Technical University Berlin

Telecommunication Networks Group

Power Consumption, Throughput and Packet Error Measurements of an IEEE 802.11 WLAN Interface

Brian Burns and Jean-Pierre Ebert

blburns@hotmail.com, ebert@ee.tu-berlin.de

Berlin, August 2001

TKN Technical Report TKN-01-007

TKN Technical Reports Series

Editor: Prof. Dr.-Ing. Adam Wolisz

Abstract

Current simulation results of WLAN power consumption and network performance rely on assumptions and percentages. Realistic values are needed to parameterize simulations and verify results. Power consumption, throughput, PER, and energy/bit were measured on a WLAN assuming a simple network scenario. Values of power consumption versus packet size, data rate, and antenna transmission power were obtained for the different operating modes of a WLAN (idle, sleep, receive, transmit). Additionally, network performance measurements like throughput, packet error rate and energy consumption are given for different packet sizes, data rates, antenna transmission power, and distance between nodes. These measurements can be used to verify simulation models.

Table of Contents

Figures and Tables

1 Purpose

The reduction of power consumption in Wireless LANs has been addressed extensively in recent research literature. The problems with the current results are that they are often presented using percentages of increased idle or sleep times, or they address only a single part of the wireless network interface card. Obviously, this does not give a clear figure of how the mechanism to reduce power consumption operates in realistic scenarios. Furthermore, there are no clear figures that tell how much power is drawn in the different working modes of a WLAN network interface card. This is necessary to draw conclusions about which power saving strategies should be used.

Therefore, the purpose of this work was to determine the power consumption of a network interface card assuming different working modes of operation and parameter settings of the WLAN NIC (network interface card). The results of this work can be used to parameterize WLAN simulation models with realistic values and to draw further conclusions on how energy saving strategies should operate.

In addition to operating mode measurements, power consumption measurements in conjunction with measurements of the PER (packet error rate) and throughput in some simple network scenarios were conducted. This was to verify later results of simulations that assume the same simple network scenarios.

2 Preconsiderations

Prior to gathering any data and the experimental setup design, a number of preconsiderations had to be performed. These preconsiderations dealt with the hardware, software, and the IEEE 802.11 specification. What follows are the preconsiderations we took into account before any work was begun on the project and what decisions were made about the setup because of them.

2.1 Hardware Preconsiderations

2.1.1 IEEE 802.11

The IEEE 802.11 specification [1] is a wireless LAN standard developed by the IEEE (Institute of Electrical and Electronic Engineering) committee in order to specify a LAN type "over the air" interface. The IEEE 802.11 standard places specifications on the mechanisms and parameters of both the physical (PHY) and medium access control (MAC) layers of the network. The PHY layer, which actually handles the transmission of data between nodes, can use either direct sequence spread spectrum, frequency-hopping spread spectrum, or infrared (IR) pulse position modulation. The MAC layer, supported by the underlying PHY layer, is concerned primarily with rules for accessing the wireless medium. Two network types are defined: the Infrastructure Network and the Ad Hoc Network. An Infrastructure Network is a network architecture for providing communication between wireless clients and an Access Point. The transition of data from the wireless to the wired medium is via the Access Point. An Ad Hoc network is an architecture that is used to support mutual communication among wireless clients. Typically created spontaneously, an ad hoc network does not need an Access Point to be part of the network but can support access to the wired network via dedicated nodes.

2.1.2 Aironet PC4800B PCMCIA card

All measurements in these experiments were taken using Aironet PC4800B wireless network interface cards (WNIC). Two cards were used to send and receive packets over an ad hoc network while the instantaneous power consumption, throughput, and PER of the transmitter were recorded. There is an open source driver for the Linux operating system [2], which was used in these experiments. This allowed for easy modification of the WNIC parameters and the driver itself as the experiments were conducted. It made the throughput and PER directly accessible by kernel hooks.

The PC4800B WNIC operates in the 2.4 GHz Industrial, Scientific, and Medical (ISM) band. Data is transmitted over a half-duplex radio channel operating up to 11 Mbit/s. It uses Direct Sequence Spread Spectrum (DSSS) transmission, which provides enough redundancy built into the signal that the PC 4800 WNIC will usually be successful during transmission within ranges of several tens of meters.

2.1.3 Aironet PC 4800 operation modes

The operating modes of the Aironet PC 4800B PCMCIA NIC studied in these experiments were idle, sleep, transmit, and receive. In transmit mode, the device is transmitting data; in receive mode, the receiver is receiving data; in idle mode, the device is neither sending nor receiving but scanning for a valid signal which makes it similar to receive mode; in sleep mode, the transceiver circuitry is powered down but not completely switched off which allows for a relatively fast bring-up of the device. Off mode was not studied in these experiments. Here the WNIC is completely switched off. No power is consumed and a power-on of the NIC can take up to several seconds.

The chipset Aironet used to build the PC4800B PCMCIA card is based on the Intersil PRISM I 11Mbit/s chipset [3]. A schematic of it is shown below. It consists of seven main chips. From right to the left, they are the MAC processor, the baseband processor, the quadrature IF modem, the dual frequency synthesizer, the RF/IF converter, the low noise amplifier, and the RF power amplifier.

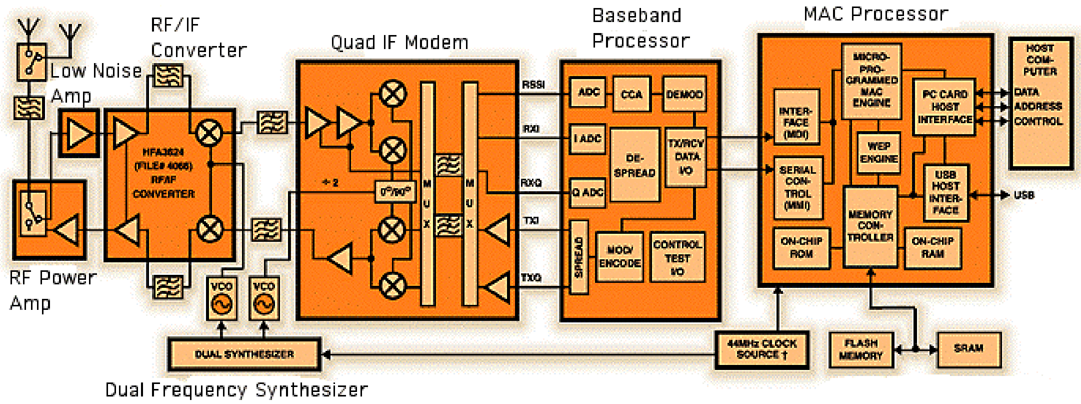


Figure 1: Intersil PRISM I chipset[1]

Each chip operates in a different power mode when the card is in idle, sleep, transmit, or receive. The MAC and baseband processors have power modes for sleep, idle, transmit, and receive. The quadrature IF modem, dual fequency synthesizer, and RF/IF converter have power modes for transmit, receive, and a single low power mode for idle and sleep. The low noise amplifier operates only during reception and idle, and the RF power amp operates only during transmission. The table below summarizes each operating mode of the PC4800B WNIC and the operating modes of the chipset. The power consumption values below represent estimates based on the specifications for each IC in the chipset, their purpose is as a reference only.

[1] The figure source is [3].

Chip	Sleep	Idle	Transmit	Receive
MAC Processor	on, 5mW	on, 40mW	on, 125mW	on, 125mW
Baseband Processor	on, 2mW	on, 23mW	on, 33mW	on, 100mW
Quadrature IF Modem	on, 10mW	on, 10mW	on, 400mW	on, 500mW
Dual Frequency Synth.	on, 0.075mW	on, 0.075mW	on, 40mW	on, 40mW
RF/IF Converter	on, 0.05mW	on, 0.05mW	on, 300mW	on, 100mW
Low Noise Amp	off	on, 35mW	off	on, 35mW
RF Power Amp	off	off	on, 1.6W	off
maximum total power:	**~20mW**	**~110mW**	**~2.4W**	**~1.0W**

Table 1: Chip power consumption and operating modes (V_{dd}=3V or 5V, I_{dd}=max)

The values above show that power consumption of the Aironet PC 4800 is higher while transmitting than while receiving, and are much higher than either idle or sleep modes. The values used represent the maximum power consumption by an IC in the four operating modes. The estimated total power consumption takes into account the primary circuitry of the Intersil PRISM I chipset and does not consider secondary ICs or power savings in the protocols. Actual values will vary greatly from the estimates above.

2.1.4 Aironet PC 4800 Control Parameters

We differentiated between parameters that were set to a fixed value throughout the measurement campaign and parameters that were modified before running the next measurement. The first set of parameters is referred to as constant parameters, and the latter is referred to as modified parameters.

The Aironet PC4800B WNIC has a number of parameters that are conveniently controllable through the /proc file system under Linux (/proc/aironet/eth1/Config). Newer versions of the driver support modification of the parameters through I/O controls.

2.1.4.1 Constant Parameters

During the course of these experiments, some of the NIC parameters were held constant. Their values and definitions as defined in the Aironet PC4800 WNIC manual (see [4]) follow:

MTU = 2400
Mode = adhoc
Channel = 6
LongRetryLimit = 1
ShortRetryLimit = 1
RTSThreshold = 2312
FragThreshold = 2312
TXDiversity = both
RXDiversity = both
PowerMode = CAM
Modulation = cck
SSID = Brian

- **MTU**

Specifies the Maximum Transferable Unit (MTU) size. Packets greater than this value are fragmented at the IP level and transmitted as two or more packets. This value is determined by the actual networking hardware. Theoretically, the PC4800B WNIC is able to transmit packets much larger than this. We chose a value of 2400Bytes and changed the driver accordingly to support MAC packets with at least 2312 byte payload, which is the maximum payload as defined in [1].

- **Mode**

Specifies the network type the node is communicating on. There are two options for the Mode in the NIC driver:

> **Infrastructure Mode:** This mode can be used to set up a connection to a wired network, such as Ethernet or Token Ring. This mode requires an Access Point to gain access to the wired network.
>
> **Ad Hoc Mode:** This mode is used to set up a small, temporary network between two or more computers without the necessity of an Access Point. We used this mode throughout the measurements.

- **Channel**

This parameter specifies the channel identifier the unit will use if it must start its own network (e.g. ad hoc mode). For all other situations (infrastructure mode), the radio will scan for the proper frequency. We chose a channel that was not in use by any other wireless network in the neighborhood to avoid interferences and unwanted influences during the measurements.

- **Long Retry Limit**

Specifies the number of times a long packet will be retried before the packet is dropped and a transmit error is reported to the driver (default is 16). We set the value to one to ensure correct packet error measurements. Otherwise, a packet could be reported as successfully transmitted when the success is based on several retransmissions, skewing results.

- **Short Retry Limit**

Specifies the number of times that a short packet will be retried before a packet is dropped and a transmit error is reported to the driver (default is 16). We set the value to one to ensure correct packet error measurements for the same reason as stated above.

- **RTS Threshold**

This parameter controls what size data packet the low level RF protocol issues to an RTS packet. There are several trade-offs to consider when setting this parameter. Setting this parameter to a small value causes RTS packets to be sent more often, consuming more of the available bandwidth, therefore reducing the apparent throughput of other network packets. However, the more often RTS packets are sent, the quicker the system can recover from interference or collisions (e.g., in hidden terminal scenarios). The RTS threshold value was set to 2312 (no use of RTS) since the measurements were conducted in a simple two-node setup where one node was sending and the other receiving. This avoids hidden terminals as well as collisions. Additionally, the RTS threshold distinguishes long and short packets. Packets with a payload smaller than the RTS threshold are counted as short packets and vice versa.

- **Fragment Threshold**

This parameter defines a threshold above which RF packets will be split up or fragmented. If a packet is fragmented or transmission of part of it is interfered with, only the portion that was unsuccessful would need to be resent. The throughput will generally be lower for fragmented packets since fragmentation consumes a higher portion of the RF bandwidth in a scenario with relatively good channel condition. Fragmentation can improve the performance to a certain extend if the radio channel is bad. We used a value of 2312 (no fragmentation) to achieve objective packet error measurements.

- **TXDiversity/RXDiversity**

This allows the WNIC to use the stronger signal from the two antenna ports. Diversity can help the radio maintain the RF connection in areas of interference. Due to the nature of how RF signals are affected by the surroundings, one antenna may be in an RF “null” where the signal is very weak, but the other antenna (even though it is only a small distance away) may have a stronger signal strength. The PC Card automatically selects the antenna that has the highest signal strength. We used ‘both’ since we believe that this is the most used operation setup.

- **PowerMode**

The Power Saving Protocol allows computers (usually portable computers) to power up only part of the time to conserve energy. If a client node is using the Power Saving Protocol to communicate with the network, the Aironet Access Point must be aware of this mode and implement additional features such as message store and forward. If the client node is powered from an AC line, PSP should not be used. Although an ad hoc Power Saving Protocol is defined in [1] the cards used support it in infratructure mode only. For completeness we give a short explanation of the possible modes:

Constant Awake Mode (CAM): Constant Awake Mode is the normal mode for desktop machines or other machines where power consumption is not an issue. It keeps the radio powered up continuously so there is little latency for responding to messages. This mode is recommended for devices where high availability is desired.

Power Save Mode: Power Save Mode is recommended for devices where power consumption is a major concern, such as small battery powered devices. If the client node is powered from an AC line, PSP should not be used. Power Save Mode causes the Access Point to buffer incoming messages. The Aironet 4000 Series Wireless LAN Adapter must wake up periodically and poll the Access Point to see if there are any buffered messages waiting. The PC Card can request each message and then go back to sleep.

Fast Power Save Mode: Fast Power Save Mode (Fast PSP Mode) switches between PSP and CAM based on network traffic. When retrieving a high number of packets, Fast PSP Mode will switch to CAM to retrieve the packets. Once the packets are retrieved, it switches back to PSP.

Maximum Power Save Mode: Maximum Power Save Mode (Max PSP Mode) can only be used in conjunction with PS or Fast PSP Modes. This mode allows the Aironet 4000 Series Wireless LAN Adapter to conserve the most power while still maintaining an infrastructure connection. Using Max PSP Mode conserves power but will reduce throughput.

- **SSID**

The Service Set Identifier (SSID) controls access to a given wireless network. This value MUST match the SSID of any/all Access Points a wireless node wants to communicate with. If the value does not match, access to the system is not granted. The SSID can be up to 32 characters (case sensitive).

2.1.4.2 Modified Parameters

The following WNIC parameters were modified throughout the experiments by automated scripts. Each packet size was sent at each data rate and transmit power. All of the parameter values used in the experiment are shown below:

DataRates = 150 0 0 0 0 0 0 0, 132 0 0 0 0 0 0 0, 139 0 0 0 0 0 0 0, 130 0 0 0 0 0 0 0
PacketSize = 1, 64, 192, 448, 704, 960, 1216, 1472, 1728, 1984, 2248
XmitPower = 1, 5, 20, 50

- **DataRates**

Specifies the data rates that will be supported by a given radio device in the Basic Service Set (BSS). The options available are 1 Mbit/s, 1 and 2 Mbit/s, 1 and 5.5 Mbit/s, 1 and 11 Mbit/s, 2 Mbit/s, 2 and 5.5 Mbit/s, 2 and 11 Mbit/s, 5.5 Mbit/s, 5.5 and 11 Mbit/s, or 11 Mbit/s. The values specified above correspond to 11Mbit/s, 5.5Mbit/s, 2Mbit/s, and 1Mbit/s, respectively. By following a special notation, the card was configured to support exactly one data rate in a single measurement. The basic data rate and the supported data rate were set to the same (single) value.

- **PacketSize**

Determines the length in bytes (+ 64 bytes fixed packet overhead) of data packets transmitted over network

- **XmitPower**

Selects the next highest programmed power level for transmit in mW (default is 50mW)

2.1.5 Wireless End Systems (Laptops)

There were two laptops used as network nodes in these experiements. They were both Sony Vaio PCG-F304s with a 366 Mhz Pentium II processor, 64 Mbyte RAM and a 6 GB Harddisk.

2.1.6 Digital Oscilloscope

The oscilloscope used to take the power consumption traces was the National Instruments PCI 5102 oscilloscope. The NI 5102 is a dual-channel 20 MS/s digitizer for use with PCI, PXI/CompactPCI, USB, PCMCIA, or ISA bus computers. It features two analog input channels, each with 15 MHz of analog input bandwidth. The analog voltage input has a range of ±50 mV to ±5/50 V using the NI5102 X1/10 probes. The NI 5102 uses a pair of 20 MS/s, 8-bit resolution ADCs to digitize the input signals. The real-time sampling rate ranges from 20 MS/s down to 1 kS/s. The relative accuracy is ±1 LSB typically and 1.8 LSB at maximum. The NI 5102 has 16,777,088 samples of onboard acquisition memory per channel if it is acquired post-trigger. Data is acquired into the onboard memory before being transferred to the host PC system memory. The PCI and PXI/CompactPCI versions of the NI 5102 can transfer acquisition samples across the PCI, PXI/CompactPCI bus system memory in real time.

2.2 Software Preconsiderations

2.2.1 Linux

Linux is an open source operating system based heavily on the POSIX and UNIX API's. It was used as the operating system for the two network nodes in these experiments. It was chosen because of its open source kernel, easy access to system drivers, and reliability. Perl and Awk scripts are also easily written and executed in the Linux environment. This allowed us to write a number of scripts to automate much of the experiments and analysis. We installed Linux kernel version 2.2.13 with the SuSE 6.2 distribution on each laptop. Version 3.1.19 of the Linux PCMCIA drivers were also installed to support the Aironet PC 4800 WNICs.

Access to the device drivers was essential during these experiments. The drivers of the Aironet PC 4800 WNICs had to be modified in order to monitor the network traffic on a very low level and to support packets with a 2312 Byte payload[1]. Linux allows the user to change source code and recompile the drivers for almost any device on the system.

2.2.2 Aironet PC 4800 Drivers and Snuffle

Snuffle[2] is a network trace program that allows the user to easily record traffic on any network interface on a Linux machine. It was used in these experiments to monitor the network traffic (time stamps, MAC packet sizes and MAC packet status) across the transmitting machine. Snuffle is mainly a library that provides network trace objects to by placed in the kernel. It provides functions to pass the traced data from the kernel to the user space and to save the data into trace files. Configuration, start/stop of measurements, selection of kernel objects to be traced, as well as saving traced data is performed via a graphical user interface. A command-line version of Snuffle was developed during the course of these experiments, therefore it was possible to start and stop the Snuffle trace program through automated scripts. This proved useful because of the large volume of traces made.

Linux allows the user to easily modify the drivers of any device on the system. Many of the parameters modified in these experiments are explained in the “Hardware Preconsiderations” section above. However, for Snuffle to monitor the data traffic across the WNICs, Snuffle trace objects were placed in the driver source code of the Aironet PC 4800 wireless NICs. Snuffle trace objects are programming constructs placed in the driver to specify what data to record and which data types they should be. Trace objects were placed in the airo.c driver of the Aironet PC 4800 NIC. In order to acquire data on successful placement of packets in the driver’s packet queue or packets dropped because of queue overflows, two trace objects were placed in the queuing routine of the driver.

---

[1] At this time, the driver we used did only support packets with 1518 Bytes at maximum. Newer versions of the driver have a limit at 2312 byte as defined in the IEEE 802.11 standard.

[2] Snuffle was developed in house in the telecommunications group at the TU-Berlin

The other two trace objects were placed into the interrupt service routine to acquire data on successful and non-successful transmission of a MAC packet, respectively (note: this is meaningful only if the retransmission counter is set to one). This ensures that a packet will be sent only once without any automatic MAC retransmission. Snuffle was then able to generate trace files that contained an entry for each packet sent over the ad hoc network. Each entry contained the time stamp, packet size, and an indicator of transmission success or failure for the packet. These trace files were used to calculate the Throughput and packet error rate of each series of tests.

2.2.3 Load generation program Netperf

Netperf [6] is a network-benchmarking program that uses bulk data transfer to measure certain aspects of performance. It was used in these experiments to generate the necessary traffic over the ad hoc network. Data streams can be sent using either TCP or UDP and the Berkeley Sockets interface. It works using the client/server model. One station on the network acts as the server, receiving data and sending acknowledgements back to the sender for each packet received. Another station acts as the client, sending data streams to the server and measuring throughput according to the number of acknowledgements received. In Netperf, no control messages are sent in the data connection. Exchange of control messages is done at the beginning and at the end of the data connection. This ensures that there is no influence by the control connection on the measurement. For our measurements we used the UDP protocol. It is possible through Netperf to specify the packet size and the duration of the UDP stream. These options are controlled through the command-line, so it was easily done through the automation scripts. For the sake of accuracy and detail, the measurements taken by Netperf were ignored, only the aspect of its bulk data transfer was used. The measurements were taken by Snuffle as described above.

Below is the command used to execute Netperf during these experiments. The four switches specify the destination IP, the type of data stream, the length of the measurement in seconds, and the packet size, respectively. Prior to this however, the Netperf server is started on the remote machine.

```
netperf -H (remote IP) -t UDP_STREAM -l 20 -- -m (packet size in bytes)
```

2.2.4 Oscilloscope Software

The NI 5102 comes with an extensive software package to be run on Windows 3.1,95,98, or NT. VirtualBench-Scope, which is shipped with the NI 5102, is a soft front panel that controls the NI 5102 with no programming required. All hardware features of the NI 5102 are accessible by the software and it is used just as you would use a stand-alone instrument. VirtualBench was used initially to test and debug the experimental setup, but it proved inadequate during the experiment itself. VirtualBench is a purely graphical interface and was useless because it could not be controlled over the command line.

The NI 5102 also comes with a very broad programmable software base that allows command line interfacing to the scope. It supports the LabVIEW[1], BrigdeVIEW[2], LabWindows/CVI[3], MSVC++, Borland C++, and Visual Basic development environments.

In this experiment, MSVC++ was used to fully program the scope and interact with it through the command-line. Numerous examples are also provided with the NI 5102 scope. Due to the complexity of the C++ API that came with the scope, an example program was modified to suit the purposes of these tests. It proved to be much more reliable and simpler to take this approach than to write a program attempting to utilize the API. MultiChannelAcquire was the example chosen because it best suited the experiment. It takes a simultaneous dual-channel trace for an allotted amount of time and streams the data to disk. The input voltage range was set ±5V at an 8 bit resolution which resulted in a granularity of .019V for every bit. These settings lead to sufficiently accurate results. MultiChannelAquire was further modified to take a one-second trace of data at a sampling rate of 1MHz from the wireless NICs and to write the trace to an output file specified through the command line. This resulted in a trace file of about 10MB that proved to be sufficient for use in later evaluations. The calculations of current and power were done on a point-by-point basis during the trace and only the instantaneous power was written to file. This allowed the oscilloscope to be run from the automation script and output the desired data without further calculation.

2.3 IEEE 802.11 Preconsiderations

What follows is a lengthy explanation of which parameters were modified in the NIC drivers and which IEEE 802.11 parameters had to be considered in our experiments. Much of the information is theoretical and is used as background for the discussion of the results later in the paper. Those familiar with IEEE 802.11 or wireless communication theory in general can skim this section.

2.3.1 UDP vs. TCP

Transmission Control Protocol (TCP) provides a reliable byte-stream transfer service between two endpoints on an Internet. TCP depends on IP to move packets around the network on its behalf. IP is inherently unreliable, so TCP protects against data loss, data corruption, packet reordering and data duplication by adding checksums and sequence numbers to transmitted data and, on the receiving side, sending back packets that acknowledge the receipt of data.

Before sending data across the network, TCP establishes a connection with the destination via an exchange of management packets. The connection is destroyed, again via an exchange of management packets, when the application that was using TCP

---

[1] LabView is a graphical environment used to program the NI oscilloscopes

[2] BridgeVIEW is the version of LabView used for industrial automation

[3] LabWindows is graphical ANSI-C programming environment for the NI 5102

indicates that no more data will be transferred. This scheme provides timely, reliable transfer of data between two points on a network.

User Datagram Protocol (UDP) provides an unreliable packetized data transfer service between endpoints on an Internet. UDP depends on IP to move packets around the network on its behalf as well. However, UDP does not actually guarantee delivery of the data to the destination, as does TCP. It does not guarantee that data packets will be delivered to the destination in the order in which they were sent by the source, nor does it guarantee that only one copy of the data will be delivered to the destination. UDP does guarantee data integrity, and it does this by adding a checksum to the data before transmission.

TCP is a timer-oriented protocol, waiting for acknowledgements from the receiver before sending the next packet, and retransmitting when the wait period for acknowledgements times out or an acknowledgement is received but corrupted. UDP is a send-and-forget protocol. Packets are sent and acknowledgements received but reception of an acknowledgement is not a prerequisite for sending the following packet.

In this experiment, UDP was chosen as the transfer protocol. The maximum throughput was desired and UDP was the obvious choice for this. Its ability to send large amounts of data without regard for the reception of each packet flooded the receiver with data, allowing for very little idle time and the highest possible throughput. Wireless transmissions are unreliable and packet loss or corruption is common. TCP would require an acknowledgement for each packet sent and frequent time-outs would leave too much idle time on the receiver side to get viable throughput and packet error rate readings.

2.3.2 Ad hoc vs. Infrastructure

Wireless technologies have two network modes under IEEE 802.11: ad hoc and infrastructure. Both modes require the use of a wireless NIC for access to the network and use standard protocols (IP, UDP, TCP, etc.), but it is the network configuration that changes from one to the other.

In infrastructure mode, computers communicate wirelessly with one another through a central base station and typically have access to resources outside of the network. The base station consists of a radio receiver, a wired NIC, and bridging software between the two. It is the link in the wireless network for any communication between two peers or outside to the network through wired Ethernet. All data in the wireless network must pass through the base station to either another wireless peer or the wired network.

In ad hoc mode the base station does not exist; all communication is done peer-to-peer without a central access point. Computers communicate directly with one another. Typically, this setup is used for networks that will not require access to machines outside of the ad hoc network, where access to the wired network is restricted, or where a temporary network is to be established.

For the purposes of this experiment, ad hoc mode was chosen. It provides the cleanest environment for testing and is easily controlled from the sender/receiver. External interference from the wired network and base station are nullified, and readings can be made (throughput and packet error rate) that will reflect only the peer-to-peer communication instead of a two-way communication (e.g., node1 – AP – node2). A bottleneck at the base station doesn't affect the peer-to-peer throughput and errors made by the base station do not affect the packet error rate. Therefore, the readings will be representative of the communication from one wireless device to another without external network factors playing a part.

In IEEE 802.3 (wired Ethernet Standard), the maximum transferable unit is limited to 1518 bytes. The base station must conform to this standard as it is the bridge to the wired network and will drop any packets in the wireless network greater than 1518 bytes if it is directed to the wired interface of the Access Point. In ad hoc mode, packets of sizes up to 2312 bytes are allowed. The 1518 byte threshold does not apply to the ad hoc network because all of the communication is peer-to-peer and never passes through the base station or a wired network. It is a purely wireless environment. This allows a broader range of packet sizes to test and thus makes the results more definitive. Thus, the MTU of each NIC was set to 2400 bytes (the maximum allowable by the NIC) in order to take advantage of this.

2.3.3 Acknowledgements

In IEEE 802.11, an acknowledgement is sent from the receiver back to the sender upon packet reception and verification. Acknowledgement is a function that is built into the standard's error handling in the IEEE 802.11 MAC protocol and cannot be overridden by the PCMCIA NIC driver. The standard acknowledgement packet is 14 bytes in length plus a fix 96•s overhead for the physical header.

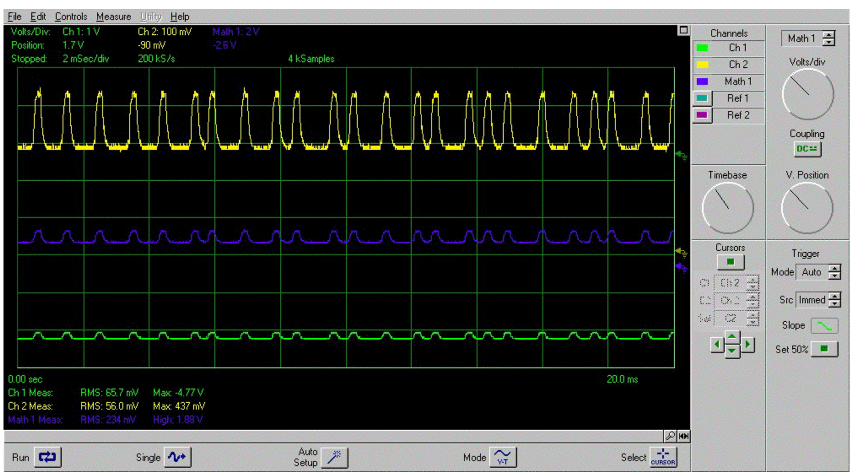


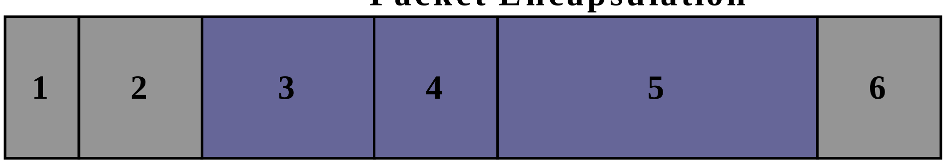
Figure 2: Reception at 11Mb/s of 64 byte packets

In the above oscilloscope trace, one can see the power spikes caused by the receiver sending an acknowledgement following reception of a packet. Reception occurs during the flat, low areas of the trace and consumes very little power with respect to sending. Reception phases are hardly different to idle phases; therefore they can not be seen in this plot. The acknowledgment accounts for a considerable amount of consumed power. This function becomes quite parasitic, in particular for small packets, when measurements of reception are being made and proper steps must be taken during measurement and analysis to nullify its effects as much as possible. This is explained in the "Calculation of Instantaneous Power" section of this paper in more detail.

2.3.4 Packet Size

As specified in IEEE 802.3(Ethernet standard) and IEEE 802.11(Wireless Ethernet standard), each packet sent over an ad hoc or infrastructure network has a given encapsulation. The packet is generally separated into four parts: the physical header, the wireless MAC header, the payload, and the trailer. The physical header gives the receiver information on the source and destination physical addresses, a checksum for data integrity, and packet size. The wireless MAC header contains the receiver and destination MAC addresses, checksums, and other packet control data. The wireless payload is further broken down into three more parts: a UDP/IP header, an Ethernet header (on some systems if an Ethernet packet is simply encapsulated to form a wireless packet), and the packet data. The trailer that follows the payload contains the FCS. The UDP/IP header contains information about the source and destination IP addresses, checksums, and packet length. The Ethernet header contains information about the source and destination Ethernet address, which is the same as the physical address. The packet data is the information being sent across the connection by the sender. The figure below illustrates an example of a UDP/IP datagram and the sizes of each of the headers of a PHY packet.

Packet Encapsulation



1) Physical Header: 96ns
2) Wireless MAC Header: 30 bytes
3) IP/UDP Header: 42 bytes
4) Ethernet Header: 22 bytes
5) data: payload – 64 bytes
6) FCS: 4 bytes

Figure 3: Datagram of Packet Encapsulation

The payload is the area of the packet that is the most relevant to this project. The physical header and wireless MAC header are not contained in the payload; thus they will not be discussed. The size of the payload is what is generally referred to as the size of the packet being sent. Together, the UDP/IP and Ethernet headers are 64 bytes in length on the Aironet wireless 4800 PCMCIA card. When specifying the sizes of packets being sent, the sizes of the headers in the payload have to be taken under consideration[1]. Therefore, a 2312 byte packet contains 2312 – 64 bytes = 2248 bytes of data on the Aironet NICs. Below is an oscilloscope snap shot of a 2312 byte packet that was specified to have payload data of 2248 bytes.

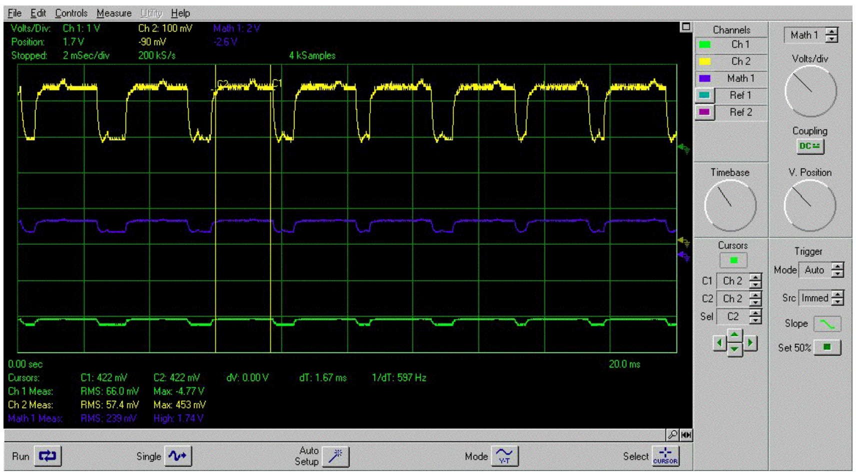


Figure 4: Transmission at 11Mb/s with a 2248 byte payload

When the IEEE 802.11 standard was drafted, they neglected to specify what to do with the Ethernet header. Some vendors encapsulate the Ethernet header in the 802.11 packet, while others strip it. It contains much of the information in the wireless MAC header, thus it is redundant. Packets whose payload data is specified as 2248 bytes could have a payload of 2312 bytes or 2334 bytes depending on whether or not the Ethernet header was stripped. After some anomalous readings were taken, it was discovered that the Aironet 4800 PCMCIA wireless NICs do not strip the payload of the Ethernet header and packets are 22 bytes larger than expected. This could result in fragmentation and skewed power consumption readings. Below is an oscilloscope snap shot of a packet of size 2313 bytes. The payload data was specified to be 2249 bytes, 1 more than the maximum allowable packet size without fragmentation occurring.

---

[1] The packet, which is given to the PC4800B itself is an Ethernet packet. We assume that the Ethernet-Header is not stripped and replaced by an wireless header. Instead we assumed, that the whole Ethernet packet is simply encapsulated.

Figure 5: Transmission at 11Mb/s with a 2249 byte payload

The smaller spikes in the top trace are the 1 byte fragments plus header information that resulted from the Ethernet header remaining in the payload data. The fragmentation of packets occurring at payload sizes 22 bytes smaller than expected results in a very large amount of overhead in measurments that could have easily gone undetected.

2.3.5 Fragmentation

The Maximum Tranferable Unit (MTU) is 1518 bytes for an Ethernet type network. Packets larger than this threshold are fragmented into smaller packets at the IP level and sent as individual packets, each smaller than the MTU. The receiver (connection endpoint) then rebuilds the packet with the fragments. 1518 bytes was the maximum MTU the Aironet PC4800 Linux driver could handle initially. A modification was made to the driver that allowed larger packet sizes.

This produces a very large amount of overhead on both the sender and receiver sides of the connection. In addition, it would change performance figures in terms of throughput and PER. Therefore the Aironet NIC driver was modified to allow the MTU to be set to values as high as 2312 bytes. In an ad hoc network this is possible due to the fact that the packets are never sent through the base station and will therefore not conflict with the MTU set in IEEE 802.3.

Fragmentation of packets can also be done on a link base in IEEE 802.11, which is handled by the MAC. This operation takes place in a transparent way, the immediate receiver of a fragment must put them together before passing them to the IP layer. This feature is used to combat bad channel conditions, since shorter packets are less likely to have bit errors. We set the MAC level fragmentation threshold to 2312 bytes, the maximum value accepted by the NIC, to avoid fragmentation.

2.3.6 CSMA/CA and RTS/CTS

For wired networks conforming to IEEE 802.3, the Carrier Sense Multiple Access and Collision Detection (CSMA/CD) protocol is used for error handling and conflict resolution when two nodes on the network attempt to transmit at the same time. Nodes wishing to send data over the network first sense the medium then listen for collisions while the transmission is in progress. This system cannot be used for 802.11 wireless networks because a station must be able to transmit and listen at the same time. Radio systems are unable to 'hear' while they are transmitting due to the fact that the transmission drowns out its ability to listen for collisions.

IEEE 802.11 uses a slightly modified error handling mechanism in order to deal with collisions and transmission errors. The Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol uses explicit acknowledgements for each packet sent over the link in order to ensure its reception. It works as follows: the station wishing to send data over the link senses the air for any activity, if none is detected, it waits an additional randomly selected amount of time and transmits if the medium is still free. If the packet is received intact, an acknowledgement is sent back to the sender, completing the process. If the acknowledgement is not received in time or is received incomplete, a collision is assumed and the packet is sent over the link after another randomly selected amount of time. Thus, CSMA/CA allows the sharing of access over the air and also effectively handles interference and other radio related problems that could disrupt transmission.

Another problem faced by wireless networks that is addressed by IEEE 802.11 is the 'hidden node' problem. Many times in a wireless network, stations are able to hear direct neighbors, but are unable to hear activity from other nodes due to interference, a physical obstruction, a large distance, or something else interfering with communication. This renders the CSMA/CA protocol useless as packets and acknowledgements are sent over the network without being able to hear all of the activity on the network. To solve this problem, Ready to Send/Clear to Send (RTS/CTS) is an optional protocol specified in 802.11 on the MAC layer. A sending station transmits an RTS packet and waits for the CTS response. Since all nodes can hear the sender and the receiver, other nodes will delay transmission attempts until the communication is over. This adds a considerable amount of overhead to the CSMA/CA protocol because the RTS/CTS packets are sent for all data packets. Therefore, there is a threshold value to be specified. RTS/CTS should be generally used for large packets that will occupy the transmission medium for extended periods of time (including collisions).

In these experiments, RTS/CTS was not needed because the ad hoc network consisted of only two stations and the extra overhead associated with the protocol would interfere with measurements. The RTS Threshold parameter, the minimum packet size with which the RTS/CTS protocol is to be used, was set to 2312 bytes in order to turn RTS/CTS off. The MTU was also set to 2312 bytes, therefore any transmitted packets larger than this value would be fragmented and RTS packets would never be sent.

3 Hardware Setup

3.1 Overview

The experimental setup consisted of two Sony VAIO Pentium II laptops each with an Aironet PC 4800 WNIC. They were both connected to a control PC by standard wired Ethernet. The wired and wireless NICs were given very distinct IP addresses on both machines and appropriate routing entries were made in order to isolate the wireless communication from the wired communication. This allowed commands to be sent to each of the laptops without interfering with the measurements taken of the wireless NICs. The third computer, the control PC, was a Pentium III 750MHz with 1GB main memory in order to be fast enough and to be able to take write the trace data from the "on-board" digital oscilloscope, which was written in real time. It ran several Perl scripts that sent commands to and from each of the laptops that facilitated data being sent across the wireless network. It was equipped with the NI5102 PCI oscilloscope card to take power measurements of the NICs as data was sent over the network.

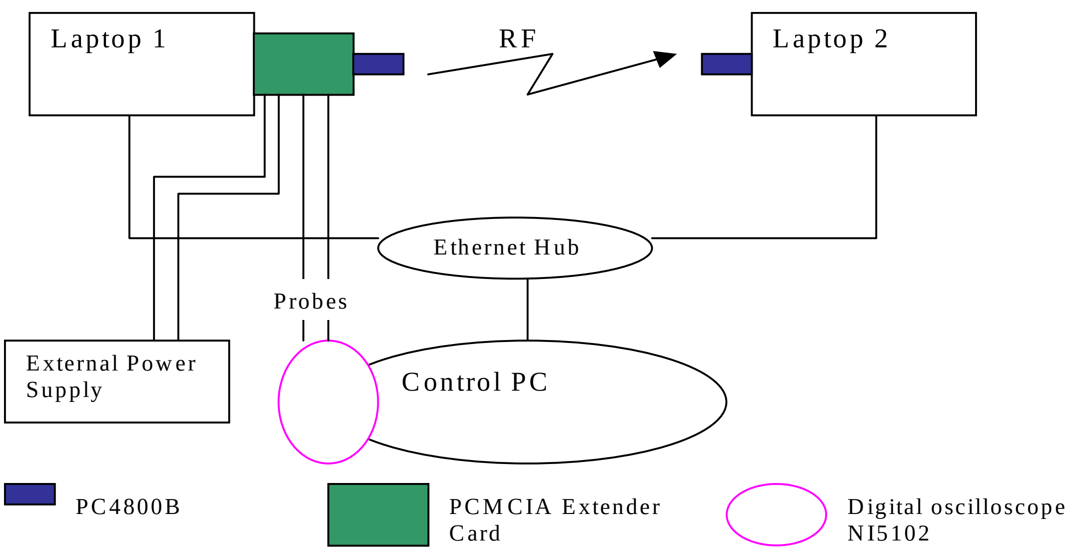


Figure 6: Measurement setup

3.2 PCMCIA Bus Extender:

One of the laptops used a Sycard Technology PCCextend 140 CardBus Extender [5] to connect to the Aironet wireless NIC. The CardBus Extender is used to extend the PCMCIA bus to the wireless NIC so that Vcc can be isolated and measurements taken of both Vcc and current consumed by the card. Measuring both these values allowed for the calculation of the instantaneous power consumption. Below is a diagram of the card.

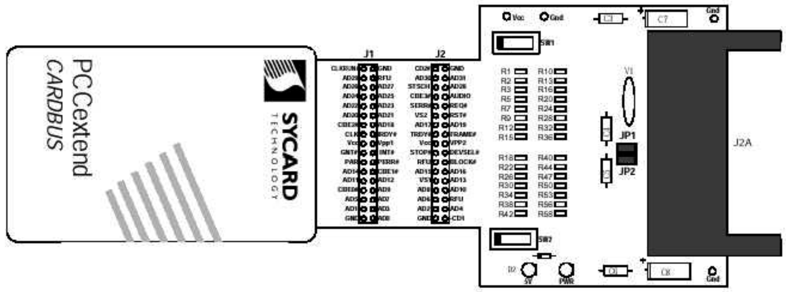


Figure 7: PCCExtend 140 CardBus Extender card

On the right is the PCMCIA socket for the wireless NIC, to the left of that are two jumpers used to isolate Vcc. By removing both jumpers, Vcc is broken and no current is supplied to the card. Both jumpers need to be removed in order to break the circuit because they are in parallel and removing only one leaves the circuit intact. Thus, both jumpers were removed during these experiments to allow for an external power supply as explained later.

The instantaneous power dissipated across two points in a circuit is the voltage between those points multiplied by the current between those points. Therefore, in order to measure the current drawn by the card, measurments of both the voltage and the current had to be taken. The voltage was measured as Vcc to ground throughout the experiment, and the value used for current was the value across Vcc. The current was measured by placing a resistor across one of the jumpers in the diagram above. The voltage drop across that resistor was then measured and divided by the value of the resistor (1.07Ω) to yield the current. Both the voltage and current were measured simultaneously to yield the instantaneous power consumed by the card. We used a resistor of 1.07Ω because it was small enough to have only a little influence on the PCMCIA card but sufficiently large to get accurate results with our measurement equipment.

3.3 Voltage Concerns

At an average operating voltage of 5 Volts and a maximum current of .5 Amps drawn across Vcc, the drop across the resistor is over .5 Volts. This corresponds to Vcc having a value of less than 4.5 Volts on the low voltage end of the resistor. The PCMCIA card has a minimum operating voltage of 4.75 Volts. With the resistor in place, the card receives insufficient voltage from the PCMCIA bus to power itself. Additionally, the power supply of the laptop varies to a certain extend, which would make the measurements inaccurate. Thus, Vcc had to be picked up by an external power source on the high voltage end of the resistor to compensate for the drop[1]. Below is a diagram of the

[1] This setup comes from Kunlun Ma and Sebastian Althen (Students of Technical University Berlin) for preliminary work of this project

PCMCIA bus, the wireless NIC, and the wiring to pick the voltage up to the appropriate level.

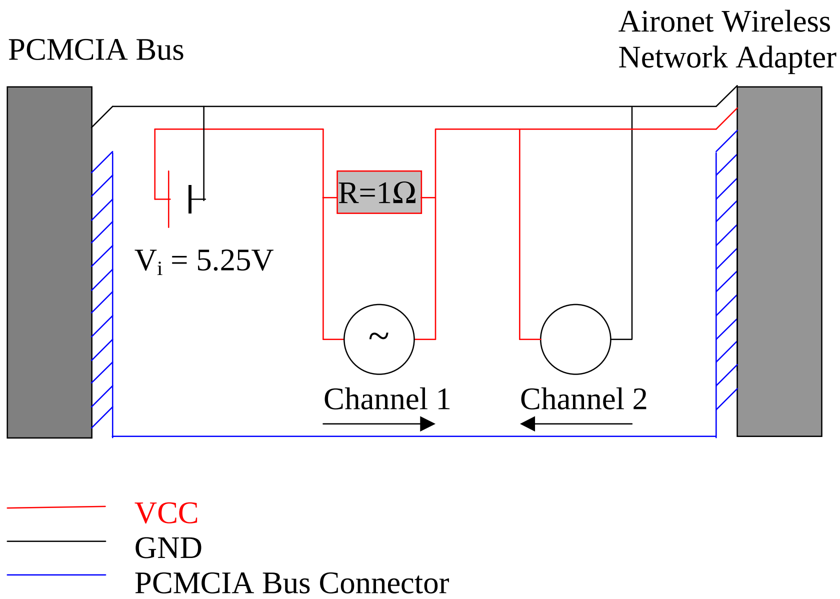


Figure 8: Hardware diagram of voltage pick-up and oscilloscope probes

The resistor is placed in series with Vcc as was stated before. 5.25 Volts was supplied by the external power source from the card's ground to Vcc on the high voltage end of the resistor. The value 5.25V was chosen because the drop would be at a maximum of < 0.5 Volts when the card draws the maximum current of 0.5 Amps. Thus, the low voltage end of the resistor would be > 4.75Volts even at maximum current draw which is above the minimum voltage necessary to power the PCMCIA NIC.

3.4 NI5102 Oscilloscope

The oscilloscope used to measure the voltage and current across Vcc was the NI 5102 PCI oscilloscope card from National Instruments [7]. It has a maximum sampling rate of $20x10^6$ samples/second and an 8 bit resolution. A sampling rate of only $1x10^6$ samples/second was used because 1 second of data translated into a 10 Mbyte data file. Due to the large volume of traces taken, it was not feasible to take longer traces or at larger sampling rates. The amount of data would have become unmanagable. The sampling rate chosen was high enough to get accurate values of the voltage and current from the traces without sacrificing managability of the trace file produced.

In the above diagram, both channels of the oscilloscope are shown measuring Vcc. Channel 1 is the voltage drop across the resistor in series with Vcc, and Channel 2 is the value of Vcc from ground. The polarity of the channels are opposite one another to have a common measurement reference point.

3.5 Remote Control

In order to make the large number of trials during this experiment, the Windows PC was used as a remote control PC for the two laptops. Each of the laptops had in addition to the wireless NIC, a standard IEEE 802.3 NIC with different IP addresses. Commands were sent over standard Ethernet to both laptops from the PC. A Perl script was written that sent commands to the laptops to modify NIC parameters, to send data over the ad hoc network, to start and stop packet traces on the ad hoc network, and to start oscilloscope measurements on the PC.

4 Software Setup

4.1 Overview

As stated before, there were two machines running SuSE Linux kernel version 2.2.13 and a PC running Windows 98. Both of the Linux machines had installed the traffic generation tool, Netperf, and one laptop had the packet trace program, Snuffle, installed. Netperf was used to send UDP streams of data across the wireless network, while Snuffle recorded the traffic. The PC was able to communicate with the Linux machines using a Linux emulator, Cygwin, with the ssh package installed. The ssh package allowed commands to be sent remotely from the PC to the Linux machines and centralized all operations to one PC. Perl 5.6.0 was also installed in Cygwin on the PC in order to run scripts that automated the experiments. Perl scripts were written that modified NIC parameters on the laptops, initialized the Snuffle trace program and traffic generation, and ran the oscilloscope on the PC. Operations were timed so that that parameter was changed on the sending/receiving laptop, the Snuffle tracer was then started, Netperf began generating traffic, the oscilloscope took a one second trace during traffic generation, and lastly, the Snuffle tracer was stopped after Netperf was through sending data.

4.2 Cygwin

One of the key components of the experimental setup was the ability to remotely control the Linux laptops from the PC. It greatly simplified data gathering and decreased the level of human error in the experiment. It was a formidable task, however, to implement since the ability to communicate between Linux running on the laptops and Windows 98 used on the control PC would have been difficult to implement (at least in the way we needed it). It was decided to use a uniform Unix environment, even if the control PC runs Windows 98, to facilitate an easy communication between the laptops and the control PC. For that purpose we used Cygwin and some extensions on the control PC.

Cygwin is a UNIX-compatibility library that can be used to port UNIX software to the Win32 operating system. It emulates a UNIX terminal on the Windows desktop and supports a large subset of the UNIX system commands. Additional packages are available from 3rd party developers for commands that aren't already supported. The Perl 5.6.0 [8] and openSSH packages were downloaded to add support for the Perl scripting language and a simple way of command exchange (ssh).

Even though Cygwin is a UNIX shell for the Windows desktop, it will run WIN32 applications (non-GUI) from the BASH prompt. This proved useful in the automation scripts. It was necessary to be able to send commands to either laptop through ssh, a UNIX command, then to be able to initialize the oscilloscope to take a trace, a WIN32 application. Without Cygwin's ability to run applications from both environments, the automation scheme would not have worked.

4.3 openSSH

In order to communicate between the machines in the experimental setup, ssh was utilized. SSH [9] is a program for securely logging into a remote Linux machine and executing commands on that remote machine. It uses 4 methods of authentication to verify the identity of the remote user, but not simultaneously. Each method has different levels of security and lower security authentication is often attempted if the higher security connection fails. In this experiment, RSA based authentication was utilized to verify the identity of the control PC when it attempted to connect to either of the laptops. This method relies on public and private keys on both the remote and host machines and can be configured to work without password authentication to log in as a given user. This allowed single commands to be sent to both laptops without disruption by a password prompt.

Commands were sent to both laptops through the wired Ethernet in order not to interfere with the wireless setup. The wireless NICs and the wired Ethernet had separate IP's. Thus, it was possible to control each laptop through the Ethernet IP and send data across the wireless network separately.

4.4 Automated Testing Scripts

Testing scripts were written in Perl to automate the experiments. The primary script was run on the Windows 98 machine and sent commands through ssh to both Linux laptops. The script initialized the WNIC's modifiable parameters to those values discussed in the pre-considerations. Each of the experimental variables (data rate, packet size, and transmit power) was changed one at a time between tests. A test consisted of three bursts of data being sent over the medium while recording the Snuffle and power consumption data. Snuffle and the oscilloscope were started by the script before each of the bursts of data and recorded information for the length of the burst. Each of these bursts was separate and gave us three values of throughput, PER, and power consumption for a given data rate, packet size, and transmit power. The oscilloscope trace files and snuffle data files were saved to disk and named according to data rate, packet size, transmit power, and test number. The data from these experiments was archived and is available at the TU-Berlin.

5 Experimental Procedures

5.1 Distance Considerations

Distance between network nodes was the fourth parameter modified in these experiments (besides data rate, packet size, and transmit power). The tests were run at distances of 5 meters, 10 meters, and 15 meters between nodes. Each of the experiments was run in completion at each of the distances. Packets were sent at each data rate, packet size, and transmission power in order to illustrate the effects of distance on the PER and throughput measurements. Additionally, the power consumption was recorded. Out of these data we were able to compute the energy that was used to transmit one “goodput” bit. Due to archiving problems with the data at 10 meters that lead to a loss of these data, only the data gathered at 5 and 15 meters can be discussed in this paper.

Below is a simple diagram of the relationship between the network nodes during the measurements. The red ‘X’ in room 1 is the transmitting machine attached to the oscilloscope, running the Snuffle tracer, and sending data over the ad hoc network with Netperf. Each of the blue ‘X’s represents the receiving machine at each of the three distances tested (5m, 10m, and 15m). Both machines were in separate rooms during the measurements. However, the doors were kept open for the control Ethernet cable through which the control PC and the laptops were connected.

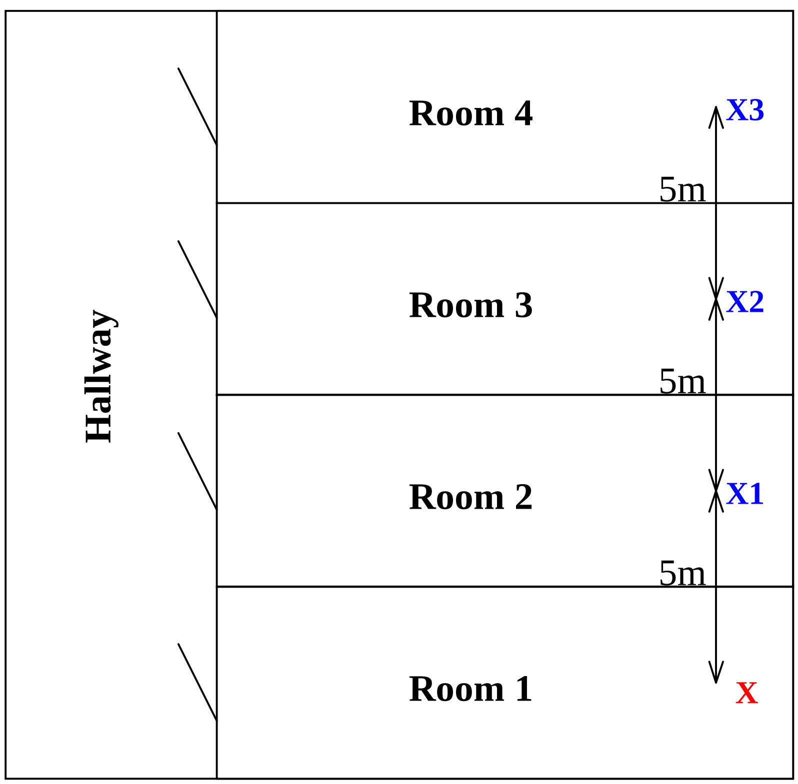


Figure 9: Experimental Setup for Distance Measurements

We also conducted measurements where both nodes were in room 1 and in close proximity. These measurements were concerned with baseline power consumption (e.g.

how much power is consumed during the send/receive/idle/sleep phases only) without consideration for distance, throughput, and PER. These data might be used to parameterize simulations later. These measurements are referred to as single shot measurements throughout the rest of the paper.

5.2 Transmitting Measurements

The transmission measurements of packet error rate, throughput and power were completed using the Perl automation scripts. The scripts were executed, at each of the distances above and during the single shot measurements as well. The scripts initialized and verified the ad hoc IP's of the machines, initialized Snuffle and Netperf on both machines, and then performed the measurements. The measurements consisted of modifying one of the network parameters on the transmitting machine (data rate, transmission power) and then sending three bursts of data over the channel at each of the packet sizes (1, 64, 192, 448, 704, 960, 1216, 1472, 1728, 1984, 2248 + 64 bytes for UDP/IP overhead). Snuffle and the oscilloscope created trace files for each packet size sent at each data rate and transmission power. All measurements were taken on the transmitting machine.

5.3 Receiving Measurements

The single shot measurements were performed exactly as the transmitting measurements, except the oscilloscope took readings on the receiving machine instead of the transmitting machine. Snuffle was not necessary in the single shot tests, because they were only concerned with the power consumption.

5.4 Calculation of Instantaneous Power Consumption

We refer to the power consumption during the respective operating modes (transmit/receive/idle/sleep) as instantaneous power consumption. The oscilloscope created trace files of power consumption that included the transmission of data packets, the idle phases, and the reception of acknowledgements in different measurement runs. A low pass filter filtered the trace files before evaluation to smooth the data because random spikes (inaccuracies of the oscilloscope and measurement approach) would have made post processing of data difficult. The filtering was done by a sliding window, averaging mechanism. The window size was set to 40 samples, which provided the best results of all values tested. The filtering was comparable to a Fourier transformation, followed by a cut off of the high frequencies, and then a back transformation with an inverse Fourier transformation. Afterwards, the transmission phase as well as the reception phase was cut out of the smoothed data by using threshold values that were determined by visual inspection of the smoothed traces. Every value that was greater than:

1.81W at 50mW RF transmit power
1.65W at 20mW RF transmit power
1.45W at 5mW RF transmit power
1.36W at 1mW RF transmit power

was counted as transmit power. We took a similar approach for cutting the reception power out of the reception power traces. All values smaller than the threshold were considered receiving power. This cut out the effect of the parasitic ACKs that consumed a considerable amount of power. The receiving thresholds are below, receiving power was effected by the data rate. Thus, the threshold varies among the data rates.

1.40W at 1 and 2Mbit/s
1.45W at 5.5Mbit/s
1.49W at 11Mbit/s

All data points cut out of the traces were averaged to find the average power consumption per packet transmitted or received. This was performed for each of the three measurement traces at each packet size. The results from the three traces were again averaged to find the final value of power consumption.

We measured the idle and sleep mode power consumption similarly. For idle mode power consumption we took traces where no packets were sent. However, for sleep mode measurements we had to change the operation of the wireless network from ad hoc to infrastructure using an Aironet Access Point, since power saving was not supported in ad hoc mode.

5.5 Calculation of Throughput

The throughput was computed using the data from Snuffle trace files. For each of the trace files, all of the packets of a fixed length successfully transmitted over the network were summarized. Thus, the total payload data transmitted was computed and converted to bits (1 byte = 8 bits). This value was then divided by the difference between the first time stamp and the last time stamp recorded in the trace. This yielded values of bits per second for the throughput. Three throughput values were calculated out of the three trace files for every packet size. The three values were then averaged to find the final throughput value.

5.6 Calculation of Packet Error Rate

The packet error rate (PER) was computed using the data from Snuffle trace files. The Snuffle trace files stored status information of each packet transmitted over the network (i.e. successful or unsuccessful transmission). The number of unsuccessful packets and the total number of sent packets of the fixed packet size were determined. All control messages (generated by netperf at the beginning and end of the trace) were ignored. The PER is the number of unsuccessful packets transmitted divided by the total number of transmitted packets. There were three values of PER calculated for each packet size. The three values were then averaged to find the final value of PER. For traces with a bad transmission channel and a zero throughput, the PER was normalized to 1. All packets were lost during transmission.

5.7 Calculation of Energy per Goodput bit

The energy/bit was calculated using the average power consumption from a power trace file in conjunction with the corresponding Snuffle trace file. We divided the average power consumption by the throughput as calculated in section 5.5. This yielded values of J/byte, thus it was multiplied by 8 to convert the value to J/bit (8 bits = 1 byte). This was done for each packet size sent at each of the four data rates and transmission powers. For traces with a bad transmission channel and zero throughput, the energy/bit was set to 0. All packets were lost during transmission, thus the energy consumption readings for those transmissions are infinite which are presented as a 0 in the following graphs.

5.8 Inaccuracies of in the Calculations

We assume that the power consumption results for large packets and low data rates are more accurate than for small packets and high data rates. We sampled the power consumption of the NIC at 1MHz in order to keep the amount of data and time for the measurements manageable. This results in fewer samples to compute an average instantaneous receive/transmit power for short packets due to either the packet size or a high data rate. Additionally, we used a threshold method to cut the data (samples) of interest out of the trace file. A part of the slope at the start and the end of each packet is included in the power consumption calculations. These slope samples have a larger influence if there are less power samples within one packet.

6 Results and Discussion

6.1 Instantaneous Power Consumption

Measurements of the average instantaneous power consumption of the WNICs versus the packet size were taken without consideration for distance. It was assumed that the close proximity of the nodes resulted in a PER of zero at each data rate. Data was taken for each of the transmit powers on the Aironet PC4800B WNIC (1mW, 5mW, 20mW and 50mW) and for each of the data rates (1Mbit/s, 2Mbit/s, 5.5Mbit/s and 11Mbit/s). Results represent a base line measurement to determine the power consumption in different operation modes of the NIC as well as for different parameter settings.

6.1.1 Average Instantaneous Power Consumption during Transmission

The following three graphs represent the average power consumption of the NICs during the transmission phase. The average consumed power for each transmit power and data rate were computed by averaging the power consumption of all the measurements for different packet sizes. Thus, the average power consumption for a data rate and transmit power was calculated.

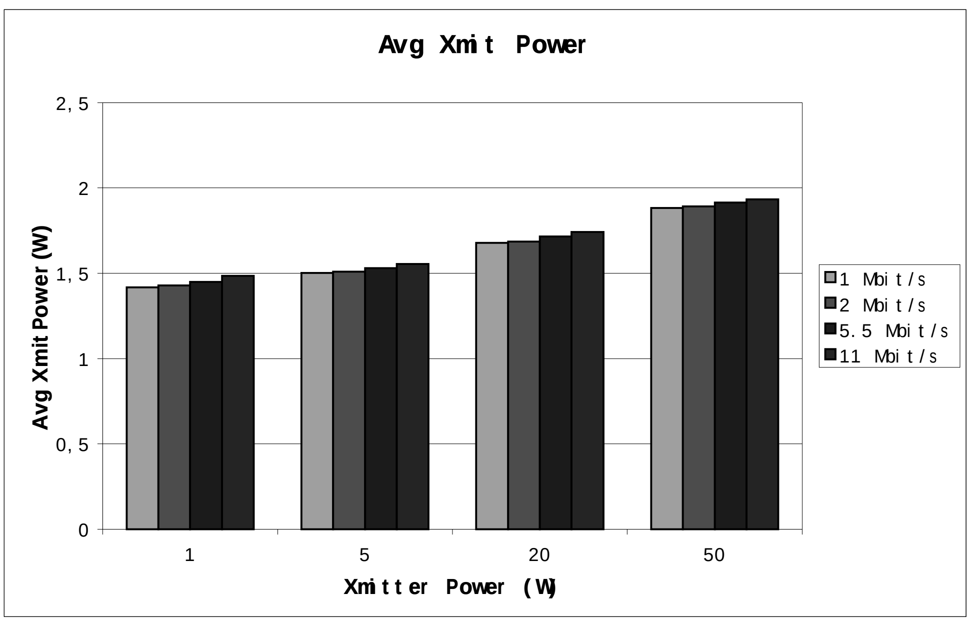


Figure 10: Average transmitting power consumed versus RF power

The graph above shows the average instantaneous transmit power for each data rate, grouped by RF transmission power. The power consumption during transmission increases with the RF transmission power and data rate. Thus on average, a packet sent at 11Mbit/s and a 50mW transmission power would consume more power than a packet sent at 1Mbit/s and 1mW (assuming a 0 PER). While increasing the data rate adds a marginal increase to the power consumption, the RF output power increases the power consumption an over-proportionate amount. Although the latter fact might be design specific to a certain extent, it shows that there is a direct dependency between the RF output power and the power consumption of the NIC. We suspect that the RF amplifier is the main reason for an increase in the consumed power during transmission when the RF output power is increased. Furthermore, we believe that the base band processor and the MAC processor cause the increase in power consumption when the data rate is increased.

6.1.2 Average Instantaneous Power Consumption during Reception

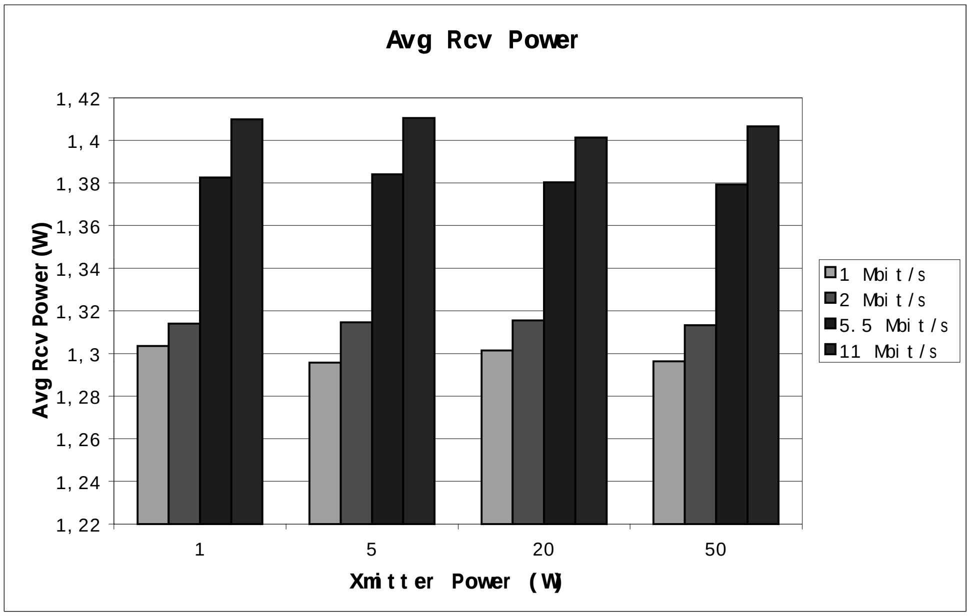


Figure 11: Average receiving power consumed versus RF power

The graph above shows the average instantaneous power during reception of a packet at each data rate, grouped by RF transmission power. Again, the average power consumed by the NIC increases with data rate, but is the same for each RF transmission power. This is because the power consumed sending the ACK and during idle phases has been cut out of the trace. The values above are independent of the transmission power because there is nothing being transmitted in the graph above. The values of receive power are less than the corresponding values for transmission. As was discussed in the pre-considerations, the RF amplifier chip consumes a great deal of power during transmission and is not operating during reception (but turns shortly “on” when the ACK has to be sent). The consumed power during reception is below the power needed during sending. The dependency on the data rates results from the fact that the base band processor and the MAC consume little more power if operating at higher data rates.

6.1.3 Average Instantaneous Power Consumption for the Idle Phase, Sleep Phase and Acknowledgement Transmission

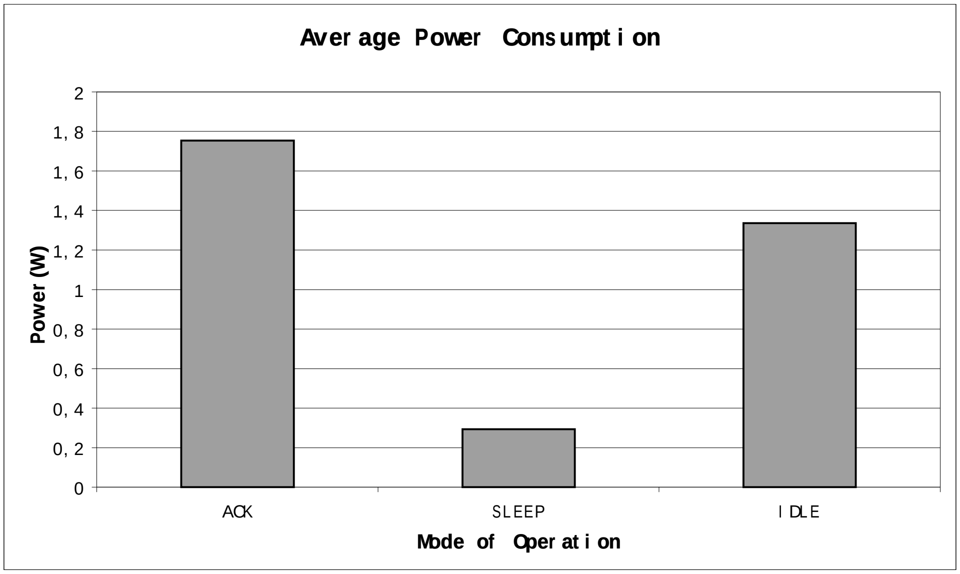


Figure 12: Modes and ACK power consumption

The graph above shows the average power consumed during sleep mode, idle mode, and for each ACK packet sent. It became obvious from the oscilloscope traces that the ACKs are sent at the highest RF transmission power regardless of the driver setting. We assume therefore that all response control messages (e.g. CTS) are sent at the highest RF power. This is most likely due to the hidden terminal problem, which causes the receiver to “shout“ control messages in order to avoid collisions and to make the responses more error resistant. Sleep mode consumes significantly less power than transmission or reception, which is consistent with the expectations set forth in the pre-considerations. This mode is for power saving purposes and a large amount of the Intersil PRISM I chipset is powered down (refer to table 1a in the preconsiderations for specifics). The power consumption in idle mode is very similar to the receive mode power consumption. This is because the chips in the Intersil PRISM I chipset used for receiving packets have to remain powered on to scan for valid signals.

6.1.4 Average Instantaneous Power Consumption Per Packet During Reception

The graphs below show the average power consumption of the WNIC per packet during reception. Data was taken for each of the transmit powers on the Aironet PC 4800 WNIC (1mW, 5mW, 20mW and 50mW); and for each of the data rates (1Mbit/s, 2Mbit/s, 5.5Mbit/s and 11Mbit/s). Again, the distance between the nodes was neglected because of their close proximity. These measurements were used as the base measurement to compute the average instantaneous power consumption during reception as shown in section 6.1.2. They are given here for completeness.

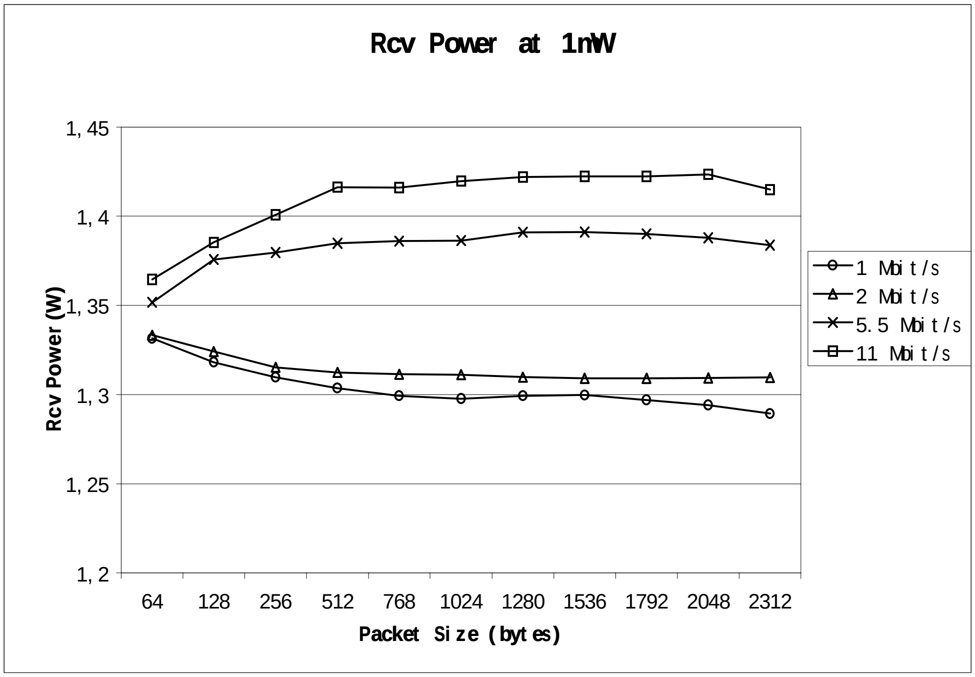


Figure 13: Receive power versus packet size at 1mW Xmit power

The graph above is the power consumed (W) by the WNIC versus packet size at each of the four data rates during reception. The RF transmission power is 1mW, the lowest value. The power consumed by the WNIC increases slightly at higher data rates and is roughly the same for packets above 512 bytes in each data rate. The reception power seems to be independent of packet size. For packets larger than 512 bytes, the curves remain on the same power consumption level. We assume there are inaccuracies for smaller packets as a result of the measurement and evaluation approach explained in section 5.8.

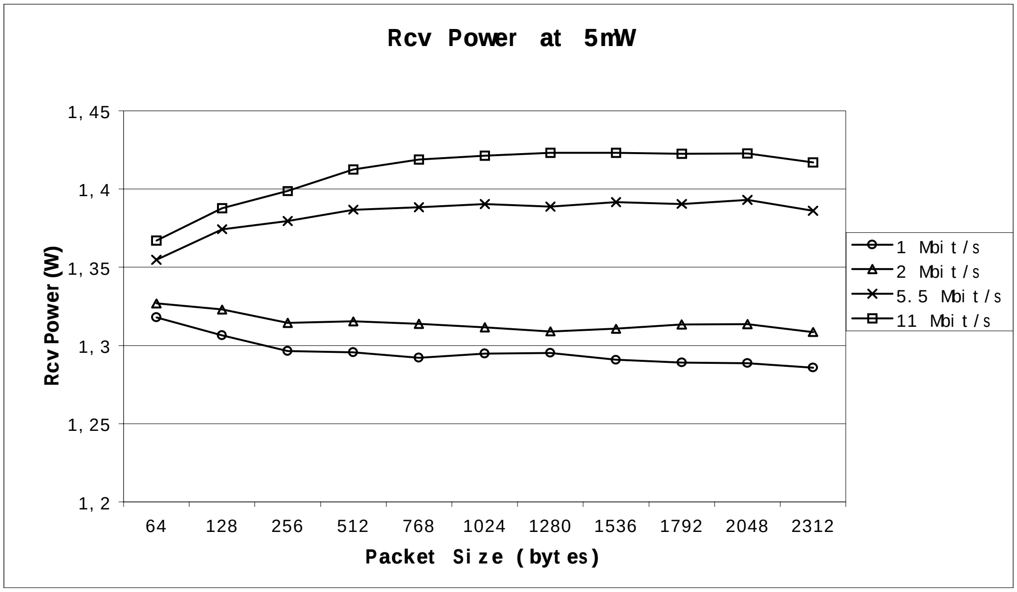


Figure 14: Receive power versus packet size at 5mW Xmit power

The graphs above and below are the power consumption of the WNIC versus packet size at each of the four data rates during reception. They are very similar to each other and the graph on the previous page even though the RF transmission powers are 5 and 20mW, respectively. This is because the power consumed by the NIC during reception is not affected by the antenna transmission power. The RF power amplifier in the Intersil chipset is not operating while the WNIC is receiving data.

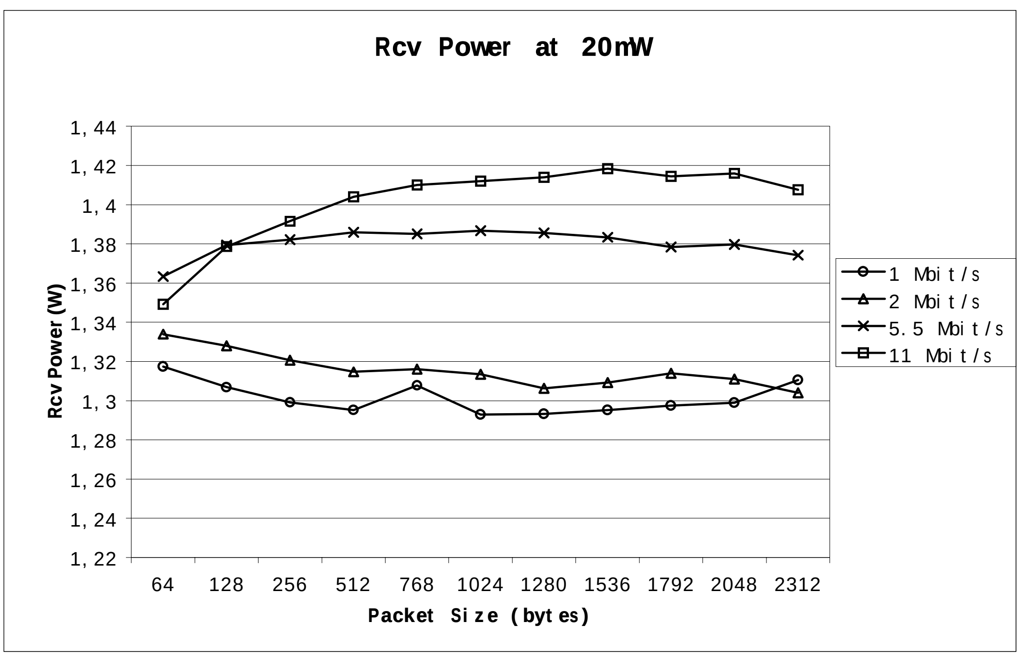


Figure 15: Receive power versus packet size at 20mW Xmit power

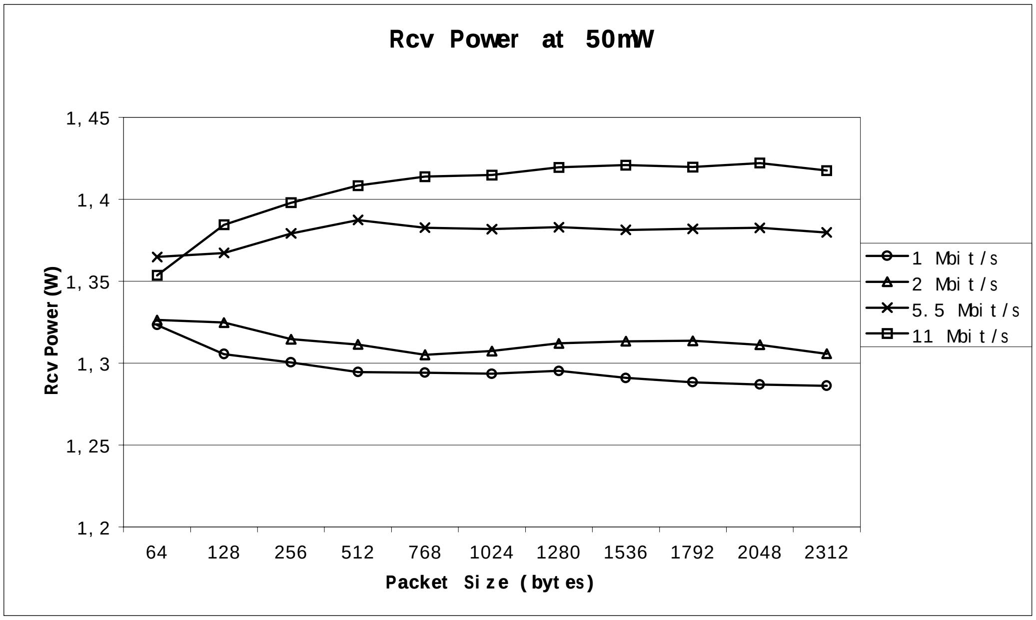


Figure 16: Receive power versus packet size at 50mW Xmit power

The graph above is of the power consumed by the NIC during reception versus packet size at each of the four data rates. Again, the graph is very similar to the previous three graphs of power consumption.

6.1.5 Average Instantaneous Power Consumption Per Packet During Transmission

The graphs below show the average power consumption of the NIC per packet during transmission. Data was taken for each of the transmit powers on the Aironet PC4800 wireless NIC (1mW, 5mW, 20mW and 50mW); and for each of the data rates, (1Mbit/s, 2Mbit/s, 5.5Mbit/s and 11Mbit/s). Again, the distances between the nodes was neglected because of their close proximity. These measurements were used as the base measurement to compute the average instantaneous power consumption during transmission as shown in section 6.1.1.

The antenna transmission power does have an affect on the results, as is apparent in the following graphs. The power consumption increases slightly with the data rate and does not vary drastically with the packet size. Packets sent at higher data rates consume more power. This is because the MAC and baseband processors consume slightly more power at higher transmission rates. The power consumption increasing with higher RF transmission powers is clearly a result of the higher power consumption of the RF amplifier at higher RF transmit powers.

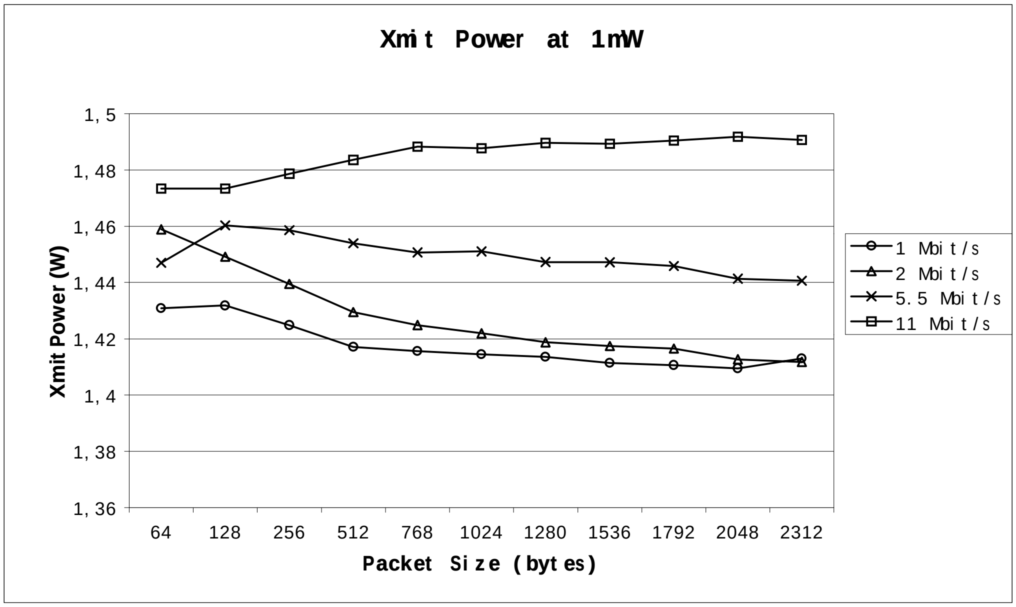


Figure 17: Transmit power versus packet size at 1mW Xmit power

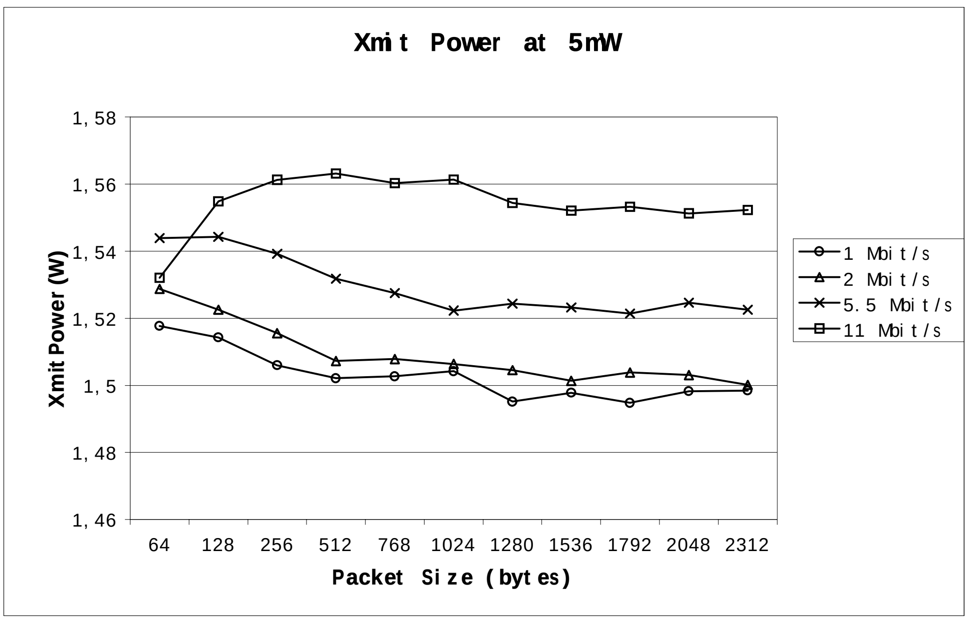


Figure 18: Transmit power versus packet size at 5mW Xmit power

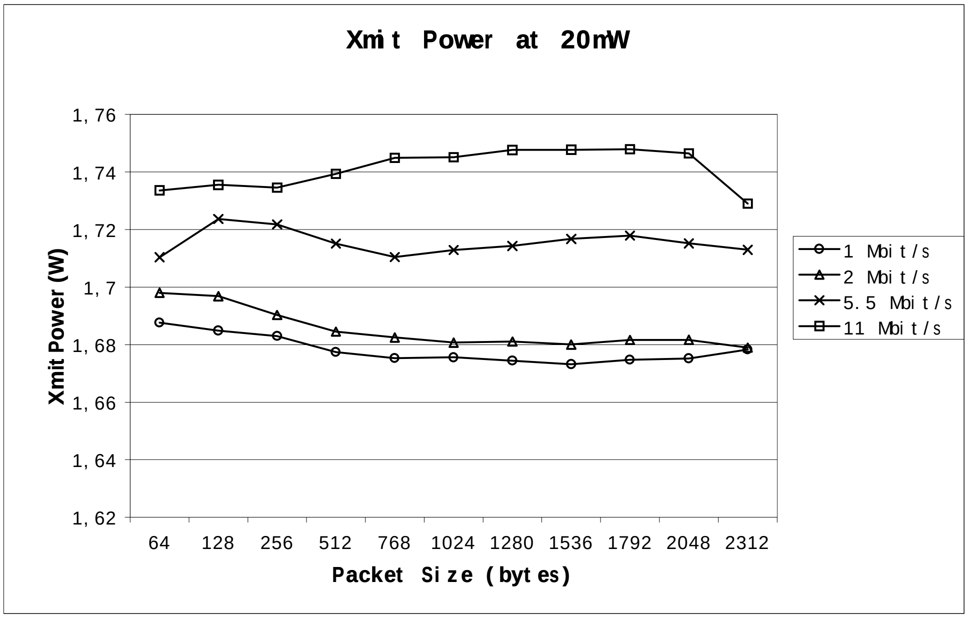


Figure 19: Transmit power versus packet size at 20mW Xmit power

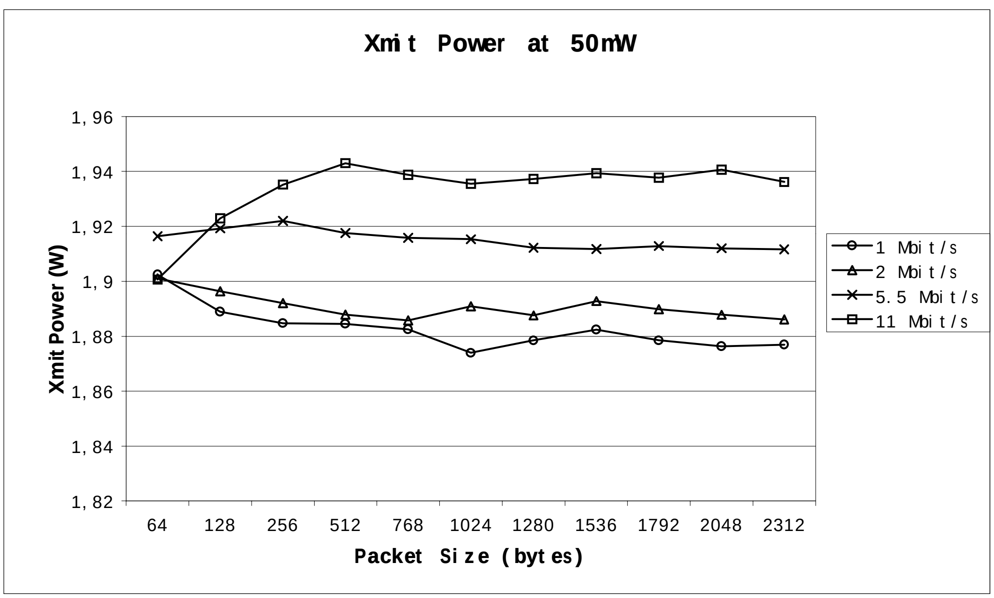


Figure 20: Transmit power versus packet size at 50mW Xmit power

6.2 Transmit Time versus Packet Size

The graph below represents the average time in seconds that a packet of the given size takes to transmit at 1Mbit/s, 2Mbit/s, 5.5Mbit/s, or 11Mbit/s. These are simple values and do not take into account the associated channel access overhead (e.g. guard times, back-off intervals, ACKs, etc.) This is purely for reference.

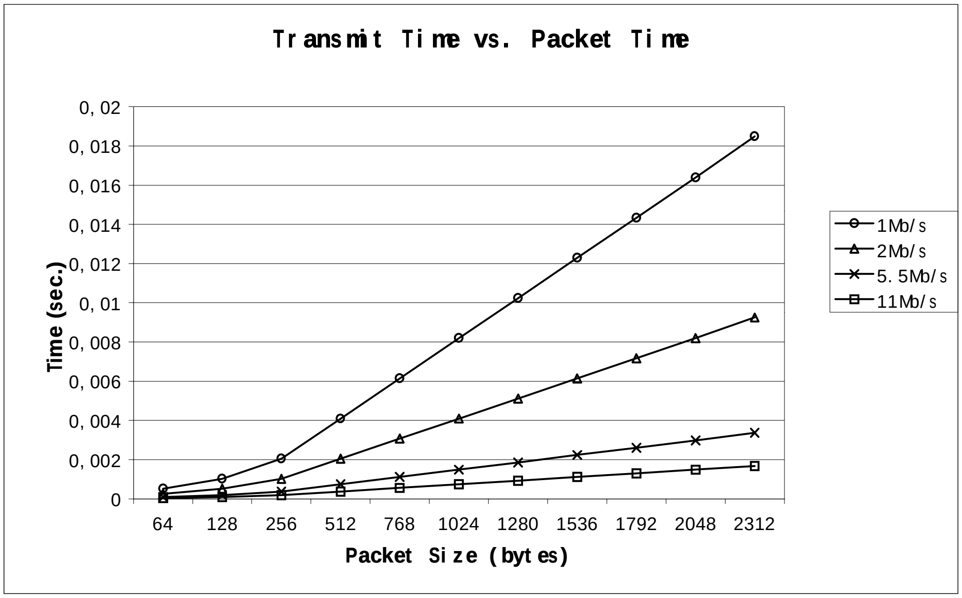


Figure 21: Time in medium versus packet size

It is obvious from the graph that it takes more time to transmit larger packets and at lower data rates. Thus, it takes much longer to transmit a 2312 byte packet at 11 Mbit/s than a 64byte packet at 11Mbit/s. In higher traffic networks, where the transmission channel is used by more than two transmitting nodes, the efficiency of the network goes down causing higher energy consumption particulary for large packets. The channel is occupied for longer amounts of time by larger packets. For good channel conditions and low collision rates, higher data rates should result in more energy efficient transmission and reception because packets occupy the channel for less time. This might reverse the effects of a bad radio channel (longer distances), since higher data rates are more error prone and can cause more retransmissions. This would result in higher energy consumption per packet and a decrease in efficiency.

6.3 Throughput versus Packet Size

Measurements of the throughput were taken at two distances[1] between terminals: 5 meters and 15 meters. The packet size, transmit power and data rate were modified as in the power consumption measurements above, but the distance was taken under consideration during these tests. It was expected that as the distance between nodes increased, so would the PER. This is due to decreased signal strength at farther distances, and results in lost packets. There was no retransmission of lost packets in these

[1] Measurements were actually taken at three distances (5m, 10m, and 15m) but as said before the 10m data archieve was corrupted and had to be thrown out.

experiments. Lost packets decrease the throughput. Thus, the throughput is expected to decrease at increased distances.

6.3.1 Transmission Throughput at 5 meters.

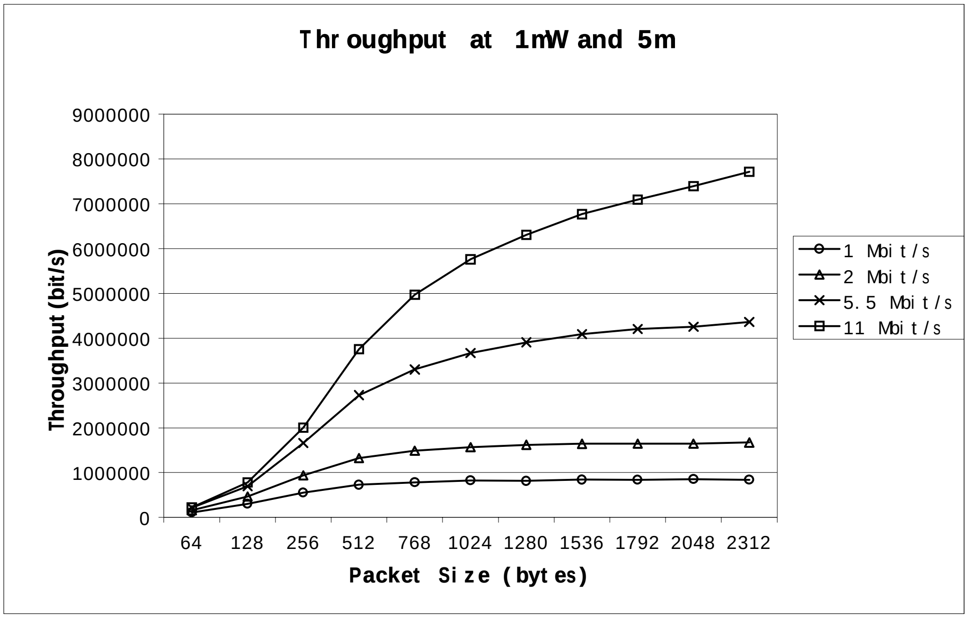


Figure 22: Throughput versus packet size at 1mW RF power and 5m distance

The graph above is of the transmission throughput versus packet size for each of the four data rates. The RF transmission power is 1mW and the distance between nodes is 5m. The throughput increases with the packet size and the data rate in our two-node network. Thus, the throughput is highest for 2312 byte packets transmitted at 11Mbit/s and is lowest for 64 byte packets transmitted at 1Mbit/s. It is expected that higher data rates will have a higher throughput at close distances because a higher rate of transfer will yield a higher throughput. There is very little signal degradation between nodes at 5m; thus there are no lost packets (or, at least so few that they do not affect the throughput by a large amount). Our results are consistent with expectations from data rate to data rate.

Higher packet sizes yield a higher throughput. This is because of the smaller protocol overhead ratio for large packets. The results for higher RF powers in the following graphs are very similar to the 1mW RF power graph above. This is consistant with our expectation, since at 5m there is very little signal degradation to cause packet losses.

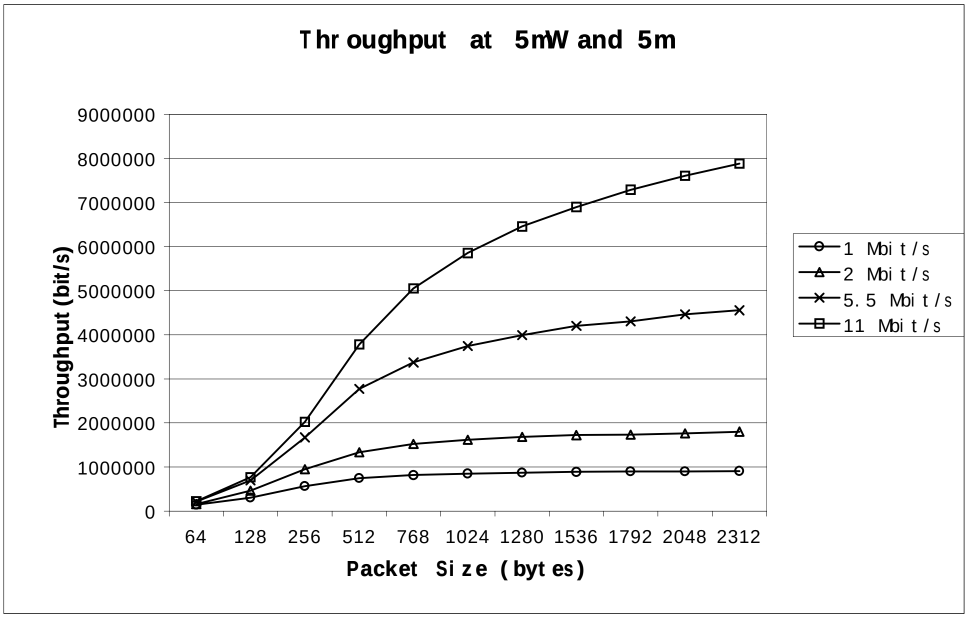


Figure 23: Throughput versus packet size at 5mW RF power and 5m distance

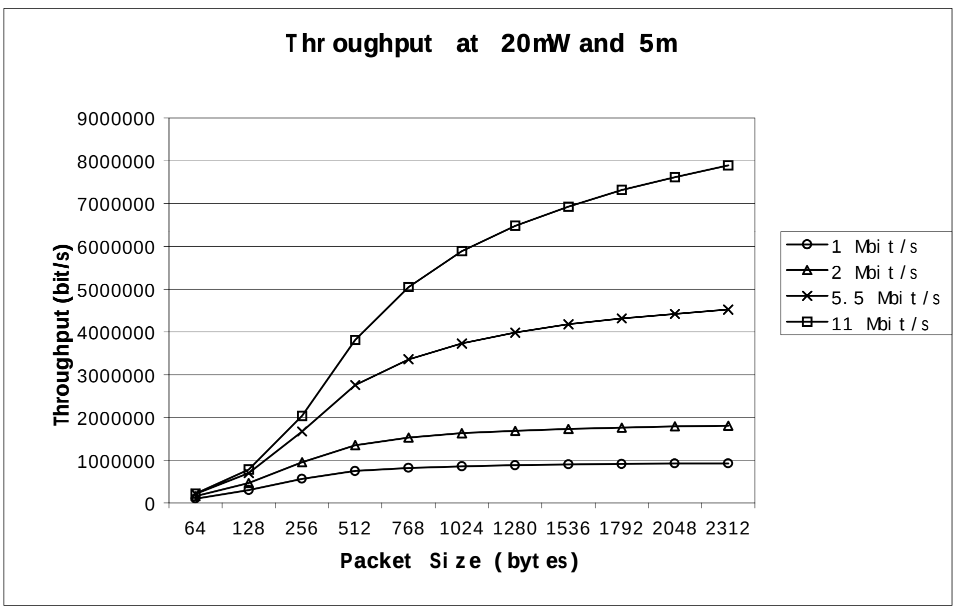


Figure 24: Throughput versus packet size at 20mW RF power and 5m distance

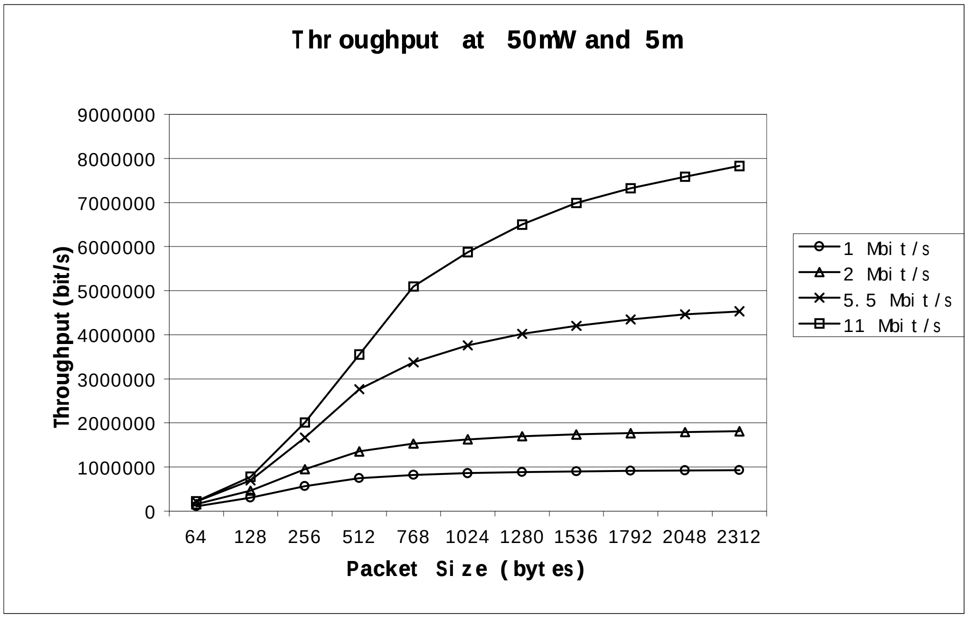


Figure 25: Throughput versus packet size at 50mW RF power and 5m distance

6.3.2 Transmission Throughput at 15 meters

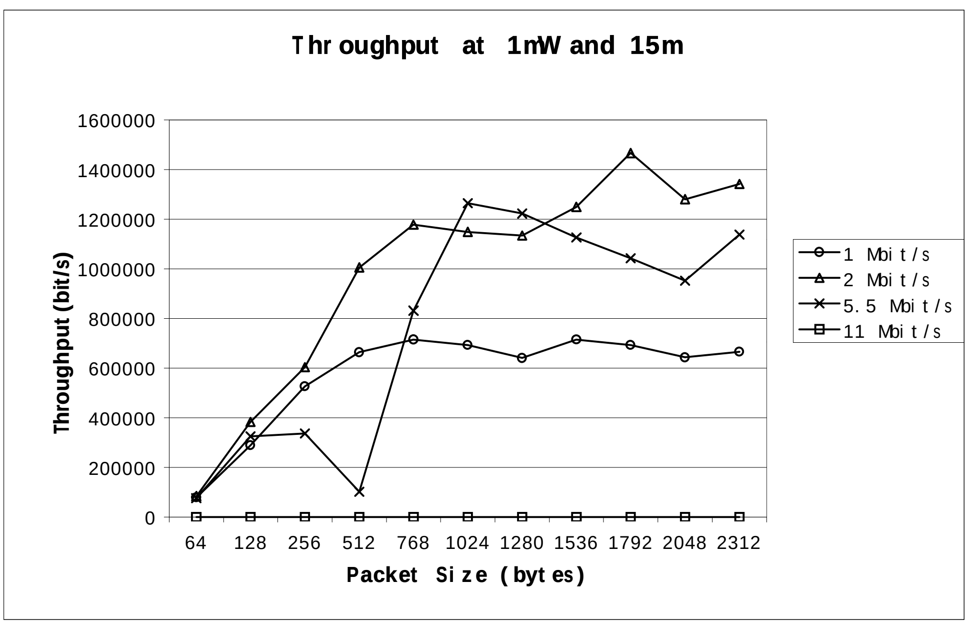


Figure 26: Throughput versus packet size at 1mW Xmit power and 15m

The graphs above and below are of the transmission throughput versus packet size at each of the four data rates. The distance between nodes is 15m, and the antenna transmission powers are 1mW and 5mW, respectively. The graph above does not contain any information for 11 Mbit/s because the data contained too many discontinuities to be useful in the plot. The throughput was zero for many of the packet sizes and non-zero for very few. The plots above still show higher throughput at the higher data rates and packet sizes; however, there are a number of discontinuities in the plots when compared to the 5m tests. The 5.5 Mbit/s plot is below the 2 Mbit/s plot for many of the packet sizes when it should be much greater, and the values of many of the data points are much lower than the 5m, 1mW cases. The variances and discontinuities at the higher data rates and the fact that the 11 Mbit/s data points were zero for many of the tests suggest that the throughput is affected by increased distance between nodes resulting in a high bit error rate.

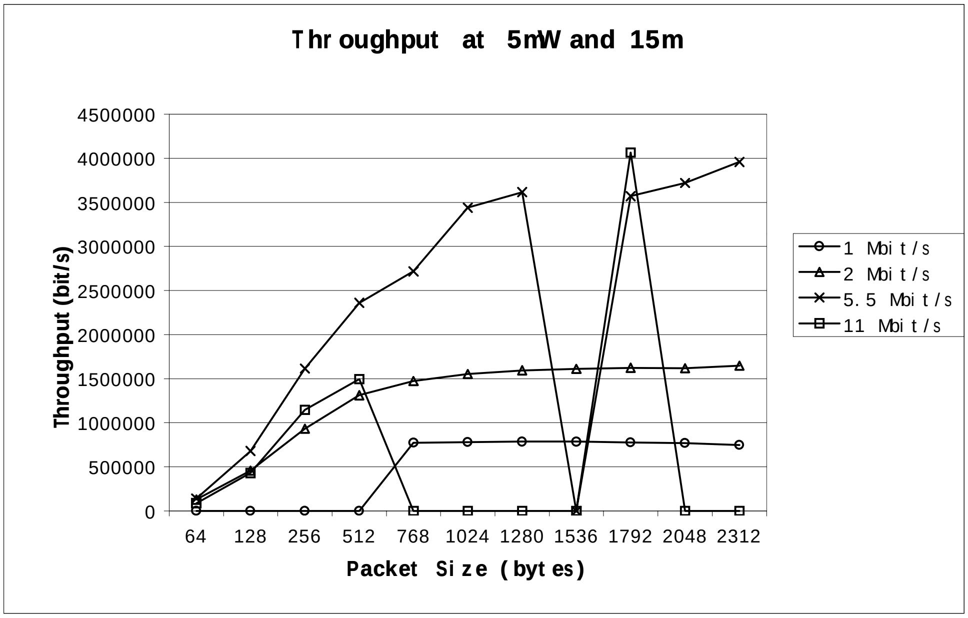


Figure 27: Throughput versus packet size at 5mW Xmit power and 15m

The throughput plot at 15m with a 5mW antenna power shows much of the same behavior as the previous plot. There are still a number of discontinuities at each of the data rates and many of the data points are lower than their counterparts in the 5m, 5mW graph. The 11Mbit/s plot still contains many zero values, as does the 5.5 Mbit/s plot. This suggests the signal degradation caused by the larger distance between nodes is balanced by the higher antenna transmission power. However, there are still a number of lost packets and synchronization problems in the transmission channel.

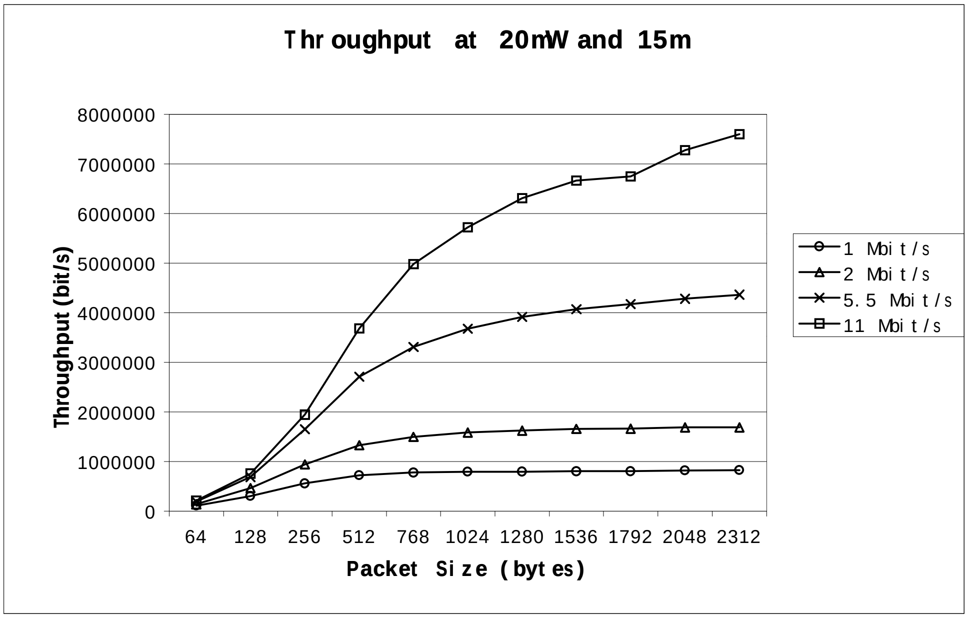


Figure 28: Throughput versus packet size at 20mW Xmit power and 15m

The graphs above and below are of the transmission throughput versus packet size at each of the four data rates. The distance between nodes is still 15m; however, the antenna transmission powers have increased to 20mW and 50mW, respectively.

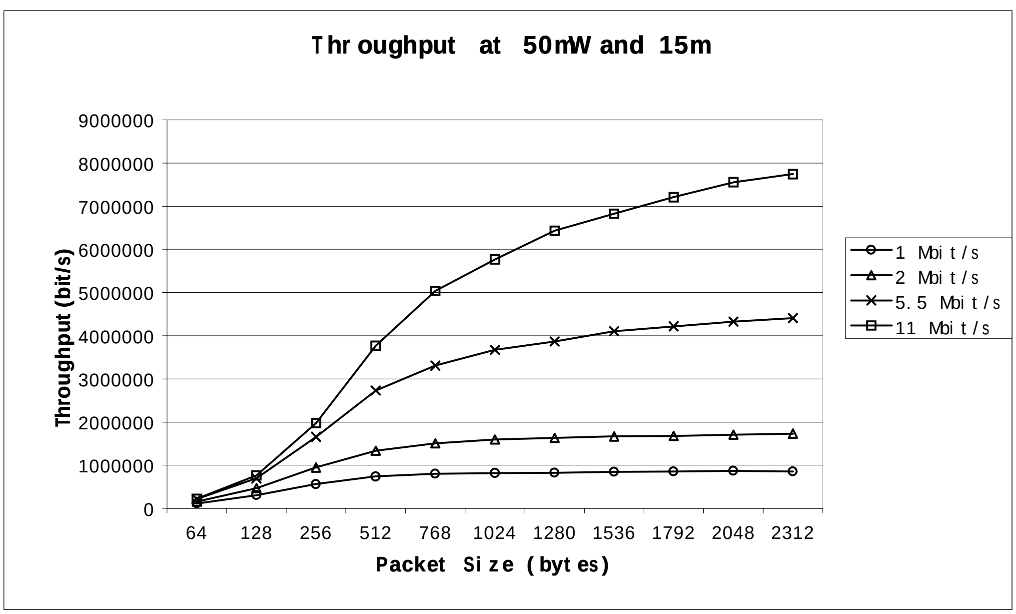


Figure 6.3h: Throughput versus packet size at 50mW Xmit power and 15m

The 20mW and 50mW throughput graphs at 15m are very similar to the 5m graphs. The higher data rates and larger packet sizes have the highest throughput and there are no discontinuities as there are in the 5mW and 1mW plots. This suggests that the higher antenna transmission power balances the effects of the increased distance on the transmission channel. As the increased distance would normally degrade the channel, the higher transmission power increases the channel strength and quality.

The values of the throughput in the 5m plots and in the higher power 15m plots are almost identical. One would expect the increased distance to increase the PER and decrease the throughput. However, it can be seen from these results that the effects of the increased PER (see later sections) due to bit errors are small if any.

6.4 PER versus packet size

Measurements of the Packet Error Rate (PER) versus the packet size were taken at 5 meters and 15 meters[1]. Data was taken for each of the transmit powers on the Aironet PC4800 wireless NIC (1mW, 5mW, 20mW and 50mW) and for each of the data rates (1Mbit/s, 2Mbit/s, 5.5Mbit/s, and 11Mbit/s). The distance between nodes was taken under consideration as it was for the throughput, because it was assumed that the increased distance would degrade the transmission channel.

6.4.1 Transmission PER at 5 meters

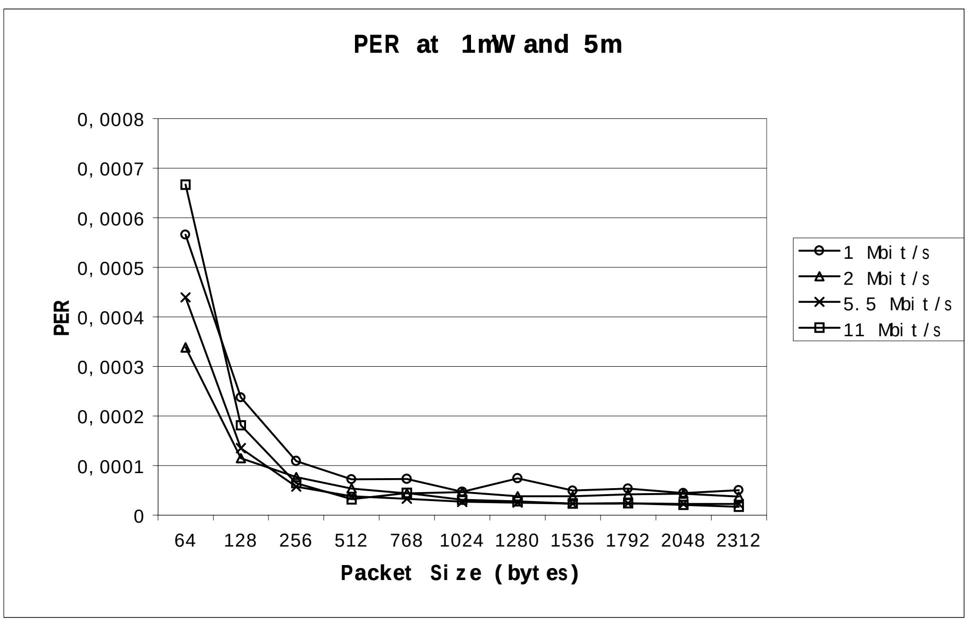


Figure 29: PER versus packet size at 1mW Xmit power and 5m

[1] Again, there as data taken at 5m, 10m, and 15m between nodes, but the 10m data was corrupted and had to be thrown out.

The graph above is of the PER versus packet size for each of the four data rates. The antenna transmission power is 1mW and the distance between nodes is 5m. The graph shows the expected low PERs at this distance between nodes; however, the PER of smaller packets is considerably higher than the PER of larger packets. This seems contrary to what one would expect under normal circumstances in a wireless environment. The packet error rate should be proportional to the packet size. Larger packets are typically corrupted more often during transmission because of the higher number of bits and the statistical probability of a bit being corrupted is higher.

Several factors might play a role in this statistical anomaly. First, the traces for short packet sizes contained more than three times the number of samples than very long packets because of the fixed run length. The number of samples in each trace was below 15000, which might be too low to be statistically relevant. Also, we do not know the details of the implementation of the NIC, which might be influenced by the packet size (e.g., the shorter the packet, the more often the card has to switch between the idle/receive/send mode). The curves' shapes may indicate that packet synchronization is an issue, which has to be done more often for shorter packets. We still assume that the PER will be higher for large packets in a very long measurement, where the number of samples for short and long packets are comparable. Unfortunately, we could not prove this because of time and hard disk capacity limitations.

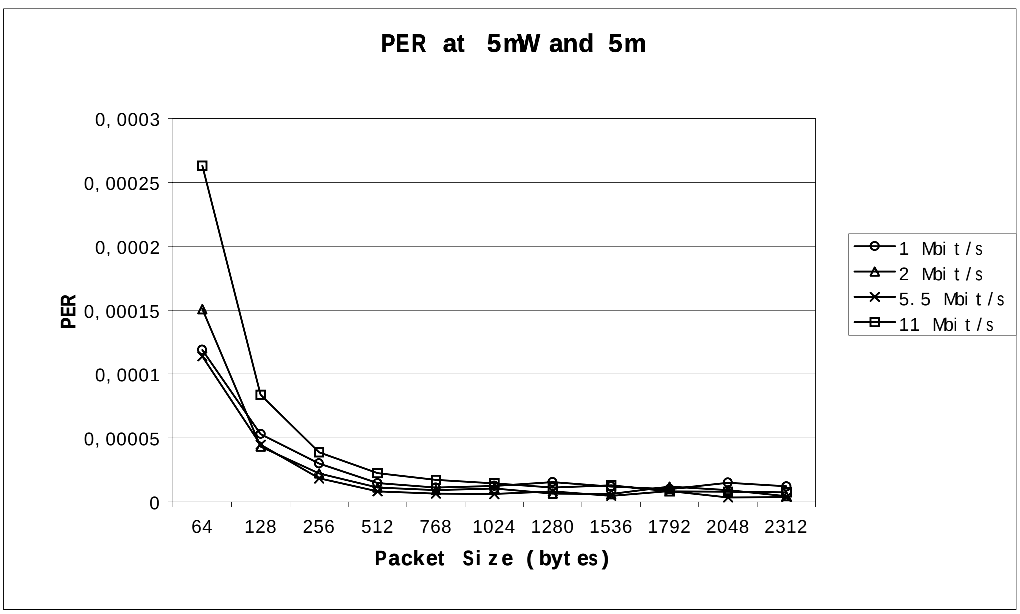


Figure 30: PER versus packet size at 5mW Xmit power and 5m

The graph above is of the PER versus packet size for each of the four data rates. The antenna transmission power is 5mW and the distance between nodes is 5m. Again, the sync problems force the PER of smaller packets to be much higher than the PER of larger packets. However, the values of the PER are smaller than they were in the previous graph. This suggests that the antenna transmission power increases the channel strength, which decreases the errors associated with sync problems and bit errors. The same behavior can be seen for the 20mW and 50mW graphs.

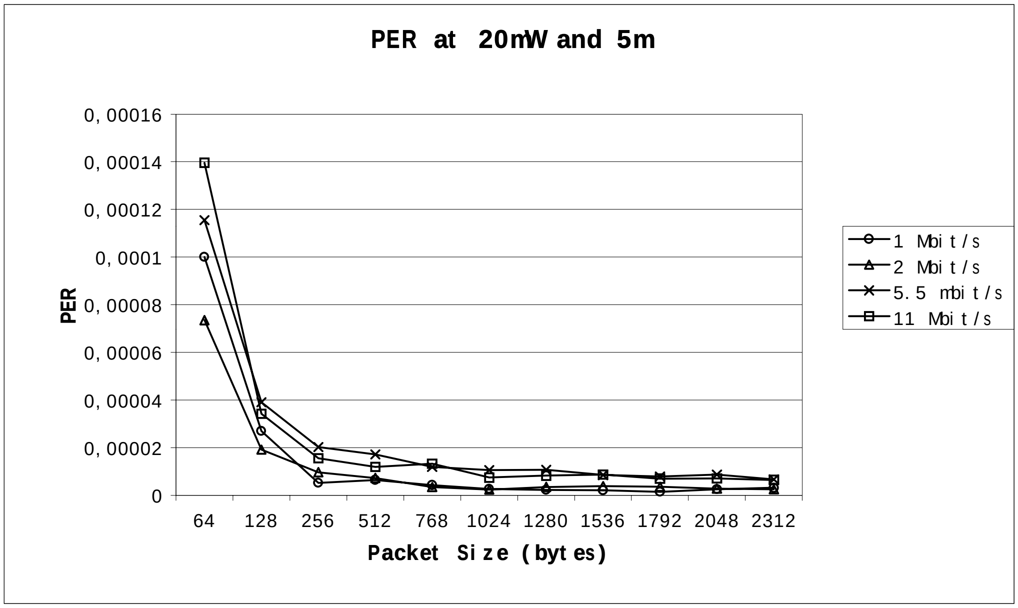


Figure 31: PER versus packet size at 20mW Xmit power and 5m

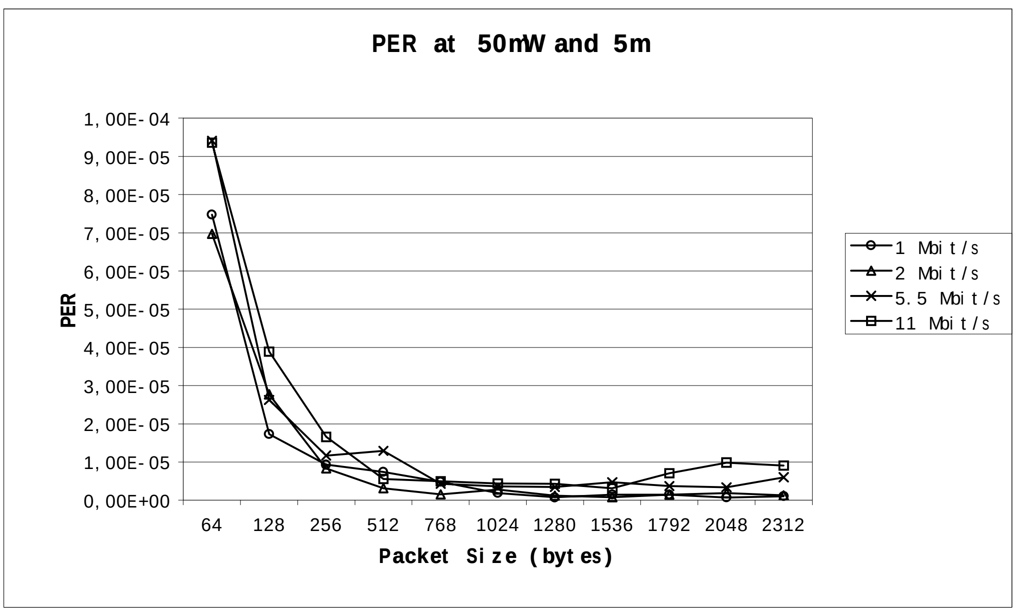


Figure 32: PER versus packet size at 50mW Xmit power and 5m

6.4.2 Transmission PER at 15 meters

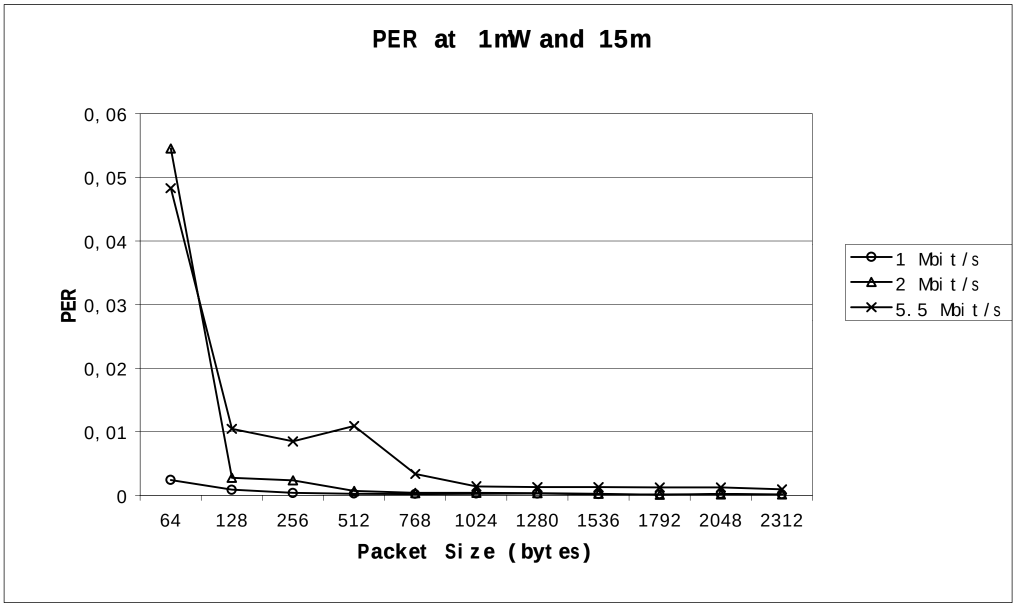


Figure 33: PER versus packet size at 1mW Xmit power and 15m

Above is a graph of the PER versus packet size for three of the four data rates. The antenna transmission power is 1mW and the distance between nodes is 15m. There is no data for 11 Mbit/s because there were many discontinuities in the data that would have made the graph unreadable. The discontinuities force the PER to be equal to one, which is far greater than other values in the graph. There is also no graph of the PER with a 5mW transmission power because all of the data rates contained many discontinuities and were unreadable.

The graph shows the same behavior that was seen in the previous graphs at 5m. The PER increases with the smaller packet sizes and decreases with the larger packet sizes. However, the values of the small packet PERs are significantly larger with the increased distance between nodes. The PER of the smallest packet size, 64 bytes, is nearly 100 times the value at 5m. The PER at higher packet sizes does not increase nearly as much as the PER of the smaller packets. This suggests that the transmission channel degradation affected packet synchronization more than the bit error probability. From the previous graphs, we know that packet synchronization skewed our results at smaller packet sizes. Normally, the PER should increase as the packet size increases. This is due to the increased bit error probability of larger packets. Since the PER increased so much more for smaller packet sizes, it can be reasoned that the increase is mainly due to sync problems. It can be concluded, that the increased distance increases the PER dramatically. Additionally, the PER for 15m was not reproducible because the channel conditions were bad and varied strongly for low transmission powers.

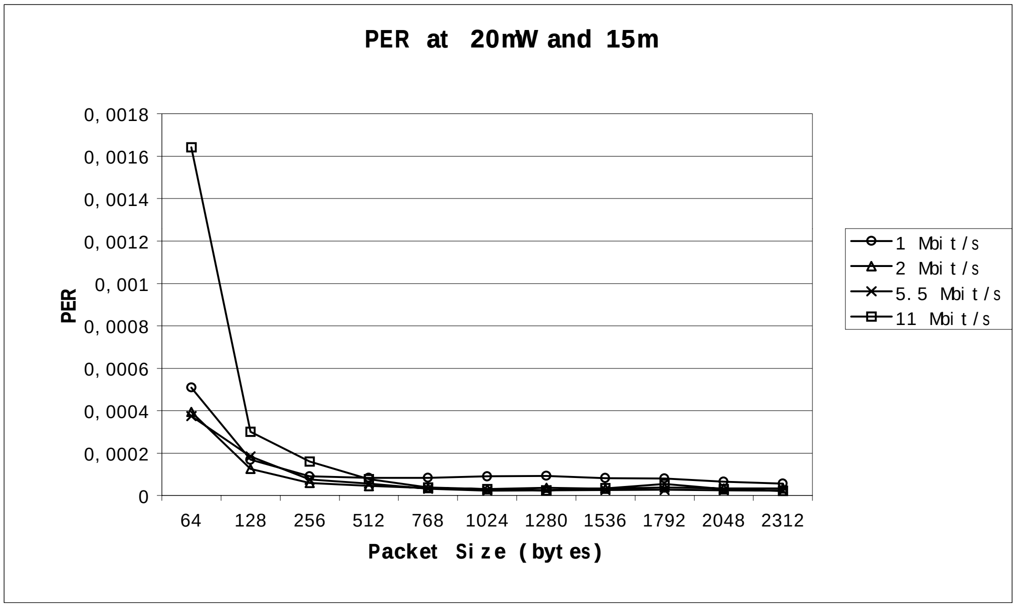


Figure 34: PER versus packet size at 20mW Xmit power and 15m

The graphs above and below are the PER versus packet size for each of the four data rates. The distance between nodes is still 15m, but the antenna transmission powers are 20mW and 50mW, respectively. The PER behavior is very similar to the previous PER graphs; it increases as the packet size is decreased. The PER decreases at higher antenna transmission powers as before, but the increase in the smaller packet sizes from the shorter distance is 10 and 4 times, respectively. This suggests that the sync problems at higher distances can be resolved effectively with higher transmission powers. This is consistent with the throughput measurements that increased at farther distances with a higher transmission power. The PER and throughput are inversely proportional.

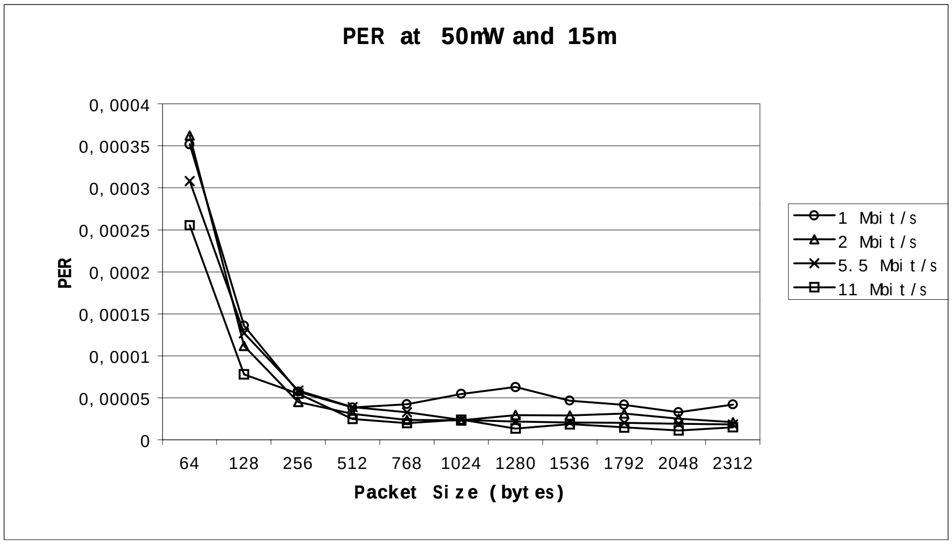


Figure 35: PER versus packet size at 50mW Xmit power and 15m

6.5 Average power consumption

The graphs in this section represent the average power consumption for transmission. Therefore, as opposed to the instantaneous power measurements above, the following power values represent the average power consumed sending packets of a certain size. This includes sending the data packet, receiving the acknowledgement, and idle times in between. The graphs are given for the four data rates, various RF transmission power levels, and at distances of 5m and 15m.

6.5.1 Average power consumption at 5m

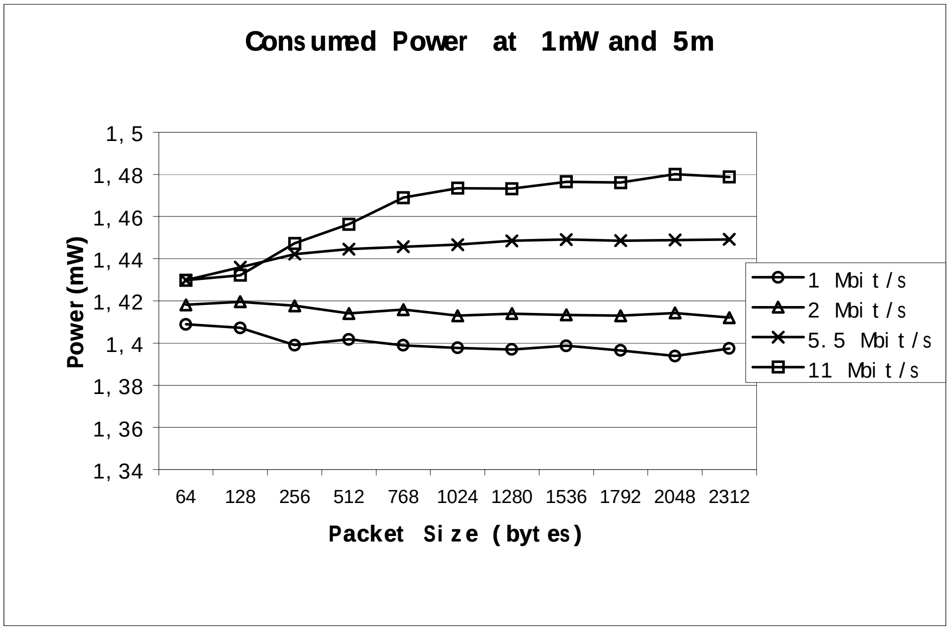


Figure 36: Power consumption versus packet size at 1mW Xmit power and 5m

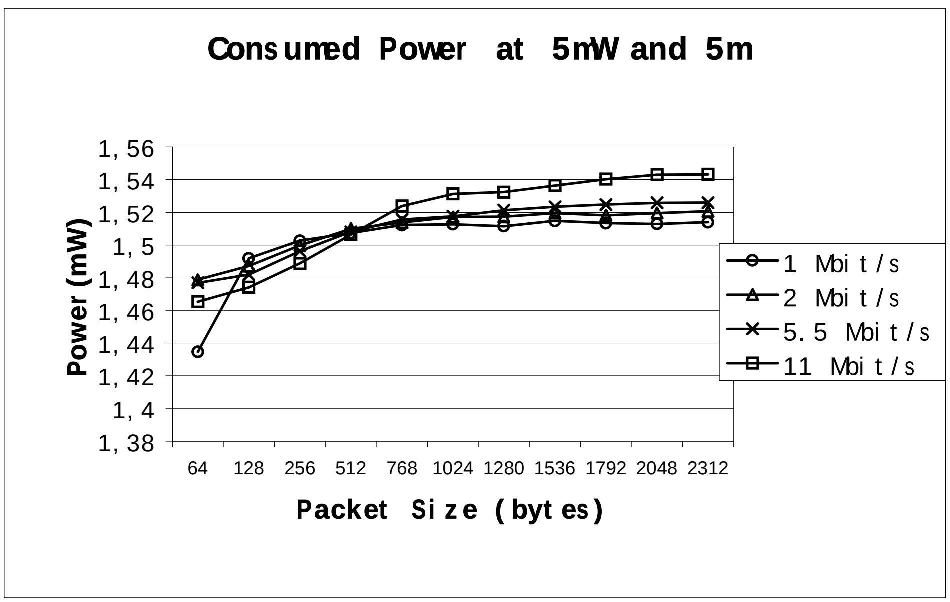


Figure 37: Power consumption versus packet size at 5mW Xmit power and 5m

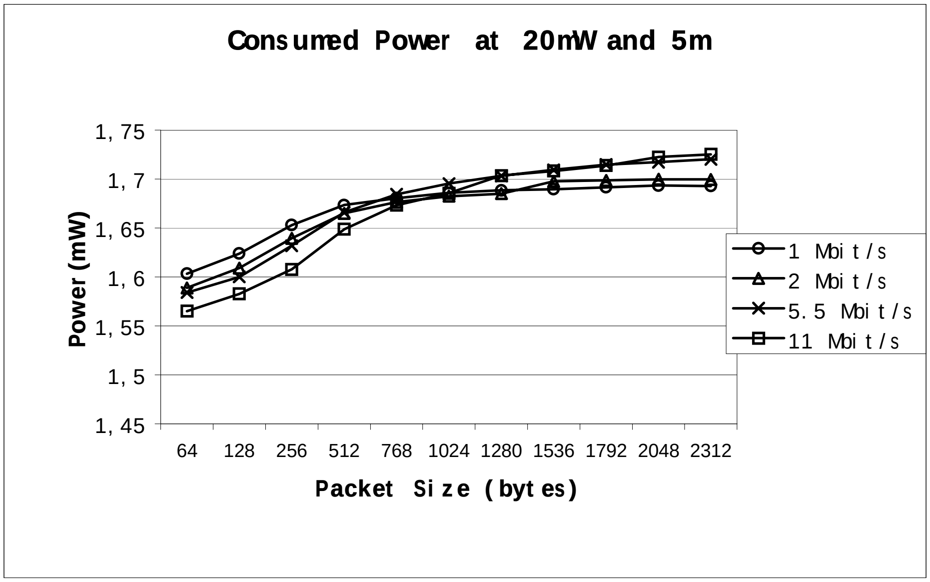


Figure 38: Power consumption versus packet size at 20mW Xmit power and 5m

It can be seen in the graphs above and below that the power consumption increases with the packet size. This is because the NIC is in the send state (relative to idle states) much longer for long packets. The power consumption also increases with the RF transmit power because the RF amplifier consumes more power in order to power the antennae. The data rate plays a less significant role when the power consumption is averaged.

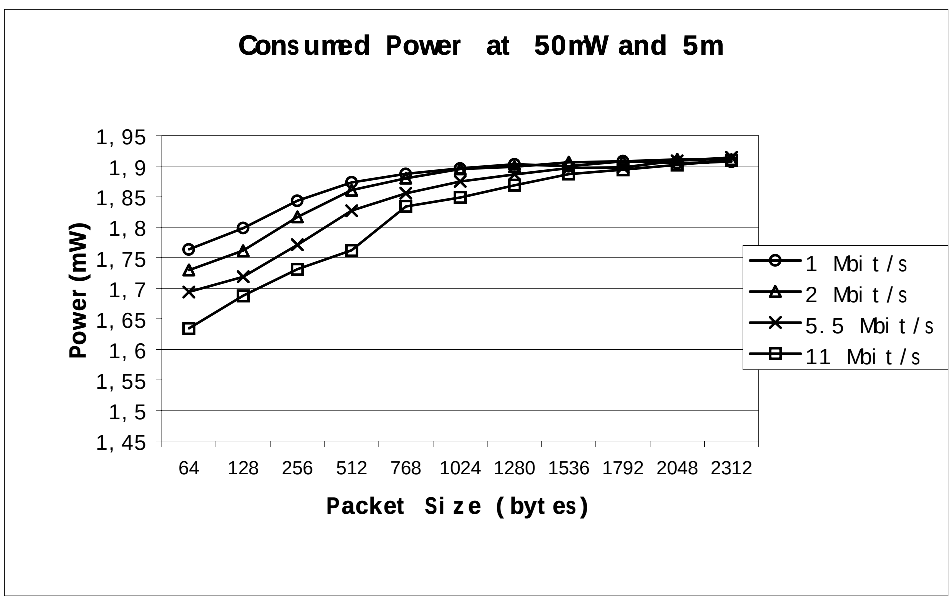


Figure 39: Power consumption versus packet size at 50mW Xmit power and 5m

6.5.2 Average power consumption for 15m

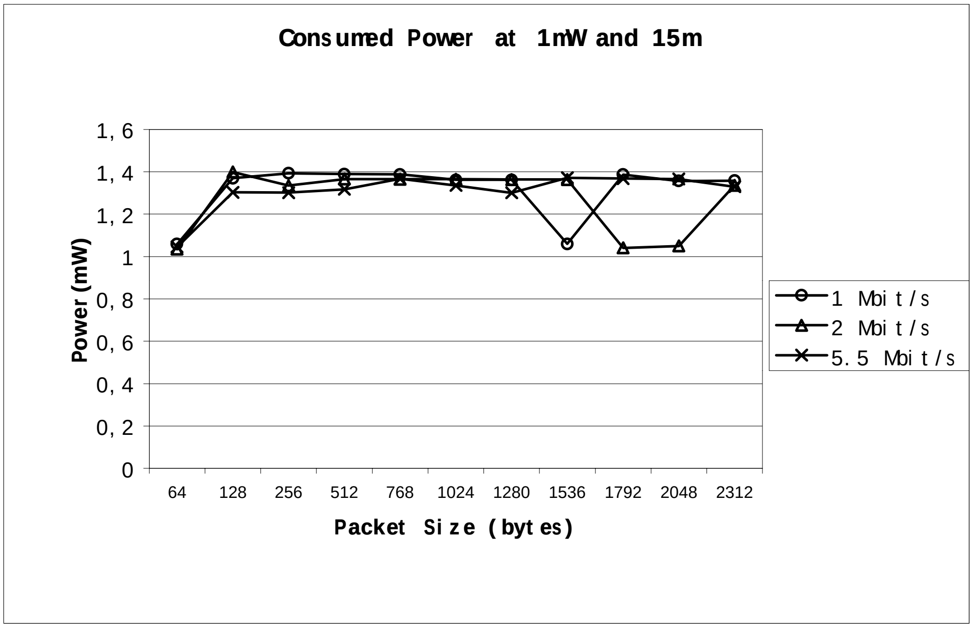


Figure 40: Power consumption versus packet size at 1mW Xmit power and 15m

Above is a graph of the average power consumption versus packet size for three of the four data rates. The antenna transmission power is 1mW and the distance between nodes is 15m. There is no data shown for 11 Mbit/s because there were many discontinuities in the data that would have made the graph unreadable (see also PER and throughput measurements for 15m). For the same reason there is also no graph of the power consumption for the 5mW transmission power.

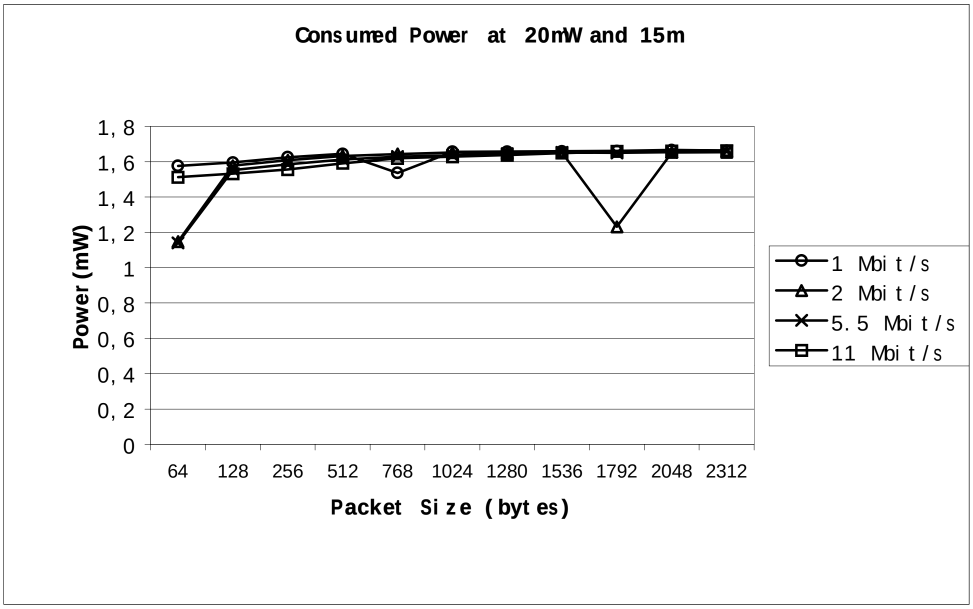


Figure 41: Power consumption versus packet size at 20mW Xmit power and 15m

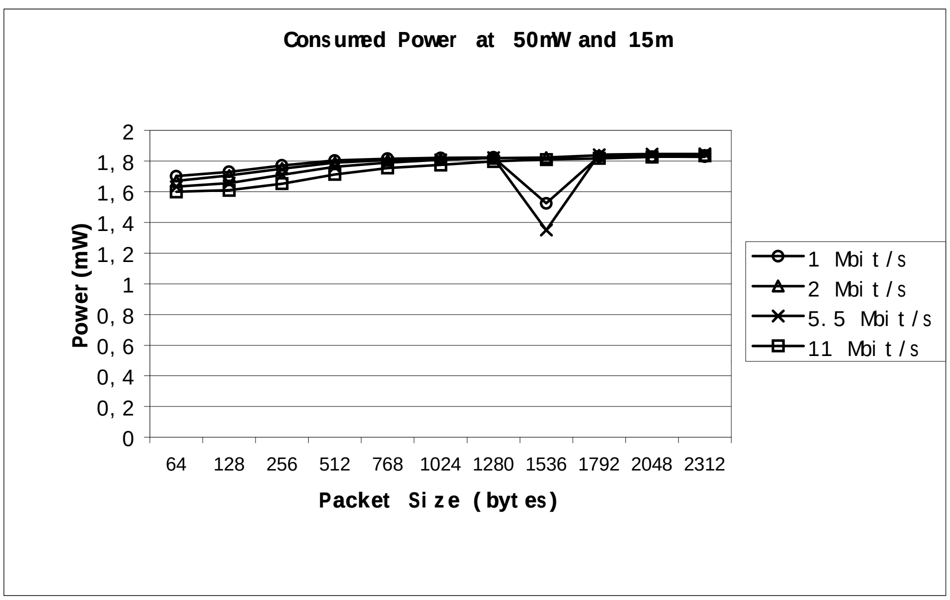


Figure 42: Power consumption versus packet size at 20mW Xmit power and 15m

The graphs above show the same behavior as the 5m plots. However, the most significant difference is that they both have anomalous values and the average power consumption at 5m is higher. We discovered that the card remained idle for a short amount of time after a transmission error before sending the next packet. That increase in the amount of idle time lead to a lower average power consumption at the higher distances.

6.6 Transmit Energy per Good Bit Transmitted

It was discovered that power measurements alone do not sufficiently reflect the power/energy efficiency of the WNIC for certain parameter settings. Therefore, we used a biased measure of power consumption in these experiments. The average power consumption was biased with the throughput (goodput) as shown below. This lead to the measure of *Energy per successfully transmitted payload bit.*

E_{bit_succ} [J/bit] = Average Power Consumption [W] / Throughput [bit/s]

Measurements of the power consumption per packet were taken as well as the throughput for each of the distances. Again, the measurements were taken at the four transmit powers (1mW, 5mW, 20mW and 50mW) and for each of the data rates (1Mbit/s, 2Mbit/s, 5.5Mbit/s and 11Mbit/s). The energy per bit was calculated by dividing the power consumption measurements by the corresponding throughput measurement. This yielded values of J/bit for the energy/bit transferred at the given data rate, packet size, and transmit power. The distance between nodes was taken under consideration because it affected the throughput and power measurements that were used to calculate these values.

6.6.1 Transmission Energy/bit at 5 meters

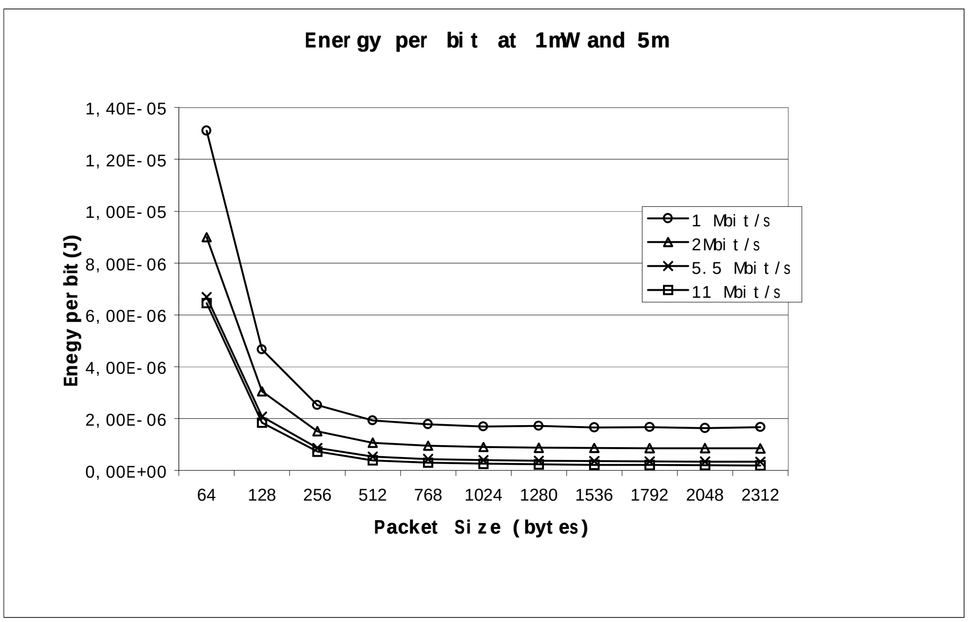


Figure 43: Energy/bit versus packet size at 1mW Xmit power and 5m

The graph above is the energy per bit versus packet size at each of the four data rates. The distance between nodes is 5m and the antenna transmission power is 1mW. The energy per bit is the amount of energy consumed by the WNIC to transmit a single bit of payload data. All of the channel overhead associated with each packet is taken into consideration in this calculation.(included in the average power consumption values). The NIC consumes energy to transmit the fixed packet overhead and this must be considered when calculating the energy consumed per good bit of data sent.

The energy/bit decreases as the packet size is increased. Therefore, a 64 byte packet sent at 11Mbit/s consumes more energy per good bit of data transferred than a 2312 byte packet sent at 11Mbit/s. This is consistent with the throughput and power consumption measurements. The power consumption varied very little as the packet size was increased and the throughput increased as the packet size was increased. Thus, the energy/bit = (power)/(throughput) should have the reciprocal relationship to packet size that the throughput has.

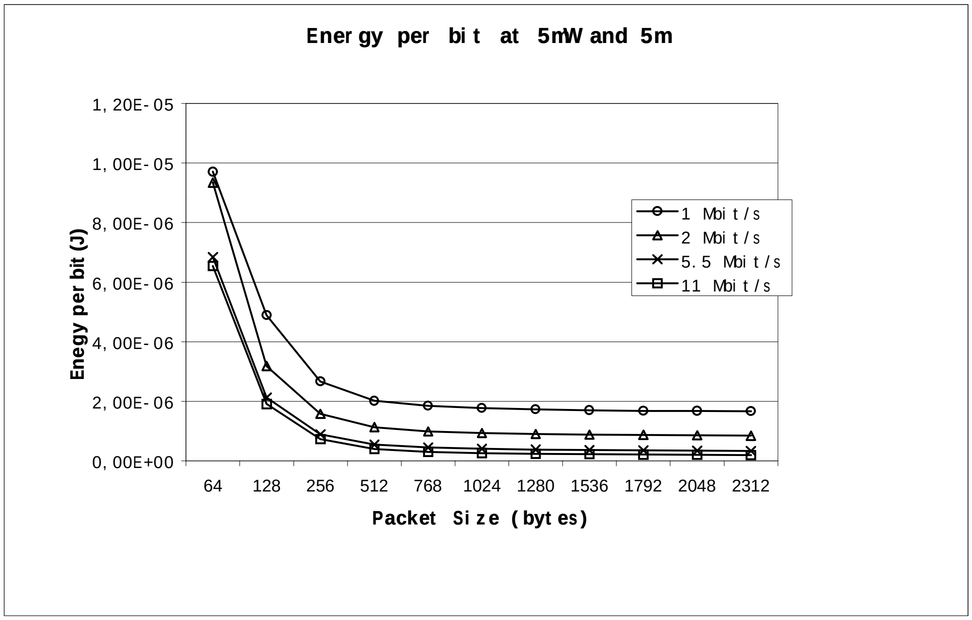


Figure 44: Energy/bit versus packet size at 5mW Xmit power and 5m

The graphs above and below are of the energy/bit versus packet size for each of the four data rates. The distance between nodes is 5m and the antenna transmission powers are 5mW and 20mW respectively. The graphs show the same trends as the previous graph. The energy/bit decreases as the packet size is increased or as the data rate is increased. The energy/bit also increases from 5mW to 20mW which is to be expected. The power consumption per packet increased as the antenna tranmssion power was increased, while the throughput does not change at this distance. The energy/bit should show the same correlation as the power consumption measurements.

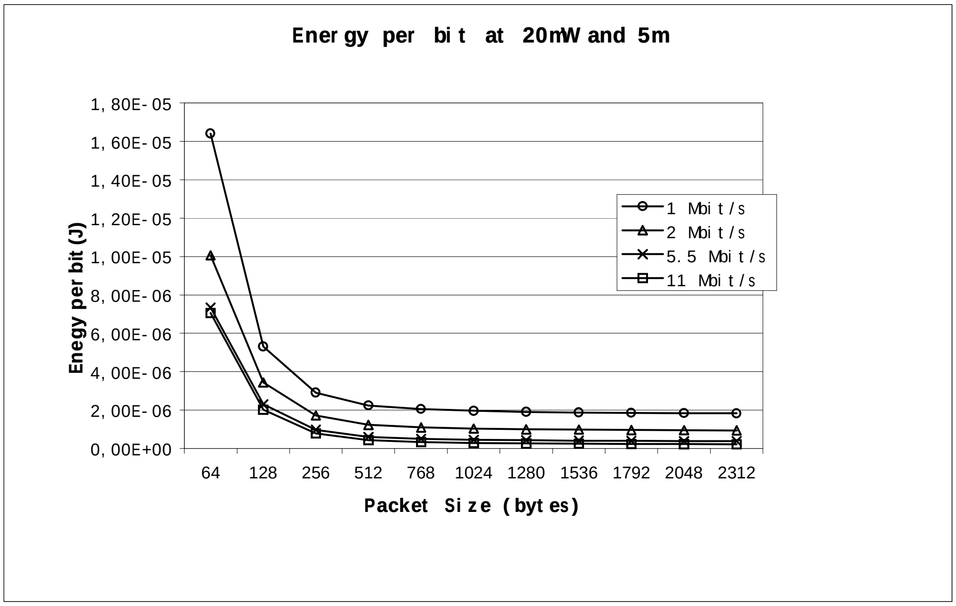


Figure 45: Energy/bit versus packet size at 20mW Xmit power and 5m

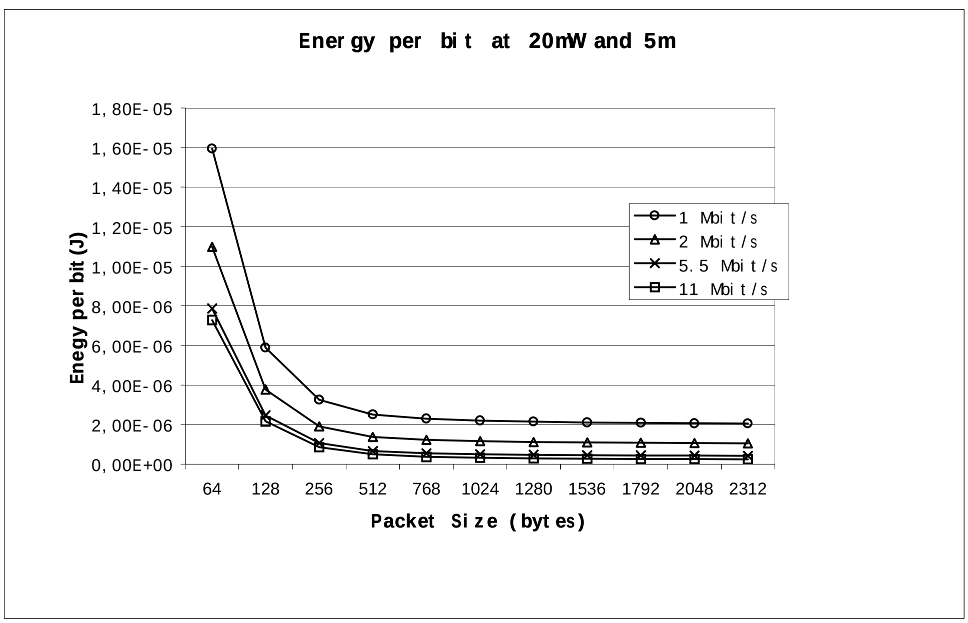


Figure 6.5d: Energy/bit versus packet size at 50mW Xmit power and 5m

The graph above is the energy/bit versus packet size for each of the four data rates. The distance between nodes is 5m and the antenna transmission power is 50mW. As the previous three graphs indicated, the energy/bit decreases for larger packet sizes and data rates. The WNIC consumes the most energy/bit when this antenna power is used, however, which is consistent with the power consumption measurements. The power consumed by the WNIC increased with the antenna transmission power. There are no throughput gains at this distance, thus the increase in energy/bit is wasted energy.

6.6.2 Transmission Energy/bit at 15 meters

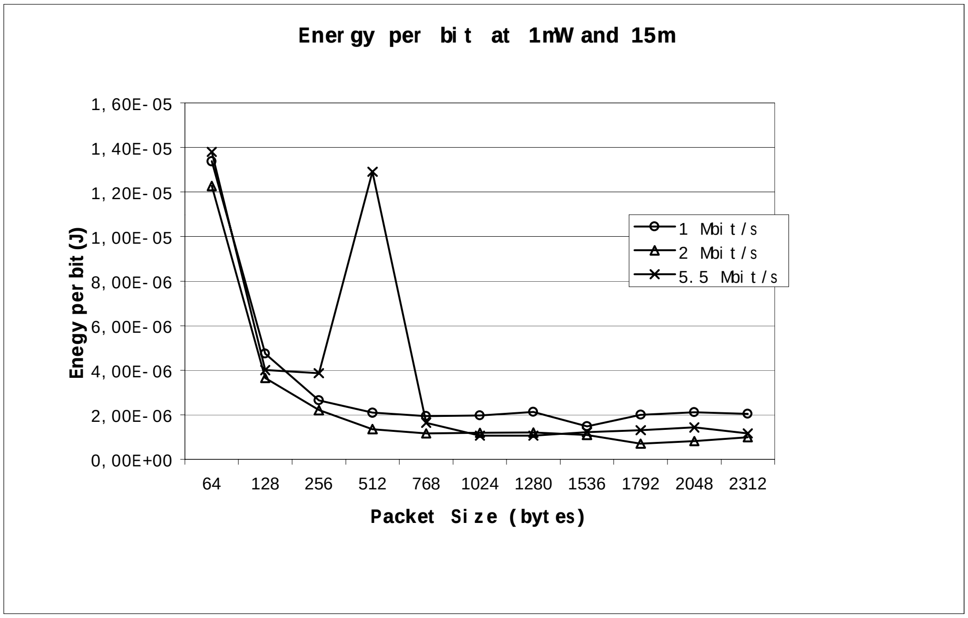


Figure 46: Energy/bit versus packet size at 1mW Xmit power and 15m

The graph above is the energy/bit versus packet size for three of the four data rates. The distance between nodes is 15m and the antenna transmission power is 1mW. The energy/bit for 11Mbit/s is not on the graph because it contained too many discontinuities and was unreadable[1]. As in the graphs before, the energy/bit decreases as the packet size is increased. The drop in throughput for 512 byte packets transmitted at 5.5Mbit/s is translated into a jump in the energy/bit at that point. This suggests that channel degradation will increase the energy/bit as it decreases the throughput.. There is a slight increase in energy/bit when compared to the 5m, 1mW graph. This increase is due to an increase in bit errors (useless transmitted packets) and a small decrease in throughput.

[1] Actually, the throughput was zero for many of the packet sizes.

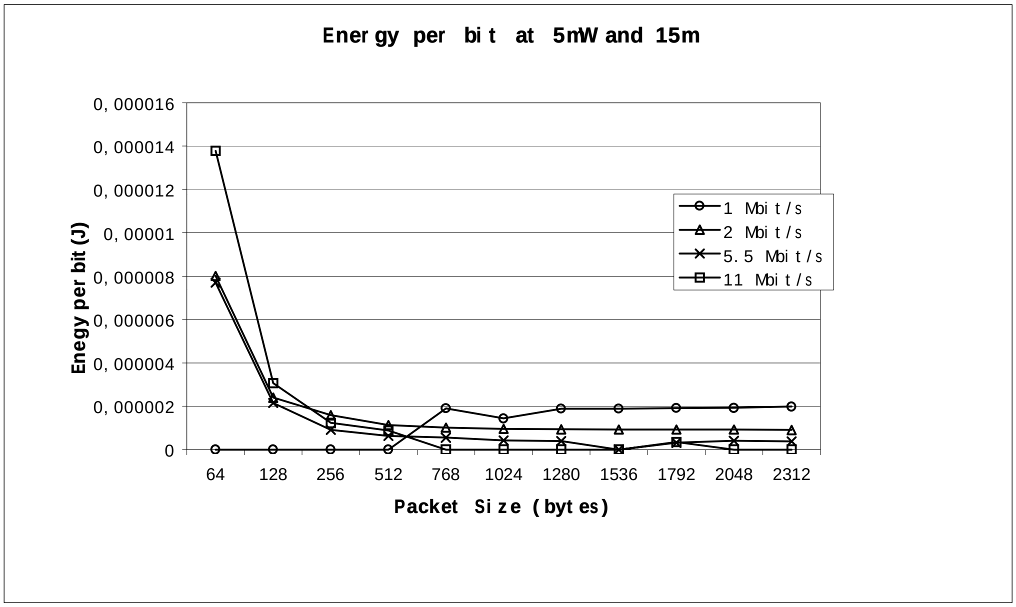


Figure 47: Energy/bit versus packet size at 5mW Xmit power and 15m

The graphs above and below are the energy/bit versus packet size for each of the four data rates. The distance between nodes is 15m and the antenna transmission powers are 5mW and 20mW, respectively. The energy/bit still decreases as the packet size is increased or as the data rate is increased. Discontinuities can be seen in the graph above for packet sizes between 768 and 1792 bytes. This is because the throughput is zero for those points. This could be interpreted mathematically as infinite energy/bit because there is non-zero power consumption at those points, but is incorrect. It is due to sync problems in the channel and the packets were never received, so those data points are set to zero.

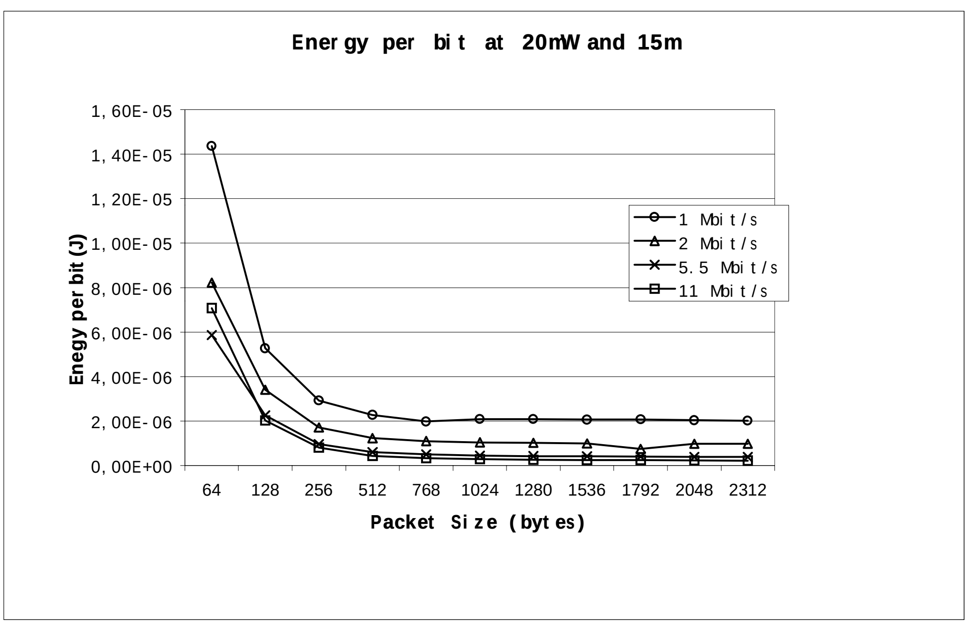


Figure 48: Energy/bit versus packet size at 20mW Xmit power and 15m

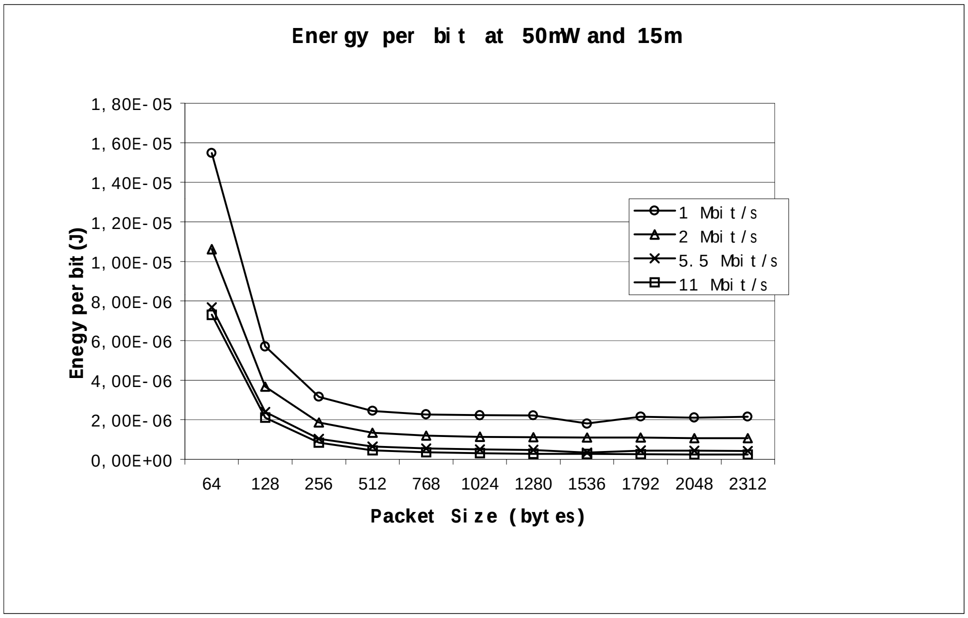


Figure 49: Energy/bit versus packet size at 50mW Xmit power and 15m

The graph above is the energy/bit versus packet size for each of the four data rates. The distance between nodes is 15m and the antenna transmission power is 50mW. The WNIC still consumes the most energy/bit at the highest antenna power, which is consistent with the power consumption measurements. However, there is no increase in the energy/bit from 5m to 15m as one might expect. The throughputs at 5m and 15m for the 50mW transmission power are equivalent. The channel is always good, thus there are very few PERs due to sync errors in this scenario. PERs due to bit errors do increase with distance, but at the highest transmission power it does not affect the average energy/bit significantly. The bit errors are too few.

7 Conclusions:

From the power consumption graphs in section 6.1, it is clear that the average power consumed by the WNIC while transmitting increases slightly with the data rate and more significantly with antenna transmitter power. The power consumed by the Intersil PRISM I chipset is outlined in the pre-considerations. During transmission, the RF power amplifier consumes the most power of all IC's by far. This component operates only during transmit mode. Its power consumption increases over-proportionally as the antenna transmitter power is increased. Thus, transmission power consumption increases with RF transmission power.

The average power consumption of the Aironet PC 4800 during reception increases slightly with the data rate but does not increase with the antenna transmitter power. The antenna transmitter power does not affect the instantaneous and average power measurements. They were not considered for the power measurments of the working modes of the WNIC (instantaneous measurements). They were taken into account for the measurements of the average power consumption during reception, but ACKs belong to control response messages which are only tranmitted at the highest RF power level. Therefore, they do not influence the measurements. The omission of the RF power amplifier power while receiving greatly reduces the power consumption of the WNIC. The instantaneous power consumption per packet during transmission is not affected greatly by the packet size. Results are very similar in the average power consumption plots for each data rate and transmission power. Therefore, there is no packet size – power consumption dependency.

The throughput graphs in 6.3 show that the 11Mb/s data rate has by far the highest throughput while 1Mb/s has the lowest. At both distances, the throughput increased with the packet size because of the lower amount of bandwidth taken up by the fixed packet overhead. These experiments were performed on a two-node network and had no collisions. In an environment with more nodes using the same transmission channel, collisions might greatly affect the results. Larger packets occupy the medium for a larger amount of time as the graph in section 6.2 demonstrates. Large packets are more prone to collisions and bit errors in higher traffic networks. Thus, they would lower the throughput in networks with more network nodes. Our two-network node is a simplification to determine basic power consumption values of the PC4800 card.

At the first distance, the throughput was not affected by the antenna transmit power. The throughput was roughly the same for 1mW as it was for 50mW. However, when the distance was increased, the throughput dropped to almost zero at lower transmit powers but increased to the same values as before when the transmit power was increased. It appears that the throughput is either all or nothing. This is because the transmission channel was either good or bad during these experiments. This indicates that the throughput and energy consumption can be controlled significantly by the RF transmission power. In good channel conditions (sufficient RF transmission power) higher data rates lead to better results with respect to throughput and energy consumption.

The PER measurements showed unexplainable results at first glance. They increased at smaller packet sizes, but still showed the lower PER values that one would expect at close distances. We believe that longer traces would provide more statistically viable measurements for the PER. In our measurements, we transmitted too few packets over-all and many more small packets than large packets. This skewed the PER results because sync problems probably caused many errors at the smaller packet sizes. This shows that in Direct Sequence Spread Spectrum, sync problems on a bad transmitting channel can be a large problem.

Other than problems with the PER at smaller packet sizes, the PER showed expected results. The PER due to bit errors decreased with the antenna transmitter power and increased with the distance between nodes. If traces for longer packets had been taken, the PER should have shown certainly that bit errors increase with the packet size.

The energy consumed per good bit of data transferred over the medium decreased as the packet size and data rate increased, and increased as the transmit power increased. Thus, 2312 byte packets transmitted at 11Mb/s at a transmit power of 1mW consumed the least energy per bit and 64 byte packets transmitted at 1Mb/s with a transmit power of 50mW consumed the most. There was a slight increase in the energy/bit at larger distances, however, this was due to an increase in bit errors and a decrease in the throughput.

Lastly, the measurements at 15m have a low statistical relevance. Longer traces and a higher number of measurement runs are needed to achieve stable results. However, this was unrealistic because of limitations in time and disk space. The results given for 15m are for informational and trend estimation purposes only. Further work is needed to determine the results for higher distances. It would be possible to accomplish this through simulations.

The main purpose of these experiments was to parameterize WNIC operating modes with realistic values of power consumption in sleep, idle, transmit, and receive. These measurements are of the total power consumption by the WNIC and create a clearer picture of how network parameters affect the power consumption. The values of throughput, PER, and energy/bit are to parameterize and verify simulation results that assume the same simple network scenarios. Therefore, future and current simulations that have relied on percentages and assumptions can be verified with realistic values of network performance and power consumption.

8 References

[1] The Editors, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications," Standard 802.11, Institute of Electrical and Electronic Engineers, Inc., November 1997

[2] http://sourceforge.net/projects/airo-linux/

[3] http://www.intersil.com/design/prism/ser-pi-11mbps.asp

[4] "Users Guide and Technical Reference Manual – Aironet Wireless LAN Adapter: PC4500 and PC4800, " DOC-710-004239-B0, Aironet Wireless Communications, Inc., 3875 Embassy Parkway, Akron, Ohio 44333-8357, http://www.princeton.edu/wireless/downloads/aironet/pc4800/Documentation/PC4800_manual.pdf

[5] Sycard Technolgy PCCExtend 140: http://www.sycard.com/

[6] Netperf: http://www.netperf.org/netperf/DownloadNetperf.html

[7] National Instruments: http://www.ni.com/

[8] Perl 5.6.0: http://cygutils.netpedia.net/V1.1/perl-5.6.0/index.html

[9] openSSH: ftp://ftp.franken.de/pub/win32/develop/gnuwin32/cygwin/porters/Vinschen_Corinna/V1.1.1/openssh-2.10p3.tar.gz