State-action based Link Layer Design for IEEE 802.11b Compliant MATLAB-based SDR

Ramanathan Subramanian, Eric Doyle, Benjamin Drozdenko, Miriam Leeser, Kaushik R. Chowdhury
rsubramanian@coe.neu.edu, doyle.e@husky.neu.edu, bdrozdenko@coe.neu.edu, mel@coe.neu.edu, krc@ece.neu.edu

***Abstract*—Software defined radio (SDR) allows unprecedented levels of flexibility by transitioning the radio communication system from a rigid hardware platform to a more user-controlled software paradigm. However, it can still be time consuming to design and implement such SDRs as they typically require thorough knowledge of the operating environment and a careful tuning of the program. In this work, we describe a systems contribution and outline strategies on how to create a state-action based design in implementing the CSMA/CA/ACK MAC layer in MATLAB® that runs on the USRP® platform, a commonly used SDR. Our design allows optimal selection of the parameters so that all operations remain functionally compliant with the IEEE 802.11b standard (1Mbps specification). The code base of the system is enabled through the Communications System Toolbox™ and incorporates channel sensing and exponential random back-off for contention resolution. The current work provides a testbed to experiment with and enables creation of new MAC protocols starting from the fundamental IEEE 802.11b compliant standard. Our system design approach guarantees the consistent performance of the bi-directional link and we include the experimental results for the three node system to demonstrate the robustness of the MAC layer in mitigating packet collisions and enforcing fairness among nodes.**

I. Introduction

Software defined radios (SDRs) enable fine-grained control of their operation by executing the processing steps in user-modifiable software [1]. SDRs form the building block for applications needing high levels of reconfigurability, such as wireless access points that support multiple wireless standards, or for systems like cognitive radios that employ situational intelligence to evolve with the radio frequency (RF) environment [2]. The design concept of the SDR is advantageous because it reduces the need for special purpose hardware and allows the developer to add new functionality to the radio by modifying the software.

This paper details our approach in realizing a link layer on a SDR platform using a common operating environment: MATLAB software and Ettus USRP® N210 hardware [3]. To facilitate quick deployment, it includes an initialization script for the setting and tuning of the reconfigurable parameters based on the specific channel measurements at the chosen experimental site. Importantly, it complies with the processing definitions in the IEEE 802.11b specification, though hardware limitations increase the time to completion of the entire transmission/reception cycle compared to an off-the-shelf hardware-only Network Interface Card.

Our approach advances the state of the art and contributes to the research community in a number of ways. First, our approach faithfully models the DATA and ACK packet structure and implements both PHY- and MAC-layer protocols according to the IEEE 802.11b specifications [4]. Our work provides a testbed to experiment with new MAC protocols. We use common tools for our design methodology, including the Ettus Research Universal Software Radio Peripheral (USRP) hardware, a radio front end commonly used in wireless research. We model our system using a finite state machine (FSM) that transitions only on the clock cycles derived from the USRP. As the basis for our designs, we use MATLAB R2015b and its USRP support package [5]. Using a software-only approach allows full parameter flexibility for the most important variables, allowing the user to reconfigure the system as needed to adapt to environmental changes. Finally, our software is publicly available, released for research purposes under the GNU Public License (GPL) and available for download directly from GitHub [6] and MATLAB Central [7]. The modularity of our code makes it relatively easy to manage, which allows extensibility by the community.

II. System Architecture Overview

Our system architecture and operational steps are shown in Fig. 1. To clearly identify the transmitting and receiving node for a given SDR pair, we use the terms designated transmitter (DTx) and designated receiver (DRx). This terminology avoids ambiguity in describing a bi-directional communication link, where the transmitter must complete its packet transfer and then switch to a receiver role to get the acknowledgement (ACK). Thus, in the subsequent discussion, the DTx alternates between its transmit and receive functions, and the DRx alternates between receive and transmit functions.

In the initialization step, the system is preset with recommended parameters and allows the user to modify a number of parameters for the entire transceiver chain. The user then initiates a parameter exploration stage wherein the transmitter and receiver codes are executed with the user-supplied parameters as constants, and the code cycles through other possible variations (both in terms of the settings of processing blocks as well as entire algorithms themselves), each time identifying the performance that results from these settings.

From this data set, the user determines a feasible set of parameter settings. These parameter settings result in less than 5% packet loss at the receiver. Note that this is the *best case* scenario, as the actual wireless channel will introduce further channel outages. Once the user selects one of the possible feasible configurations returned by the search, the

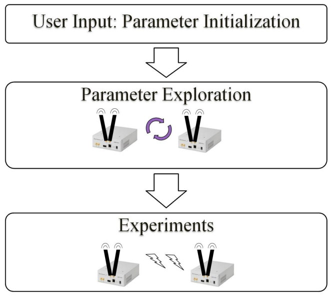

Fig. 1. System Architecture

code is ready for driving the USRP radios for over-the-air experiments. Hence, using a software-only approach and parameterizing the most important variables allows the user to reconfigure the parameter settings as needed to adapt to changes in the radio's environment.

A. Background

We adopt the IEEE 802.11b medium access control (MAC) layer packet structure specifications in our implementation [8]. Our approach collects all the bits in the packet in multiples of 8 octets, which forms one USRP frame. This makes it easy for us to work with the MATLAB system objects, with PHY and MAC header fields in the DATA/ACK having sizes that are multiples of 8 octets. Multiple USRP frames will compose the standard-compliant 802.11b packet.

III. Related Work

A. SDR Platforms

Specialized software is needed to effectively work with the SDR systems and perform the signal processing tasks needed to instantiate wireless communications, such as modulation, preamble detection, encoding, and filtering. [9] describes a bi-directional transceiver that implements a DBPSK PHY layer in MATLAB using standard tools like MATLAB Coder and MEX to speed up processing steps. It employs a *transceive* function that uses the USRP clock to define and bring about slot time synchronized operations. GNU Radio [10] is one of the most widely used SDR programs, owing to the fact that it's open source, hardware-independent, and modifiable. Its GUI, GNU Radio Companion, allows the user to build block diagrams to represent complex encoding and decoding schemes. Built-in modules allow the user to perform various types of modulation (e.g. PSK, QAM, OFDM) and error-correcting codes (e.g. Reed Solomon, Viterbi). An SDR-based testbed that implements a full-duplex OFDM physical layer and a CSMA link layer along with some strategies for establishing bidirectional communications is described in [11]. It involves MATLAB R2013a, MATLAB Coder on USRP-N210 and USRP2 hardware. The PHY layer, based on IEEE 802.11a, incorporates timing recovery, frequency recovery, frequency equalization, and error checking. The CSMA link layer involves carrier sensing based on energy detection and stop-and-wait ARQ. However, this approach requires additional

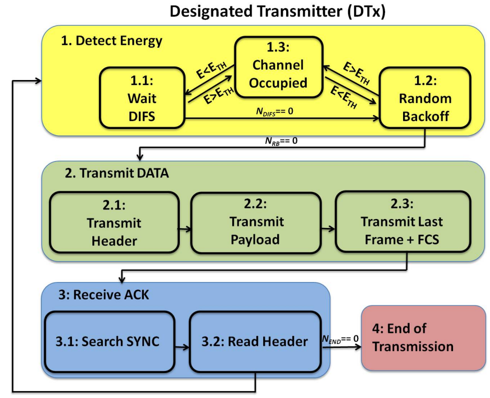

Fig. 2. States for the Designated Transmitter (DTx)

development efforts for improving speed and enabling full-duplex. WARP is scalable, extensible programmable wireless platform produced by Rice University to prototype advanced wireless networks [12]. This platform has been used to build, among many other things, a full duplex IEEE 802.11 network with OFDM and a MAC protocol [13], and a distributed energy-conserving cooperation MAC protocol for MIMO performance improvements [14].

IV. State-action based System Design

We model our system using FSMs. Our approach involves first designing a number of (i) state diagrams to reflect the logical and time-dependent operational steps of our system, and (ii) block diagrams reflecting the sequential order of operations.

A. Designated Transmitter State Machine

To implement the MAC layer carrier sense multiple access with collision avoidance (CSMA/CA) protocol, we identify 4 main states for the DTx, as shown in Fig. 2.

1) Detect Energy: At START, a new USRP frame arrives, and gets stored in a receive buffer. The DTx begins sensing energy in the channel. The DTx decides to move either to a back-off state or to a transmit state depending on whether the channel is busy or not. A random amount of time is chosen uniformly from a progressively increasing time interval. DTx continually senses the channel, and only when the channel is free does it decrement the back-off time; otherwise, it stalls. Only when the back-off time counts down to zero does the DTx attempt to transmit.

2) Transmit DATA: Upon entering this state, the DTx prepares the DATA packet and then, by calling the transceive function continually, places it in the transmit buffer of the USRP which then gets transmitted over the air. After transmitting the DATA, two possibilities exist. The transmission is successful with the reception of an ACK, or the transmission is not successful due to packet collision with other DTxs.

3) Receive ACK: As soon as the transmission is completed, the DTx moves into the Receive ACK state, searching for the Physical Layer Convergence Procedure (PLCP) header in the

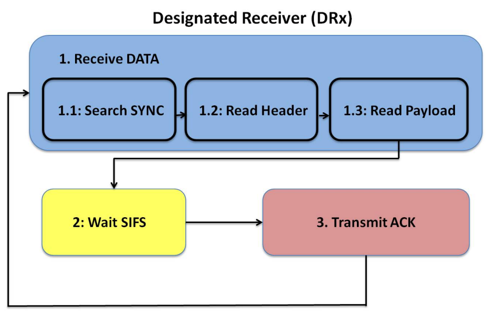

Fig. 3. States for the Designated Receiver (DRx)

received ACK. If that is successful, then the frame control and the address fields are checked for accuracy. The DTx then repeats the above mentioned sequence of steps until the last frame is successfully transmitted. On the other hand, if no ACK is received, the packet is considered lost and the DTx backs-off for an increased random back-off time and re-attempts transmission.

4) End Of Transmission: When there are no more DATA to be sent, the DTx arrives at end of transmission (EOT) state.

B. Designated Receiver State Machine

For the DRx, we identify 3 states as shown in Fig. 3.

1) Receive DATA: When the DRx succeeds in detecting the Preamble and the Start Frame Delimiter (SFD), it reads into the PHY and MAC header and then progresses to extract the payload. Post extracting the last set of payload bits, Frame Check Sequence (FCS) is checked.

2) Wait SIFS: The DRx waits for a fixed interval of time, referred to as Short Inter-frame Space (SIFS), before sending out an ACK packet, post receiving the DATA.

3) Transmit ACK: The DRx sends out an ACK addressed to the DTx when all the payload bits have been received.

V. MAC Layer Design

We first implement the CSMA/CA protocol that allows the nodes to sense the channel and attempt to transmit only when the channel is idle to avoid packet collisions. Then, we modify this base implementation with the standards-specific functions, as described below.

A. MAC Overview

Our Medium Access Control (MAC) layer employs the Distributed Coordination Function (DCF) strategy incorporating the CSMA/CA mechanism as it is described in the IEEE 802.11b specification [8]. Our implementation incorporates the key features of CSMA/CA, namely, (i) carrier sensing via energy detection, (ii) DCF interframe spacing (DIFS) duration, and (iii) exponential random back-off. An illustration of the overall steps of the operation is shown in Fig. 4 and Fig. 5.

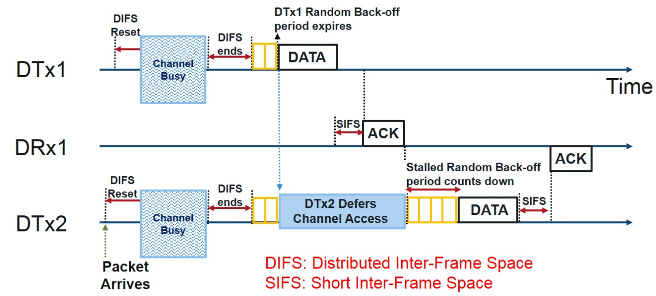

Fig. 4. CSMA/CA/ACK Timeline Chart - Energy Detection

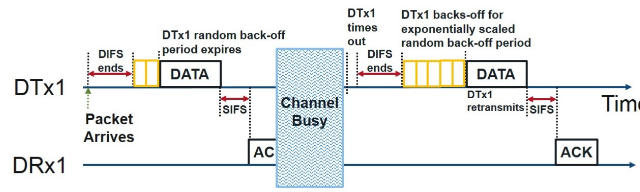

Fig. 5. CSMA/CA/ACK Timeline Chart - Exponential Random Back-off and Retransmission

1) Energy Detection: Channel occupancy can be identified by detecting RF energy in the channel. Energy in the channel is computed using equation (1).

$$Energy = \sum_{n=1}^{n=N} |x(n)|^2 \tag{1}$$

In our implementation, $x(n)$ represents the samples in the USRP frame retrieved from the receive buffer of the USRP.

2) DIFS Period: The standard specifies that when a packet is prepared by the DTx and ready to be sent to the intended DRx, the DTx must actively listen to the channel for a fixed specified amount of time known as the DIFS period. If during this period, the DTx senses RF signal energy from other transmitting devices (i.e. when the channel is found busy), it defers the transmission and enters a *Channel Occupied* state. In this state, the DTx stays idle as long as the ambient RF energy is above a specified threshold. When the energy drops below the threshold (i.e. the medium is sensed to be free), the DTx resets the DIFS duration and starts counting down again.

3) Binary Exponential Random Back-off: This method of random back-off is used to schedule retransmissions after collisions. Essentially, when the DIFS duration runs out, the DTx transitions to the Exponential Random Back-off state wherein the retransmissions are delayed by an amount of time determined by a minimum contention window, c_{min}, and the number of attempts to retransmit the DATA. With this increased number of retransmit attempts, the delay can increase exponentially. As an example, after k collisions, a random number of slot-times is chosen at random from $[0, 2^k\text{-}1]$ as described in equation (2).

$$Random\ Backoff\ Delay = randi[0, 2^k\text{-}1] \times Slot\text{-}time \tag{2}$$

The MATLAB $randi(\cdot)$ function picks uniformly at random an integer from the specified interval. In our implementation,

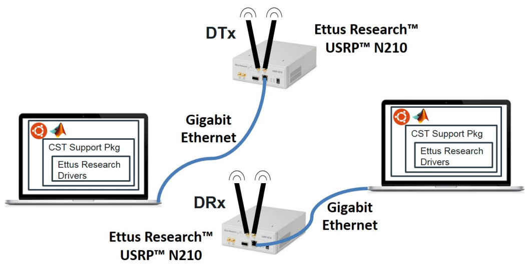

Fig. 6. Transceiver Hardware Setup

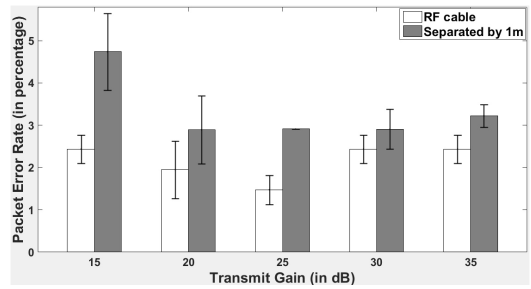

Fig. 7. Two Node Performance: Packet Error Rate

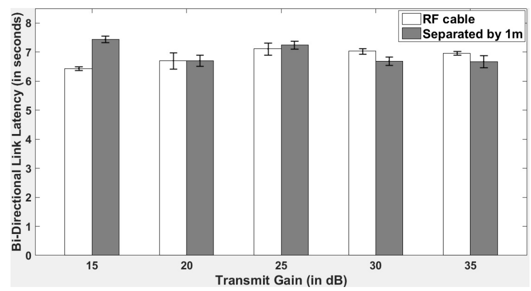

Fig. 8. Two Node Performance: Bi-directional Link Latency

the frame time represents the basic slot-time for the system, within which, we can detect another DTx transmitting.

VI. Experimental Setup

We use the Ettus USRP N210 platform [3], a radio front end commonly used in wireless research, connect it to a PC host using a gigabit Ethernet cable, and to program it using MATLAB. The communication between the USRP and host computer is established in MATLAB using the Communications System Toolbox USRP Radio support package [5]. We use the Ubuntu OS for its speed and availability of native tools towards developmental efforts. The overall setup is shown in Fig. 6. The most common undesirable behaviors that can occur during the course of the experiments are *underflow* and *overflow*. *Underflow* occurs when the radio requests for a frame of data from the receive buffer, but the host is not yet ready to provide it. *Overflow* occurs when the receive buffer becomes full and buffered data must be overwritten. We take adequate diagnosing steps to avoid the above mentioned behaviors.

VII. Experimental Results

A. Two Node Performance (1 DTx and 1 DRx)

Link layer contention resolution and other MAC layer functions depends on the ability to reliably generate alternating DATA-ACKs between the sender and receiver. In this regard, determining the performance of this basic link is important. Packet error rate (PER) and Bi-directional link latency are key performance indicators of the two node system. Of particular interest is the performance of the system when the transmit power level of the DTx is decreased below standard levels. The DTx was set up to send IEEE 802.11b compliant packets each with a large payload of random binary bits (2012 octets). The DRx receives the DATA and acknowledges the receipt of the same by transmitting an ACK. The experiment was designed to be statistically significant, and hence, 100 packets were transmitted for each of the 5 different transmit gain settings. The results were averaged over 5 runs. The DTx and DRx were kept about a meter apart.

1) Packet Error Rate: A packet is in error if the ACK for the same is not received in time by the DTx. An ideal system must recover quickly from such errors and, best trade-off PER and bi-directional link latency. PER is measured on average in percentage reflecting how many packets might be received in error for every 100 packets sent.

2) Bi-directional Link Latency: Bi-directional link latency is the average time taken for the DTx in sending a DATA and receiving the corresponding ACK. Note that since the MAC layer code runs during the course of the experiment, the Bi-directional link latency includes the DIFS duration and the random back-off period both set at 20ms. The MAC layer functionality however is largely dormant in the two node system due to the lack of contention. Bi-directional link latency is averaged for a packet in seconds. From Fig. 7 and Fig. 8, we can infer that the system guarantees a consistent $\leq 5\%$ packet error rate and approximately 7 secs of bi-directional link latency (DATA-ACK packet exchange inclusive of the MAC functions) over a wide range of transmit gains (15-30dB). Importantly, varying the distance between the two nodes does not significantly affect performance. Even moving the two nodes farther apart while still in LOS (e.g. by 15 meters), the PER and bi-directional link latency stayed consistent. However, the presence of many metallic surfaces, such as in our lab setting, give rise to multi-path reflections that can be strong and result in packet errors. The fact that the performance was significantly better when the nodes were connected by RF cables confirms the case.

B. Profile of Time Elapsed in DTx States

At the DTx, we measured the time elapsed in each state for a DATA-ACK exchange. The stacked plots shown in Fig. 9 shows the breakdown of the time spent in each substate. The plot at the top shows the small contributors to the overall processing time, and the one below shows the large contributors. Note that (1) the time spent in the MAC portion of the code includes the time elapsed to detect energy in the channel continually together with the DIFS and random back-off duration, and (2) the time taken to send the IEEE 802.11b DATA includes the time to prepare the packet.

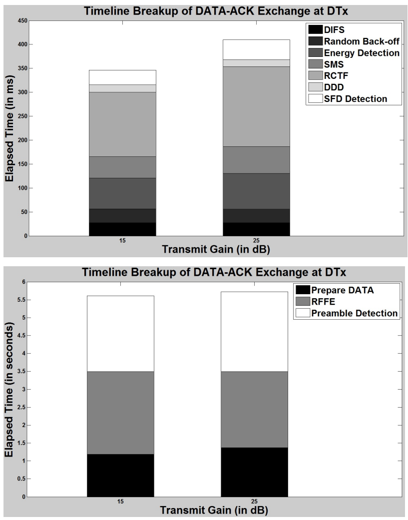

Fig. 9. Timeline Breakup of DATA-ACK Exchange at DTx

Owing to hardware constraints, packet exchanges in this system are in the order of seconds. However, we argue that this is acceptable because our system adds the feature of software definition, which requires additional time for execution.

C. Three Node Experimental Setup (2 DTxs and 1 DRx)

Given that without the MAC layer, the DATA/ACK collisions and the link latencies will be unacceptably high, we performed experiments to assess the MAC performance with a set of 3 USRPs (three nodes: 2 DTxs and 1 DRx). We expect to see increased bi-directional link latency and PER as the DTxs contend to gain access to the channel leading to packets collisions and subsequent retransmits. In our two node experiments, we confirmed that for a wide range of transmit gains, the performance remains consistent. We now have two independent links incident on one shared DRx, and hence, we do not expect to see much difference in the performance of the two links when varying the transmit gains here in the three node system. Instead, we measured bi-directional Link Latency and Packet Error Rate for DATA-ACK exchange in the two links as shown in Fig. 10 by varying the payload size in the DATA.

Essentially, the experiments let us compare the individual performances of the two links and further establish the MAC layer's role in enforcing fairness among the DTxs in accessing the channel.

1) Implemented MAC functions: The MAC header format for DATA and ACK is shown in Fig. 11 and Fig. 12 respectively will aid in discussion of the MAC layer functions. The DRx determines the DTx address from the MAC header of the received DATA and sends out an ACK addressed to that DTx. Furthermore, the DRx can reject DATA packets not addressed to it. Note that steps right from preamble detection,

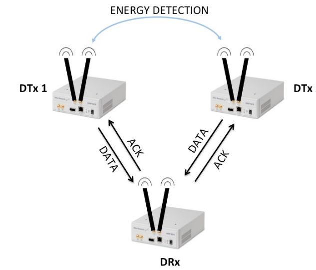

Fig. 10. Three Node System with 2 DTxs and 1 DRx

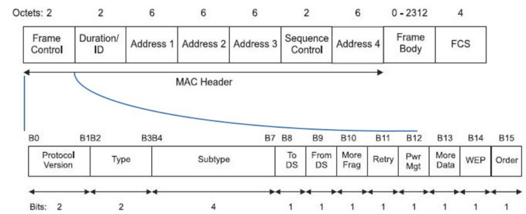

Fig. 11. MAC Header - DATA packet [8]

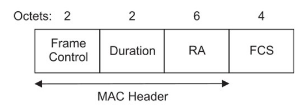

Fig. 12. MAC Header - ACK packet [8]

SFD detection, all the way up to reading into the IP address of the DTx from the MAC header, are carried out at the DRx, preceding the rejection of that DATA. On the other hand, the DTxs can identify the DRx from the MAC header of the received ACK and can go on to either accept or reject the ACK based on the IP Address. With the two node system, we had the DTxs re-transmitting DATA only towards lost ACKs.

2) MAC parameters: We learned from our initial set of experiments that the DATA/ACK processing takes significantly more time compared to time taken in transmitting a DATA packet. The experiments helped us fine-tune the DIFS duration, random back-off duration, and timeout-for-ACK duration towards fewer packet collisions. As a result, we performed our experiments with DIFS duration, minimum contention window, and ACK timeout duration set at 0.75, 0.5, and 0.5 seconds, respectively.

D. Three Node Performance: Experimental Results

Packet error rate and bi-directional link latency for DATA-ACK exchanges in the two links varying the payload size in the DATA are shown in Fig. 13 and Fig. 14, respectively. Four different payload sizes were used for the experiment, 500, 1000, 1500, and 2000 octets, to measure three node performance. The link latency and the packet error rate in the latter is bound to increase as larger packets incur higher processing delay at the DRx and more collisions necessitating increased packet retransmits.

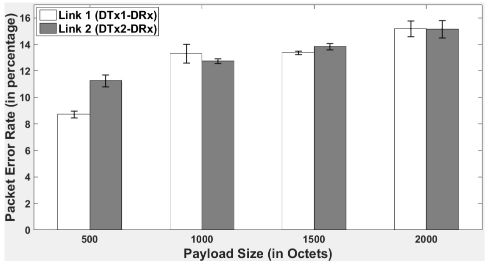

Fig. 13. Three Node Performance - Packet Error Rate of the Links

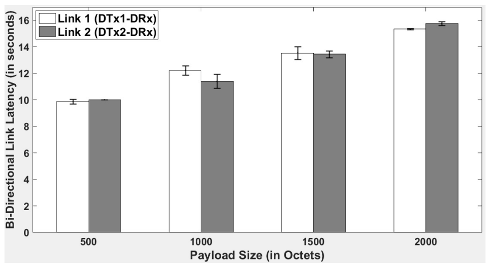

Fig. 14. Three Node Performance - Bi-directional Link Latencies

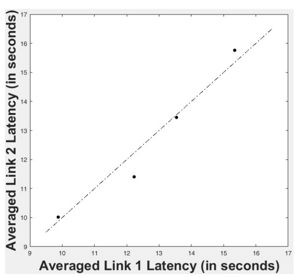

Fig. 15. MAC Layer Fairness - Averaged Link Latencies

1) Fairness: The dashed line shown in Fig. 15 represents an ideal system, in which the two DTx's access the channel equally often, such that their bi-directional link latencies are identical. Fairness is an important feature for the system to have, and is enabled by the MAC layer. Notice that the latencies of the two links deviate by only a small amount from the ideal line for varying payload sizes. This result establishes the role and efficacy of the MAC layer in enabling and enforcing fairness among the two DTxs when accessing the common channel.

VIII. Conclusion

Building our system around the concept of state-action based design helped realize an IEEE 802.11b standard compliant link layer. The state machine based design allows for modularity of our code making it relatively easy to manage and should allow for extensibility by the community. Through our experiments we have established the role and efficacy of the implemented MAC layer towards mitigating packet collisions and enforcing fairness among DTxs in accessing a common channel. We had to overcome a number of implementation challenges. Foremost, we had trouble realizing slot-synchronized operations, one of the most crucial issues in real-time testbeds. Second, it was difficult to pick the right energy threshold to deal with a variable noise floor due to environmental noise effects. Finally, our system required a thorough calibration step prior to running experiments. The minimum receive gain settings at the devices are always different. These experimental results have provided us with performance benchmarks that will focus future work on further optimization and sophistication of the MATLAB-based MAC layer for full real-time operation. As part of our future work, we will use this framework to perform evaluation studies on the co-existence of LTE and 802.11 Wi-Fi.

Acknowledgment

This work was supported in part by MathWorks under the Development-Collaboration Research Grant A#: 1-945815398. We would like to thank Mike McLernon and Ethem Sozer for their continued support on this project.

References

[1] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, 2006.

[2] K. R. Chowdhury and T. Melodia, "Platforms and testbeds for experimental evaluation of cognitive ad hoc networks," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 96–104, 2010. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2010.5560593

[3] Ettus Research, Inc. (2015) Usrp n200/n210 networked series. [Online]. Available: http://www.ettus.com

[4] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band*, IEEE Std. 802.11b-1999, 1999.

[5] MathWorks, Inc. (2016) Usrp support from communications system toolbox. [Online]. Available: http://www.mathworks.com/hardware-support/usrp.html

[6] R. Subramanian. (2016) Ieee 802.11b standard compliant link layer code for matlab-based sdr. [Online]. Available: https://github.com/80211bSDR/LinkLayer-MATLAB

[7] ——. (2016) Ieee 802.11b link layer for matlab-based sdr. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/55784-ieee-802-11b-link-layer-for-matlab-based-sdr

[8] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. 802.11, 1999.

[9] B. Drozdenko, R. Subramanian, K. Chowdhury, and M. Leeser, "Implementing a matlab-based self-configurable software defined radio transceiver," in *Cognitive Radio Oriented Wireless Networks - 10th International Conference, CROWNCOM 2015, Doha, Qatar, April 21-23, 2015, Revised Selected Papers*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, M. Weichold, M. Hamdi, M. Z. Shakir, M. M. Abdallah, G. K. Karagiannidis, and M. Ismail, Eds., vol. 156. Springer, 2015, pp. 164–175. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-24540-9

[10] GNU Radio Project. (2015) Gnuradio: The free and open source radio ecosystem. [Online]. Available: http://www.gnuradio.org

[11] T. Collins, "A green approach to a multi-protocol wireless communications network," Ph.D. dissertation, Citeseer, 1911.

[12] R. U. O. WARP Project. (2015) Wireless open-access research platform. [Online]. Available: http://warp.rice.edu/index.php

[13] M. Duarte, A. Sabharwal, V. Aggarwal, R. Jana, K. Ramakrishnan, C. W. Rice, and N. Shankaranarayanan, "Design and characterization of a full-duplex multiantenna system for wifi networks," *Vehicular Technology, IEEE Transactions on*, vol. 63, no. 3, pp. 1160–1177, 2014.

[14] C. Hunter, L. Zhong, and A. Sabharwal, "Leveraging physical-layer cooperation for energy conservation," *IEEE T. Vehicular Technology*, vol. 63, no. 1, pp. 131–145, 2014. [Online]. Available: http://dx.doi.org/10.1109/TVT.2013.2271121