On Power and Energy Trends of IEEE 802.11n PHY

Balaji V. Iyer[†]
Sensors E&M Applications Laboratory
Georgia Tech Research Institute, Atlanta, GA
Balaji.Iyer@gtri.gatech.edu

Thomas M. Conte
School of Computer Science
Georgia Institute of Technology, Atlanta, GA
conte@cc.gatech.edu

ABSTRACT

The main contribution of this work is to decipher the power and energy characteristics of IEEE 802.11 PHY. In this work, we implement an IEEE 802.11n receiver and transmitter benchmark and measure power using an HDL-based scalable clustered-processor called CLAW. Second, we show that such a scalable processor can help conserve power and energy without sacrificing performance. We were able to extract 28% to 43% energy reduction with minimal overhead.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

General Terms

Algorithms, Design, Measurement, Performance

Keywords

IEEE 802.11n, Power and Energy Characterization

1. INTRODUCTION

Communication systems, especially WLAN, are one of the most developing fields today [7, 13, 11]. Most of the WLAN today use some flavor of IEEE 802.11 standard [13]. With the necessity of higher throughput and data-rates, IEEE Communications Society set up an IEEE 802.11 High-Throughput Study Group (HTGC) to come up with a new standard for WLAN communication [13].

HTGC outlined the shortcomings of physical layer (PHY) in the current 802.11 standard. They proposed a new MIMO WLAN scheme that is able to provide high throughput and data-rate called IEEE 802.11n [13]. Currently, a 3rd draft of the standard has been released [15]. Industry generally agrees to implement the standard after the 2nd draft.

[†]This work was done as part of Balaji's Ph.D. work at North Carolina State University

MSWiM'09, October 26–29, 2009, Tenerife, Canary Islands, Spain.

One of the biggest advantages of fast and high data-rate WLAN is that it makes telecommunication systems nomadic [8]. Many such systems use batteries as the sole energy source. Thus, understanding the energy consumption and reducing unwanted energy wastage becomes compulsory requirements of such systems.

There are two major contributions of this work. First, we study the power and energy characteristics of IEEE 802.11n PHY standard as proposed in [15]. We perform this task by building a model of the physical layer as per [15] and using the parameters given by IEEE 802.11 experts in [11, 2, 13, 6, 12]. Second, we show that a dynamic length-adaptive processor called CLAW [5] is able to reduce energy-wastage with minimal overhead on IEEE 802.11n PHY.

2. IEEE 802.11N PHY ARCHITECTURE

Fig. 1 show the major components of the IEEE 802.11n PHY transmitter and receiver. For simplicity, we show a system with two antennas. There are 6 major components in the transmitter: FEC encoder, interleaver, OFDM symbol mapper, MIMO encoder, iFFT, and digital to analog converter. The receiver complements the actions of the transmitter using the inverse of these six components. In this work, we maintain all our work in the digital domain, thus the analog to digital or digital to analog conversion is not modelled.

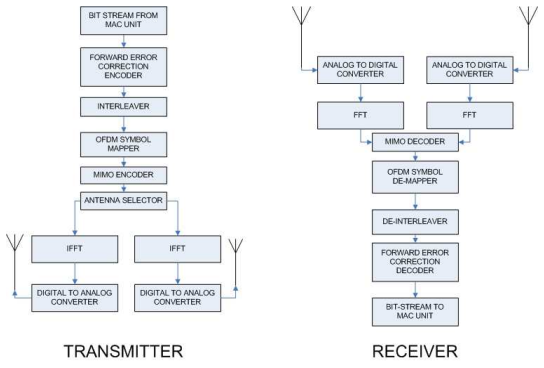

Figure 1: IEEE 802.11n PHY Block Diagram

2.1 FEC Encoding and Decoding

LDPC and Convolutional Encoding are proposed as a possible FEC for IEEE 802.11n [3, 6]. For the best performance [15] advice the use the convolutional encoding with constraint length of 7 and generator polynomials $91(113_8)$ and

$121(171_8)$. Viterbi decoder was chosen for decoding these values. The value of K in the decoder was also kept at 7.

2.2 Interleaving

IEEE 802.11n standard proposes using block-interleaving. Even though this is one of the required units, as per [6], the array size of this unit does not create a performance changes for a normal AWGN channel.

2.3 OFDM Symbol Mapping

Authors of [6] experimented with four schemes: BPSK, QPSK, 16-bit QAM and 64-bit QAM. The authors prove that a 64-bit QAM with a code-rate of 3/4 seem to give "'slightly-better"' results than others.

2.4 MIMO Encoding and Decoding

MIMO encoding and decoding is the heart of the IEEE 802.11n PHY. Almost all papers we encountered agree using Space-Time Block Coding (STBC) for this step [13, 6, 15]. The most popular STBC scheme is the Alamouti Scheme [6, 11, 14]. This scheme has a spatial rate of 1 and the received data can be easily decoded. Alamouti scheme can be used for any number of receiver and transmitter antennas. Authors of [6] show that using more than two antennas just adds extra complexity without any noticable performance improvement.

2.5 Fast Fourier Transform

In this system, the iFFT and FFT are used for modulating and demodulating the signals. The authors of [6] recommend using a radix-2 DIT FFT algorithm.

3. EXPERIMENTAL FRAMEWORK

3.1 Processor Architecture

For this work, we used a clustered scalable VLIW processor called CLAW. Issue-width of CLAW can be modified with the feedback from the architect, the programmer and/or the user during run-time or compile-time [4]. In CLAW, two-issue cores are combined together to create the desired width. In this work, we study the transmitter and receiver characteristics on a 2-Cluster (4-Issue) and 4-Cluster (8-Issue) CLAW machine. Two-cluster machine is the smallest processor domain where the shutoff and code-steering capability of the processor and the compiler can be demonstrated. Four-cluster is shown as an example case where if the architect chose to overdesign the processor, they can still save energy using the cluster-shutoff mechanism of CLAW. For more details about CLAW, the reader is referred to [4, 5].

3.2 Synthesis and Power Measurement

CLAW is synthesized by Synopsys Design Compiler using the Artisan SAGE-X 90 nm Regular-VT (RVT) standard-cell library (1V, 25^oC with typical operating characteristics). International Technology Roadmap for Semiconductors (ITRS) recommends using a RVT library for embedded processor synthesis. The synthesized processor is then place and routed using Cadence Design Encounter. The output of this step is a parasitic file (SPEF format) that provides the wire-lengths and their capacitance.

The executable is simulated using Verilog-XL HDL simulator. The text section and the data sections (read-only data and read-write data) are extracted from the executable and stored in appropriate text files. The test-bench reads these two files and stores them in the appropriate memory array. Memory is modeled as an array inside the test bench. When the reset is set on the processor, the program counter (PC) is pointed to the "main" function. During execution, the processor requests the testbench to provide the appropriate instruction at the specific address location (pointed by PC). Loads and stores are handled in the similar way by the LSU.

The VCD, SPEF, and the synthesized Verilog files are analyzed by Synopsys Primetime to provide power values. The energy values are computed by multiplying this power-value with the cycle-time provided by Verilog-XL.

4. IEEE 802.11(PHY) IMPLEMENTATION

To date, we did not find any public open-source implementation for IEEE 802.11n PHY. Thus, we had to implement an unbiased IEEE 802.11n transmitter and receiver. We found that the best way to accomplish this is to construct the algorithm using published benchmarks.

Convolutional-encoder, Viterbi decoder, FFT, and iFFT were taken from EEMBC telecom suite [1]. Block-interleaver and QAM were implemented using algorithms given in [10]. Reference input given by EEMBC with the convolutional encoder was used as input to transmitter. AWGN is added to the output of iFFT and is used as the input to the receiver.

We were unable to find any public C-language implementation of Alamouti's algorithm. The authors of [11] have submitted a Matlab version of this algorithm to Mathworks. We used their source-code and decided to manually convert it to C-language. For matrix-multiplication in the Alamouti scheme, we used matrix-multiply kernel available in DSP-Stone benchmark [16].

5. RESULTS

5.1 Instruction Distribution

One of the important characteristics in any benchmark is the instruction-distribution in the dynamic trace. Fig. 2 and 3 shows this information for the transmitter and the receiver. In these graphs, we have eliminated the NOP count, since they do not provide any insights into the algorithms.

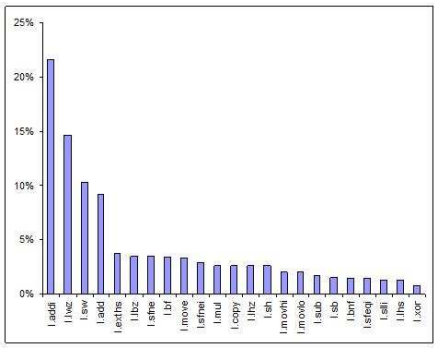

Figure 2: Transmitter Dynamic Inst. Distribution

The most commonly occurring instruction is the "add immediate" (*l.addi*) instruction. A common usage of this instruction is to increase or decrease the stack size. This in-

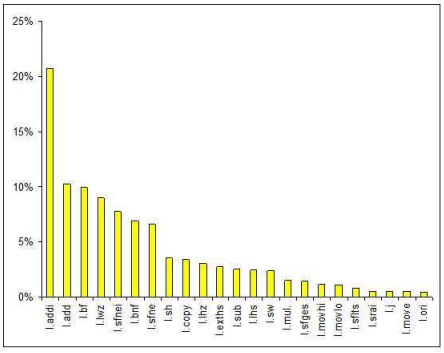

Figure 3: Receiver Dynamic Inst. Distribution

struction is also used when a constant value, known at compile time, is added to a variable stored in a register.

In convolutional encoder and Viterbi-decoder, an array of data is passed into the function to do the appropriate transformation. This array is stored in the memory and accessed using load and store instructions. The majority of load and store instructions are contributed by these two units. The convolutional encoder and Viterbi decoder also performs several shift and rotate operations. Many of the extends (e.g. *l.extbs*) and shifts (e.g. *l.srai*, *l.slli*) occured in these units. Similarly, block-interleaving and de-interleaving write information into an array in row-major and column-major format. Such tasks are again performed using load and store instructions.

l.Movlo and *l.movhi* are a pair of instructions used to move a 32-bit value into a register. These two instructions are synonymous to *movw* and *movt* instruction-pair in the ARM architecture. These instructions are used to move address of a global variable or a function. In our benchmark, several global variables are used for communicating data among functions. Moving values to and from these variables requires the use of this instruction pair. Finally, the multiply instructions (*l.mul*) occurred inside the STBC unit.

5.2 Parallelism

Another important aspect is to see the amount of parallelism emitted by these two algorithms. Table 1 shows the instruction per cycle(IPC) and cycle-count.

Table 1: Parallelism Information

		4-Issue	8-Issue
Transmitter	IPC	1.17	1.32
	Cycle-count	73184	541953
Receiver	IPC	0.98	1.12
	Cycle-count	1299829	1137351

Overall these benchmarks are not very parallel. Viterbi decoder and the convolutional encoder are one of the least parallel benchmarks in the EEMBC benchmark suite [5, 1]. FFT is also well known for not providing high-levels of parallelism in software. STBC transmitter contains a lot of matrix multiplications which can be easily parallelized. This is one of the main reason for using Alamouti's scheme to perform STBC [2, 6]. STBC decoder has significant amount of serial comparisons in the maximum-likelihood(ML) detecting phase that helped reduce the IPC of the receiver. Similarly, the task of the STBC decoder is much more intensive than the STBC encoder, which contributed to the increase in the cycle-count.

5.3 Power Dissipation

Fig. 4 and 5 shows the power distribution of the transmitter and the receiver. In both the units, the STBC consumed the most amount of power. This is because they used a lot of multiply instructions which uses the multiply and accumulate unit. This unit is the most power hungry unit in the processor [5].

In the transmitter, the STBC unit dissipated 47% of the total power. The convolutional encoder and iFFT dissipated 23% and 20%. The QAM and the interleaver each dissipated 10% of the total power. In the receiver, the STBC unit dissipated 55% of the total power. The FFT and the Viterbi decoder each dissipated 20% of the total power. The QAM and de-interleaver dissipated 5% and 2%, respectively.

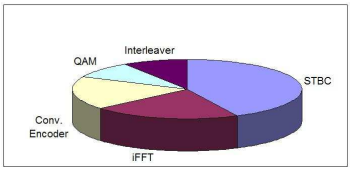

Figure 4: Transmitter Power-Dissipation Percent

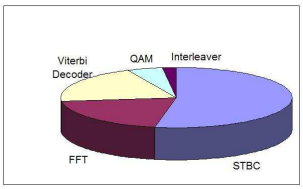

Figure 5: Receiver Power-Dissipation Percentage

5.4 Energy Consumption

Fig. 6 shows the energy consumption of the transmitter and receiver. We report the energy consumption without any shutoff insertion (the base case) and using the shutoff capability of CLAW. The dominating component is the dynamic energy caused by switching activity. Static (or leakage) energy contributed only 13% of the total energy since Artisan 90nm RVT libraries are inherently low-leakage libraries [9].

The receiver consumed significantly more energy than the transmitter due to the complexity of the STBC receiver. This increased the cycle-time, which lead to energy-increase. Using the shutoff feature, a 29% dynamic energy reduction was seen on the transmitter. The convolutional encoder was the least parallel unit and the profiling compiler was able to find several holes in this routine to shutoff the 2nd cluster. This is one of the leading contributor of this energy reduction in the transmitter. In the receiver, we were able to achieve a 28% energy reduction. The profiler was able to shutoff unused units in the ML detector inside the STBC decoder and the Viterbi decoder. Since the ML detector has several comparisons and jumps, many of these were scheduled on cluster 1. This freed up cluster 2 and the processor was able

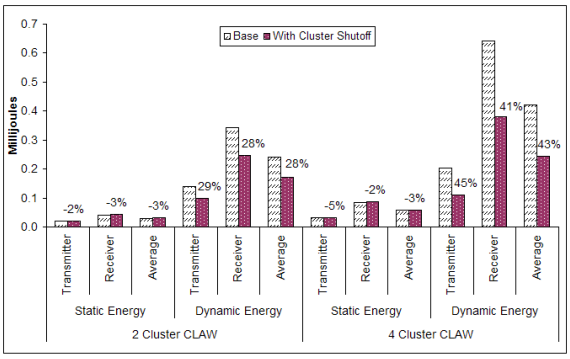

Figure 6: Energy Consumption

to shutoff this cluster. The Viterbi decoder, which is also a ML detection algorithm, also did not emit high amount of parallelism and the profiling-compiler was able to find holes in the schedule to free up one cluster. Inserting the shutoffs increased the cycle-time of the transmitter and the receiver by 2.5% and 2.3%. Static power barely changed in both the receiver and the transmitter. Even though we are gating unwanted units, it is impossible to physically shutoff a unit on HDL-based designs. The increase in energy is proportional to cycle-time increase.

Similar trends were noticed in four-cluster CLAW machine. Using the shutoff mechanism, a 45% dynamic energy reduction was achieved in the transmitter. In the receiver, we were able to save 41% of dynamic energy. Our cluster-assignment algorithm sits on top of the scheduler and assigns instructions given by the scheduler. When the number of clusters were changed from two to four, the issue-rate was doubled from four to eight. This helped the scheduler have a deeper view into the instruction list, and the cluster-assignment procedure was able to assign more instructions to cluster 2 and 3. Moreover, since we are inserting the shutoff-instructions during compile-time, a more conservative approach was taken. The bottleneck of the algorithm (in terms of parallelism) remained the same for both the cluster-configurations. Static energy trends also remained the same across the cluster-configurations.

6. RELATED WORK

The main aim of this study is to characterize IEEE 802.11n as described in the current draft in [15]. We do not propose any modifications or enhancements. To our best knowledge, there has not been another study that have studied the energy and power trends of the IEEE 802.11n standard. Xiao in [13] has pointed out the shortcomings of the current IEEE 802.11 standard and have proposed several changes that were implemented in [15].

In this study, we take the parameters and recommendations made by several WLAN researchers. Otefa,et al. in [6] have worked out all the equations specified in the standard and provided numerical parameters that will help extract high performance from the existing standard. We use this paper as a base to model the transmitter and receiver.

Tarokh, et al. in [11] have provided the performance values for STBC techniques. They have also submitted a Matlab implementation to Mathworks. We use this implementation to model STBC scheme. Zheng and Tse have also performed an indepth study on STBC algorithm [14]. Similarly, Wang et al. in [12] have studied the BER and SER for systems using STBC and QAM.

Cai et al. in [3] have provided an alternative to convolutional codes by proposing the use of LDPC codes. G. Fermenias and Riera-Palou in [7] have provided details about using OFDM codes for IEEE 802.11n.

7. CONCLUSION

In this paper, IEEE 802.11n PHY transmitter and receiver were assembled and characterized. This algorithm were simulated on a HDL based processor that was synthesized using a well-characterized standard-cell library and had their energy and power measured using industrial strength tools.

The STBC unit to be most power-hungry unit in both the transmitter and receiver. STBC encoder emitted a higher parallelism and thus took lot less cycles to execute than the STBC decoder. This had adverse effects on the total energy consumption. The CLAW framework was able to analyze the algorithm and find holes in the schedule to shutoff off unwanted units and issue-widths to save energy. On average, 28% and 43% energy reduction was shown on the two and four-cluster CLAW.

This study gives an detailed and accurate view about the energy trends inside the IEEE 802.11n PHY algorithm. This information can be used by researchers to find the power and performance bottlenecks of 802.11n and make implementation decisions and power-budgets.

8. REFERENCES

[1] http://www.eembc.org

[2] S. A. Alamouti, "A Simple Transmit Diversity Technique for Wireless Communications," *IEEE JSAC*, 1998

[3] Z. Cai et al., "Efficient encoding of IEEE 802.11n LDPC Codes," *Electronic Letters*, Vol. 42, No. 25, 2008

[4] B. V. Iyer, J. G. Beu and T. M. Conte, "Length Adaptive Processors: A Solution for Energy/Performance Dilemma in Embedded Systems," *INTERACT*, 2009.

[5] B. V. Iyer, "Length Adaptive Processors: A Solution for Energy/Performance Dilemma in Embedded Systems," *Ph.D. Thesis* North Carolina State University, 2009.

[6] A. M. Otefa, et al.,"Performance Analysis of 802.11n Wireless LAN Physical layer," *5th ICICT*, 2007

[7] F. Riera-Palou, G. Fermenias, "Improving STBC performance in IEEE 802.11n using group-orthogonal frequency diversity," *WCNC*, 2008

[8] J. M. Perez, V. Fernandez, "Low Cost encoding of IEEE 802.11n," *Electronic Letters*, Vol. 44, No. 4, 2008

[9] L. Pickup and S. Tyson, "Hot Chips? . . . Not!" *Chip Design Magazine*, pp. 26-29, August/September 2004

[10] B. Sklar, "Digital Communications," *Prentice Hall*, 1999

[11] V. Tarokh, et al., "Space-Time Block Codes for Wireless Communications: Performance Results," *IEEE JSAC*, 1999

[12] C. Wang, et al., "Space-Time Block Coding Performance Results Under QAM," *IEEE TENCON*, 2002

[13] Y. Xiao, "IEEE 802.11n: Enhancement For Higher Throughput in WLANs," *IEEE Wireless Comm.*, 2005

[14] L. Zheng, D. N. C. Tse, "Diversity and Multiplexing: A Fundamental Tradeoff in Multiple-Antenna Channels," *IEEE Trans. on Info. Theory*, Vol. 49, No. 5, 2003

[15] "IEEE 802.11n Standard/D11: 'Draft 3.0: IEEE Standard for Local Metrapolitan network-specific requirements. Part 11: WLAN Medium Access Control and Physical Layer specifications: Enhancements for Higher Throughput," 2007

[16] V. Zivojnovic, et al., "DSPStone: A DSP-oriented Benchmarking Methodology," *Proc of ICSPAT*, 1994