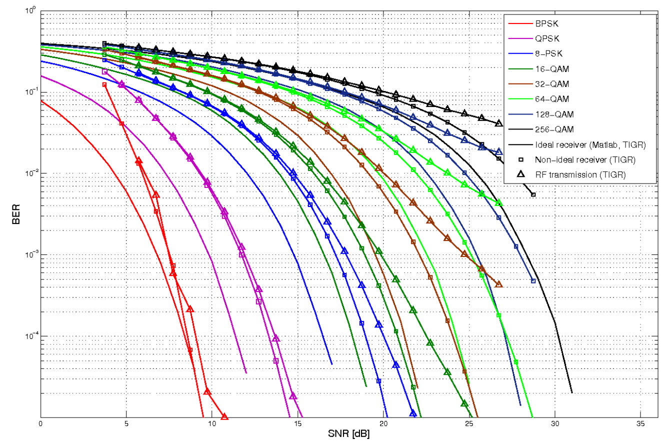

Figure 15: BER vs. SNR performance

SFO is still present in the received samples inducing linear phase increase over subcarriers as shown in upper right plot. This plot depicts the input (blue) and the output (red) of *ofdm_phase_tracking* showing the effect of phase shift tracking based on pilot subcarriers. Appropriate constellation plot in bottom right shows how phase shift is corrected in I/Q plane before deriving data symbols to demapper. The performance of the TIGR receiver in RF channel is shown in Fig. 14. The RF link between two nodes equipped with USRP devices is configured to carrier frequency of 2.45 GHz and 1 MHz bandwidth, while transmit amplitude is adjusted such that the estimated SNR at the receiver is 25 dB. Similar to the case of simulated channel, the effects of the frequency mismatch between transmitter's and receiver's oscillator (CFO) and SFO as well as their compensation by the TIGR receiver can be observed in given plots.

3. IMPLEMENTED SCENARIOS

3.1 Discrete waterfilling (bit/power loading)

The first scenario is concerned with the implementation of adaptive bit and power loading over subcarriers given the channel conditions, usually quantified by received SNR. For given Quality of Service (QoS) demands, commonly expressed through target Bit Error Rate (BER), subcarriers with higher SNR, determined by actual channel state and loaded power, can sustain higher-order modulation, thus allowing for higher data rate. Therefore, the allocation of total power can be optimized in order to improve system performance. This can be formulated as an optimization problem which can be solved by designing an efficient resource allocation algorithm.

There are basically two approaches which bring an optimal solution to this problem:

- Rate adaptive (RA) optimization, which maximizes the overall rate given a fixed total power,
- Margin adaptive (MA) optimization, where the total transmitted power is minimized maintaining certain service rate demand.

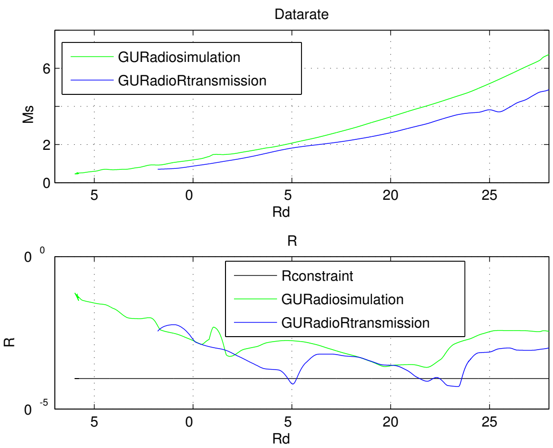

Figure 16: Datarate and BER vs. SNR performance

The authors in [10] proposed optimal solution, named discrete waterfilling, assuming uniform granularity among supported modulation schemes. The basic idea lies in iterative procedure of allocating/dealocating power from those subcarriers which requires the smallest/largest energy increase/decrease in order to switch to higher/lower modulation level, while not violating given constraints.

In order to evaluate the performance of RA Levin-Campello algorithm [10] in TIGR framework, the power-rate functions (BER performance) for supported modulation schemes, given in Table 1, are determined in order to calibrate the required SNR for different scenarios. Therefore, the BER performance of TIGR link for the case of ideal receiver, nonideal receiver and RF transmission is derived and shown in Fig. 15. The ideal receiver assumes ideal estimation of all impairments at the receiver, where only AWGN presence influences different BER values for different SNRs. It is shown that simulated performance agrees with theoretical curves. The BER performance for non-ideal receiver assumes the presence of estimation noise, i.e., imperfections in parameters estimation caused by specific implementation. It can be seen that for all modulations, except BPSK, for BER $< 10^{-2}$ there is constant gap of $2-3$ dB between the ideal and nonideal cases. Finally, power-rate functions in the case of RF transmission indicates that additional hardware impairments, such as DAC/ADC quantization noise and PAPR, influences the performance of the system in such a way that the SNR gap between ideal and derived curve increases with the decrease of BER. Those impairments are

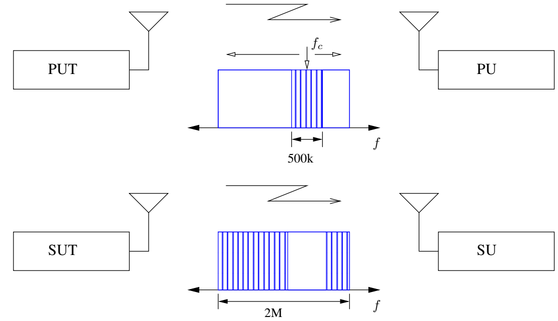

Figure 18: The DSA scenario

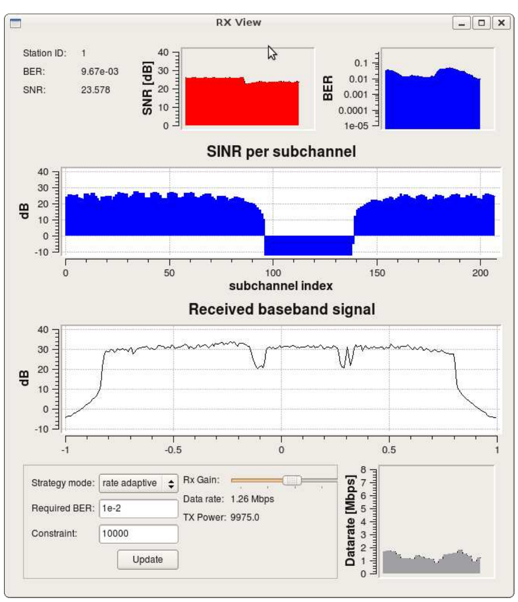

Figure 17: The receiver's GUI in DSA scenario when narrowband PU is detected

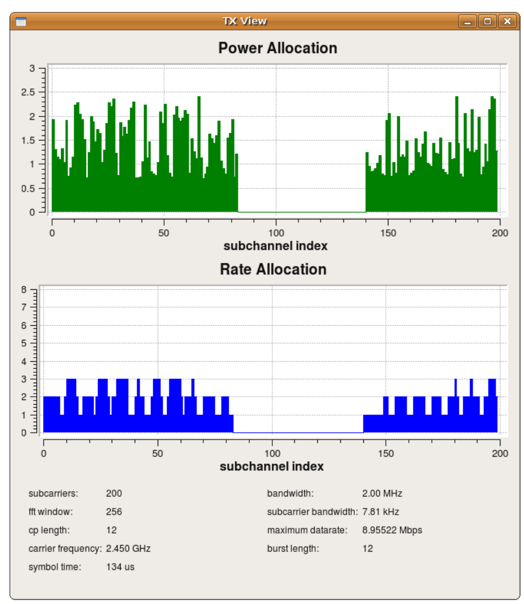

Figure 19: The transmitter's GUI in DSA scenario showing bit/power allocation in the presence of interferer

still being analyzed and modeled in order to be included in present simulation model.

Having the performance of power-rate functions, we calculated required SNR for achieving particular BER for each modulation scheme that is required for implementation of discreet waterfilling. The performance is for RA scenario when total transmitted power (amplitude) constraint is varied (thus received SNR is increased) and rate is maximized is shown in Fig. 16. Those results are averaged over 10 measurements. As it was expected, given the BER performance in Fig. 15, simulated achieved data rate is higher compared to the case of RF transmission. It can be noticed that in both cases of non-ideal receiver in simulated channel and RF transmission, BER constraint is exceeded due to additional noise introduced by estimation stages and hardware impairments. Therefore, those impairments should be appropriately modeled and included in optimization procedure.

3.2 DSA scenario

The TIGR framework is also used to demonstrate the DSA scenario that allows several standards or users to opportunistically share the available spectrum resources without introducing mutual interference. Specifically, the given scenario assumes interference-free coexistence of two OFDM-based systems within a common frequency band as shown in Fig. 18. The first system is primary user (PU) transmitter-receiver pair which operates in narrowband randomly changing the portion of occupied spectrum, and in certain way emulates the frequency hopping of GSM signal. Accordingly, SU pair operating within the whole available band, continuously monitors and detects parts of non-used spectrum by measuring SINR over subcarriers based on the method given in [11] and performs capacity achieving OFDM transmission with optimal rate and power allocation over subchannels for given system constraints. The receiver's GUI for the case when PU is detected is shown in Fig. 17, while appropriate bit/power loading is given in Fig. 19. The actual bit and power loading is determined by estimated SINR values. The DSA scenario is successfully demonstrated at DySPAN 2010 technical conference [12].

4. CONCLUSION AND FUTURE WORK

In this paper we have described the design issues and performance results for TIGR framework, a reconfigurable testbed for adaptive OFDM transmission based on GNU Radio platform and CORBA communication model which enables additional adaptivity and reconfigurability features within the system. The TIGR supports for continuous capacity achieving OFDM transmission with optimal rate and power allocation over subchannels for given system constraints. Proposed system extends PHY layer functionalities of current wireless standards and offers control and feedback mechanisms for easy reconfiguration of transmission parameters allowing evaluation of different strategies in either simulation or real-time scenarios. Basic insight into interaction between CORBA and GNU Radio software components is introduced and basic adaptivity is presented. Two experimental scenarios are described bringing general conclusion that estimation noise and hardware impairments should be appropriately modeled and included into resource allocation procedures, which is usually neglected in algorithms design.

However, additional experimental evaluation with USRP2 and different indoor conditions is to be performed in order to get larger insight into the TIGR capability. Current efforts are put on the profiling of the transmitter/receiver execution in order to optimize signal processing code, thus allowing for higher bandwidths. Additionally, TIGR is close to support an additional feature of adaptive Coded OFDM (COFDM), where the rate and coding adaptation is jointly performed, thus bringing the framework closer to support current wireless standards (WiMAX, LTE). Also MIMO implementation is considered to be added while some initial work is already done in this direction. Available flexibility also allows for inserting some advanced features as cyclostationary signatures in order to support additional cognitive radio scenarios.

5. ACKNOWLEDGMENTS

Authors would like to acknowledge Dominik Auras for his tireless and disinterested assistance during the development of TIGR.

6. REFERENCES

[1] Y. G. Li and G. L. Stuber. *Orthogonal Frequency Division Multiplexing for Wireless Communications*. Springer, 2006.

[2] ”GNU Radio”, `http://gnuradio.org/redmine/projects/gnuradio/wiki`.

[3] ”Object Management Group”, `http://www.omg.org/`.

[4] ”USRP”, `http://www.ettus.com`.

[5] T. Schmidl and D. Cox. Robust Frequency and Timing Synchronization for OFDM. *IEEE Trans. Commun.*, 45(12):1613–1621, Dec 1997.

[6] T. Rondeau, M. Ettus, and R. McGwier. Open Source Transparency for OFDM Experimentation. In *Software Defined Radio Forum Technical Conference*, 2008.

[7] M. Morelli and U. Mengali. An Improved Frequency Offset Estimator for OFDM Applications. *IEEE Commun. Lett.*, 3(3):75–77, Mar 1999.

[8] M. Zivkovic and R. Mathar. Preamble-based SNR Estimation in Frequency Selective Channels for Wireless OFDM Systems. *Proc. of IEEE VTC 2009 Spring*, 2009.

[9] M. Zivkovic and R. Mathar. An Improved Preamble-based SNR Estimation Algorithm for OFDM systems. *Proc. of IEEE PIMRC 2010, Istanbul, Turkey*, pages 172 –176, 2010.

[10] J. Campello. Practical Bit Loading for DMT. In *Communications, 1999. ICC '99. 1999 IEEE International Conference on*, volume 2, pages 801 –805 vol.2, 1999.

[11] M. Zivkovic and R. Mathar. Joint Frequency Synchronization and Spectrum Occupancy Characterization in OFDM-based Cognitive Radio Systems. To appear in *Proc. of IEEE Globecom 2011, Houston, USA*, 2011.

[12] M. Zivkovic, D. Auras, and R. Mathar. OFDM-based Dynamic Spectrum Access. In *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, pages 1 –2, april 2010.