Design and Implementation of an RF Front End for Physical Layer Wireless Network Emulation

Glenn Judd and Peter Steenkiste
Carnegie Mellon University

***Abstract*—Networking researchers have long faced a fundamental tension between the experimental realism of wireless testbeds on one hand, and the control and repeatability of simulation on the other hand. To overcome the stark tradeoff of these traditional alternatives, we have developed a wireless network emulator that enables both realistic and repeatable experimentation at network scale. A critical component in this emulator is the RF Front End that converts RF signals to lower frequencies - where they can be digitized - and vice versa.**

We discuss the unique requirements that physical layer network emulation demands of an RF Front End. We present a design that meets these demands and demonstrate its performance via measurements.

***Index Terms*—Mobile Networks and Systems, Emulation**

I. Introduction

As wireless network deployment and use become ubiquitous, it is increasingly important to make efficient use of the finite bandwidth provided. Unfortunately, research aimed at evaluating and improving wireless network protocols and applications is hindered by the inability to perform repeatable and realistic experiments.

The most direct method of attaining experimental realism is to conduct experiments using real hardware and software in various real world environments. Unfortunately, this approach faces serious repeatability and control issues since the behavior of the physical layer is tightly coupled to the physical environment and precise conditions under which an experiment is conducted. Even repeating the same experiment twice can be a daunting task when anything in the surrounding environment is in motion; remote researchers face an even bleaker situation trying to reproduce an experiment. It is also difficult to avoid affecting colocated production networks. Moreover, configurability and management of even a small number of mobile nodes distributed in three dimensions is cumbersome.

For these reasons, many researchers have understandably embraced simulation. This approach solves the problems of repeatability, configurability, manageability, modifiability, and (potentially) integration with external networks, but faces formidable obstacles in terms of realism. Even fundamental functions such as deciding what a received frame looks like [1] diverge greatly from the operation of real hardware. Evaluating real applications running over wireless networks is typically very difficult using a simulator.

We have developed [2] a wireless emulator that enables both realistic and repeatable wireless experimentation by accurately emulating wireless signal propagation in a physical space. Unlike previous approaches, this emulator supports, real devices, applications, MAC, and PHY layers on a network-wide scale while maintaining experimental control and repeatability. The key technique we use to accomplish this is digital emulation of signal propagation using an FPGA.

This emulator provides an attractive middle ground between pure simulation and wireless testbeds. To a large degree, this emulator should be able to maintain the repeatability, configurability, isolation from production networks, and manageability of simulation while retaining the support for real applications and much of the realism of hardware testbeds. As a result, this emulator should provide a superior platform for wireless experimentation.

This emulator is not, however, a complete replacement for simulation and real world evaluation. Simulation is still useful in cases where a very large-scale experiment is needed or in certain cases where functionality not available in hardware is required (e.g. changing the MAC firmware). Real world evaluation is still useful when radio channel fidelity beyond the capabilities of the emulator is required, or for verifying the operation of the emulator in real-world settings.

Designing and implementing an emulator capable of emulating a complete wireless network is a challenging task. In this paper we focus on one aspect of that task: the design of an RF Front End that allows wireless devices to interface with the emulator's data conversion module. At a high level, this RF Front End bears many similarities to RF front ends used in digital software radios [3]. There are, however, notable differences such as: strong input signals which require attenuation and not amplification, a reduced need for filtering due to the closed nature of the system, and the need to pay strict attention to signal isolation within and between front ends.

This discussion proceeds as follows: Section II gives a brief introduction to the architecture of the wireless emulator. Section III and Section IV then present the design and implementation of the emulator's RF Front End. Section V presents selected results from validation tests conducted to ensure that the RF Front End meets its basic requirements. SectionVI then compares the approach we are taking with existing approaches, and Section VII concludes this work.

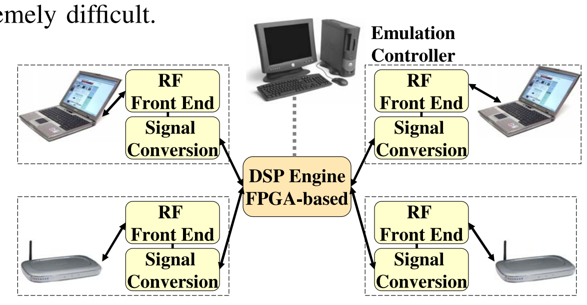
II. Emulator Architecture

Before we discuss the design of the RF Front End, we briefly discuss the architecture of the emulator.

Developing a physical layer wireless network emulator is a difficult task requiring careful design. The massive computation, low latency, high isolation, and support for diverse experimental setups require a powerful, flexible architecture.

Traditional channel emulators have utilized a monolithic ASIC-based design. In these designs analog signals from transmitters are routed via coaxial cable to a central emulator unit. Inside this unit the signals are digitized, channel emulation occurs, and then the resulting channel analog output signals are routed out via coaxial cables to receivers. Some advanced emulators provide digital inputs and outputs for use with digital baseband devices.

At an abstract level the architecture of the wireless network emulator is similar to the traditional monolithic approach. Wireless devices - "RF nodes" - are connected to the emulator; all communication between RF nodes occurs through the signal propagation environment modeled within the emulator. While sufficient for channel emulation, this traditional architecture is less than ideal for network-wide emulation. Routing a large number of analog signals to a small physical space makes the problem of isolating these signals from each other extremely difficult.



Network devices and signal conversion modules reside in shielded chassis.

Fig. 1. Detailed Emulator Architecture

Instead, we have developed the distributed emulator architecture shown in Figure 1. On transmit, the RF signal from a given RF node is passed into the RF Front End where it is shifted down to a lower frequency. This lower frequency signal is then digitized by the Signal Conversion Module, and forwarded in digital form into a central DSP Engine that is built around one or more FPGAs. By digitizing signals in a distributed fashion, there is no centralized isolation problem to solve in the DSP Engine. All signals exiting the Signal Conversion Modules for transmission to the DSP Engine (and vice versa) are digital.

The RF Front End is distinct from the Signal Conversion Module since RF components are frequency specific. Keeping the RF Front End separate allows for different RF bands to be supported by swapping RF Front End cards without the need to construct an expensive multi-band RF Front End (unless desired.)

The DSP Engine models the effects of signal propagation (e.g. large-scale attenuation and small-scale fading) on each signal path between each RF node. For each RF node, the DSP combines the processed input signals from all the other RF nodes. For each RF node, the resulting signal is then sent out to the signal conversion module which converts the digital signal back to a radio signal. It is then sent to the wireless line card through the antenna port. Using an FPGA provides both the signal processing power necessary as well as flexible allocation of computational resources.

The Emulation Controller controls RF node movement in the emulated physical environment as well as application behavior on the end hosts. Movement within the physical environment is emulated in real time; the controller coordinates this movement with the real-time modeling of the signal propagation environment.

III. RF Front End Design

The RF Front End converts transmissions from devices attached from the high frequency signals generated by the device to a low frequency that can be digitized by the Signal Conversion Module. When the device is not transmitting, signals from other devices in the emulated environment must be converted from the low frequency signals produced by the Signal Conversion Module to the high frequencies that the signals were originally transmitted at.

In addition, the RF Front End must meet the following requirements:

Signal Scale. The RF Front End must ensure that transmitted signals are presented to the Signal Conversion Module at an ideal signal strength: as close as possible to the maximum signal strength supported by the Signal Conversion Module, but not greater. Similarly, received signals must be attenuated in order to place the received signal at the desired range of signal strengths.

Flatness. At each frequency supported by the RF Front End, the scaling introduced by the RF Front End should be as close to equal as possible. That is, the transmit frequency response should be flat as should the receive frequency response.

Noise. The RF Front End should introduce as little noise as possible into the signal.

Isolation. It is critical that RF Front End not introduce signal paths between devices that affect the operation of the emulator. Thus, the RF Front End must maintain a high degree of isolation between the devices attached to it.

Spurious Signals. Finally, the RF Front End must produce as little spurious signal energy as possible.

IV. Implementation

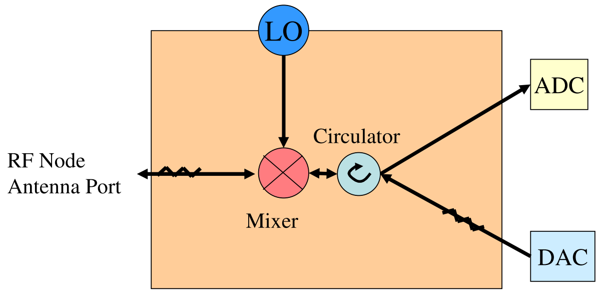


Fig. 2. Strawman RF Front End Implementation

Strawman Design. The main signal paths in the RF Front End are 1) from the RF port of the RF Node to the A/D port of the Signal Conversion Module, and 2) from the D/A port of the Signal Conversion Module to the RF port of the RF Node. Conceptually, the RF Front End could be implemented as shown in Figure 2. This figure shows the RF Front End in a shaded box; the RF Front End connects the RF Node's antenna port to the ADC and DAC which are located in the Signal Conversion Module. An off-board LO signal source is used. In this implementation a single "mixer" is used. On downconversion, the mixer shifts the high-frequency signal generated by the RF Node down to a lower frequency as required by the Signal Conversion Module. On upconversion, the mixer shifts the low-frequency signal generated by the Signal Conversion Module up to the high-frequency that it was originally transmitted at. As the RF Node may be half-duplex, these two paths must share the same RF Port. The architecture in Figure 2 solves this problem using a "circulator." This device sends signals transmitted by the RF Node to the downconversion path; signals received from the Signal Conversion Module are sent to the RF Node.

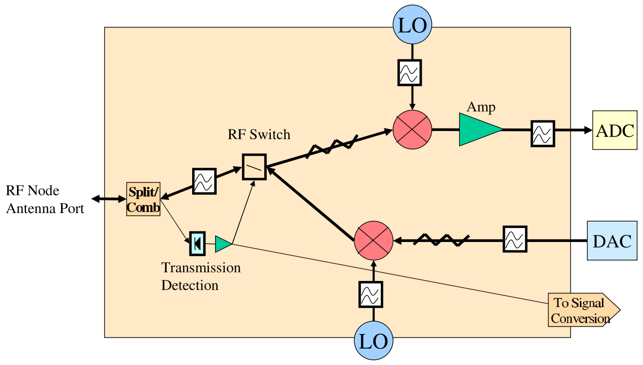


Fig. 3. Actual RF Front End Implementation

In principle, no signal would leak between the two signal paths. Unfortunately, this design has shortcomings.

- Isolation - the limited isolation of the circulator means that some signals will go in undesired directions. For example, some signal coming into the upconversion path from the Signal Conversion Module will leak through the circulator onto the downconversion path. This signal will then enter the Signal Conversion Module. The net effect during emulator operation would be to make signals being *received* by one RF Node appear as if there were being *transmitted* - weakly - from that RF Node.
  In addition, the LO distribution system can be an additional source of signal leakage. Some of the signal being transmitted will leak into the LO distribution system and arrive at another RF Front End where it will be sent for reception.
- Lack of undersampling support - the use of a single LO for both upconversion and downconversion in this design effectively precludes the use of undersampling. The use of undersampling is discussed further below.
- Signal Matching/LO Strength - The downconversion output signal strength should match the ADC input range. In practice, however, the input signal to the mixer has a limit which prevents a good match. Using a higher powered LO can achieve a better match, but such a high powered LO can be difficult to achieve.
- Spurs - All mixers produce spurious signal known as "mixer spurs." These become more severe as the mixer input signal strength approaches the mixer signal limit. In the strawman design, the input signal needs to be as close to the limit as possible to allow for a good ADC input range match. Thus, this design cannot simultaneously achieve good isolation from spurs and good ADC input range match.

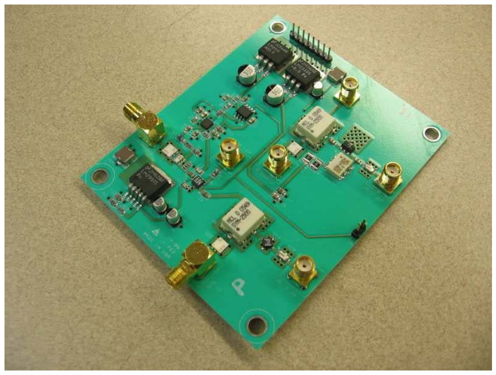


Fig. 4. RF Front End Picture

Figures 3 and 4 show the actual RF Front End implementation. The discussion below elaborates on how this implementation overcomes the shortcomings of the strawman design.

Improving Isolation. As the emulator targets half-duplex devices, isolation between upconversion and downconversion paths can be improved by using an RF switch instead of a circulator. Full-duplex devices can still be supported using two RF Front Ends. The RF switch must be very fast to detect transmissions so as to truncate as little of the transmitted signal as possible; it also must recognize the end of a transmission quickly, though this is not as critical as recognizing the start of a transmission. The use of an RF switch requires a transmit detection circuit that controls the switch. This detection circuit must also be fast for the same reasons that the switch must be fast.

Leakage through the LO distribution system is limited as follows. First separate mixers and LO distribution systems are used for downconversion and upconversion. This allows undersampling to be supported as discussed below. Moreover, this eliminates leakage between the upconversion and downconversion LO systems. To further reduce leakage, the LO circuits contain high-pass filters that eliminate spurious low-frequency signals that result from mixing and that can be mixed back to high-frequency inadvertantly.

Signal Scaling. On upconversion, the signal must be scaled into the desired dynamic range. Upconversion signal scaling can easily be accomplished by adding attenuation to the IF input port. Attenuation here affects only the upconversion path and improves spurious signal isolation.

On downconversion, in order to match the required Signal Conversion Module input level, an on-board jumper cable is added on the downconversion path. This allows variable amounts of attenuation to be added in four places on the RF Front End: the RF Port, the IF input, the IF output, and the jumper loop. As discussed below, rather than add attenuation to the IF output port, it is preferable to add attenuation to the loop. Hence attenuation is never added to the IF output port.

Reducing Spurs. Another important problem that must be solved is the reduction of "mixer spurs" on the downconversion path. On downconversion a mixer works by multiplying a local oscillator signal (LO) with an RF signal. This results in the desired signal of $RF - LO$ being output from the IF port. Unfortunately, however, the mixer also outputs numerous combinations of $m \cdot RF + n \cdot LO$ where m and n take on arbitrary integral values. (In practice, only small values are of importance.)

This can be addressed in a few ways. Using a higher power mixer (i.e. a mixer that uses a higher power LO) produces greater isolation with respect to spurs. Using a higher power LO can be inconvenient as the LO signal source must be strong enough to drive multiple RF Front Ends. More importantly, the best spurious free dynamic range (SFDR) achieved after testing several mixers using this technique was 41.8 dBc which is still less than desired. Moreover, this value was achieved with an input signal strength of 0 dBm. The output signal from the mixer will experience additional signal loss of around 7 dB. Hence the mixer output signal that achieves 41.8 dBc of isolation is -7 dBm. The signal conversion module, however, requires +7 dBm in order to match the full range of the A/D converter. Hence, some dynamic range is lost simply due to this mismatch.

Importantly, isolation of spurs - and hence SFDR - increases as the RF signal incident to the mixer becomes weaker. Hence, excellent SFDR can be obtained if a sufficiently weak RF signal is used. Unfortunately, doing this will further increase the mismatch of the required input signal strength to the Signal Conversion Module, thus simply reducing the RF signal strength is not feasible.

Part of the solution used in this work is to amplify the signal after it has passed through the mixer in order to meet the signal strength requirement of the Signal Conversion Module. This will introduce some undesired noise into the signal. However, much of this noise is unimportant since the signal is so much stronger than the noise. Thus, this noise will be "buried" in the actual noise floor received by RF nodes. Added phase noise, however, will degrade the signal.

Another alternative design would be to use an "active mixer" that amplifies as it mixes. In practice, however, active mixers were found to perform very poorly with respect to flatness due to impedance variation over frequency.

Note that on upconversion, mixer spurs are also produced. The incoming signal is weak enough, however, that mixer spurs are not an issue for upconversion.

Undersampling. Spurious signal isolation can be further improved using undersampling. In order to meet the Nyquist criteria, an A/D converter running at a sample rate of F_{sample} is typically used to sample signals where $F_{signal} < F_{sample}/2$. Signals where $F_{signal} > F_{sample}/2$ will alias into the region $F_{signal} mod F_{sample}/2$. For this reason, digital signal processing systems typically filter signals $F_{signal} > F_{sample}/2$. It is possible, however, to leverage aliasing to perform a "digital downconversion." That is, signals $F_{signal} > F_{sample}/2$ can be intentionally aliased to the desired frequency range. Now, undersampled signals where $F_{signal} mod F_{sample} > F_{sample}/2$ will be inverted in frequency; for this reason, undersampled signals typically are sampled to meet the condition $F_{signal} mod F_{sample} < F_{sample}/2$.

Undersampling is not, however, without drawbacks. The first effect is that additional noise is folded in. This effect is negligible in this case since the additional noise is so much weaker than the signal being sampled. A much more significant drawback of undersampling, in the emulator, is increased sensitivity to sampling clock phase noise - measured in the frequency domain - or sampling clock jitter - measured in the time domain.

The achievable signal-to-noise ratio of a sampled signal is limited by A/D aperture uncertainty t_{jitter} as follows: $SNR = -20 log_{10}(2\pi f_{analog} t_{jitter})$

Solving for t_{jitter} yields: $t_{jitter} = (10^{SNR/-20})/(2\pi f_{analog})$

With an LO of 2.396 GHz and a sampling clock running at 180 MHz, f_{analog} for 802.11b/g channel 11 is (2.462 GHz - 2.396 GHz + 180 MHz) = 246 MHz. Thus, to achieve a target dynamic range of 60 dB requires 0.65 ps jitter rms. Achieving 50 dB requires 2.0 ps jitter rms.

This is further discussed in the Signal Conversion discussion below.

Miscellaneous. In addition, the RF Front End contains a filter on the RF input/output that eliminates any low-frequency components that might be present on the RF path to or from the RF node. Filters are also placed on the IF input and output ports to filter out undesired signals.

The RF port also contains a coaxial connectorized attenuator. This attenuator is set at a low value (e.g. 2 dB), and reduces any impedance mismatch that might be seen by the RF node looking into the RF Front End and vice versa.

Quadrature Support. Note that the RF Front End does not employ quadrature signal paths. This support may be employed on the data conversion module and is outside the scope of this paper.

V. Validation

This section evaluates RF Front End downconversion and upconversion performance. Some of the metrics that will be measured using these tests are:

- Gain/loss - The amount of signal strength gained or lost during signal conversion. On downconversion the final signal level should match the A/D converter's input range. On upconversion the dynamic range should be placed in the desired range (e.g. the low end should extend several dB below the receiver's minimum reception threshold.)
- Flatness - The degree to which gain/loss is constant over frequency. under 2 dB is probably tolerable. A few more dB can be corrected for using a digital filter. Differences of approximately 10 dB become problematic.
- Spurious Free Dynamic Range (SFDR) - The isolation between the desired signal and the closest spur. This is not important for a single channel, but is very important for wideband experiments.

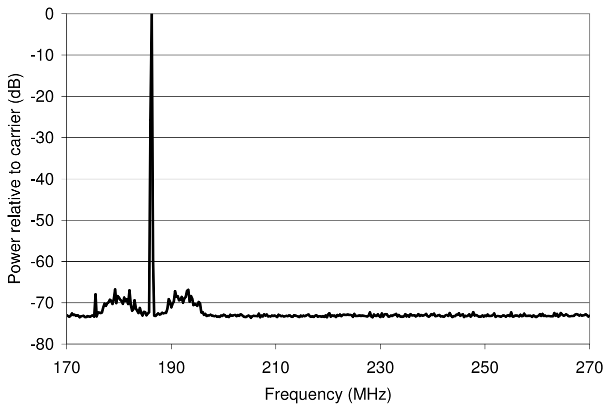


Fig. 5. SFDR Analysis - IF Above Nyquist

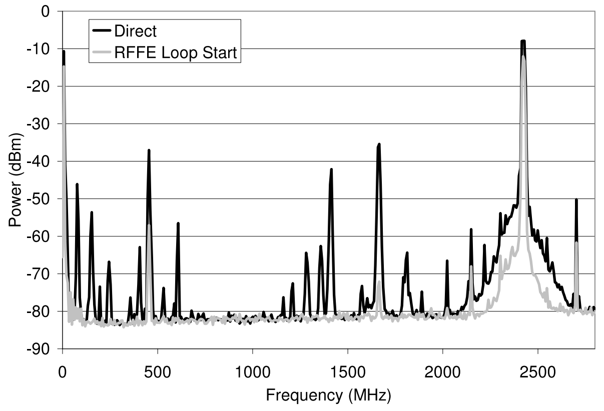


Fig. 6. Direct Vs. RFFE Loop Start

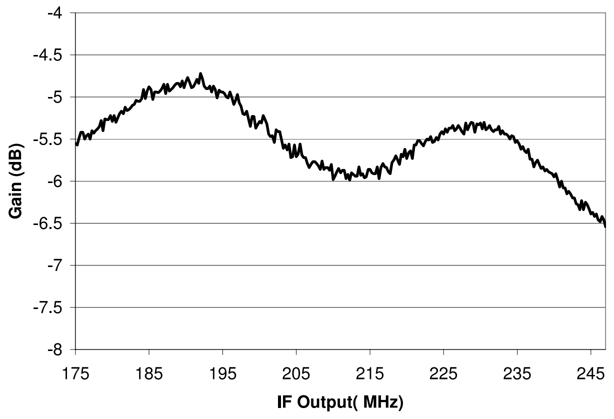


Fig. 7. RF Downconversion Flatness

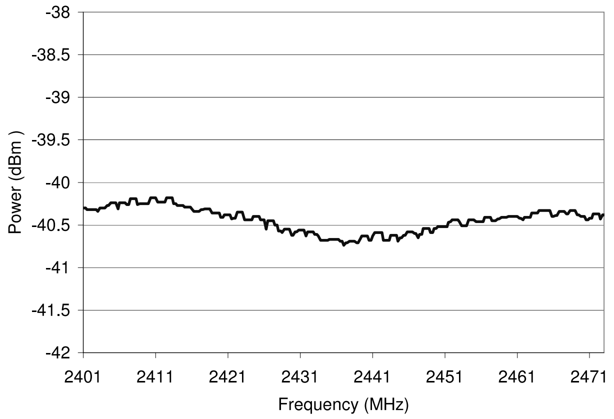


Fig. 8. RF Upconversion Flatness

A wide variety of tests can be conducted to verify fidelity. This section presents results from a few of the most important tests conducted on a single RF Front End. The behavior of other RF Front Ends is similar, but not presented here.

A. Downconversion.

This discussion first presents tests measuring RF Front End downconversion performance.

Spurious Free Dynamic Range (SFDR). Spurious free dynamic range was measured by applying a 0 dBm 2.412 GHz signal to the RF input port of the RF Front End. The LO frequency for this experiment was 2.396 GHz, and LO power was just over 7 dBm. Measurements were gathered from measurement point, the RF Front End's IF Output port, using a spectrum analyzer (resolution bandwidth was 30 kHz.) Two strong spurious signals were observed at 32 MHz and 48 MHz. SFDR in this case was 37 dB.

As discussed in Section IV, undersampling can greatly improve SFDR. Assuming that the RF Front End is targetting an SCM with a sampling rate of 170 Msps, the improved SFDR (at IF output) from undersampling is demonstrated by repeating the SFDR test with an LO of 2.226 GHz and examining IF output between 170 MHz and 270 MHz. Measurements were taken from point (c) as before with the same spectrum analyzer settings except for the new frequency range. Figure 5 shows the result. No spurs are seen in this case. (Spurious signals near the desired signal in this case are due to the signal source.)

Eliminating Spurious Device Output.

Wireless devices generate a broad range of signals in addition to the desired signal output. Unfiltered, these will be aliased into the sampled signal by the SCM. The "Direct" data series in Figure 6 shows the direct output of a Senao NL-5354MP ARIES2 broadcasting 1 Mbps data at 2.412 GHz. The figure shows the card's output between 0 and 2.8 GHz directly connected to a spectrum analyzer with a resolution bandwidth of 100 kHz.

The efficacy of the RF Front End at filtering out these undesired signals is shown in the RFFE Loop start series in Figure 6. This data was obtained by connecting the Senao NL-5354MP ARIES2 source from the previous test to the RF Port and examining the output of the RF Front End downconversion loop start port. Thus, this measurement is after filtering, transmit detection, and transmit switching have occurred. This series shows that the RF Front End successfully filters the vast majority of spurious signals. (It also shows that a small amount of attenuation occurs between the RF input port and the loop start port as expected.)

Flatness. Downconversion performance was measured using an LO of 2.226 GHz at 7 dBm (at the mixer), and an RF signal of 0 dBm swept between 2.401 GHz and 2.473 GHz (the 802.11b/g band in the United States) sent into the RF Front End's RF Port. Figure 7 shows the results measured using a specturm analyzer attached to the RF Front End's IF output port (30 kHz resolution bandwidth was used.) Gain variation during the downconversion process varies by approximately

1.7 dB. Thus, the RF Front End produces a reasonably flat frequency response.

B. Upconversion.

Upconversion performance was then measured using an LO of 2.396 GHz at 7 dBm, and an RF signal of -10 dBm swept between 5 and 77 MHz into the RF Front End's IF input port. Figure 8 shows the results measured using a spectrum analyzer attached to the RF port. Gain in this case varies by approximately 0.7 dB which is a very flat frequency response.

VI. Related Work

A. Wireless Simulators

ns-2 [4] has been the de facto standard tool of the wireless networking community for some time. Yet ns-2's wireless support has not kept pace with current technology, and contains a particularly simple physical layer [1]. As a result, some researchers use commercial simulators such as QualNet [5] and OpNet [6]. Nevertheless, even commercial simulators fail to completely capture radio and MAC behavior. More fundamentally, the real APIs, parameters, and capabilities provided by real wireless hardware are not supported by wireless simulators.

Moreover, real user applications running on real operating systems are not supported using simulators. Thus, to test simulator code in the real world often involves the development of two implementations: one for the simulator, and one for a real device. As a result, simulation-based research is frequently never ported to a real wireless device which hampers verification of results, and lessens its impact and benefit to the community.

B. Wireless Emulators

Simulation's lack of support for real devices can be overcome by combining elements of simulation with real devices. This combination of real devices and simulation is referred to as emulation. Emulation - the combination of real hardware endpoints with simulated network communication - has proven to be a useful technique in wired networking research, and it has an even larger potential in the wireless domain. The wireless MAC and physical layers of most wireless emulators, however, are only very crudely simulated.

C. Wireless Testbeds

More recently, several efforts such as Emulab [7], Orbit [8], and MiNT [9], [10] have begun using controlled wireless testbeds. Like these systems, the emulator supports real devices running, and real applications. The emulator and these systems, however, differ greatly in both their mechanism for modeling signal propagation environments and the range of environments that they are capable of modeling.

The wireless network emulator discussed in this work is - in some ways - a hybrid of traditional network simulation and emulation. Like other systems, the emulator uses real devices; unlike other systems, however, the emulator is capable of full control over the physical signal propagation between between the devices attached to the network.

D. Channel Emulators / Fading Simulators

The most functionally similar approach to the emulation technique discussed in this work is provided by commercial channel emulators [11], [12]. The goal of these emulators, however, is quite different: rather than supporting emulation of an entire wireless network, commercial channel emulators are designed to support very fine-grained emulation of the wireless channel between either a pair of devices or between a small number of base stations and a small number of mobile devices (with the total of both typically being less than 8.)

In contrast to the telecommunications networks typically targetted by channel emulators, wireless data networks are frequently contention-based with half-duplex devices. Channel emulators lack direct support for half-duplex nodes and require external components to support half-duplex nodes which limits their performance. As a result, while channel emulators are very useful for equipment vendors evaluating a new device, the limited scale, lack of support for complete interaction between all nodes, and lack of flexibility make commercial channel emulators an unattractive option.

VII. Conclusion

Physical layer wireless network emulation is a promising technique for conducting wireless network experiments. Experience has shown that this technique yields insights that are difficult to achieve using either testbeds or simulation.

A key element in physical layer wireless emulation is the RF Front End that interfaces RF devices with the data conversion units. Yet the realization of this RF Front End faces several challenges. This work has discussed the requirements of an RF Front End for physical layer wireless network behavior. A design that meets these requirements has been presented, and measurements of the implemented design confirm that it successfully meets its demanding requirements.

References

[1] M. Takai and J. Martin. Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks, October 2001.

[2] G. Judd and P. Steenkiste. Using Emulation to Understand and Improve Wireless Networks and Applications. *Proceedings of NSDI 2005*, May 2005.

[3] J. Mitla. The software radio architecture. *IEEE Communications* 33(5):26–38, 1995.

[4] S. McCanne and S. Floyd. UCB/LBNL/VINT Network Simulator - ns (version 2), April 1999, http://www.isi.edu/nsnam/ns/.

[5] Scalable Network Tech. Qualnet, http://www.scalable-networks.com/.

[6] OPNET Tech. Opnet, http://www.opnet.com.

[7] B. White, J. Lepreau, and S. Guruprasad. Lowering the barrier to wireless and mobile experimentation. *Proc. of HotNets-I*, October 2002.

[8] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols. *Proc. of WCNC 2005.*, March 2005.

[9] P. De, A. Raniwala, S. Sharma, and T. Chiueh. MiNT: A Miniaturized Network Testbed for Mobile Wireless Research. *Proc. of Infocom 2005*, March 2005.

[10] P. De, R. Krishnan, A. Raniwala, K. Tatavarthi, N. Syed, J. Modi, and T. Chiueh. MiNT-m: An Autonomous Mobile Wireless Experimentation Platform. *Proc. of Mobisys 2006*, June 2006.

[11] PROPSim. Propsim c8 wideband multichannel simulator, http://www.propsim.net/.

[12] Spirent Communications. Tas4500 flex5 rf channel emulator, http://www.spirent-communications.com/.