A Modular Reference Application for IEEE 802.11n Wireless LAN MACs

Hans-Peter Loeb
Infineon Technologies
Munich, Germany
Email: HansPeter.Loeb@infineon.com

Christian Sauer
Cadence Design Systems
Munich, Germany
Email: sauerc@cadence.com

***Abstract*—Designing efficient yet flexible medium access controllers (MAC) for wireless protocols is a challenge. Not only are these protocols still evolving, they are also increasingly demanding in terms of throughput and real-time requirements. In order to support a careful application-driven architecture development, reference applications are required that expose the full system and enable the quantitative evaluation of performance-flexibility tradeoffs.**

For this purpose, we have captured the 802.11n MAC protocol in an executable reference application which comprises the system function and its environment including traffic scenarios. We model the reference in Click, a packet processing framework. The functionally-correct model captures performance-relevant aspects such as the wireless protocol timing exactly. Leveraging extensions to Click we can use the model for the development and deployment of embedded architectures. Our 802.11n MAC model comprises between 118 and 1238 functional elements and can be simulated in real time depending on the scenario. Due to its modularity, additional scenarios can be added productively.

I. Introduction

Communication standards evolve quickly and are growing in complexity. This is especially true for the recent 802.11n wireless LAN (WLAN) standard which adds high-throughput (HT) and Quality-of-Service (QoS) extensions to the Media Access (MAC) layer. Traditionally, most of the MAC functions have been implemented in hardware to meet real-time and performance requirements. Evolving and/or coexisting standards, on the other hand, call for flexible SW-based MAC solutions similar to the software-defined radio initiatives at the physical (PHY) layer [1].

In order to find a feasible flexibility/costs tradeoff a careful and quantitative evaluation of different function splits is needed. Due to the lack of established tools and methodologies, we have developed a framework [2] in prior work. Tailored to packet processing the SystemClick framework enables a) the development of programmable platforms and b) supports their later deployment by generating efficient code for embedded processors starting from an architecture-independent reference application.

This application must capture the protocol behavior exactly and at the right granularity. Frame formats, control, and management tasks must interact correctly to represent a realistic system. Precise timing requirements, e.g., for interframe gaps, require a notion of time for system evaluation. Further usage in the development process requires an implementation that is indicative of the final performance.

Unfortunately, such a reference is not available for the 802.11n MAC layer. Only test specifications such as 802.11T exist which describe a host of traffic patterns and measurement metrics. Executable MAC models within network simulation frameworks such NS-2 are typically too abstracted to be performance indicative [3]. This in turn can lead to inconsistencies when compared with real-world systems [4]. Few models cover aspects of 802.11n such as frame aggregation [5], but no complete model is publicly available or documented. Earlier attempts of a reference application for 802.11a in SDL have been well received by the community but did not result in efficient solutions [6].

Therefore, we have developed an executable reference application of the 802.11n MAC layer which fulfills above requirements. It can serve the following purposes:

- Capture and document protocol knowledge in an executable specification.
- Serve as a golden model for test and verification or provide black-box function for network simulation.
- Allow performance-indicative benchmarking of different MAC architectures and design space exploration.
- Allow experimentation with protocol extensions.
- Serve as a starting point for efficient software-based implementations.

For a comprehensive specification of the system-level reference, three essential facets must be addressed [7]:

- Environment - Describes the surroundings of the system, its configuration, and parameters. This includes use cases and traffic setup.
- System function - Unambiguously specifies the function of the system in form of a functionally correct and executable model.
- Measurement - How are performance and the quality of the results evaluated?

Due to the limited space we focus on environment (Sec. II) and system function (Sec. III). The results section (IV) covers the measurement facet briefly and characterizes the model.

II. Environment and usage scenarios

The IEEE 802.11T specification describes 37 use cases, 19 usage models, and 24 different applications. However, such

TABLE I
MAC PERFORMANCE SCENARIOS

Name	*Description*
HT	Unidirectional high throughput (HT) setup between an access point (AP) and a single station (STA) comprising packets of different size distributions (min, max, typ, IMIX).
QoS	AP and four STAs comprising all applications of Table II (/w file) to expose queuing and channel access timing. Additional VoIP and HDTV STAs may be added.
Cmplx	AP and 32 STAs with IMIX traffic (1Mbps) for high model complexity exposing memory and configuration bounds.

TABLE II
BENCHMARK APPLICATIONS AND THEIR CHARACTERISTICS.

Application	*Throughput*	*Pkt. Size*	*Delay*	*TID*
VoIP*	0.096 Mbps	120 B	30 ms	3
Video Conf*	2 Mbps	512 B	100 ms	2
HDTV	24 Mbps	1500 B	200 ms	1
Internet/File$^{(*)}$	1/150 Mbps	IMIX[1]	BE[2]	0

* Two-way communication [1]) Internet IMIX distribution [2]) Best Effort

TABLE III
SETTINGS FOR QOS AND AGGREGATION PARAMETERS

AC/TID	*TXOP*	*A-MSDU* max. size	*A-MSDU* timeout	*A-MPDU* max. size
0	1.5 ms	4096 B	50 ms	64 Pkts.
1	3 ms	4096 B	20 ms	32 Pkts.
2	0	2300 B	10 ms	-
3	0	-	-	-

multitude is not necessary for the evaluation of different MAC architectures. We propose three basic scenarios to indicatively expose a MAC's performance, see Table I.

The scenarios are implemented in simulation setups instantiating MAC system functions (Section III) as well as host environments (traffic generators, packet sinks, triggers for management operations) and a combined Air/PHY channel module. Connecting the MACs, the latter generates both timing behavior of frame transmissions and half-duplex busy signaling. Frames aggregated at the PHY layer are transmitted back to back. Bit errors can be introduced at random or in case collisions occur.

We configure all stations to be in 802.11n configuration (greenfield mode) and assume a channel of 300Mbps with standard timing: slot time $9\mu s$, SIFS $16\mu s$, PHY preamble and header $(24 + 4 * \mathrm{nr\_mimo\_streams})\ \mu s$, time to assess the channel (CCA) $4\mu s$.

The QoS applications in Table II are associated with traffic identifiers (TIDs) and mapped directly onto access categories (AC). Parameters for transmission opportunities (TXOP) and aggregation are listed in Table III. We depart from standard settings to expose all performance-relevant functions within a static setup, e.g. in terms of aggregation. Transmission opportunities are not necessary for low-bandwidth applications such as VoIP [8].

III. WLAN SYSTEM FUNCTION

We model the WLAN system function in Click [9], a widely-used framework for describing network processing applications. The framework's application library already includes some wireless modules for functions above the MAC layer. In the past, Click has been found suitable for modeling packet processing at the MAC layer, see, e.g., [10].

We have significantly extended the library with support for the 11n protocol family and MAC level real-time frame processing capabilities. These WLAN elements together with standard elements for, e.g., classification, queueing and scheduling, allow a variety of configurations. To increase efficiency and ease model deployment, we port the application library to C within our framework [2].

A. IEEE 802.11 Protocol Function

The model captures the exact protocol timing and state of both a legacy 802.11abg and 802.11e/n MAC. We focus on performance-critical aspects of the protocol and thus implement the following features:

- In legacy mode, fragmentation and RTS/CTS protection can be used. Medium access is based on the contention window procedure, and frames are acknowledged and resend one by one. On collisions, a back-off mechanism increases the contention window.
- QoS extensions (11e) introduce four prioritized categories (AC) for medium access. Stations acquire TXOPs per AC e.g. by completing an RTS/CTS exchange and subsequently send packets. Acknowledgement of frames is done in blocks (Immediate BlockACK) with timely retransmission of failed packets.
- HT extensions (11n) introduce two mandatory aggregation schemes: The aggregation of service data units (A-MSDU) is used early in MAC processing, relieving per-frame processing. Aggregation at the PHY layer (A-MPDU) concatenates subframes protected with a CRC checksum. Total length must match the current TXOP and be pre-computed prior to transmission.

In addition, management functions such as beacons, station authentication and association as well as the exchange of configuration data are comprised in the model. Although operational and functionally correct, WEP security and rate selection modules are merely placeholders for more sophisticated schemes imposing similar loads on a system. Detailed PHY handling and higher-layer management are not performance relevant and have thus been abstracted. Similarly, advanced features such as Reverse Direction Grant (RDG) and PSMP affect network performance but can be neglected in terms of the MAC's performance requirements.

B. Model Implementation

Figure 1 shows the Click model. Processing paths for TX (upper part) and RX (lower part) cover time-critical protocol layer at the air side and pre-/postprocessing stages at the host side. The figure exemplary features three QoS paths in different configurations for AC 0 and the legacy (11abg) paths. Section III-C describes these paths in detail. All processing paths can be configured by a ReconfigurationManager which writes a global StationInfoBase every time the connection state changes.

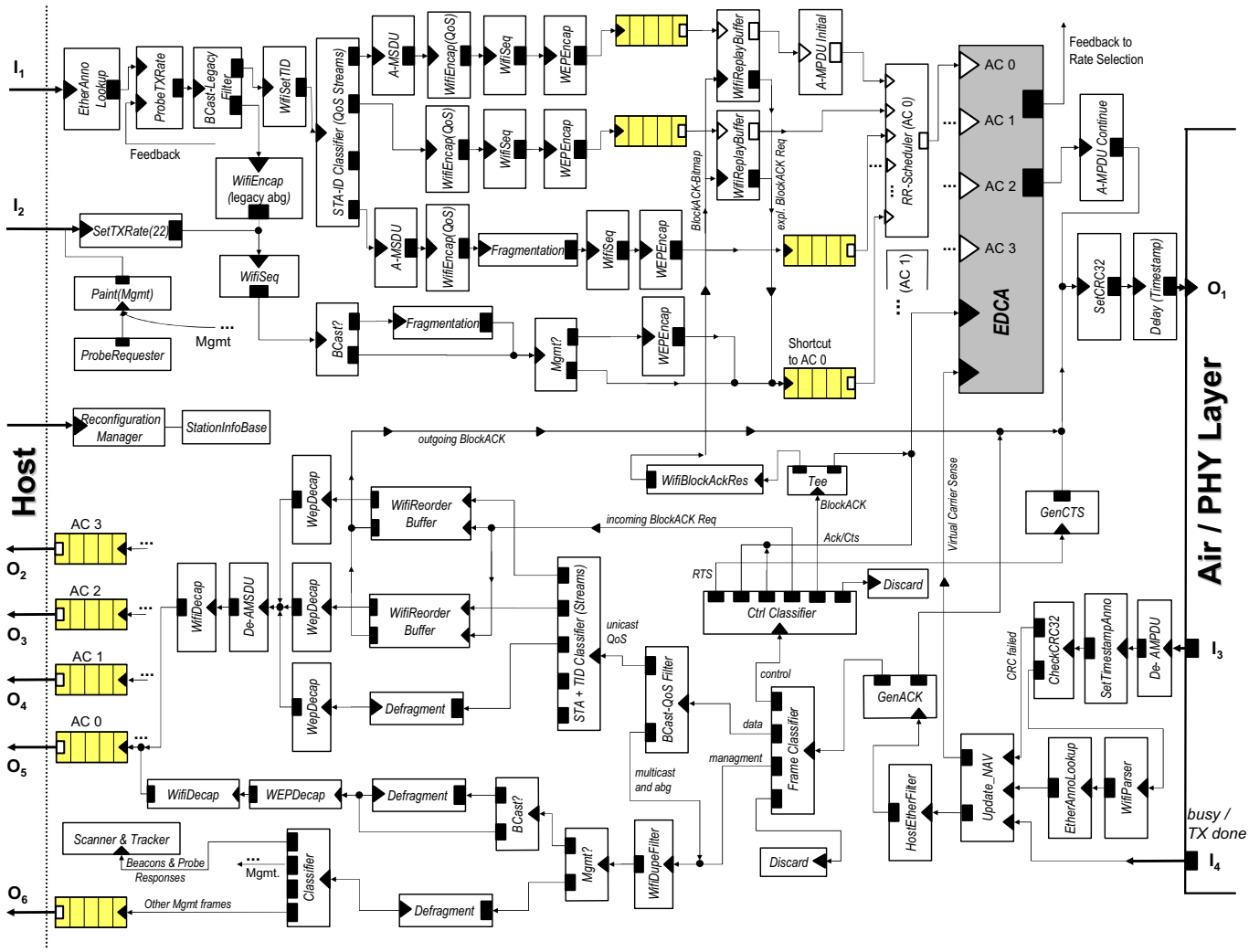

Fig. 1. Click model of a WLAN 802.11abg/en MAC with transmit paths in the upper half and receive paths below.

Exact timing of interframe gaps and timeouts relative to the air are critical for standard compliance. Click's run-to-completion model of computation and its scheduling schemes can not guarantee precise timing in the presence of processing delays. Thus, we introduce a scheduling scheme based on time stamps. Incoming packets and the CCA are stamped upon reception to have a consistent timing reference throughout their processing. Tasks are scheduled relative to this time stamp. For transmission, packets are annotated with the scheduled transmit time and delayed accordingly.

Central to the model's protocol timing is the Enhanced Distributed Channel Access (EDCA) element which schedules transmission sequences. It maintains backoff counters per AC and pulls packets from Click schedulers which select, e.g., among all streams of a given AC. It is implemented as a state machine with precisely timed transitions at timeouts and when the virtual carrier sense changes.

C. Processing Paths

This section summarizes the packet processing paths through the Click model shown in Figure 1.

A.1) Outbound 11abg data frames. All frames from the host (input I_1) pass rate selection and get an internal id (STA-ID) based on Ethernet address and annotated TID. Legacy (11abg) frames are sorted out, encapsulated, receive a sequence number, and are fragmented/encrypted as needed. Stored in the best effort traffic queue, they are pulled round-robbin by the EDCA. The EDCA sets the duration field, TX time, and schedules RTS/CTS protection if indicated. Eventually, every frame is CRC32 protected and sent as specified by an annotated time stamp to output O_2.

A.2-4) Outbound 11n data frames. 11n frames are classified into QoS paths and have additional header fields and TID set. If configured, aggregation into A-MSDUs and encryption are performed before being queued. The EDCA schedules and pulls frames for a given AC. In BlockACK mode, the WifiReplayBuffer tracks frames, replays them if not acknowledged,

and issues BlockACK requests. Depending on the TXOP length, max.-sized A-MPDUs are initially prepared by computing the aggregate length. The EDCA performs RTS/CTS protection to start TXOPs and sends CF-END frames when no more packets are available. Finally, the actual aggregation is done and CRC-protected frames are sent as scheduled.

B.1) Inbound 11abg data frames. WLAN packets from the Air/PHY (I_3) are timestamped and CRC checked. Packets are annotated with STA-ID, TID and type. The NAV timer is updated with the frames' duration. The model's state is also updated if CRC checks fail or if medium state changes (I_4). Next, frames for other destinations are discarded and unicast frames are acknowledged. After classification and duplicate removal, frames are decrypted and reassembled. After header translation to Ethernet, frames are forwarded to the host (O_6).

B.2-4) Inbound 11n data frames. A-MPDUs are deaggregated, member frames are marked and forwarded subsequently the same as 11abg frames. STA-ID classification then forwards the frames into the proper QoS path. The WifiReorderBuffer keeps track of received frames and releases them in correct sequence. Frames are then decrypted and A-MSDUs de-aggregated into single frames before forwarded to the host (O_{2-5}).

C.1) Outbound 11abg acknowledgment. ACK frames are generated upon reception of a valid abg unicast data frame or if indicated by the ACK policy. The frame is scheduled after SIFS, forwarded to CRC protection, and sent out.

C.2) Outbound 11n blockack. BlockACKs are triggered by explicit or implicit requests and generated by the WifiReorderBuffer. Explicit requests are treated like generic inbound control frames. Responses are scheduled after SIFS. The implicit mechanism extracts the trigger information from A-MPDU members and schedules the response, either after the last member is processed or if time to send runs out.

D.1) Inbound 11abg acknowledgment. Inbound ACK frames are processed as in (B.1), but classified as control frames and forwarded to EDCA. There, the stored frame is killed and the next transmission can proceed. A timeout in the EDCA detects failed transmissions and causes retransmissions according to retry counters. Feedback is given to rate selection.

D.2) Inbound 11n blockack. Inbound BlockACKs are processed as in (B.1), but are classified as 11n control frames and forwarded to the WifiBlockAckResponder. The extracted bitmap is then forwarded to the proper WifiReplayBuffer(s). There, acknowledged frames are cleared and lost frames prepared for retransmission. A timeout detects missing BlockACKs and repeats requests.

E) RTS/CTS. If an outbound frame exceeds the RTS threshold or a TXOP is to be used, EDCA issues an RTS prior to actual transmission. At the receiver of an RTS, a CTS frames is generated similar to ACK frame processing (C.1/D.1). Completing the exchange, inbound CTS are classified and trigger the transmission of the outbound frame after SIFS in the EDCA.

F) Management Frames. Management frames are generated by the model or can be provided by the host through dedicated ports (I_2). Outbound management frames are forwarded to the legacy path (B.1) without additional Wifi encapsulation. Incoming management frames are received like data frames, but assorted by classification and either directly processed or forwarded to the host.

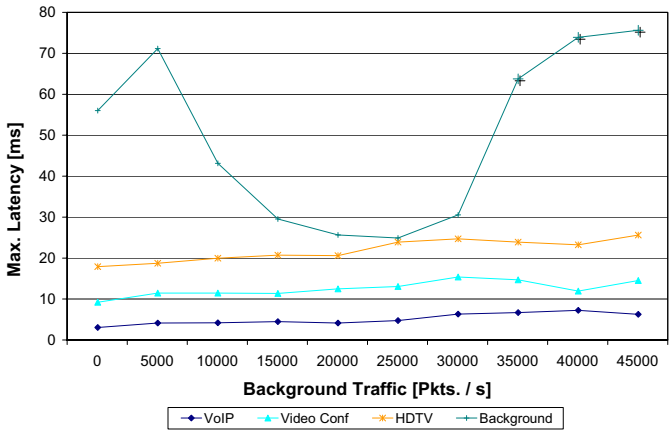

Fig. 2. Maximum end-to-end latencies in the QoS scenario as background traffic load is increased. The channel model's drop rate is 5%.

IV. Results

Measurements most importantly include protocol conformance, i.e. real-time behavior, packet sequences and frame formats must be verified. Furthermore, forwarding delays for QoS applications and total throughput are relevant. We cover these aspects briefly and also present simulation times and discuss model complexity.

A. Measurements and Verification

Using the defined scenarios we have simulated and validated the model. Runtime parameters enable fast evaluation of configurations. The framework allows detailed analysis and tracing of traffic and element execution. Applying assertion based techniques, a protocol checker monitors, e.g., response times for various protocol sequences. Data streams are checked for end-to-end packet loss, throughput and latency. In addition, packet traces have been checked by tools such as Wireshark.

The maximum delays for the QoS applications from Table II are met for all scenarios. Figure 2 shows the maximum delay incurred in the QoS scenario with two HDTV / VoIP streams and a packet drop rate of 5% as the background traffic load (AC 0) is increased. While all higher QoS classes remain unaffected, the background latency is first large due to the A-MSDU aggregation timeout, then comparable to AC 1 and lastly starts to grow as the channel capacity is reached. Packet loss only occurs for AC 0 as indicated with shaded markers.

Table VI shows for an ideal channel that the number of collisions increases with the number of clients connected. The load on the channel approaches theoretical bounds for the HT scenario.

TABLE IV
SIZE OF THE APPLICATION LIBRARY.

Library Type	*Elements*	*Lines of C-Code*[1]
WLAN-specific	41	7490
Standard	39	3232
Sum	80	10722

[1]) Physical LoC without comments and empty lines

TABLE V
FUNCTIONAL ELEMENT INSTANCES

Scenario	*AP*	*STAs*	*Env.*	*Total*
1 HT	118	123	44	285
2 QoS	230	123	97	819
3 Complex	1238	123	599	5741

B. Model Complexity

All scenarios for the reference application are generated from 18 Click source files with a total of 1188 lines of code (LoC). This includes verbose output and tracing. The source instantiates and configures elements from an application library. The library comprises 80 different elements with a total of 10722 LoCs, see Table IV.

The number of instantiated elements in the AP varies with the number of STAs as shown in Table V. Most elements are inferred by hierarchy and macros. Despite the size of the models, the effort for customizing stations and modeling new scenarios is well within a few hours.

During implementation, we found PHY-layer aggregation to be a main source of complexity besides real-time behavior and project it to become a performance bottleneck. Also, concerning the model's size, the full configuration of every stream leads to overly large number of elements. For example, support for aggregation is typically limited to few streams. This could be addressed by implementing multi-stream elements or by changing the graph structure at runtime.

C. Simulation Speed

Depending on model complexity, one minute traffic can be simulated in between 23 and 432 seconds on a 1.7Ghz Pentium M system running Cygwin. As detailed in Table VI, this is sufficient to verify most scenarios in real time and even allows testing of large network configurations.

TABLE VI
SIMULATION PERFORMANCE

Scenario	Simulation[1](1min)	Load on air[2]	Collisions
HT (IMIX/AC 0)	47s	253Mbps	-
HT (IMIX/AC 3)	23s	17Mbps	-
QoS	58s	185Mbps	145/s
Cmplx	432s	38Mbps	404/s

[1]) CRC and WEP disabled [2]) 300Mbps channel

V. CONCLUSION

We have presented an executable reference application for the 802.11n MAC layer. The application is modular and captures performance-relevant aspects and timing behavior of the IEEE standard in a concise way. We have identified three scenarios relevant for architecture exploration. Further setups can be created and customized productively leveraging the application library and the component-based approach. The implementation is efficient such that simulation of most setups in real-time is possible. Using tracing techniques, this allows fast verification and quantitative evaluation of parameter settings, e.g., in terms of throughput or number of collisions.

Future work will aim at a flexible MAC architecture. For this purpose, the application library has to be characterized on a range of platform building blocks so that the reference application can drive the design space exploration.

ACKNOWLEDGMENT

This work was partly supported by the European Community's Seventh Framework Programme under grant agreement No. 213311, project Omega - Home Gigabit Networks. We thank F. Beckmann of Infineon for his contributions to the model.

REFERENCES

[1] U. Ramacher, "Software-defined radio prospects for multistandard mobile phones," *Computer*, vol. 40, no. 10, pp. 62–69, 2007.

[2] C. Sauer, M. Gries, and H.-P. Loeb, "SystemClick - a domain-specific framework for early exploration using functional performance models," *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, pp. 480–485, June 2008.

[3] Qi Chen et al., "Overhaul of IEEE 802.11 modeling and simulation architecture in NS-2," in *10th Int. Symp. on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWIM)*, Oct. 2007.

[4] D. Malone, I. Dangerfield, and D. Leith, "Verification of common 802.11 MAC model assumptions," in *Passive and Active Network Measurement Conference (PAM 2007, LNCS 4427)*, 2007.

[5] D. Skordoulis, Q. Ni, H.-H. Chen, A. Stephens, C. Liu, and A. Jamalipour, "IEEE 802.11n MAC frame aggregation mechanisms for next-generation high-throughput WLANs," *Wireless Communications, IEEE*, vol. 15, no. 1, pp. 40–47, February 2008.

[6] G. Panic, D. Dietterle, *et al.*, "A system-on-chip implementation of the IEEE 802.11a MAC layer," in *Euromicro DSD*, 2003.

[7] M. Tsai, C. Kulkarni, *et al.*, "A benchmarking methodology for network processors," in *Network Processor Design: Issues and Practices*. Morgan Kaufmann Publishers, 2002, pp. 141–165.

[8] A. Salhotra, R. Narasimhan, and R. Kopikare, "Evaluation of contention free bursting in IEEE 802.11e wireless LANs," *Wireless Communications and Networking Conference, 2005 IEEE*, vol. 1, pp. 107–112 Vol. 1, March 2005.

[9] E. Kohler, R. Morris, B. Chen, *et al.*, "The Click modular router," *ACM Trans. on Computer Systems*, vol. 18, no. 3, Aug. 2000.

[10] R. Dhar, G. George, A. Malani, and P. Steenkiste, "Supporting integrated MAC and PHY software development for the USRP SDR," *Networking Technologies for Software Defined Radio Networks, 2006. SDR '06.1st IEEE Workshop on*, pp. 68–77, Sept. 2006.