CHAPTER 9

# Physical Layer Overview

*Any girl can be glamorous. All you have to do is stand still and look stupid.*
—Hedy Lamarr

Protocol layering allows for research, experimentation, and improvement on different parts of the protocol stack. The second major component of the 802.11 architecture is the physical layer, which is often abbreviated PHY. This chapter introduces the common themes and techniques that appear in each of the radio-based physical layers and describes the problems common to all radio-based physical layers; it is followed by more detailed explanations of each of the physical layers that are standardized for 802.11.

## Physical-Layer Architecture

The physical layer is divided into two sublayers: the *Physical Layer Convergence Procedure* (PLCP) sublayer and the *Physical Medium Dependent* (PMD) sublayer. The PLCP (Figure 9-1) is the glue between the frames of the MAC and the radio transmissions in the air. It adds its own header. Normally, frames include a preamble to help synchronize incoming transmissions. The requirements of the preamble may depend on the modulation method, however, so the PLCP adds its own header to any transmitted frames. The PMD is responsible for transmitting any bits it receives from the PLCP into the air using the antenna. The physical layer also incorporates a *clear channel assessment* (CCA) function to indicate to the MAC when a signal is detected.

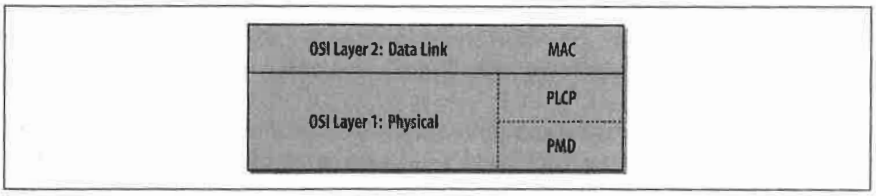


*Figure 9-1. Physical layer logical architecture*



DELL-1031, Part 2
10,291,449

## The Radio Link

Three physical layers were standardized in the initial revision of 802.11, which was published in 1997:

- Frequency-hopping (FH) spread-spectrum radio PHY
- Direct-sequence (DS) spread-spectrum radio PHY
- Infrared light (IR) PHY

In 1999, two further physical layers based on radio technology were developed:

- 802.11a: Orthogonal Frequency Division Multiplexing (OFDM) PHY
- 802.11b: High-Rate Direct Sequence (HR/DS or HR/DSSS) PHY

This book discusses the four physical layers based on radio waves in detail; it does not discuss the infrared physical layer, which isn't widely used.

### The IR PHY

802.11 also includes a specification for a physical layer based on infrared (IR) light. Using infrared light instead of radio waves seems to have several advantages. IR ports are less expensive than radio transceivers—in fact, the cost is low enough that IR ports are standard on practically every laptop.

IR is extremely tolerant of radio frequency (RF) interference because radio waves operate at a totally different frequency. This leads to a second advantage: IR is unregulated. Product developers do not need to investigate and comply with directives from several regulatory organizations throughout the world.

Security concerns regarding 802.11 are largely based on the threat of unauthorized users connecting to a network. Light can be confined to a conference room or office by simply closing the door. IR-based LANs can offer some of the advantages of flexibility and mobility but with fewer security concerns. This comes at a price. IR LANs rely on scattering light off the ceiling, so range is much shorter.

This discussion is academic, however. No products have been created based on the IR PHY. The infrared ports on laptops comply with a set of standards developed by the Infrared Data Association (IrDA), not 802.11. Even if products were created around the IR PHY, the big drivers to adopt 802.11 are flexibility and mobility, which are better achieved by radio's longer range and ability to penetrate solid objects.

## Licensing

The classic approach to radio communications is to confine an information-carrying signal to a narrow frequency band and pump as much power as possible (or legally allowed) into the signal. Noise is simply the naturally present distortion in

the frequency band. Transmitting a signal in the face of noise relies on brute force—you simply ensure that the power of the transmitted signal is much greater than the noise.

In the classic transmission model, avoiding interference is a matter of law, not physics. With high power output in narrow bands, a legal authority must impose rules on how the RF spectrum is used. In the United States, the Federal Communications Commission (FCC) is responsible for regulating the use of the RF spectrum. Many FCC rules are adopted by other countries throughout the Americas. European allocation is performed by CEPT's European Radiocommunications Office (ERO) and the European Telecommunications Standards Institute (ETSI). Other allocation work is performed by the International Telecommunications Union (ITU).

For the most part, an institution must have a license to transmit at a given frequency. Licenses can restrict the frequencies and transmission power used, as well as the area over which radio signals can be transmitted. For example, radio broadcast stations must have a license from the FCC. Likewise, mobile telephone networks must obtain licenses to use the radio spectrum in a given market. Licensing guarantees the exclusive use of a particular set of frequencies. When licensed signals are interfered with, the license holder can demand that a regulatory authority step in and resolve the problem, usually by shutting down the source of interference.

### Frequency allocation and unlicensed frequency bands

Radio spectrum is allocated in bands dedicated to a particular purpose. A band defines the frequencies that a particular application may use. It often includes guard bands, which are unused portions of the overall allocation that prevent extraneous leakage from the licensed transmission from affecting another allocated band.

Several bands have been reserved for unlicensed use. For example, microwave ovens operate at 2.45 GHz, but there is little sense in requiring homeowners to obtain permission from the FCC to operate microwave ovens in the home. To allow consumer markets to develop around devices built for home use, the FCC (and its counterparts in other countries) designated certain bands for the use of "industrial, scientific, and medical" equipment. These frequency bands are commonly referred to as the *ISM bands*. The 2.4-GHz band is available worldwide for unlicensed use.* Unlicensed use, however, is not the same as unlicensed sale. Building, manufacturing, and designing 802.11 equipment does require a license; every 802.11 card legally sold in the U.S. carries an FCC identification number. The licensing process requires the manufacturer to file a fair amount of information with the FCC. All this information is a matter of public record and can be looked up online by using the FCC identification number.

* The 2.4-GHz ISM band is reserved by the FCC rules (Title 47 of the Code of Federal Regulations), part 15.247. ETSI reserved the same spectrum in ETSI Technical Specifications (ETS) 300-328.

## The Nonexistent Microwave Absorption Peak of Water

It is often said that microwave ovens operate at 2.45 GHz because it corresponds to a particular excitation mode of water molecules. This is sometimes even offered as a reason why 802.11 cannot be used over long distances. Atmospheric water vapor would severely attenuate any microwave signals in rain or in humid climates.

The existence of a water excitation mode in the microwave range is a myth. If there was an excitation mode, water would absorb a significant amount of the microwave energy. And if that energy was absorbed effectively by water, microwave ovens would be unable to heat anything other than the water near the surface of food, which would absorb all the energy, leaving the center cold and raw. An absorption peak would also mean that atmospheric water vapor would disrupt satellite communications, which is not an observed phenomenon. NASA Reference Publication 1108(02), *Propagation Effects on Satellite Systems at Frequencies Below 10 GHz*, discusses the expected signal loss due to atmospheric effects, and the loss is much more pronounced at frequencies above 10 GHz. The microwave absorption peak for water, for example, is at 22.2 GHz.

Microwave ovens do not work by moving water molecules into an excited state. Instead, they exploit the unusually strong dipole moment of water. Although electrically neutral, the dipole moment allows a water molecule to behave as if it were composed of small positive and negative charges at either end of a rod. In the cavity of a microwave oven, the changing electric and magnetic fields twist the water molecules back and forth. Twisting excites the water molecules by adding kinetic energy to the entire molecule but does not change the excitation state of the molecule or any of its components.

Use of equipment in the ISM bands is generally license-free, provided that devices operating in them do not emit significant amounts of radiation. Microwave ovens are high-powered devices, but they have extensive shielding to restrict radio emissions. Unlicensed bands have seen a great deal of activity in the past three years as new communications technologies have been developed to exploit the unlicensed band. Users can deploy new devices that operate in the ISM bands without going through any licensing procedure, and manufacturers do not need to be familiar with the licensing procedures and requirements. At the time this book was written, a number of new communications systems were being developed for the 2.4-GHz ISM band:

- The variants of 802.11 that operate in the band (the frequency-hopping layer and both spread spectrum layers)
- Bluetooth, a short-range wireless communications protocol developed by an industry consortium led by Ericsson
- Spread-spectrum cordless phones introduced by several cordless phone manufacturers
- X10, a protocol used in home automation equipment that can use the ISM band for video transmission

Unfortunately, "unlicensed" does not necessarily mean "plays well with others." All that unlicensed devices must do is obey limitations on transmitted power. No regulations specify coding or modulation, so it is not difficult for different vendors to use the spectrum in incompatible ways. As a user, the only way to resolve this problem is to stop using one of the devices; because the devices are unlicensed, regulatory authorities will not step in.

### Other unlicensed bands

Additional spectrum is available in the 5-GHz range. In the United States, the following three bands are called the Unlicensed National Information Infrastructure (UNII) bands:*

- 5.15–5.25 GHz
- 5.25–5.35 GHz
- 5.725–5.825 GHz

Devices operating in the UNII bands must obey limitations on radiated power, but there are no further constraints imposed on them. European regulatory authorities have set aside the same frequency bands, but the first two bands are dedicated to HiperLAN technology; the third band is the only one potentially available for 802.11 networks.

## Spread Spectrum

Spread-spectrum technology is the foundation used to reclaim the ISM bands for data use. Traditional radio communications focus on cramming as much signal as possible into as narrow a band as possible. Spread spectrum works by using mathematical functions to diffuse signal power over a large range of frequencies. When the receiver performs the inverse operation, the smeared-out signal is reconstituted as a narrow-band signal, and, more importantly, any narrow-band noise is smeared out so the signal shines through clearly.

Use of spread-spectrum technologies is a requirement for unlicensed devices. In some cases, it is a requirement imposed by the regulatory authorities; in other cases, it is the only practical way to meet regulatory requirements. As an example, the FCC requires that devices in the ISM band use spread-spectrum transmission and impose acceptable ranges on several parameters.

Spreading the transmission over a wide band makes transmissions look like noise to a traditional narrowband receiver. Some vendors of spread-spectrum devices claim that the spreading adds security because narrowband receivers cannot be used to pick up the full signal. Any standardized spread-spectrum receiver can easily be used, though, so additional security measures are mandatory in nearly all environments.

* The UNII bands are defined by FCC part 15.407.

This does not mean that spread spectrum is a "magic bullet" that eliminates interference problems. Spread-spectrum devices can interfere with other communications systems, as well as with each other; and traditional narrow-spectrum RF devices can interfere with spread spectrum. Although spread spectrum does a better job of dealing with interference within other modulation techniques, it doesn't make the problem go away. As more RF devices (spread spectrum or otherwise) occupy the area that your wireless network covers, you'll see the noise level go up, the signal-to-noise ratio decrease, and the range over which you can reliably communicate drop.

To minimize interference between unlicenced devices, the FCC imposes limitations on the power of spread-spectrum transmissions. The legal limits are one watt of transmitter output power and four watts of effective radiated power (ERP). Four watts of ERP are equivalent to 1 watt with an antenna system that has 6-dB gain, or 500 milliwatts with an antenna of 9-dB gain, etc.* The transmitters and antennas in PC Cards are obviously well within those limits—and you're not getting close even if you use a commercial antenna. But it is possible to cover larger areas by using an external amplifier and a higher-gain antenna. There's no fundamental problem with doing this, but you must make sure that you stay within the FCC's power regulations.

### Types of spread spectrum

The radio-based physical layers in 802.11 use three different spread-spectrum techniques:

*Frequency hopping (FH or FHSS)*
: Frequency-hopping systems jump from one frequency to another in a random pattern, transmitting a short burst at each subchannel. The 2-Mbps FH PHY is specified in clause 14.

*Direct sequence (DS or DSSS)*
: Direct-sequence systems spread the power out over a wider frequency band using mathematical coding functions. Two direct-sequence layers were specified. The initial specification in clause 15 standardized a 2-Mbps PHY, and 802.11b added clause 18 for the HR/DSSS PHY.

*Orthogonal Frequency Division Multiplexing (OFDM)*
: OFDM divides an available channel into several subchannels and encodes a portion of the signal across each subchannel in parallel. The technique is similar to the Discrete Multi-Tone (DMT) technique used by some DSL modems. Clause 17, added with 802.11a, specifies the OFDM PHY.

* Remember that the transmission line is part of the antenna system, and the system gain includes transmission line losses. So an antenna with 7.5-dB gain and a transmission line with 1.5-dB loss has an overal system gain of 6 dB. It's worth noting that transmission line losses at UHF freqencies are often very high; as a result, you should keep your amplifier as close to the antenna as possible.

## The Unlikely Invention of Spread Spectrum

Spread spectrum was patented in the early 1940s by Austrian-born actress Hedy Lamarr. She was certainly better known for other reasons: appearing in the first nude scene on film in the Czech film *Ecstasy*, her later billing as "the most beautiful woman in the world" by Hollywood magnate Louis Mayer, and as the model for Catwoman in the Batman comics.

Before fleeing the advance of Nazi Germany, she was married to an Austrian arms merchant. While occupying the only socially acceptable role available to her as a hostess and entertainer of her husband's business clients, she learned that radio remote control of torpedoes was a major area of research for armaments vendors. Unfortunately, narrowband radio communications were subject to jamming, which neutralized the advantage of radio-guided weapons. From these discussions, she first hit on the idea of using a complex but predetermined hopping pattern to move the frequency of the control signal around. Even if short bursts on a single frequency could be jammed, they would move around quickly enough to prevent total blockage. Lamarr worked out everything except how to precisely control the frequency hops.

After arriving in the United States, she met George Antheil, an avant-garde American composer known as the "bad boy of music" for his dissonant style. His famous *Ballet mécanique* used (among many outrageous noisemakers) 16 player pianos controlled from a single location. Performing the piece required precisely controlled timing between distributed elements, which was Lamarr's only remaining challenge in controlling the hopping pattern. Together, they were granted U.S. patent number 2,292,387 in 1942. The patent expired in 1959 without earning a cent for either of them, and Lamarr's contributions went unacknowledged for many years because the name on the patent was Hedy Kiesler Markey, her married name at the time. The emerging wireless LAN market in the late 1990s led to the rediscovery of her invention and widespread recognition for the pioneering work that laid the foundation for modern telecommunications.

Frequency-hopping techniques were first used by U.S. ships blockading Cuba during the Cuban Missile Crisis. It took many years for the electronics underpinning spread-spectrum technology to become commercially viable. Now that they have, spread-spectrum technologies are used in cordless and mobile phones, high-bandwidth wireless LAN equipment, and every device that operates in the unlicensed ISM bands. Unfortunately, Hedy Lamarr died in early 2000, just as the wireless LAN market was gaining mainstream attention.

Frequency-hopping systems are the cheapest to make. Precise timing is needed to control the frequency hops, but sophisticated signal processing is not required to extract the bit stream from the radio signal. Direct-sequence systems require more sophisticated signal processing, which translates into more specialized hardware and higher electrical power consumption. Direct-sequence techniques also allow a higher data rate than frequency-hopping systems.

# RF and 802.11

802.11 has been adopted at a stunning rate. Many network engineers accustomed to signals flowing along well-defined cable paths are now faced with a LAN that runs over a noisy, error-prone, quirky radio link. In data networking, the success of 802.11 has inexorably linked it with RF engineering. A true introduction to RF engineering requires at least one book, and probably several. For the limited purposes I have in mind, the massive topic of RF engineering can be divided into two parts: how to make radio waves and how radio waves move.

## RF Components

RF systems complement wired networks by extending them. Different components may be used depending on the frequency and the distance that signals are required to reach, but all systems are fundamentally the same and made from a relatively small number of distinct pieces. Two RF components are of particular interest to 802.11 users: antennas and amplifiers. Antennas are of general interest since they are the most tangible feature of an RF system. Amplifiers complement antennas by allowing them to pump out more power, which may be of interest depending on the type of 802.11 network you are building.

### Antennas

Antennas are the most critical component of any RF system because they convert electrical signals on wires into radio waves and vice versa. In block diagrams, antennas are usually represented by a triangular shape, as shown in Figure 9-2.

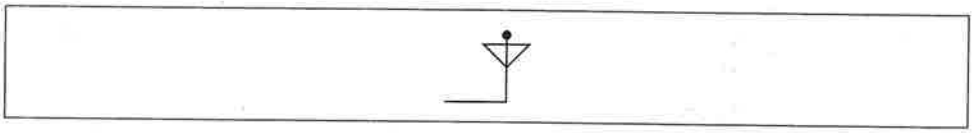

*Figure 9-2. Antenna representations in diagrams*

To function at all, an antenna must be made of conducting material. Radio waves hitting an antenna cause electrons to flow in the conductor and create a current. Likewise, applying a current to an antenna creates an electric field around the antenna. As the current to the antenna changes, so does the electric field. A changing electric field causes a magnetic field, and the wave is off.

The size of the antenna you need depends on the frequency: the higher the frequency, the smaller the antenna. The shortest simple antenna you can make at any frequency is 1/2 wavelength long (though antenna engineers can play tricks to reduce antenna size further). This rule of thumb accounts for the huge size of radio broadcast antennas and the small size of mobile phones. An AM station broadcasting at 830 kHz has a wavelength of about 360 meters and a correspondingly large antenna, but

an 802.11b network interface operating in the 2.4-GHz band has a wavelength of just 12.5 centimeters. With some engineering tricks, an antenna can be incorporated into a PC Card, and a more effective external antenna can easily be carried in a backpack.

Antennas can also be designed with directional preference. Many antennas are omnidirectional, which means they send and receive signals from any direction. Some applications may benefit from directional antennas, which radiate and receive on a narrower portion of the field. Figure 9-3 compares the radiated power of omnidirectional and directional antennas.

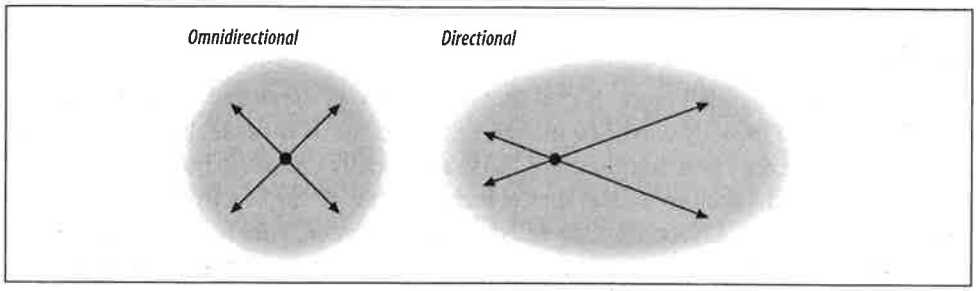


*Figure 9-3. Radiated power for omnidirectional and directional antennas*

For a given amount of input power, a directional antenna can reach farther with a clearer signal. They also have much higher sensitivity to radio signals in the dominant direction. When wireless links are used to replace wireline networks, directional antennas are often used. Mobile telephone network operators also use directional antennas when cells are subdivided. 802.11 networks typically use omnidirectional antennas for both ends of the connection, though there are exceptions—particularly if you want the network to span a longer distance. Also, keep in mind that there is no such thing as a truly omnidirectional antenna. We're accustomed to thinking of vertically mounted antennas as omnidirectional because the signal doesn't vary significantly as you travel around the antenna in a horizontal plane. But if you look at the signal radiated vertically (i.e., up or down) from the antenna, you'll find that it's a different story. And that part of the story can become important if you're building a network for a college or corporate campus and want to locate antennas on the top floors of your buildings.

Of all the components presented in this section, antennas are the most likely to be separated from the rest of the electronics. In this case, you need a transmission line (some kind of cable) between the antenna and the transceiver. Transmission lines usually have an impedance of 50 ohms.

In terms of practical antennas for 802.11 devices in the 2.4-GHz band, the typical wireless PC Card has an antenna built in. As you can probably guess, that antenna will do the job, but it's mediocre. Most vendors, if not all, sell an optional external

antenna that plugs into the card. These antennas are decent but not great, and they will significantly increase the range of a roaming laptop. You can usually buy some cable to separate the antenna from the PC Card, which can be useful for a base station. However, be careful—coaxial cable (especially small coaxial cable) is very lossy at these frequencies, so it's easy to imagine that anything you gain by better antenna placement will be lost in the cable. People have experimented with building high-gain antennas, some for portable use, some for base station use. And commercial antennas are available—some designed for 802.11 service, some adaptable if you know what you're doing.

### Amplifiers

Amplifiers make signals bigger. Signal boost, or *gain*, is measured in decibels (dB). Amplifiers can be broadly classified into three categories: low-noise, high-power, and everything else. Low-noise amplifiers (LNAs) are usually connected to an antenna to boost the received signal to a level that is recognizable by the electronics the RF system is connected to. LNAs are also rated for *noise factor*, which is the measure of how much extraneous information the amplifier introduces. Smaller noise factors allow the receiver to hear smaller signals and thus allow for a greater range.

High-power amplifiers (HPAs) are used to boost a signal to the maximum power possible before transmission. Output power is measured in dBm, which are related to watts (see the sidebar). Amplifiers are subject to the laws of thermodynamics, so they give off heat in addition to amplifying the signal. The transmitter in an 802.11 PC Card is necessarily low-power because it needs to run off a battery if it's installed in a laptop, but it's possible to install an external amplifier at fixed access points, which can be connected to the power grid where power is more plentiful.

This is where things can get tricky with respect to compliance with regulations. 802.11 devices are limited to one watt of power output and four watts effective radiated power (ERP). ERP multiplies the transmitter's power output by the gain of the antenna minus the loss in the transmission line. So if you have a 1-watt amplifier, an antenna that gives you 8 dB of gain, and 2 dB of transmission line loss, you have an ERP of 4 watts; the total system gain is 6 dB, which multiplies the transmitter's power by a factor of 4.

## RF Propagation

In fixed networks, signals are confined to wire pathways, so network engineers do not need to know anything about the physics of electrical signal propagation. Instead, there are a few rules used to calculate maximum segment length, and as long as the rules are obeyed, problems are rare. RF propagation is not anywhere near as simple.

## Decibels and Signal Strength

Amplifiers may boost signals by orders of magnitude. Rather than keep track of all those zeroes, amplifier power is measured in decibels (dB).

$$dB = 10 \times \log10 \text{ (power out/power in)}$$

Decibel ratings are positive when the output is larger than the input and negative when the output is smaller than the input. Each 10-dB change corresponds to a factor of 10, and 3-dB changes are a factor of 2. Thus, a 33-dB change corresponds to a factor of 2000:

$$33 \text{ dB} = 10 \text{ dB} + 10 \text{ dB} + 10 \text{ dB} + 3 \text{ dB} = 10 \times 10 \times 10 \times 2 = 2000$$

Power is sometimes measured in dBm, which stands for dB above one milliwatt. To find the dBm ratio, simply use 1 mW as the input power in the first equation.

It's helpful to remember that doubling the power is a 3-dB increase. A 1-dB increase is roughly equivalent to a power increase of 1.25. With these numbers in mind, you can quickly perform most gain calculations in your head.

### Multipath interference

One of the major problems that plague radio networks is multipath fading. Waves are added by superposition. When multiple waves converge on a point, the total wave is simply the sum of any component waves. Figure 9-4 shows a few examples of superposition.

In Figure 9-4c, the two waves are almost exactly the opposite of each other, so the net result is almost nothing. Unfortunately, this result is more common than you might expect in wireless networks. Most 802.11 equipment uses omnidirectional antennas, so RF energy is radiated in every direction. Waves spread outward from the transmitting antenna in all directions and are reflected by surfaces in the area. Figure 9-5 shows a highly simplified example of two stations in a rectangular area with no obstructions.

This figure shows three paths from the transmitter to the receiver. The wave at the receiver is the sum of all the different components. It is certainly possible that the paths shown in Figure 9-5 will all combine to give a net wave of 0, in which case the receiver will not understand the transmission because there is no transmission to be received.

Because the interference is a delayed copy of the same transmission on a different path, the phenomenon is called multipath fading or multipath interference. In many cases, multipath interference can be resolved by changing the orientation or position of the receiver.

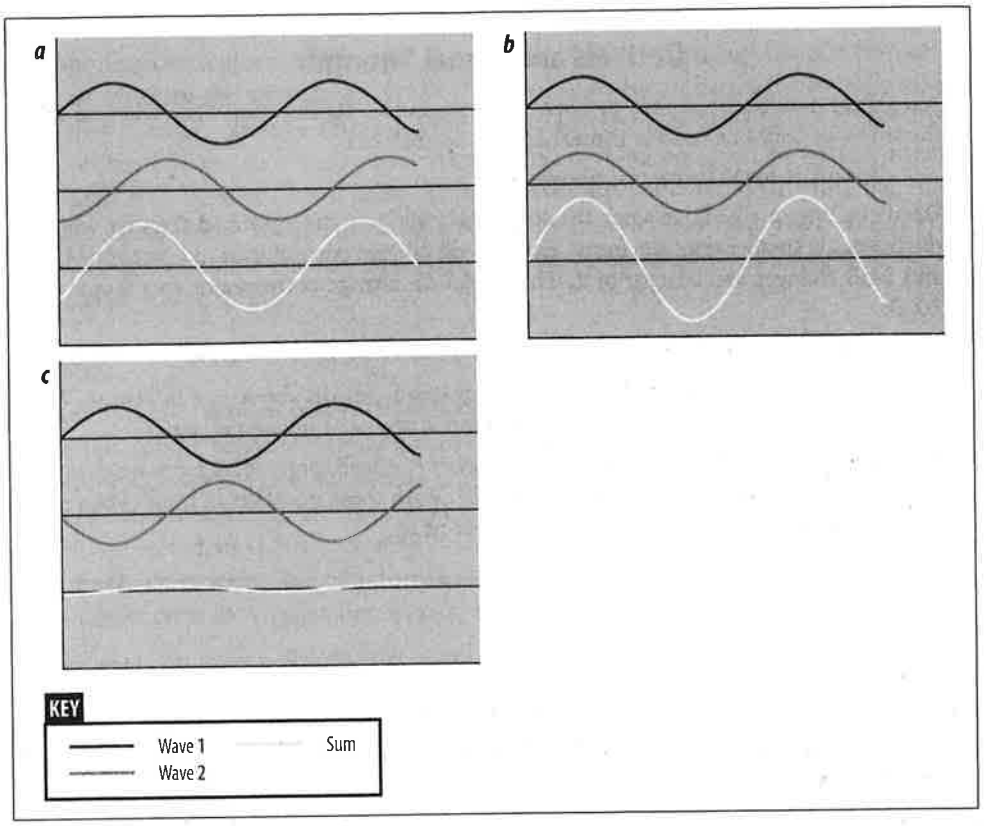


*Figure 9-4. Wave combination by superposition*

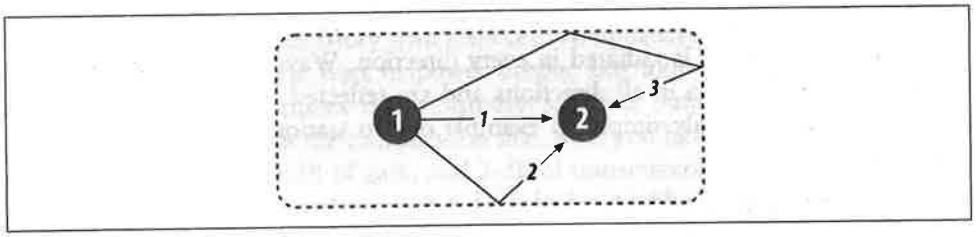


*Figure 9-5. Multiple paths*

### Inter-symbol interference (ISI)

Multipath fading is a special case of inter-symbol interference. Waves that take different paths from the transmitter to the receiver will travel different distances and be delayed with respect to each other, as in Figure 9-6. Once again, the two waves combine by superposition, but the effect is that the total waveform is garbled. In real-world situations, wavefronts from multiple paths may be added. The time between

the arrival of the first wavefront and the last multipath echo is called the *delay spread*. Longer delay spreads require more conservative coding mechanisms. 802.11b networks can handle delay spreads of up to 500 ns, but performance is much better when the delay spread is lower. When the delay spread is large, many cards will reduce the transmission rate; several vendors claim that a 65-ns delay spread is required for full-speed 11-Mbps performance at a reasonable frame error rate. A few wireless LAN analysis tools can directly measure delay spread.

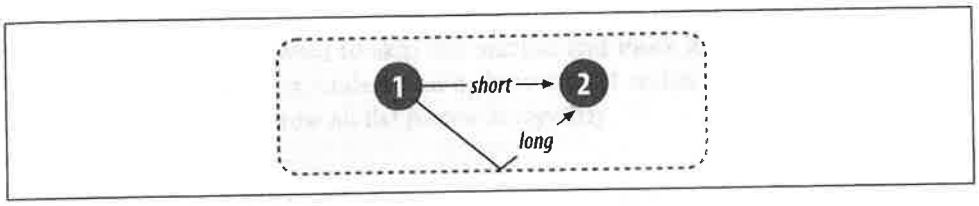


*Figure 9-6. Inter-symbol interference*

CHAPTER 10

# The ISM PHYs: FH, DS, and HR/DS

This chapter goes into detail about the physical layers specified by the 802.11 standards for use in the microwave ISM band at 2.4 GHz. As such, it is one of the most difficult, most interesting, and least useful chapters in the book. Feel free to skip it—everything you're likely to need to know about the physical layer is covered in Chapter 9. But if you want a challenge, or if you find the internals of wireless networks fascinating, read on.

The current version of the 802.11 standard specifies three physical layers in the 2.4-GHz ISM band:

*FH PHY*
: A low-rate, frequency-hopping layer

*DS PHY*
: A low-rate, direct-sequence layer

*HR/DSSS PHY*
: A high-rate, direct-sequence layer added by 802.11b

The last of these is probably the only one you'll see in use, particularly if you don't have a lot of older equipment. But if you're taking the trouble to dive through this chapter, you might as well see how the technology developed. There is one additional physical layer, a very high-speed layer standardized in 802.11a; it is discussed in Chapter 11. Standardization work has begun on a fourth physical layer for the 2.4-GHz ISM band that promises speeds of up to 54 Mbps.

## 802.11 FH PHY

Of all the physical layers standardized in the first draft of 802.11 in 1997, the frequency-hopping, spread-spectrum (FH or FHSS) layer was the first layer to see widespread deployment. The electronics used to support frequency-hopping modulation are relatively cheap and do not have high power requirements. Initially, the main

advantage to using frequency-hopping networks was that a greater number of networks could coexist, and the aggregate throughput of all the networks in a given area was high. With the advent of higher-rate, direct-sequence systems, the aggregate throughput advantage of frequency-hopping has been demolished.

This chapter describes the basic concepts and modulation techniques used by the FH PHY. It also shows how the physical layer convergence procedure prepares frames for transmission on the radio link and touches briefly on a few details about the physical medium itself. At this point, the FH PHY is largely a footnote in the history of 802.11, so you may want to skip this section and move ahead to the next section on the DS PHY. However, understanding how 802.11 technology developed will give you a better feeling for how all the pieces fit together.

## Frequency-Hopping Transmission

Frequency hopping depends on rapidly changing the transmission frequency in a predetermined, pseudorandom pattern, as illustrated in Figure 10-1. The vertical axis of the graph divides the available frequency into a number of slots. Likewise, time is divided into a series of slots. A hopping pattern controls how the slots are used. In the figure, the hopping pattern is {2,8,4,7}. Timing the hops accurately is the key to success; both the transmitter and receiver must be synchronized so the receiver is always listening on the transmitter's frequency.

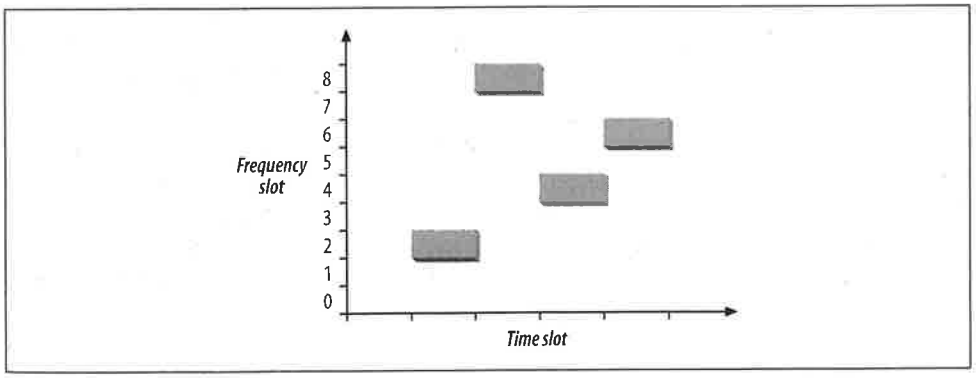


*Figure 10-1. Frequency hopping*

Frequency hopping is similar to frequency division multiple access (FDMA) but with an important twist. In FDMA systems, each device is allocated a fixed frequency. Multiple devices share the available radio spectrum by using different frequencies. In frequency-hopping systems, the frequency is time-dependent rather than fixed. Each frequency is used for a small amount of time, called the *dwell time*.

Among other things, frequency hopping allows devices to avoid interfering with primary users assigned to the same frequency band. It works because primary users are assigned narrow frequency bands and the right to transmit at a power high enough to override the wireless LAN. Any interference caused by the secondary user that affects the primary user is transient because the hopping sequence spreads the energy out over a wide band.* Likewise, the primary user only knocks out one of the spread-spectrum device's slots and looks like transient noise. Figure 10-2 shows the result when frequency slot 7 is given to a primary user. Although the transmission in the fourth time slot is corrupted, the first three transmissions succeed.

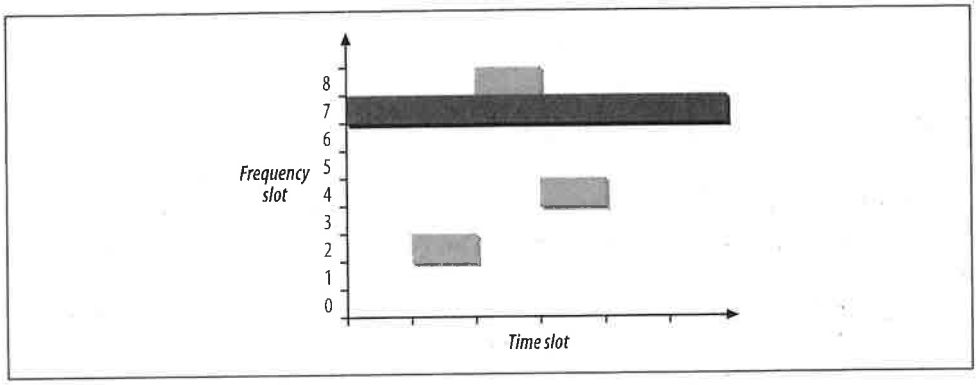


*Figure 10-2. Avoiding interference with frequency hopping*

If two frequency-hopping systems need to share the same band, they can be configured with different hopping sequences so they do not interfere with each other. During each time slot, the two hopping sequences must be on different frequency slots. As long as the systems stay on different frequency slots, they do not interfere with each other, as shown in Figure 10-3. The gray rectangles have a hopping sequence of {2,8,4,7}, as in the previous figures. A second system with a hopping sequence of {6,3,7,2} is added. Hopping sequences that do not overlap are called *orthogonal*. When multiple 802.11 networks are configured in a single area, orthogonal hopping sequences maximizes throughput.

### 802.11 FH details

802.11 divides the microwave ISM band into a series of 1-MHz channels. Approximately 99% of the radio energy is confined to the channel. The modulation method used by 802.11 encodes data bits as shifts in the transmission frequency from the channel center. Channels are defined by their center frequencies, which begin at 2.400 GHz for channel 0. Successive channels are derived by adding 1-MHz steps: channel 1

* If the primary user of a frequency band notices interference from secondary users, regulators can (and will) step in to shut down the secondary user, hence the low power used by spread-spectrum modulation techniques.

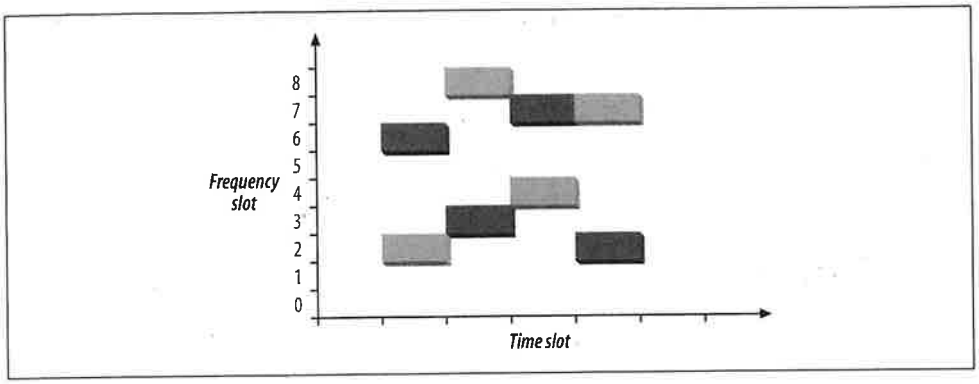


*Figure 10-3. Orthogonal hopping sequences*

has a center frequency of 2.401 GHz, channel 2 has a center frequency of 2.402 GHz, and so on up to channel 95 at 2.495 GHz. Different regulatory authorities allow use of different parts of the ISM band; the major regulatory domains and the available channels are shown in Table 10-1.

*Table 10-1. Channels used in different regulatory domains*

| Regulatory domain | Allowed channels |
|---|---|
| US (FCC) | 2 to 79 (2.402–2.479 GHz) |
| Canada (IC) | 2 to 79 (2.402–2.479 GHz) |
| Europe (excluding France and Spain) (ETSI) | 2 to 79 (2.402–2.479 GHz) |
| France | 48 to 82 (2.448–2.482 GHz) |
| Spain | 47 to 73 (2.447–2.473 GHz) |
| Japan (MKK) | 73 to 95 (2.473–2.495 GHz) |

The dwell time used by 802.11 FH systems is 390 time units, which is almost 0.4 seconds. When an 802.11 FH PHY hops between channels, the hopping process can take no longer than 224 microseconds. The frequency hops themselves are subject to extensive regulation, both in terms of the size of each hop and the rate at which hops must occur.

### 802.11 hop sequences

Mathematical functions for deriving hop sets are part of the FH PHY specification and are found in clause 14.6.8 of the 802.11 specification. As an example, hopping sequence 1 for North America and most of Europe begins with the sequence {3, 26, 65, 11, 46, 19, 74, 50, 22, ...}. 802.11 further divides hopping sequences into non-overlapping sets, and any two members of a set are orthogonal hopping sequences. In Europe and North America, each set contains 26 members. Regulatory authorities

in other areas have restricted the number of hopped channels, and therefore each set has a smaller number of members. Table 10-2 has details.

*Table 10-2. Size of hop sets in each regulatory domain*

| Regulatory domain | Hop set size |
|---|---|
| US (FCC) | 26 |
| Canada (IC) | 26 |
| Europe (excluding France and Spain) (ETSI) | 26 |
| France | 27 |
| Spain | 35 |
| Japan (MKK) | 23 |

### Joining an 802.11 frequency-hopping network

Joining a frequency-hopping network is made possible by the standardization of hop sequences. Beacon frames on FH networks include a timestamp and the FH Parameter Set element. The FH Parameter Set element includes the hop pattern number and a hop index. By receiving a Beacon frame, a station knows everything it needs to synchronize its hopping pattern.

Based on the hop sequence number, the station knows the channel-hopping order. As an example, say that a station has received a Beacon frame that indicates that the BSS is using the North America/Europe hop sequence number 1 and is at hop index 2. By looking up the hop sequence, the station can determine that the next channel is 65. Hop times are also well-defined. Each Beacon frame includes a Timestamp field, and the hop occurs when the timestamp modulo dwell time included in the Beacon is 0.

### ISM emission rules and maximum throughput

Spectrum allocation policies are the limiting factor of frequency-hopping 802.11 systems. As an example, consider the three major rules imposed by the FCC in the U.S.:*

1. There must be at least 75 hopping channels in the band, which is 83.5-MHz wide.
2. Hopping channels can be no wider than 1 MHz.
3. Devices must use all available channels equally. In a 30-second period, no more than 0.4 seconds may be spent using any one channel.

Of these rules, the most important is the second one. No matter what fancy encoding schemes are available, only 1 MHz of bandwidth is available at any time. The frequency

* These rules are in rule 247 of part 15 of the FCC rules (47 CFR 15.247).

at which it is available shifts continuously because of the other two rules, but the second rule limits the number of signal transitions that can be used to encode data.

With a straightforward, two-level encoding, each cycle can encode one bit. At 1 bit per cycle, 1 MHz yields a data rate of 1 Mbps. More sophisticated modulation and demodulation schemes can improve the data rate. Four-level coding can pack 2 bits into a cycle, and 2 Mbps can be squeezed from the 1-MHz bandwidth.

The European Telecommunications Standards Institute (ETSI) also has a set of rules for spread-spectrum devices in the ISM band, published in European Telecommunications Standard (ETS) 300-328. The ETSI rules allow far fewer hopping channels; only 20 are required. Radiated power, however, is controlled much more strictly. In practice, to meet both the FCC and ETSI requirements, devices use the high number of hopping channels required by the FCC with the low radiated power requirements of ETSI.

### Effect of interference

802.11 is a secondary use of the 2.4-GHz ISM band and must accept any interference from a higher-priority transmission. Catastrophic interference on a channel may prevent that channel from being used but leave other channels unaffected. With approximately 80 usable channels in the U.S. and Europe, interference on one channel reduces the raw bit rate of the medium by approximately 1.25%. (The cost at the IP layer will be somewhat higher because of the interframe gaps, 802.11 acknowledgments, and framing and physical-layer covergence headers.) As more channels are affected by interference, the throughput continues to drop. See Figure 10-4.

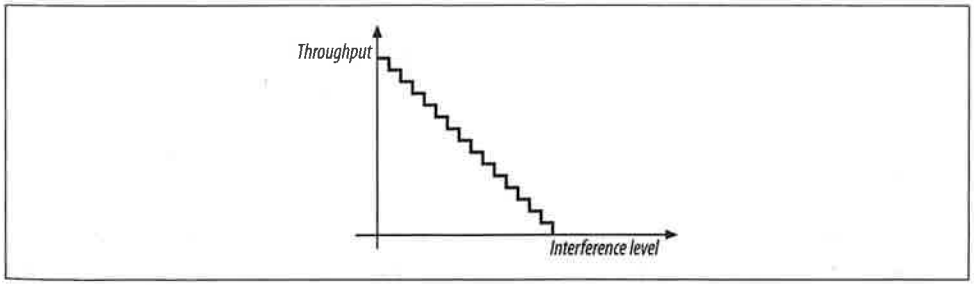


*Figure 10-4. Throughput response to interference in FHSS systems*

## Gaussian Frequency Shift Keying (GFSK)

The FH PHY uses Gaussian frequency shift keying (GFSK).* Frequency shift keying encodes data as a series of frequency changes in a carrier. One advantage of using

* The term *keying* is a vestige of telegraphy. Transmission of data across telegraph lines required the use of a key. Sending data through a modern digital system employs modulation techniques instead, but the word keying persists.

frequency to encode data is that noise usually changes the amplitude of a signal; modulation systems that ignore amplitude (broadcast FM radio, for example) tend to be relatively immune to noise. The *Gaussian* in GFSK refers to the shape of radio pulses; GFSK confines emissions to a relatively narrow spectral band and is thus appropriate for secondary uses. Signal processing techniques that prevent widespread leakage of RF energy are a good thing, particularly for secondary users of a frequency band. By reducing the potential for interference, GFSK makes it more likely that 802.11 wireless LANs can be built in an area where another user has priority.

### 2-Level GFSK

The most basic GFSK implementation is called 2-level GFSK (2GFSK). Two different frequencies are used, depending on whether the data that will be transmitted is a 1 or a 0. To transmit a 1, the carrier frequency is increased by a certain deviation. Zero is encoded by decreasing the frequency by the same deviation. Figure 10-5 illustrates the general procedure. In real-world systems, the frequency deviations from the carrier are much smaller; the figure is deliberately exaggerated to show how the encoding works.

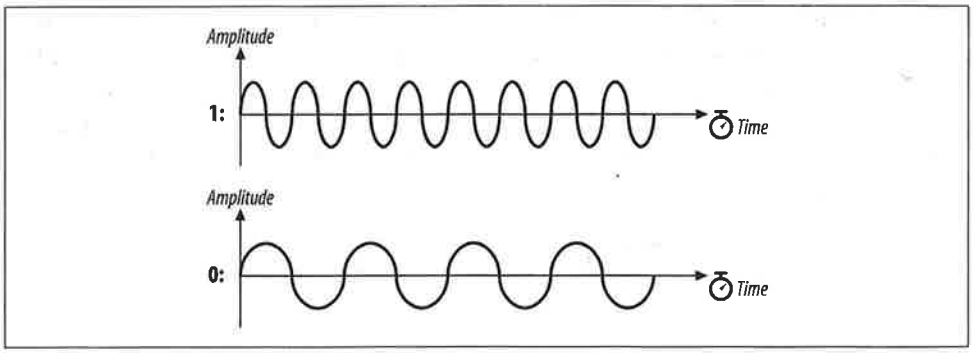


*Figure 10-5. 2-level GFSK*

The rate at which data is sent through the system is called the *symbol rate*. Because it takes several cycles to determine the frequency of the underlying carrier and whether 1 or 0 was transmitted, the symbol rate is a very small fraction of the carrier frequency. Although the carrier frequency is roughly 2.4 GHz, the symbol rate is only 1 or 2 million symbols per second.

Frequency changes with GFSK are not sharp changes. Instantaneous frequency changes require more expensive electronic components and higher power. Gradual frequency changes allow lower-cost equipment with lower RF leakage. Figure 10-6 shows how frequency varies as a result of encoding the letter M (1001101 binary)

using 2GFSK. Note that the vertical axis is the frequency of the transmission. When a 1 is transmitted, the frequency rises to the center frequency plus an offset, and when a 0 is transmitted, the frequency drops by the same offset. The horizontal axis, which represents time, is divided into symbol periods. Around the middle of each period, the receiver measures the frequency of the transmission and translates that frequency into a symbol. (In 802.11 frequency-hopping systems, the higher-level data is scrambled before transmission, so the bit sequence transmitted to the peer station is not the same as the bit sequence over the air. The figure illustrates how the principles of 2GFSK work and doesn't step through an actual encoding.)

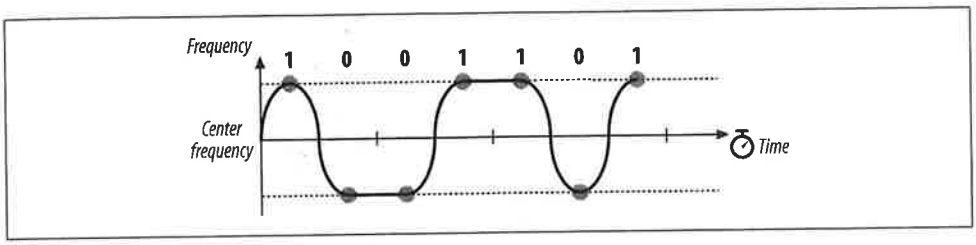


*Figure 10-6. 2GFSK encoding of the letter M*

### 4-Level GFSK

Using a scheme such as this, there are two ways to send more data: use a higher symbol rate or encode more bits of information into each symbol. 4-level GFSK (4GFSK) uses the same basic approach as 2GFSK but with four symbols instead of two. The four symbols (00, 01, 10, and 11) each correspond to a discrete frequency, and therefore 4GFSK transmits twice as much data at the same symbol rate. Obviously, this increase comes at a cost: 4GFSK requires more complex transmitters and receivers. Mapping of the four symbols onto bits is shown in Figure 10-7.

With its more sophisticated signal processing, 4GFSK packs multiple bits into a single symbol. Figure 10-8 shows how the letter M might be encoded. Once again, the vertical axis is frequency, and the horizontal axis is divided into symbol times. The frequency changes to transmit the symbols; the frequencies for each symbol are shown by the dashed lines. The figure also hints at the problem with extending GFSK-based methods to higher bit rates. Distinguishing between two levels is fairly easy. Four is harder. Each doubling of the bit rate requires that twice as many levels be present, and the RF components distinguish between ever smaller frequency changes. These limitations practically limit the FH PHY to 2 Mbps.

## FH PHY Convergence Procedure (PLCP)

Before any frames can be modulated onto the RF carrier, the frames from the MAC must be prepared by the Physical Layer Convergence Procedure (PLCP). Different

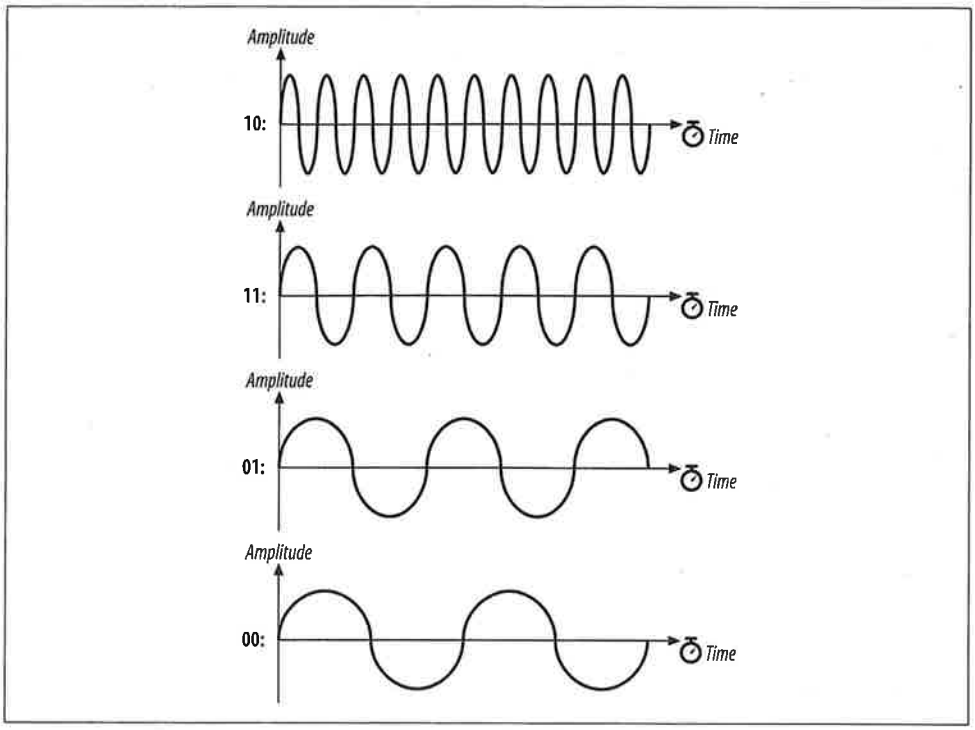


*Figure 10-7. Mapping of symbols to frequencies in 4GFSK*

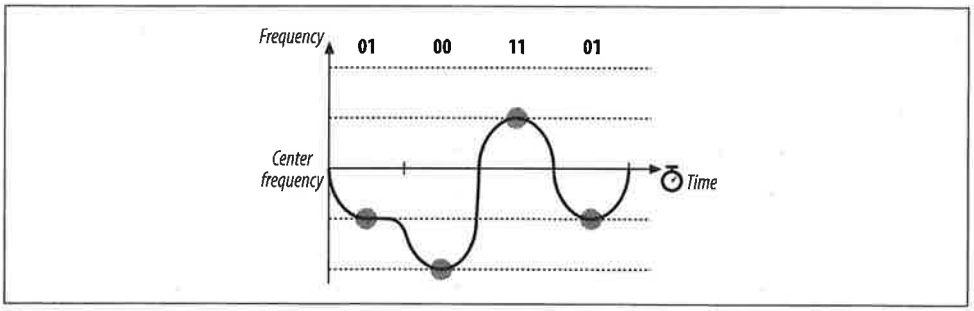


*Figure 10-8. 4GFSK encoding of the letter M*

underlying physical layers may have different requirements, so 802.11 allows each physical layer some latitude in preparing MAC frames for transmission over the air.

### Framing and whitening

The PLCP for the FH PHY adds a five-field header to the frame it receives from the MAC. The PLCP is a relay between the MAC and the physical medium dependent (PMD) radio interface. In keeping with ISO reference model terminology, frames

passed from the MAC are PLCP service data units (PSDUs). The PLCP framing is shown in Figure 10-9.

*Preamble*

As in a wired Ethernet, the preamble synchronizes the transmitter and receiver and derives common timing relationships. In the 802.11 FH PHY, the Preamble is composed of the Sync field and the Start Frame Delimiter field.

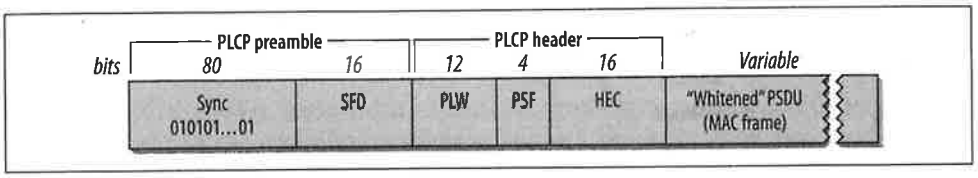


*Figure 10-9. PLCP framing in the FH PHY*

*Sync*

The sync field is 80 bits in length and is composed of an alternating zero-one sequence (010101...01). Stations search for the sync pattern to prepare to receive data. In addition to synchronizing the sender and receiver, the Sync field serves three purposes. Presence of a sync signal indicates that a frame is imminent. Second, stations that have multiple antennas to combat multipath fading or other environmental reception problems can select the antenna with the strongest signal. Finally, the receiver can measure the frequency of the incoming signal relative to its nominal values and perform any corrections needed to the received signal.

*Start Frame Delimiter (SFD)*

As in Ethernet, the SFD signals the end of the preamble and marks the beginning of the frame. The FH PHY uses a 16-bit SFD: 0000 1100 1011 1101.

*Header*

The PLCP header follows the preamble. The header has PHY-specific parameters used by the PLCP. Three fields comprise the header: a length field, a speed field, and a frame check sequence.

*PSDU Length Word (PLW)*

The first field in the PLCP header is the PLW. The payload of the PLCP frame is a MAC frame that may be up to 4,095 bytes long. The 12-bit length field informs the receiver of the length of the MAC frame that follows the PLCP header.

*PLCP Signaling (PSF)*

Bit 0, the first bit transmitted, is reserved and set to 0. Bits 1–3 encode the speed at which the payload MAC frame is transmitted. Several speeds are available, so this field allows the receiver to adjust to the appropriate demodulation scheme. Although the standard allows for data rates in increments of 500 kbps from 1.0

Mbps to 4.5 Mbps, the modulation scheme has been defined only for 1.0 Mbps and 2.0 Mbps.* See Table 10-3.

*Table 10-3. PSF meaning*

| Bits (1-2-3) | Data rate |
|---|---|
| 000 | 1.0 Mbps |
| 001 | 1.5 Mbps |
| 010 | 2.0 Mbps |
| 011 | 2.5 Mbps |
| 100 | 3.0 Mbps |
| 101 | 3.5 Mbps |
| 110 | 4.0 Mbps |
| 111 | 4.5 Mbps |

*Header Error Check (HEC)*
: To protect against errors in the PLCP header, a 16-bit CRC is calculated over the contents of the header and placed in this field. The header does not protect against errors in other parts of the frame.

No restrictions are placed on the content of the Data field. Arbitrary data may contain long strings of consecutive 0s or 1s, which makes the data much less random. To make the transmitted data more like random white noise, the FH PHYs apply a *whitening* algorithm to the MAC frame. This algorithm scrambles the data before radio transmission. Receivers invert the process to recover the data.

## Frequency-Hopping PMD Sublayer

Although the PLCP header has a field for the speed at which the MAC frame is transmitted, only two of these rates have corresponding standardized PMD layers. Several features are shared between both PMDs: antenna diversity support, allowances for the ramp up and ramp down of the power amplifiers in the antennas, and the use of a Gaussian pulse shaper to keep as much RF power as possible in the narrow frequency-hopping band. Figure 10-10 shows the general design of the transceiver used in 802.11 frequency-hopping networks.

### PMD for 1.0-Mbps FH PHY

The basic frequency-hopping PMD enables data transmission at 1.0 Mbps. Frames from the MAC have the PLCP header appended, and the resulting sequence of bits is

* It is unlikely that significant further work will be done on high-rate, frequency-hopping systems. For high data rates, direct sequence is a more cost-effective choice.

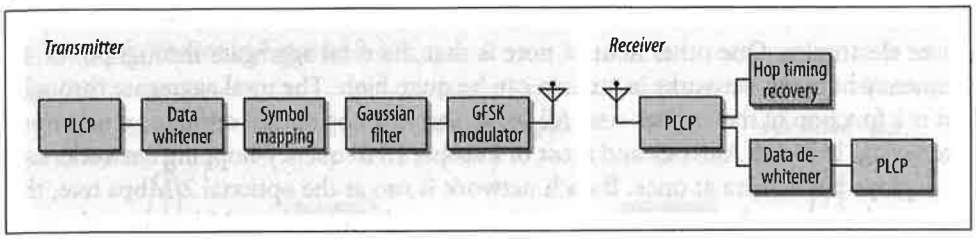


*Figure 10-10. Frequency-hopping transceiver*

transmitted out of the radio interface. In keeping with the common regulatory restriction of a 1-MHz bandwidth, 1 million symbols are transmitted per second. 2GFSK is used as the modulation scheme, so each symbol can be used to encode a single bit. 802.11 specifies a minimum power of 10 milliwatts (mW) and requires the use of a power control function to cap the radiated power at 100 mW, if necessary.

### PMD for 2.0-Mbps FH PHY

A second, higher-speed PMD is available for the FH PHY. As with the 1.0-Mbps PMD, the PLCP header is appended and is transmitted at 1.0 Mbps using 2GFSK. In the PLCP header, the PSF field indicates the speed at which the frame body is transmitted. At the higher data rate, the frame body is transmitted using a different encoding method than the physical-layer header. Regulatory requirements restrict all PMDs to a symbol rate of 1 MHz, so 4GFSK must be used for the frame body. Two bits per symbol yields a rate of 2.0 Mbps at 1 million symbols per second. Firmware that supports the 2.0-Mbps PMD can fall back to the 1.0-Mbps PMD if signal quality is too poor to sustain the higher rate.

### Carrier sense/clear channel assessment (CS/CCA)

To implement the CSMA/CA foundation of 802.11, the PCLP includes a function to determine whether the wireless medium is currently in use. The MAC uses both a virtual carrier-sense mechanism and a physical carrier-sense mechanism; the physical layer implements the physical carrier sense. 802.11 does not specify how to determine whether a signal is present; vendors are free to innovate within the required performance constraints of the standard. 802.11 requires that 802.11-compliant signals with certain power levels must be detected with a corresponding minimum probability.

## Characteristics of the FH PHY

Table 10-4 shows the values of a number of parameters in the FH PHY. In addition to the parameters in the table, which are standardized, the FH PHY has a number of parameters that can be adjusted to balance delays through various parts of an 802.11 frequency-hopping system. It includes variables for the latency through the MAC, the

PLCP, and the transceiver, as well as variables to account for variations in the transceiver electronics. One other item of note is that the total aggregate throughput of all frequency-hopping networks in an area can be quite high. The total aggregate throughput is a function of the hop set size. All sequences in a hop set are orthogonal and noninterfering. In North America and most of Europe, 26 frequency-hopping networks can be deployed in an area at once. If each network is run at the optional 2-Mbps rate, the area can have a total of 52-Mbps throughput provided that the ISM band is relatively free of interference.

*Table 10-4. FH PHY parameters*

| Parameter | Value | Notes |
|---|---|---|
| Slot time | 50 µs | |
| SIFS time | 28 µs | The SIFS is used to derive the value of the other interframe spaces (DIFS, PIFS, and EIFS). |
| Contention window size | 15–1,023 slots | |
| Preamble duration | 96 µs | Preamble symbols are transmitted at 1 MHz, so a symbol takes 1 sto transmit; 96 bits require 96 symbol times. |
| PLCP header duration | 32 µs | The PLCP header is 32 bits, so it requires 32 symbol times. |
| Maximum MAC frame | 4,095 bytes | 802.11 recommends a maximum of 400 symbols (400 bytes at 1 Mbps, 800 bytes at 2 Mbps) to retain performance across different types of environments. |

# 802.11 DS PHY

Direct-sequence modulation has been the most successful modulation technique used with 802.11. The initial 802.11 specification described a physical layer based on low-speed, direct-sequence spread spectrum (DS or DSSS). Direct-sequence equipment requires more power to achieve the same throughput as a frequency-hopping system. 2-Mbps direct-sequence interfaces will drain battery power more quickly than 2-Mbps frequency-hopping interfaces. The real advantage to direct-sequence transmission is that the technique is readily adaptable to much higher data rates than frequency-hopping networks.

This section describes the basic concepts and modulation techniques used by the initial DS PHY. It also shows how the PLCP prepares frames for transmission on the radio link and touches briefly on a few details of the physical medium itself.

## Direct-Sequence Transmission

Direct-sequence transmission is an alternative spread-spectrum technique that can be used to transmit a signal over a much wider frequency band. The basic approach of direct-sequence techniques is to smear the RF energy over a wide band in a carefully controlled way. Changes in the radio carrier are present across a wide band, and

receivers can perform correlation processes to look for changes. The basic high-level approach is shown in Figure 10-11.

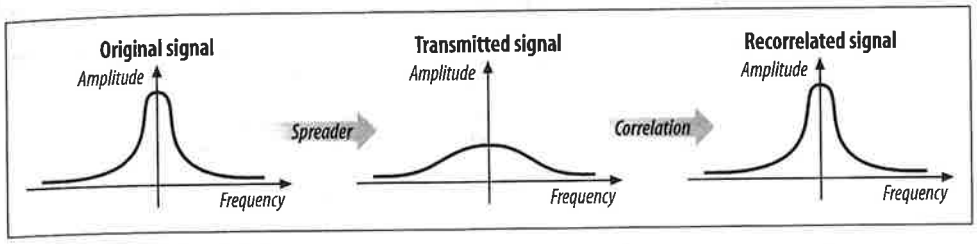


*Figure 10-11. Basic DSSS technique*

At the left is a traditional narrowband radio signal. It is processed by a *spreader*, which applies a mathematical transform to take a narrowband input and flatten the amplitude across a relatively wide frequency band. To a narrowband receiver, the transmitted signal looks like low-level noise because its RF energy is spread across a very wide band. The key to direct-sequence transmission is that any modulation of the RF carrier is also spread across the frequency band. Receivers can monitor a wide frequency band and look for changes that occur across the entire band. The original signal can be recovered with a *correlator*, which inverts the spreading process.

At a high level, a correlator simply looks for changes to the RF signal that occur across the entire frequency band. Correlation gives direct-sequence transmissions a great deal of protection against interference. Noise tends to take the form of relatively narrow pulses that, by definition, do not produce coherent effects across the entire frequency band. Therefore, the correlation function spreads out noise across the band, and the correlated signal shines through, as illustrated in Figure 10-12.

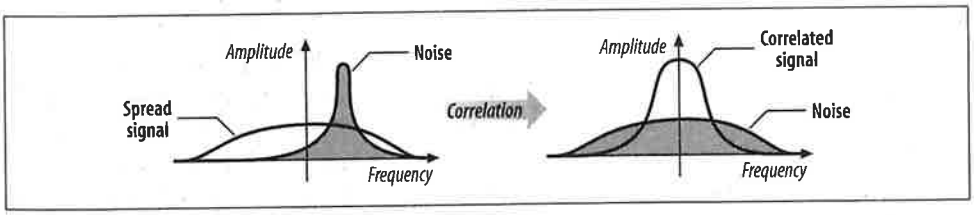


*Figure 10-12. Spreading of noise by the correlation process*

Direct-sequence modulation works by applying a chipping sequence to the data stream. A *chip* is a binary digit used by the spreading process. Bits are higher-level data, while chips are binary numbers used in the encoding process. There's no mathematical difference between a bit and a chip, but spread-spectrum developers have adopted this terminology to indicate that chips are only a part of the encoding and transmission process and do not carry any data. Chipping streams, which are also called pseudorandom noise codes (PN codes), must run at a much higher rate than the underlying data. Figure 10-13 illustrates how chipping sequences are used in the transmission of data using direct-sequence modulation. Several chips are used to

encode a single bit into a series of chips. The high-frequency chipped signal is transmitted on an RF carrier. At the other end, a correlator compares the received signal to the same PN sequence to determine if the encoded bit was a 0 or a 1.

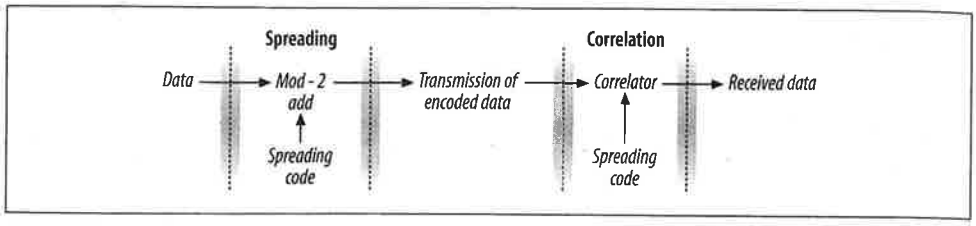


*Figure 10-13. Chipping*

The process of encoding a low bit rate signal at a high chip rate has the side effect of spreading the signal's power over a much wider bandwidth. One of the most important quantities in a direct-sequence system is its *spreading ratio*, which is the number of chips used to transmit a single bit.* Higher spreading ratios improve the ability to recover the transmitted signal but require a higher chipping rate and a larger frequency band. Doubling the spreading ratio requires doubling the chipping rate and doubles the required bandwidth as well. There are two costs to increased chipping ratios. One is the direct cost of more expensive RF components operating at the higher frequency, and the other is an indirect cost in the amount of bandwidth required. Therefore, in designing direct-sequence systems for the real world, the spreading ratio should be as low as possible to meet design requirements and to avoid wasting bandwidth.

Direct-sequence modulation trades bandwidth for throughput. Compared to traditional narrowband transmission, direct-sequence modulation requires significantly more radio spectrum and is much slower. However, it can often coexist with other interference sources because the receiver's correlation function effectively ignores narrowband noise. It is easier to achieve high throughput using direct-sequence techniques than with frequency hopping. Regulatory authorities do not impose a limit on the amount of spectrum that can be used; they generally set a minimum lower bound on the processing gain. Higher rates can be achieved with a wider band, though wider bands require a higher chip rate.

### 802.11 direct-sequence details

For the PN code, 802.11 adopted an 11-bit Barker word. Each bit is encoded using the entire Barker word as a chipping sequence. Detailed discussion of Barker words and their properties are well beyond the scope of this book. The key attribute for

* The spreading ratio is related to a figure known as the *processing gain*. The two are sometimes used interchangeably, but the processing gain is slightly lower because it takes into account the effects of using real-world systems as opposed to perfect ideal systems with no losses.

802.11 networks is that Barker words have good *autocorrelation* properties, which means that the correlation function at the receiver operates as expected in a wide range of environments and is relatively tolerant to multipath delay spreads.

Regulatory authorities require a 10-dB processing gain. Using an 11-bit spreading code for each bit allows 802.11 to meet the regulatory requirements with some margin of safety, but it is small enough to allow as many overlapping networks as possible. Longer spreading codes allow higher processing gains but require wider frequency channels.

### Encoding in 802.11 direct-sequence networks

802.11 uses the Barker sequence {+1, –1, +1, +1, –1, +1, +1, +1, –1, –1, –1}. As used in 802.11, +1 becomes 1, and –1 becomes 0, so the Barker sequence becomes 10110111000. It is applied to each bit in the data stream by a modulo-2 adder.* When a 1 is encoded, all the bits in the spreading code change; for 0, they stay the same. Figure 10-14 shows the encoding process.

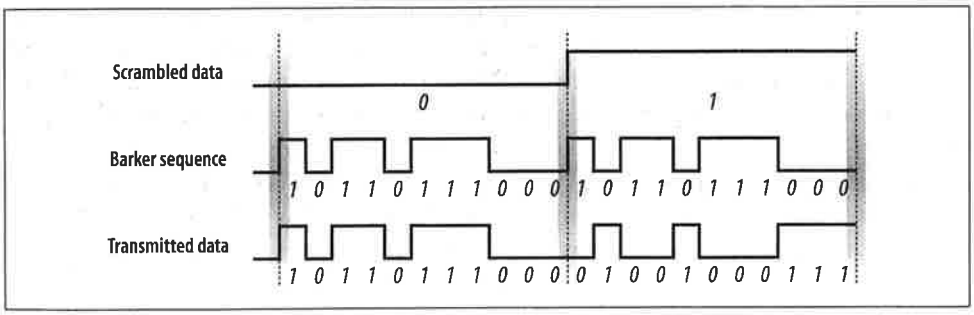


*Figure 10-14. Encoding with the Barker word*

Receivers can look at the number of 1s in a received bit time. The Barker sequence has six 1s and five 0s. An 11-bit sequence with six 1s must therefore correspond to a transmitted 0, and an 11-bit sequence with six 0s must correspond to a transmitted 1. In addition to counting the numbers of 1s and 0s, the receiver can analyze the pattern of received bits to infer the value of the transmitted bit.

### Operating channels

Channels for the DS PHY are much larger than the channels for the FH PHY. The DS PHY has 14 channels in the 2.4-GHz band, each 5 MHz wide. Channel 1 is placed at

* Encoding with the Barker sequence is similar to a number of other techniques. Some cellular systems, most notably in North America, use code division multiple access (CDMA) to allow several stations to access the radio medium. CDMA exploits some extremely complex mathematics to ensure that transmissions from each mobile phone look like random noise to every other mobile phone in the cell. The underlying mathematics are far more complicated than a simple fixed pseudo-random noise code.

2.412 GHz, channel 2 at 2.417 GHz, and so on up to channel 14 at 2.484 GHz. Table 10-5 shows which channels are allowed by each regulatory authority. Channel 10 is available throughout North America and Europe, which is why most products use channel 10 as the default operating channel.

*Table 10-5. Channels used in different regulatory domains*

| Regulatory domain | Allowed channels |
|---|---|
| US (FCC)/Canada (IC) | 1 to 11 (2.412–2.462 GHz) |
| Europe, excluding France and Spain (ETSI) | 1 to 13 (2.412–2.472 GHz) |
| France | 10 to 13 (2.457–2.472 GHz) |
| Spain | 10 to 11 (2.457–2.462 GHz) |
| Japan (MKK) | 14 (2.484 GHz) |

### Channel energy spread

Within a channel, most of the energy is spread across a 22-MHz band. Because the DS PHY uses an 11-MHz chip clock, energy spreads out from the channel center in multiples of 11 MHz, as shown in Figure 10-15. To prevent interference to adjacent channels, the first side lobe is filtered to 30 dB below the power at the channel center frequency, and additional lobes are filtered to 50 dB below the power at the channel center. This corresponds to reducing the power by a factor of 1,000 and 100,000, respectively. These limits are noted in Figure 10-15 by the use of dBr, which means dB relative to the power at the channel center. Figure 10-15 is not to scale: –30 dBr is only one thousandth, and –50 dBr is one hundred thousandth.

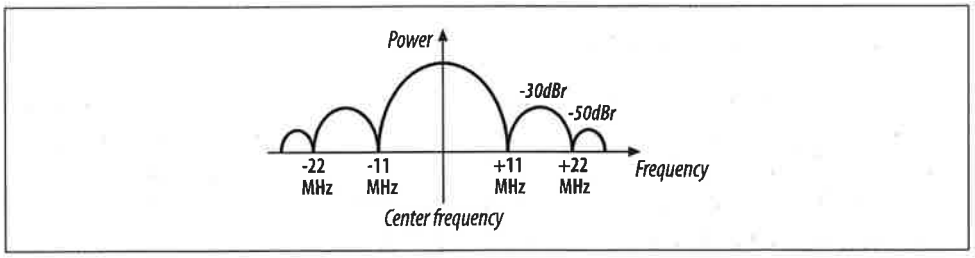


*Figure 10-15. Energy spread in a single 802.11 DS transmission channel*

With the transmit filters in place, RF power is confined mostly to 22-MHz frequency bands. European regulators cap the maximum radiated power at 100 mW; the FCC in the U.S. allows a substantially higher radiated power of 1,000 mW, but most products fall far below this in practice.

To prevent interference from networks operating on adjacent channels, 802.11 direct-sequence equipment must be separated by a frequency band of at least 22 MHz between channel center frequencies. With a channel spacing of 5 MHz, networks must be separated by five channel numbers to prevent interference, as illustrated in

Figure 10-16. If directly adjacent channels were selected, there would be a great deal of overlap in the center lobes.

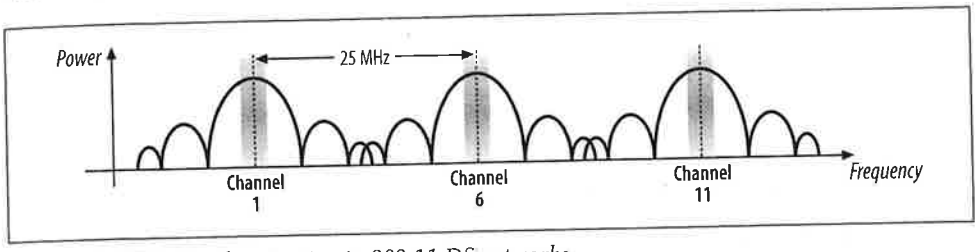


*Figure 10-16. Channel separation in 802.11 DS networks*

### Maximum theoretical throughput

If the signal processing techniques used by the DS PHY are used, then the maximum throughput would be a function of the frequency space used. Roughly speaking, the ISM band is 80-MHz wide. Using the same spreading factor of 11 would lead to a maximum bit rate of slightly more than 7 Mbps. However, only one channel would be available, and products would need to have an oscillator running at 77 MHz to generate the chipping sequence. High-frequency devices are a tremendous drain on batteries, and the hypothetical high-rate encoding that uses the entire band makes terrible use of the available spectrum. To achieve higher throughput, more sophisticated techniques must be used. 802.11b increases the symbol rate slightly, but it gets far more mileage from more sophisticated encoding techniques.

### Interference response

Direct-sequence–modulated signals are more resistant to interference than frequency-hopping signals. The correlation process enables direct-sequence systems to work around narrowband interference much more effectively. With 11 chips per bit, several chips can be lost or damaged before a single data bit is lost. The disadvantage is that the response of direct-sequence systems to noise is not incremental. Up to a certain level, the correlator can remove noise, but once interference obscures a certain amount of the frequency band, nothing can be recovered. Figure 10-17 shows how direct-sequence systems degrade in response to noise.

Direct-sequence systems also avoid interfering with a primary user more effectively than frequency-hopping systems. After direct-sequence processing, signals are much wider and have lower amplitudes, so they appear to be random background noise to traditional narrowband receivers. Two direct-sequence users in the same area can cause problems for each other quite easily if the two direct-sequence channels are not

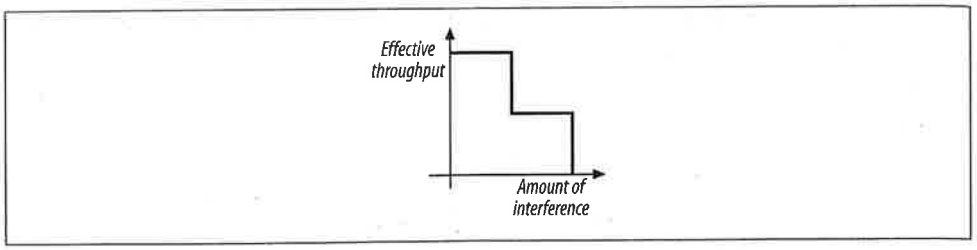


*Figure 10-17. Throughput response to interference in DSSS systems*

separated by an adequate amount. Generally speaking, interference between two direct-sequence devices is a problem long before a primary band user notices anything.

## Differential Phase Shift Keying (DPSK)

Differential phase shift keying (DPSK) is the basis for all 802.11 direct-sequence systems. As the name implies, phase shift keying (PSK) encodes data in phase changes of the transmitted signal. The absolute phase of a waveform is not relevant in PSK; only changes in the phase encode data. Like frequency shift keying, PSK resists interference because most interference causes changes in amplitude. Figure 10-18 shows two identical sine waves shifted by a small amount along the time axis. The offset between the same point on two waves is the phase difference.

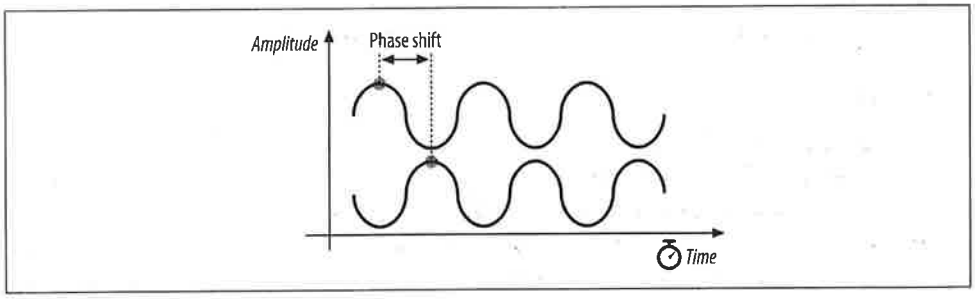


*Figure 10-18. Phase difference between two sine waves*

### Differential binary phase shift keying (DBPSK)

The simplest form of PSK uses two carrier waves, shifted by a half cycle relative to each other. One wave, the reference wave, is used to encode a 0; the half-cycle shifted wave is used to encode a 1. Table 10-6 summarizes the phase shifts, and Figure 10-19 illustrates the encoding as a phase difference from a preceding sine wave.

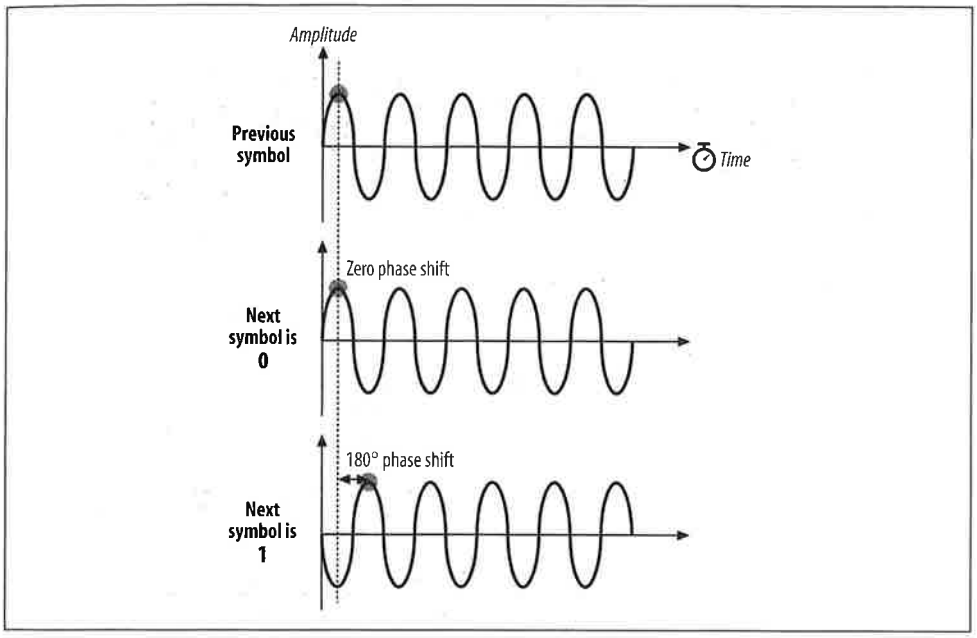


*Figure 10-19. DBPSK encoding*

*Table 10-6. DBPSK phase shifts*

| Symbol | Phase shift |
|---|---|
| 0 | 0 |
| 1 | 180° ($\pi$ radians) |

To stick with the same example, encoding the letter M (1001101 in binary) is a matter of dividing up the time into seven symbol times then transmitting the wave with appropriate phase shift at each symbol boundary. Figure 10-20 illustrates the encoding. Time is divided into a series of symbol periods, each of which is several times the period of the carrier wave. When the symbol is a 0, there is no change from the phase of the previous symbol, and when the symbol is a 1, there is a change of half a cycle. These changes result in "pinches" of the carrier when 1 is transmitted and a smooth transition across the symbol time boundary for 0.

### Differential quadrature phase shift keying (DQPSK)

Like 2GFSK, DBPSK is limited to one bit per symbol. More advanced receivers and transmitters can encode multiple bits per symbol using a technique called differential quadrature phase shift yeying (DQPSK). Rather than a fundamental wave and a half-cycle shifted wave, DQPSK uses a fundamental wave and three additional waves, each shifted by a quarter cycle, as shown in Figure 10-21. Table 10-7 summarizes the phase shifts.

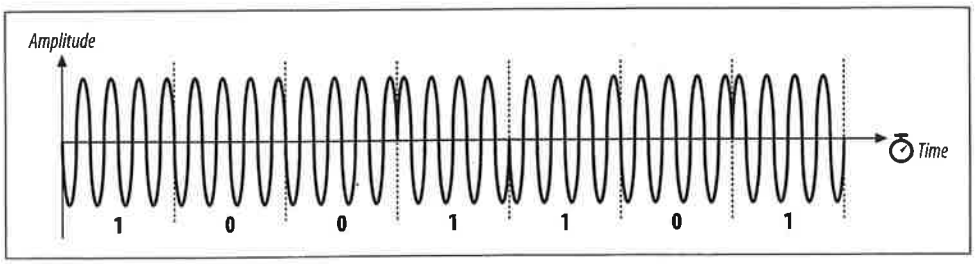


*Figure 10-20. The letter M encoded in DBPSK*

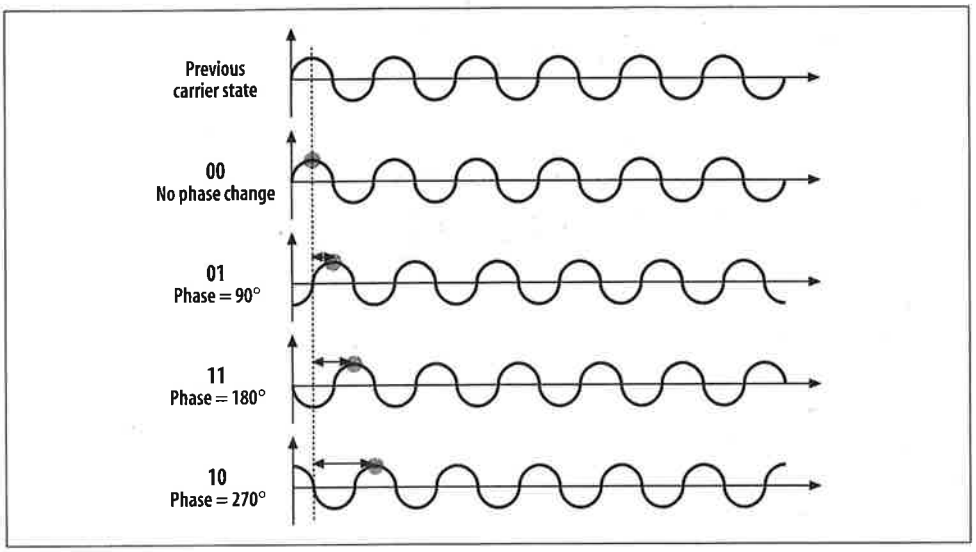


*Figure 10-21. DQPSK encoding*

*Table 10-7. DQPSK phase shifts*

| Symbol | Phase shift |
|---|---|
| 00 | 0 |
| 01 | 90° ($\pi/2$ radians) |
| 11 | 180° ($\pi$ radians) |
| 10 | 270° ($3\pi/2$ or $-\pi/2$ radians) |

Now encode M in DQPSK (Figure 10-22). In the UTF-8 character set, M is represented by the binary string 01001101 or, as the sequence of four two-bit symbols, 01-00-11-01. In the first symbol period, there is a phase shift of 90 degrees; for clarity, the figure shows the phase shift from a pure sine wave. The second symbol results in no phase shift, so the wave continues without a change. The third symbol causes a phase shift of 180 degrees, as shown by the sharp change from the highest amplitude to the lowest amplitude. The final symbol causes a phase shift of 90 degrees.

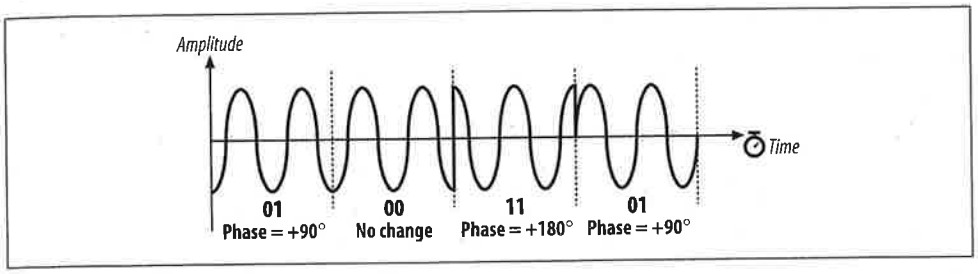


*Figure 10-22. The letter M encoded in DQPSK*

The obvious advantage of DQPSK relative to DBPSK is that the four-level encoding mechanism can have a higher throughput. The cost of using DQPSK is that it cannot be used in some environments because of severe multipath interference. Multipath interference occurs when the signal takes several paths from the transmitter to the receiver. Each path has a different length; therefore, the received signal from each path has a different delay relative to the other paths. This delay is the enemy of an encoding scheme based on phase shifts. Wavefronts are not labeled or painted different colors, so a wavefront could arrive later than expected because of a long path or it could simply have been transmitted late and phase shifted. In environments where multipath interference is severe, DQPSK will break down much quicker than DBPSK.

## DS Physical-Layer Convergence (PLCP)

As in the FH PHY, frames must be processed by the PLCP before being transmitted into the air.

### Framing and scrambling

The PLCP for the DS PHY adds a six-field header to the frames it receives from the MAC. In keeping with ISO reference model terminology, frames passed from the MAC are PLCP service data units (PSDUs). The PLCP framing is shown in Figure 10-23.

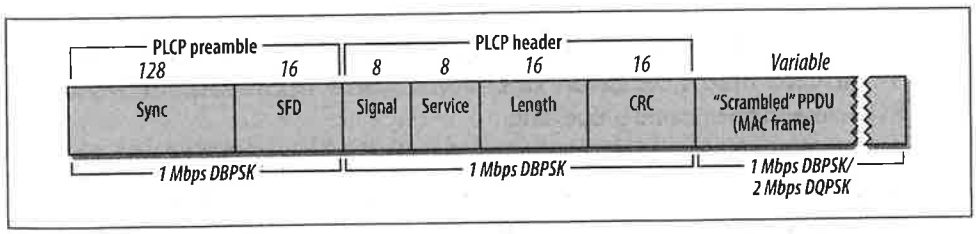


*Figure 10-23. DS PLCP framing*

The FH PHY uses a data whitener to randomize the data before transmission, but the data whitener applies only to the MAC frame trailing the PLCP header. The DS PHY

has a similar function called the *scrambler*, but the scrambler is applied to the entirety of the direct-sequence frame, including the PLCP header and preamble.

*Preamble*
: The Preamble synchronizes the transmitter and receiver and allows them to derive common timing relationships. It is composed of the Sync field and the Start Frame Delimiter field. Before transmission, the preamble is scrambled using the direct-sequence scrambling function.

*Sync*
: The Sync field is a 128-bit field composed entirely of 1s. Unlike the FH PHY, the Sync field is scrambled before transmission.

*Start Frame Delimiter (SFD)*
: The SFD allows the receiver to find the start of the frame, even if some of the sync bits were lost in transit. This field is set to 0000 0101 1100 1111, which is different from the SFD used by the FH PHY.

*Header*
: The PLCP header follows the preamble. The header has PHY-specific parameters used by the PLCP. Five fields comprise the header: a signaling field, a service identification field, a Length field, a Signal field used to encode the speed, and a frame check sequence.

*Signal*
: The Signal field is used by the receiver to identify the transmission rate of the encapsulated MAC frame. It is set to either 0000 1010 (0x0A) for 1-Mbps operation or 0001 0100 (0x14) for 2-Mbps operation.

*Service*
: This field is reserved for future use and must be set to all 0s.

*Length*
: This field is set to the number of microseconds required to transmit the frame as an unsigned 16-bit integer, transmitted least significant bit to most significant bit.

*CRC*
: To protect the header against corruption on the radio link, the sender calculates a 16-bit CRC over the contents of the four header fields. Receivers verify the CRC before further frame processing.

No restrictions are placed on the content of the Data field. Arbitrary data may contain long strings of consecutive 0s or 1s, which makes the data much less random. To make the data more like random background noise, the DS PHY uses a polynomial scrambling mechanism to remove long strings of 1s or 0s from the transmitted data stream.

## DS Physical Medium Dependent Sublayer

Unlike the FH PHY, the DS PHY uses a single PMD specification. This is a complex and lengthy specification that incorporates provisions for two data rates (1.0 and 2.0 Mbps). Figure 10-24 shows the general design of a transceiver for 802.11 direct-sequence networks.

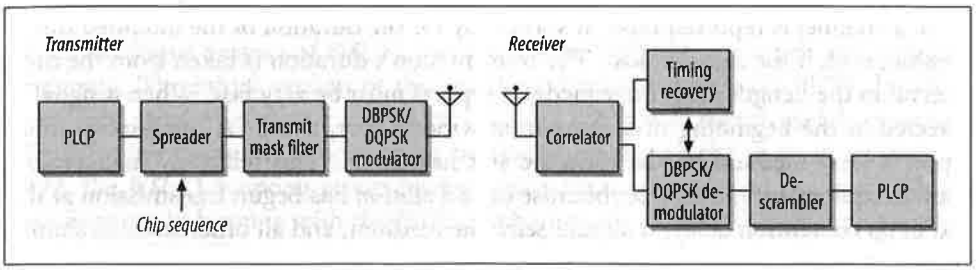


*Figure 10-24. Direct-sequence transceiver*

### Transmission at 1.0 Mbps

At the low data rate, the direct-sequence PMD enables data transmission at 1.0 Mbps. The PLCP header is appended to frames arriving from the MAC, and the entire unit is scrambled. The resulting sequence of bits is transmitted from the physical interface using DBPSK at a rate of 1 million symbols per second. The resulting throughput is 1.0 Mbps because one bit is encoded per symbol. Like the FH PMD, the DS PMD has a minimum power requirement and can cap the power at 100 mW if necessary to meet regulatory requirements.

### Transmission at 2.0 Mbps

Like the FH PHY, transmission at 2.0 Mbps uses two encoding schemes. The PLCP preamble and header are transmitted at 1.0 Mbps using DBPSK. Although using a slower method for the header transmission reduces the effective throughput, DBPSK is far more tolerant of noise and multipath interference. After the preamble and header are finished, the PMD switches to DQPSK modulation to provide 2.0-Mbps service. As with the FH PHY, most products that implement the 2.0-Mbps rate can detect interference and fall back to lower-speed 1.0-Mbps service.

### CS/CCA for the DS PHY

802.11 allows the CS/CCA function to operate in one of three modes:

*Mode 1*

When the energy exceeds the energy detection (ED) threshold, it reports that the medium is busy. The ED threshold depends on the transmit power.

*Mode 2*

Implementations using Mode 2 must look for an actual DSSS signal and report the channel busy when one is detected, even if the signal is below the ED threshold.

*Mode 3*

Mode 3 combines Mode 1 and Mode 2. A signal must be detected with sufficient energy before the channel is reported busy to higher layers.

Once a channel is reported busy, it stays busy for the duration of the intended transmission, even if the signal is lost. The transmission's duration is taken from the time interval in the Length field. Busy medium reports must be very fast. When a signal is detected at the beginning of a contention window slot, the CCA mechanism must report a busy medium by the time the slot has ended. This relatively high performance requirement must be set because once a station has begun transmission at the end of its contention delay, it should seize the medium, and all other stations should defer access until its frame has concluded.

## Characteristics of the DS PHY

Table 10-8 shows the values of a number of parameters in the DS PHY. In addition to the parameters in the table, which are standardized, the DS PHY has a number of parameters that can be adjusted to balance delays through various parts of an 802.11 direct-sequence system. It includes variables for the latency through the MAC, the PLCP, and the transceiver, as well as variables to account for variations in the transceiver electronics. One other item of note is that the total aggregate throughput of all direct-sequence networks in an area is much lower than the total aggregate throughput of all nonoverlapping frequency-hopping networks in an area. The total aggregate throughput is a function of the number of nonoverlapping channels. In North America and most of Europe, three direct-sequence networks can be deployed in an area at once. If each network is run at the optional 2-Mbps rate, the area can have a total of 6-Mbps throughput, which is dramatically less than the frequency-hopping total aggregate throughput.

*Table 10-8. DS PHY parameters*

| Parameter | Value | Notes |
|---|---|---|
| Slot time | 20 μs | |
| SIFS time | 10 μs | The SIFS is used to derive the value of the other interframe spaces (DIFS, PIFS, and EIFS). |
| Contention window size | 31 to 1,023 slots | |
| Preamble duration | 144 μs | Preamble symbols are transmitted at 1 MHz, so a symbol takes 1 s to transmit; 144 bits require 144 symbol times. |
| PLCP header duration | 48 μs | The PLCP header is 48 bits, so it requires 48 symbol times. |
| Maximum MAC frame | 4–8,191 bytes | |

Like the FH PHY, the DS PHY has a number of attributes that can be adjusted by a vendor to balance delays in various parts of the system. It includes variables for the latency through the MAC, the PLCP, and the transceiver, as well as variables to account for variations in the transceiver electronics.

# 802.11b: HR/DSSS PHY

When the initial version of 802.11 was ratified in 1997, the real work was only just beginning. The initial version of the standard defined FH and DS PHYs, but they were only capable of data rates up to 2 Mbps. 2 Mbps is barely useful, especially when the transmission capacity must be shared among all the users in an area. In 1999, the 802.11 working group released its second extension to the basic 802.11 specification. In keeping with the IEEE numbering convention, the second extension was labeled 802.11b.

802.11b adds another physical layer into the mix. It uses the same MAC as all the other physical layers and is based on direct-sequence modulation. However, it enables transmission at up to 11 Mbps, which is adequate for modern networks. Higher data rates led to a stunning commercial success. 802.11b has blazed new trails where other wireless technologies failed to make an impact. The 802.11b PHY is also known as the high-rate, direct-sequence PHY, abbreviated HR/DS or HR/DSSS. Even though the modulation is different, the operating channels are exactly the same as the channels used by the original low-rate direct sequence.

## Complementary Code Keying

802.11 direct-sequence systems use a rate of 11 million chips per second. The original DS PHYs divided the chip stream up into a series of 11-bit Barker words and transmitted 1 million Barker words per second. Each word encoded either one bit or two bits for a corresponding data rate of 1.0 Mbps or 2.0 Mbps, respectively. Achieving higher data rates and commercial utility requires that each code symbol carry more information than a bit or two.

Straight phase shift encoding cannot hope to carry more than a few bits per code word. DQPSK requires that receivers distinguish quarter-cycle phase differences. Further increasing the number of bits per symbol would require processing even finer phase shifts, such as an eighth-cycle or sixteenth-cycle shift. Detecting smaller phase shifts is more difficult in the presence of multipath interference and requires more sophisticated (and thus expensive) electronics.

Instead of continuing with straight phase-shift keying, the IEEE 802.11 working group turned to an alternate encoding method. Complementary code keying (CCK) divides the chip stream into a series of 8-bit code symbols, so the underlying transmission is based on a series of 1.375 million code symbols per second. CCK is based on sophisticated mathematical transforms that allow the use of a few 8-bit sequences

to encode 4 or even 8 bits per code word, for a data throughput of 5.5 Mbps or 11 Mbps. In addition, the mathematics underlying CCK transforms allow receivers to distinguish between different codes easily, even in the presence of interference and multipath fading. Figure 10-25 illustrates the use of code symbols in CCK. It is quite similar to the chipping process used by the slower direct-sequence layers; the difference is that the code words are derived partially from the data. A static repeating code word such as the Barker word is not used.

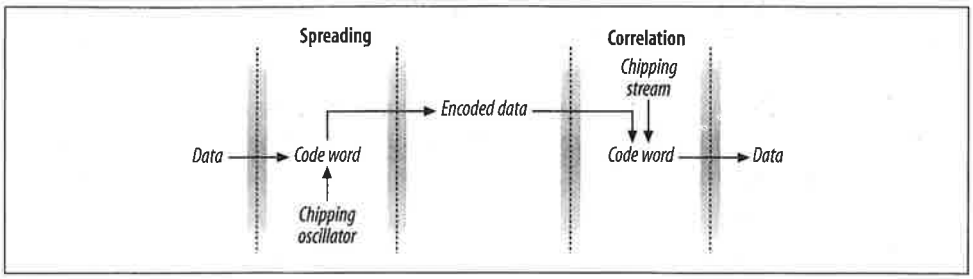


*Figure 10-25. Code symbols in CCK*

Barker spreading, as used in the lower-rate, direct-sequence layers, uses a static code to spread the signal over the available frequency band. CCK uses the code word to carry information, as well as simply to spread the signal. Several phase angles are used to prepare a complex code word of eight bits.

## High-Rate, Direct-Sequence PLCP

Like the other physical layers, the HR/DSSS PHY is split into two parts. As with the other physical layers, the PLCP adds additional framing information.

### Framing and scrambling

Unlike the other physical layers, two options exist for the PLCP framing. Both are shown in Figure 10-26. The "long" frame format is identical to the classic DS PLCP format and must be supported. For efficiency and improved throughput, stations may also support the optional "short" PLCP format.

Naturally, the optional short format may be used only if all stations support it. To prevent networks configured for the short format from disappearing, 802.11b requires that stations answering Probe Requests from an active scan return a response using the same PLCP header that was received. If a station that supports only the long PLCP header sends a Probe Response, an access point returns a response using the long header, even if the BSS is configured for the short header.

*Preamble*
: Frames begin with the preamble, which is composed of the Sync field and the SFD field. The preamble is transmitted at 1.0 Mbps using DBPSK.

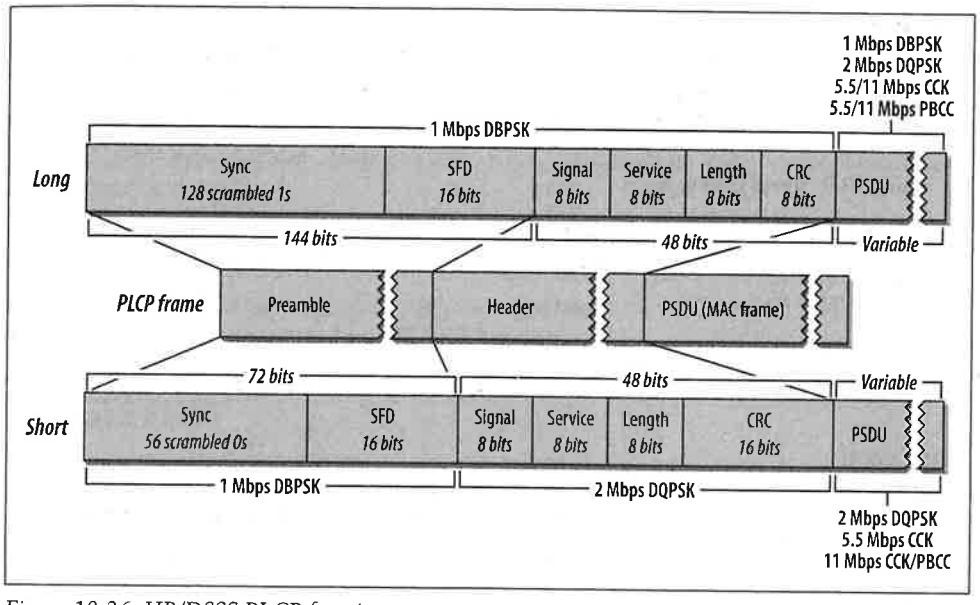


*Figure 10-26. HR/DSSS PLCP framing*

*Long Sync*
: The Long Sync field is composed of 128 1 bits. It is processed by the scrambler before transmission, though, so the data content varies. High-rate systems use a specified seed for the scrambling function but support backwards compatibility with older systems that do not specify a seed.

*Short Sync*
: The Short Sync field is composed of 56 0 bits. Like the Long Sync, it is also processed by the scrambler.

*Long SFD*
: To indicate the end of the Sync field, the long preamble concludes with a Start of Frame Delimiter (SFD). In the long PLCP, the SFD is the sequence 1111 0011 1010 0000. As with all IEEE specifications, the order of transmission from the physical interface is least-significant bit first, so the string is transmitted right to left.

*Short SFD*
: To avoid confusion with the Long SFD, the Short SFD is the reverse value, 0000 0101 1100 1111.

The PLCP header follows the preamble. It is composed of the Signal, Service, Length, and CRC fields. The long header is transmitted at 1.0 Mbps using DBPSK. However, the short header's purpose is to reduce the time required for overhead transmission so it is transmitted at 2.0 Mbps using DQPSK.

*Long Signal*

The Long Signal field indicates the speed and transmission method of the enclosed MAC frame. Four values for the 8-bit code are currently defined and are shown in Table 10-9.

*Table 10-9. Signal field values*

| Speed | Value (msb to lsb) | Hex value |
|---|---|---|
| 1 Mbps | 0000 1010 | 0x0A |
| 2 Mbps | 0001 0100 | 0x14 |
| 5.5 Mbps | 0011 0111 | 0x37 |
| 11 Mbps | 0110 1110 | 0x6E |

*Short Signal*

The Short Signal field indicates the speed and transmission method of the enclosed frame, but only three values are defined. Short preambles can be used only with 2 Mbps, 5.5 Mbps, and 11 Mbps networks.

*Service*

The Service field, which is shown in Figure 10-27, was reserved for future use by the first version of 802.11, and bits were promptly used for the high-rate extensions in 802.11b. First of all, the Length field describes the amount of time used for the enclosed frame in microseconds. Above 8 Mbps, the value becomes ambiguous. Therefore, the eighth bit of the service field is used to extend the Length field to 17 bits. The third bit indicates whether the 802.11b implementation uses locked clocks; clock locking means that transmit frequency and symbol clock use the same oscillator. The fourth bit indicates the type of coding used for the packet, which is either 0 for CCK or 1 for PBCC. All reserved bits must be set to 0. The Service field is transmitted from left to right (b0 to b7), which is the same in both the short and long PLCP frame formats.

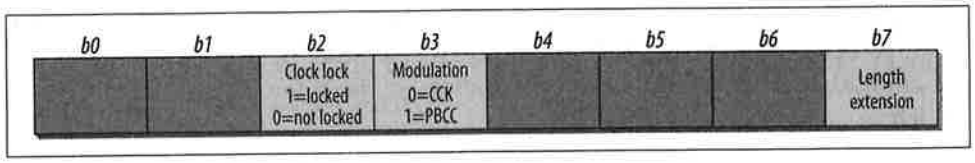


*Figure 10-27. Service field in the HR/DSSS PLCP header*

*Length*

The Length field is the same in both the short and long PLCP frame formats and is the number of microseconds required to transmit the enclosed MAC frame. Approximately two pages of the 802.11b standard are devoted to calculating the value of the Length frame, but the details are beyond the scope of this book.

*CRC*

The CRC field is the same in both the short and the long PLCP frames. Senders calculate a CRC checksum using the Signal, Service, and Length fields. Receivers can use the CRC value to ensure that the header was received intact and was not damaged during transmission. CRC calculations take place before data scrambling.

The data scrambling procedure for the HR/DSSS PHY is nearly identical to the data scrambling procedure used with the original DS PHY. The only difference is that the scrambling function is seeded to specified values in the HR/DSSS PHY. Different seeds are used for short and long PLCP frames.

## HR/DSSS PMD

Unlike the FH PHY, the DS PHY uses a single PMD specification. The general transceiver design is shown in Figure 10-28.

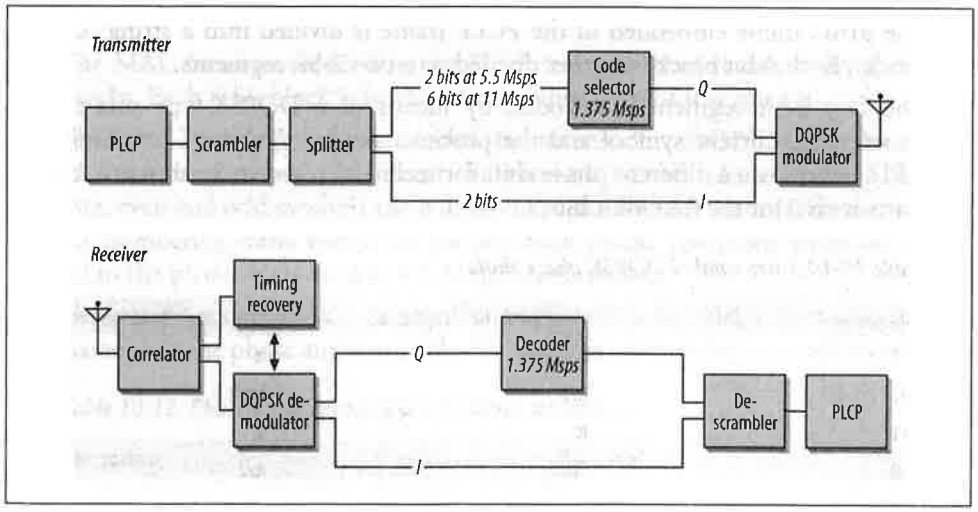


*Figure 10-28. HR/DSSS transceiver*

### Transmission at 1.0 Mbps or 2.0 Mbps

To ensure backwards compatibility with the installed base of 802.11-based, direct-sequence hardware, the HR/DSSS PHY can transmit and receive at 1.0 Mbps or 2.0 Mbps. Slower transmissions are supported in the same manner as the lower-rate, direct-sequence layers described in Chapter 9.

### Transmission at 5.5 Mbps with CCK

Higher-rate transmission is accomplished by building on the DQPSK-based phase shift keying techniques. DQPSK transmits two bits per symbol period, encoded as

one of four different phase shifts. By using CCK, the symbol words themselves carry additional information. 5.5-Mbps transmission encodes four data bits into a symbol. Two are carried using conventional DQPSK, and the other two are carried through the content of the code words. Figure 10-29 illustrates the overall process.

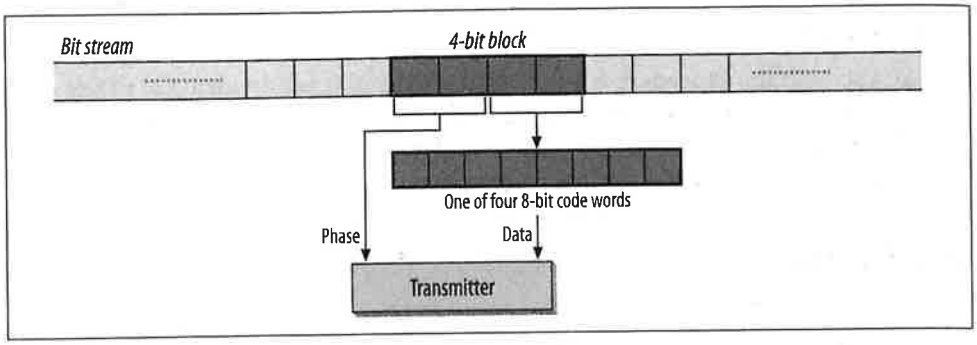


*Figure 10-29. 802.11b transmission at 5.5 Mbps*

1. The MAC frame embedded in the PCLP frame is divided into a string of 4-bit blocks. Each 4-bit block is further divided into two 2-bit segments.
2. The first 2-bit segment is encoded by means of a DQPSK-type phase shift between the current symbol and the previous symbol (Table 10-10). Even and odd symbols use a different phase shift for technical reasons. Symbol numbering starts with 0 for the first 4-bit block.

*Table 10-10. Inter-symbol DQPSK phase shifts*

| Bit pattern | Phase angle (even symbols) | Phase angle (odd symbols) |
|---|---|---|
| 00 | 0 | $\pi$ |
| 01 | $\pi/2$ | $3\pi/2$ |
| 11 | $\pi$ | 0 |
| 10 | $3\pi/2$ | $\pi/2$ |

3. The second 2-bit segment is used to select one of four code words for the current symbol (Table 10-11). The four code words can be derived using the mathematics laid out in clause 18.4.6.5 of the 802.11 standard.

*Table 10-11. Mbps code words*

| Bit sequence | Code word |
|---|---|
| 00 | i,1,i,–1,i,1,–i,1 |
| 01 | –i,–1,–i,1,1,1,–i,1 |
| 10 | –i,1,–i,–1,–i,1,i,1 |
| 11 | i,–1,i,1,–i,1,i,1 |

### Transmission at 11 Mbps with CCK

To move to a full 11 Mbps, 8 bits must be encoded with each symbol. As with other techniques, the first two bits are encoded by the phase shift of the transmitted symbol relative to the previous symbol. Six bits are encoded using CCK. Figure 10-30 illustrates the process.

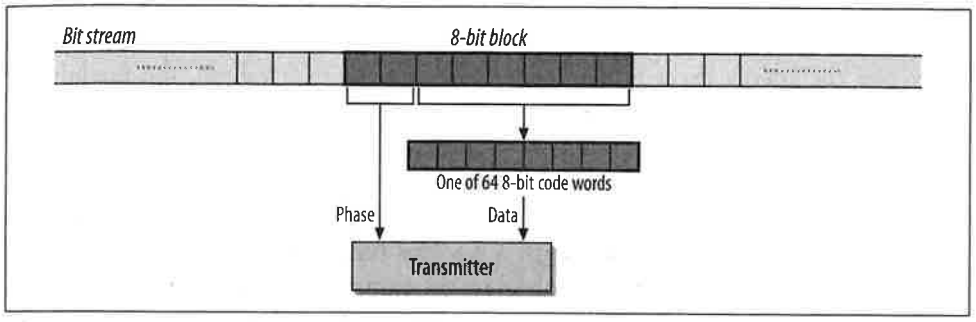


*Figure 10-30. 802.11b transmission at 11 Mbps*

1. The MAC frame embedded in the PCLP frame is divided into a string of 8-bit blocks. Each 8-bit block is further divided into four 2-bit segments.
2. The first 2-bit segment is encoded by means of a DQPSK-type phase shift between the current symbol and the previous symbol. As with the 5.5-Mbps rate, even and odd symbols use a different phase shift for technical reasons. Symbol numbering starts with 0 for the first 8-bit block. The phase shifts are identical to the phase shifts used in 5.5-Mbps transmission.
3. The remaining six bits are grouped into three successive pairs. Each pair is associated with the phase angle in Table 10-12 and is used to derive a code word.

*Table 10-12. Phase angle encoding for 11-Mbps transmission*

| Bit pattern | Phase angle |
|---|---|
| 00 | 0 |
| 01 | $\pi/2$ |
| 10 | $\pi$ |
| 11 | $3\pi/2$ |

As an example, consider the conversion of the bit sequence 0100 1101 into a complex code for transmission on an 802.11b network. The first two bits, 01, encode a phase shift from the previous symbol. If the symbol is an even symbol in the MAC frame, the phase shift is $\pi/2$; otherwise, the shift is $3\pi/2$. (Symbols in the MAC frame are numbered starting from 0, so the first symbol in a frame is even.) The last six bits are divided into three 2-bit groups: 00, 11, and 01. Each of these is used to encode an angle in the code word equation. The next step in transmission is to convert the phase angles into the complex code word for transmission.

### Clear channel assessment

Like the original DS PHY, high-rate implementers have three choices for the CS/CCA operation mode. All the direct-sequence CCA modes are considered to be part of the same list. Mode 1 is identical to the DS PHY's CCA Mode 1, and Modes 2 and 3 are used exclusively by the original DS PHY. Modes 4 and 5 are the HR/DSSS-specific CCA modes.

*Mode 1*
: When the energy exceeds the energy detection (ED) threshold, the medium is reported busy. The ED threshold depends on the transmit power used. This mode is also available for classic direct-sequence systems.

*Mode 4*
: Implementations using Mode 4 look for an actual signal. When triggered, a Mode 4 CCA implementation starts a 3.65 ms timer and begins counting down. If no valid HR/DSSS signal is received by the expiration of the timer, the medium is reported idle. 3.65 ms corresponds to the transmission time required for the largest possible frame at 5.5 Mbps.

*Mode 5*
: Mode 5 combines Mode 1 and Mode 4. A signal must be detected with sufficient energy before the channel is reported busy to higher layers.

Once a channel is reported busy, it stays busy for the duration of the intended transmission, even if the signal is lost. The channel is considered busy until the time interval in the Length field has elapsed. Implementations that look for a valid signal may override this requirement if a second PLCP header is detected.

## Optional Features of the 802.11b PHY

802.11b includes two optional physical-layer features.

### Packet Binary Convolutional Coding (PBCC)

PBCC is an optional coding method that has not been widely implemented. Proposals for further revisions to wireless LAN technology in the ISM band specified PBCC, but those proposals were rejected in the summer of 2001.

### Channel agility

To avoid interfering with existing 802.11 networks based on frequency-hopping technology, 802.11b includes the *channel agility* option. When employing the channel agility option, 802.11b networks periodically hop to a different channel. Three direct-sequence channels are used for nonoverlapping networks; the hop sequences and dwell times are designed to avoid interfering with a frequency-hopping network deployed in the same area.

## Characteristics of the HR/DSSS PHY

Table 10-13 shows the values of a number of parameters in the HR/DSSS PHY. Like the DS PHY, the HR/DSSS PHY has a number of parameters that can be adjusted to compensate for delays in any part of a real system.

*Table 10-13. HR/DSSS PHY parameters*

| Parameter | Value | Notes |
|---|---|---|
| Maximum MAC frame length | 4,095 bytes | |
| Slot time | 20 µs | |
| SIFS time | 10 µs | The SIFS is used to derive the value of the other interframe spaces (DIFS, PIFS, and EIFS). |
| Contention window size | 31 to 1,023 slots | |
| Preamble duration | 144 µs | Preamble symbols are transmitted at 1 MHz, so a symbol takes 1 s to transmit; 96 bits require 96 symbol times. |
| PLCP header duration | 48 bits | The PLCP header transmission time depends on whether the short preamble is used. |

One other item of note is that the total aggregate throughput of all HR/DSSS networks in an area is still lower than the total aggregate throughput of all nonoverlapping frequency-hopping networks in an area. The total aggregate throughput is a function of the number of nonoverlapping channels. In North America and most of Europe, three HR/DSSS networks can be deployed in an area at once. If each network is run at the optional 11-Mbps rate, the area can have a total of 33-Mbps throughput, which is less than the frequency-hopping total aggregate throughput.

# CHAPTER 11

# 802.11a: 5-GHz OFDM PHY

Many of the wireless devices that are currently on the market use the 2.4-GHz ISM band, which is rapidly becoming crowded. In an attempt to attain higher data rates and avoid overcrowding, the 802.11 working group is looking at the unlicensed bands around 5 GHz. In the U.S., the 5-GHz bands that are reserved for unlicensed use are designated as the Unlicensed National Information Infrastructure (U-NII). The U-NII bands provide more spectrum space than the 2.4-GHz bands and are much less heavily used; there are very few devices on the market that operate in these bands. The 802.11a working group is responsible for developing physical layers for high-rate wireless service on the 5-GHz bands.

802.11a hardware hit the market in late 2001. There are three main chipset vendors: Intersil, the developer of the widely used PRISM chipsets for the 2.4-GHz ISM band; the ever acquisitive Cisco, through its acquisition of chip-maker Radiata in late 2000; and Atheros Communications, a start-up founded by several notable Stanford researchers. 802.11a products hit the market about the time this book went to press, so I have not evaluated any vendor claims about increased range, throughput, or performance. Most of the products on the market look very much like the current 802.11b products: many are PC Cards and have similar configuration and installation routines.

In spite of its many advantages, 802.11a does not look like a sure-fire commercial success. Higher frequencies have higher path losses; some observers estimate that access points will have to be deployed much more densely than in an 802.11b network; some observers have estimated that one and a half times as many (or more) access points may be required. Most of the testing on 802.11a has been conducted by vendors in offices dominated by cubicles. How 11a systems will fare in other environments can be answered only by more deployment and testing. Higher-frequency RF systems require higher power, which may mean that 802.11a cards are not well-suited for mobile battery-powered systems. Many of the questions swirling around 802.11a can be answered only by the delivery of commercial products to the marketplace in

volume. The 802.11a standard as it currently stands is only for the U.S., but a task group is working on extending the standard to other regulatory domains.

This chapter begins with a qualitative introduction to the basis of OFDM. When all the mathematical formalism is stripped away, OFDM is a method for chopping a large frequency channel into a number of subchannels. The subchannels are then used in parallel for higher throughput. I anticipate that many readers will skip the first section, either because they are already familiar with OFDM or are interested only in how the frequency bands are used and how the PCLP wraps up frames for transmission.

# Orthogonal Frequency Division Multiplexing (OFDM)

802.11a is based on *orthogonal frequency division multiplexing* (OFDM). OFDM is not a new technique. Most of the fundamental work was done in the late 1960s, and U.S. patent number 3,488,445 was issued in January 1970. Recent DSL work (HDSL, VDSL, and ADSL) and wireless data applications have rekindled interest in OFDM, especially now that better signal-processing techniques make it more practical.* OFDM does, however, differ from other emerging encoding techniques such as code division multiple access (CDMA) in its approach. CDMA uses complex mathematical transforms to put multiple transmissions onto a single carrier; OFDM encodes a single transmission into multiple subcarriers. The mathematics underlying the code division in CDMA is far more complicated than in OFDM.

OFDM devices use one wide frequency channel by breaking it up into several component subchannels. Each subchannel is used to transmit data. All the "slow" subchannels are then multiplexed into one "fast" combined channel.

## Carrier Multiplexing

When network managers solicit user input on network build-outs, one of the most common demands is for more speed. The hunger for increased data transmissions has driven a host of technologies to increase speed. OFDM takes a qualitatively similar approach to Multilink PPP: when one link isn't enough, use several in parallel.

OFDM is closely related to plain old frequency division multiplexing (FDM). Both divide the available bandwidth into slices called *carriers* or *subcarriers* and make those carriers available as distinct channels for data transmission. OFDM boosts

* The lack of interest in OFDM means that references on it are sparse. Readers interested in the mathematical background that is omitted in this chapter should consult *OFDM for Wireless Multimedia Applications* by Richard van Nee and Ramjee Prasad (Artech House, 2000).

throughput by using several subcarriers in parallel and multiplexing data over the set of subcarriers.

Traditional FDM was widely used by first-generation mobile telephones as a method for radio channel allocation. Each user was given an exclusive channel, and guard bands were used to ensure that spectral leakage from one user did not cause problems for users of adjacent channels. Figure 11-1 illustrates the traditional FDM approach.

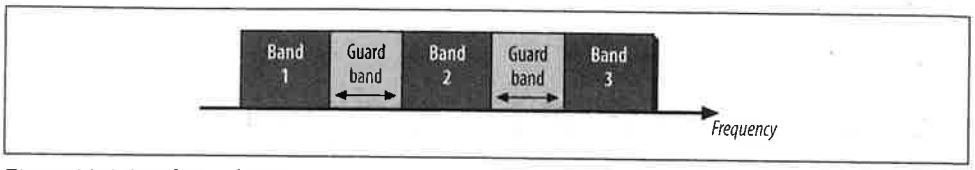


*Figure 11-1. Traditional FDM*

The trouble with traditional FDM is that the guard bands waste bandwidth and thus reduce capacity. To wasting transmission capacity with unused guard bands, OFDM selects channels that overlap but do not interfere with each other. Figure 11-2 illustrates the contrast between traditional FDM and OFDM.

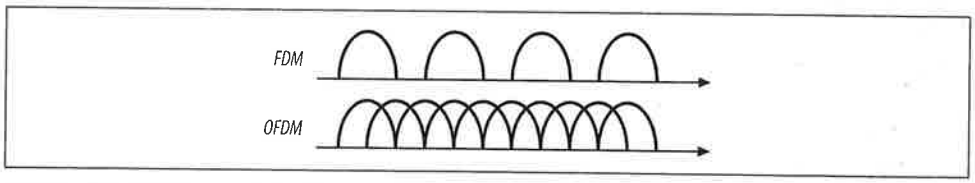


*Figure 11-2. FDM versus OFDM*

Overlapping carriers are allowed because the subcarriers are defined so that they are easily distinguished from one another. The ability to separate the subcarriers hinges on a complex mathematical relationship called *orthogonality*.

## Orthogonality Explained (Without Calculus)

*Orthogonal* is a mathematical term derived from the Greek word *orthos*, meaning straight, right, or true. In mathematics, the word "orthogonal" is used to describe independent items. Orthogonality is best seen in the frequency domain, looking at a spectral breakdown of a signal. OFDM works because the frequencies of the subcarriers are selected so that at each subcarrier frequency, all other subcarriers do not contribute to the overall waveform. One common way of looking at orthogonality is shown in Figure 11-3. The signal has been divided into its three subcarriers. The peak of each subcarrier, shown by the heavy dot at the top, encodes data. The subcarrier

set is carefully designed to be orthogonal; note that at the peak of each of the subcarriers, the other two subcarriers have zero amplitude.

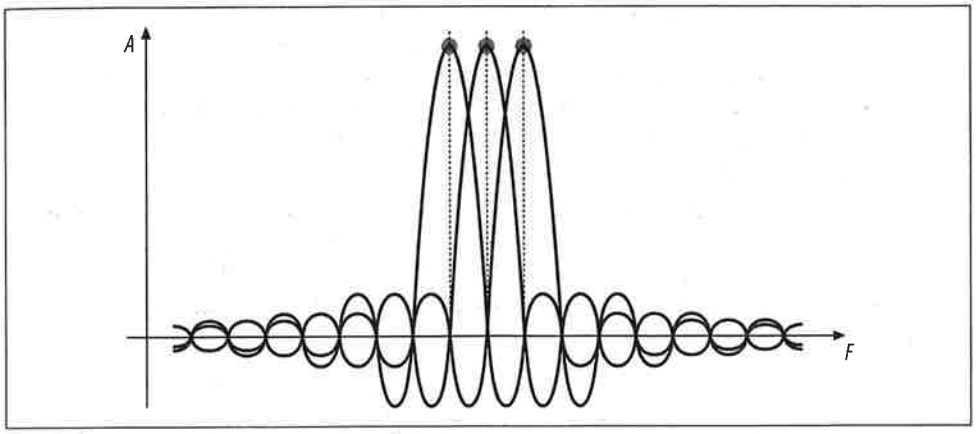


*Figure 11-3. Orthogonality in the frequency domain*

OFDM takes the coded signal for each subchannel and uses the inverse fast Fourier transform (IFFT) to create a composite waveform from the strength of each subchannel. OFDM receivers can then apply the FFT to a received waveform to extract the amplitude of each component subcarrier.

## Guard Time

With the physical layers discussed in Chapter 10, the main problem for receivers was inter-symbol interference (ISI) (Figure 11-4). ISI occurs when the delay spread between different paths is large and causes a delayed copy of the transmitted bits to shift onto a previously arrived copy.

With OFDM, inter-symbol interference does not pose the same kind of problem. The Fourier transform used by OFDM distills the received waveform into the strengths of the subcarriers, so time shifts do not cause dramatic problems. In Figure 11-4, there would be a strong peak for the fundamental low-frequency carrier, and the late-arriving high-frequency component could be ignored.

As with all benefits, however, there is a price to pay. OFDM systems use multiple subcarriers of different frequencies. The subcarriers are packed tightly into an operating channel, and small shifts in subcarrier frequencies may cause interference between carriers, a phenomenon called *inter-carrier interference* (ICI). Frequency shifts may occur because of the Doppler effect or because there is a slight difference between the transmitter and receiver clock frequencies.

To address both ISI and ICI, OFDM transceivers reserve the beginning portion of the symbol time as the *guard time* and perform the Fourier transform only on the

## Fourier Analysis, the Fourier Transform, and Signal Processing

The Fourier transform is often called "the Swiss Army knife of signal processing." Signal processing often defines actions in terms of frequency components. Receivers, however, process a time-varying signal amplitude. The *Fourier transform* is a mathematical operation that divides a waveform into its component parts. Fourier analysis takes a time-varying signal and converts it to the set of frequency-domain components that make up the signal.

Signal-processing applications often need to perform the reverse operation as well. Given a set of frequency components, these applications use them like a recipe to build the composite waveform with these frequency components. The mathematical operation used to build the composite waveform from the known ingredients in the frequency domain is the *inverse Fourier transform*.

Strictly speaking, Fourier analysis is applied to smooth curves of the sort found in physics textbooks. To work with a set of discrete data points, a relative of Fourier transform called the *discrete Fourier transform* (DFT) must be used. Like the Fourier transform, the DFT has its inverted partner, the inverse DFT (IDFT).

The DFT is a computationally intensive process of order $N^2$, which means that its running time is proportional to the square of the size of the number of data points. If the number of data points is an even power of two, however, several computational shortcuts can be taken to cut the complexity to order N log N. On large data sets, the reduced complexity boosts the speed of the algorithm. As a result, the "short" DFT applied to $2^n$ data points is called the fast Fourier transform (FFT). It also has an inverted relative, the inverse fast Fourier transform (IFFT).

Fast Fourier transforms used to be the domain of supercomputers or special-purpose, signal-processing hardware. But with the microprocessor speeds available today, sophisticated signal processing is well within the capabilities of a PC. Specialized digital signal processors (DSPs) are now cheap enough to be used in almost anything—including the chip sets on commodity 802.11 cards.

non–guard time portion of the symbol time. The non–guard time portion of the symbol is often called the *FFT integration time* because the Fourier transform is performed only on that portion of the symbol.

Delays shorter than the guard time do not cause ICI because they do not allow frequency components to leak into successive symbol times. Selecting the guard time is a major task for designers of OFDM systems. The guard time obviously reduces the overall throughput of the system because it reduces the time during which data transmission is allowed. A guard time that is too short does not prevent interference but does reduce throughput, and a guard time that is too long reduces throughput unnecessarily.

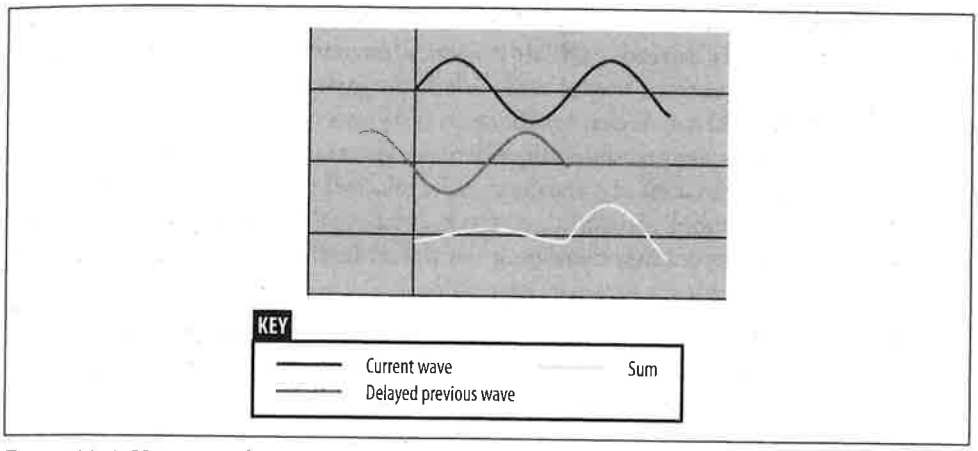


*Figure 11-4. ISI reviewed*

## Cyclic Extensions (Cyclic Prefixes)

The most straightforward method of implementing the guard time would be simply to transmit nothing during the guard time, as shown in Figure 11-5.

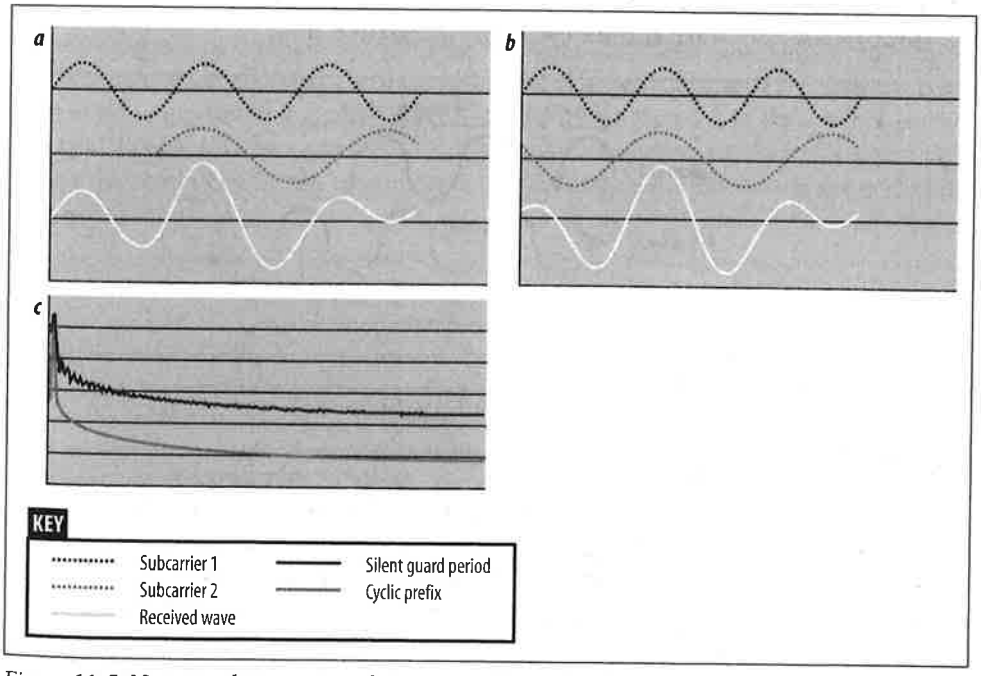


*Figure 11-5. Naive implementation of guard time (do not do this!)*

Simplistic implementations of the guard time can destroy orthogonality in the presence of common delay spreads. OFDM depends on having an integer number of wavelengths between each of the carriers. When the guard time is perfectly quiet, it is easy to see how a delay can destroy this necessary precondition, as in Figure 11-5. When the two subcarriers are added together, the spectral analysis shows subcarrier 1 (two cycles/symbol) as a strong presence and a relatively smaller amount of subcarrier 2 (three cycles/symbol). In addition, the spectral analysis shows a large number of high-frequency components, muddying the waters further. These components are the consequence of suddenly turning a signal "on."

Solving the problems associated with a quiet guard time is quite simple. Each subcarrier is extended through the FFT integration period back through the preceding guard time. Extending each subcarrier (and hence the entire OFDM symbol) yields a Fourier transform that shows only the amplitudes of the subcarrier frequencies. This technique is commonly called *cyclic extension*, and it may be referred to as the "cyclic prefix extension." The guard time with the extended prefix is called the *cyclic prefix*.

In Figure 11-6, the cyclic prefix preserves the spectral analysis. Subcarrier 1 was not shifted and is not a problem. Subcarrier 2 is delayed, but the previous symbol appears only in the guard time and is not processed by the Fourier transform. Thanks to the cyclic prefix extension, when subcarrier 2 is processed by the Fourier transform, it is a pure wave at three cycles per integration time.

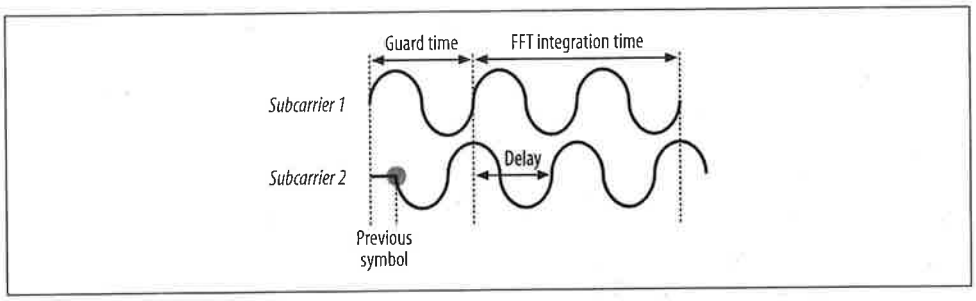


*Figure 11-6. Cyclic prefix extension*

## Convolution Coding

Strictly speaking, convolution coding is not part of OFDM. However, OFDM is used in applications for which the signal is subject to narrowband interference or frequency-specific narrowband fading, also known as *deep fading*. When fading occurs, a channel's ability to carry data may go to zero because the received amplitude is so small. To keep a few faded channels from driving the bit error rate to the sky, OFDM implementations often apply an error correction code across all the subchannels. Implementations that use an error correction code in conjunction with OFDM are sometimes called *coded OFDM* (COFDM).

One common technique applied by OFDM transceivers is *convolution coding*, which is a specialized form of a forward error-correcting code. Full details of convolution coding are beyond the scope of this book. One important point is that the convolution code is described by a rate (R) that specifies the number of data bits transmitted per code bit. A convolution code with R=1/2 transmits one data bit for every two code bits. As the code rate decreases, more code bits are available to correct errors, and the code becomes more robust. However, the price of robustness is decreased throughput. 802.11a uses convolution codes extensively. The most conservative data rates use codes with a rate of 1/2, and the aggressive codes used for the highest data rates use a coding rate of 3/4.

## Windowing

One further enhancement helps OFDM transceivers cope with real-world effects. Transitions can be abrupt at symbol boundaries, causing a large number of high-frequency components (noise). To make OFDM transmitters good radio citizens, it is common to add padding bits at the beginning and end of transmissions to allow transmitters to "ramp up" and "ramp down" from full power. Padding bits are frequently needed when error correction coding is used. Some documentation may refer to the padding as "training sequences."

Windowing is a technique used to bring the signal for a new symbol gradually up to full strength while allowing the old symbol to fade away. Figure 11-7 shows a common windowing function based on a cosine curve. At the start of the symbol period, the new function is brought up to full strength according to the cosine function. When the symbol ends, the cosine curve is used to fade out the bits at the end of the symbol.

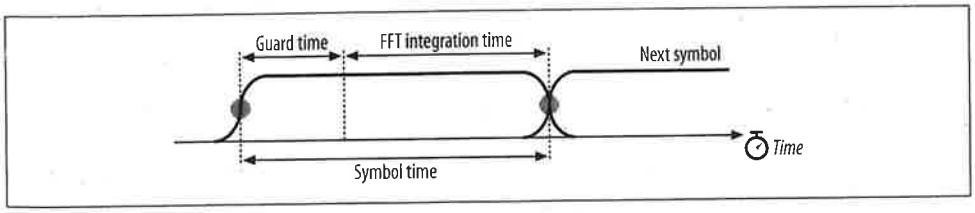


*Figure 11-7. Cosine windowing technique*

# OFDM as Applied by 802.11a

802.11a is not a radical application of OFDM. The task group responsible for standardizing OFDM took the middle ground to apply OFDM to wireless LAN.

## OFDM Parameter Choice for 802.11a

When choosing OFDM parameters, there are usually three given items of information. Bandwidth is fixed, often by regulatory authorities. Delay is determined by the environment in which the OFDM system will operate; most office buildings generally show a delay spread of 40–70 ns, though in some environments, the delay spread can approach 200 ns. Finally, the bit rate is usually a design goal, though the goal is usually "make the bit rate as high as possible, given the constraints of the other parameters."

One common guideline is that the guard time should be two to four times the average delay spread. As a result, the 802.11a designers selected a guard time of 800 ns. Symbol duration should be much larger than the guard time, but within reason. Larger symbol times mean that more subcarriers can fit within the symbol time. More subcarriers increase the signal-processing load at both the sender and receiver, increasing the cost and complexity of the resulting device. A practical choice is to select a symbol time at least five times the guard time; 802.11a matches the 800-ns guard time with a 4-μs symbol time. Subcarrier spacing is inversely related to the FFT integration time. 802.11a has a 3.2-μs integration time and a subcarrier spacing of 0.3125 MHz (1/3.2 μs).

Operating channels in 802.11a are specified as 20 MHz wide. The bandwidth of an operating channel is a design decision. Wider operating channels have higher throughput, but fewer operating channels fit into the assigned frequency spectrum. The use of a 20-MHz operating channel allows for reasonable speeds on each channel (up to 54 Mbps), as well as a reasonable number of operating channels in the assigned spectrum. 802.11a offers a wide variety of choices in modulation and coding to allow for a trade-off between robust modulation and conservative coding, which yields low, reliable throughput and finer-grained modulation and aggressive coding, resulting in higher, yet somewhat more fragile, throughput.

## Structure of an Operating Channel

Like the DS PHYs, the OFDM physical layer organizes the spectrum into operating channels. Each 20-MHz channel is composed of 52 subcarriers. Four of the subcarriers are used as *pilot carriers* for monitoring path shifts and ICI, while the other 48 subcarriers are used to transmit data. Subcarriers are spaced 0.3125 MHz apart. As shown in Figure 11-8, channels are numbered from –26 to 26. Subcarrier 0 is not used for signal-processing reasons.

Pilot subcarriers are assigned to subcarriers –21, –7, 7, and 21. To avoid strong spectral lines in the Fourier transform, the pilot subcarriers transmit a fixed bit sequence specified in 802.11a using a conservative modulation technique.

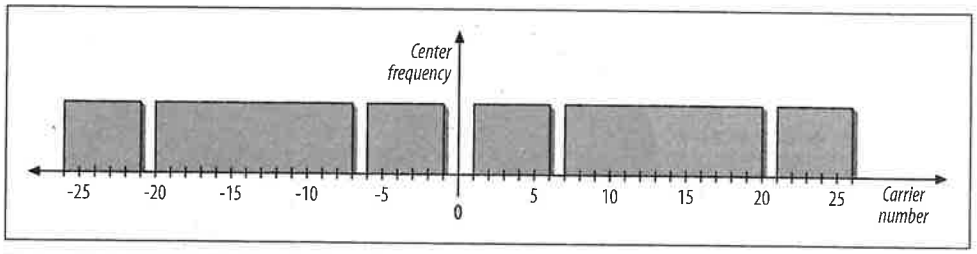


*Figure 11-8. Structure of an OFDM channel*

## Operating Channels

In the U.S., the channels in the 5-GHz band are numbered starting every 5 MHz according to the following formula:

center frequency (MHz) = 5,000 + 5 × n, n=0,1,2, .. 200

Obviously, each 20-MHz 802.11a channel occupies four channels in the U-NII bands. The recommended channel use is given in Table 11-1.

*Table 11-1. United States channels for 802.11a*

| Band | Allowed power[a] | Channel numbers | Center frequency (GHz) |
|---|---|---|---|
| U-NII lower band | 40 mW | 36 | 5.180 |
| (5.15–5.25 GHz) | (2.5 mW/MHz) | 40 | 5.200 |
| | | 44 | 5.220 |
| | | 48 | 5.240 |
| U-NII mid-band | 200 mW | 52 | 5.260 |
| (5.25–5.35 GHz) | (12.5 mW/MHz) | 56 | 5.280 |
| | | 60 | 5.300 |
| | | 64 | 5.320 |
| U-NII upper band | 800 mW | 149 | 5.745 |
| (5.725–5.825 GHz) | (50 mW/MHz) | 153 | 5.765 |
| | | 157 | 5.785 |
| | | 161 | 5.805 |

[a] The allowed power is the maximum output power using a 6-dBi antenna gain.

There is one other feature of note about the operating bands. As with the DS PHYs, a transmit mask limits power leakage into the side bands. The mask is shown in Figure 11-9.

Figure 11-10 gives an overall view of the 802.11a channels available in the U.S. Four channels are available in each of the U-NII bands in the U.S. In the two lower U-NII

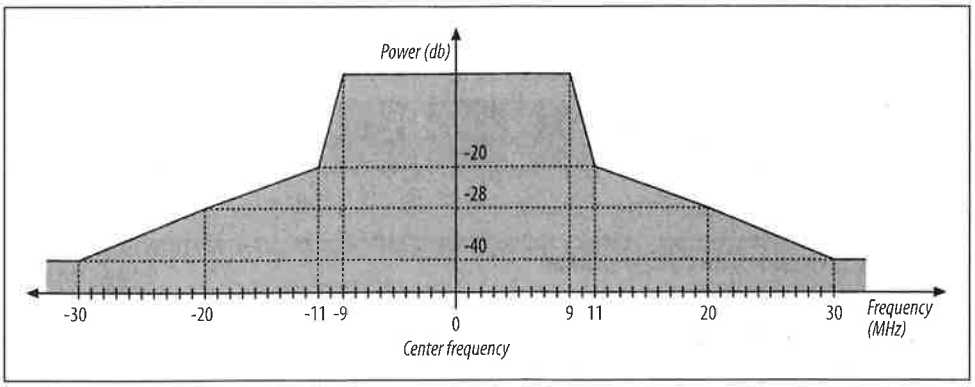


*Figure 11-9. Transmit spectrum mask for 802.11a*

bands, channels are allowed to overlap, and a 30-MHz guard band is present at both the lower end of the low U-NII band and the upper end of the mid U-NII band.

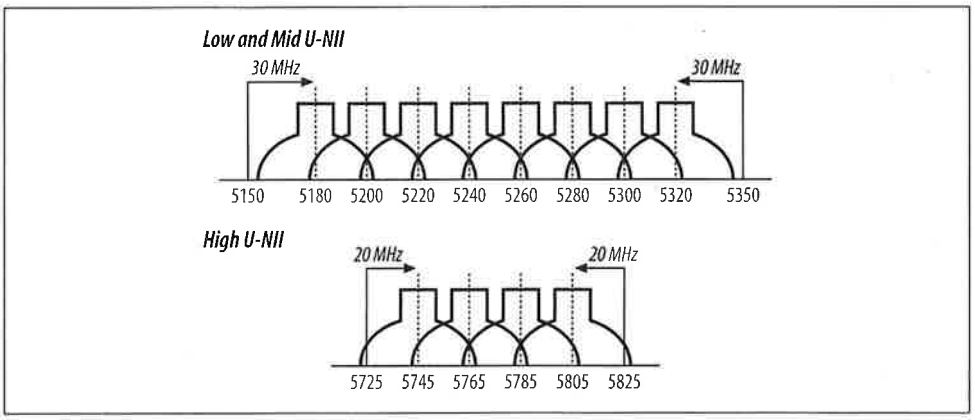


*Figure 11-10. Operating bands from Table 11-1*

# OFDM PLCP

Like all the other physical layers, the OFDM PHY includes its own PLCP, which adds physical layer–specific framing parameters.

## Framing

The OFDM PHY adds a preamble and a PLCP header. It also adds trailing bits to assist the encoding schemes used. This section divides the PLCP frame logically, but some components span different fields in the protocol unit. Figure 11-11 is the jumping-off point for discussion of the OFDM frame.

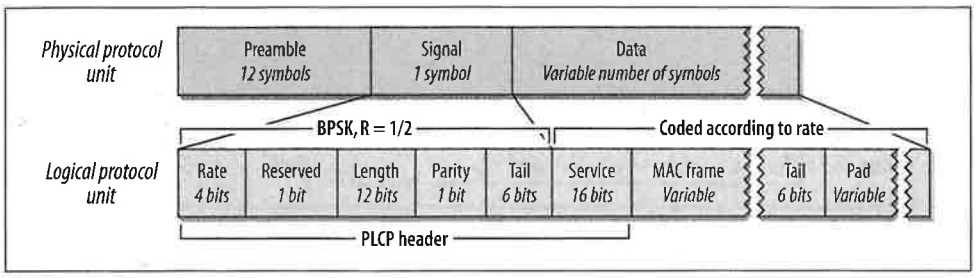


*Figure 11-11. OFDM PLCP framing format*

Figure 11-12 shows the start of a frame, but includes the guard intervals and windowing used by the transmitter. The preamble lasts 16 µs, which is evenly divided between short and long training sequences; the difference between the two is described in the next section. After the preamble, one OFDM symbol carries the Signal field, then a variable number of data symbols carry the end of the PLCP header, the MAC payload, and the trailer. All symbols use a modified cosine window to ensure smooth transitions. After the short preamble, which is used to synchronize frequencies, a guard time protects against multipath fading.

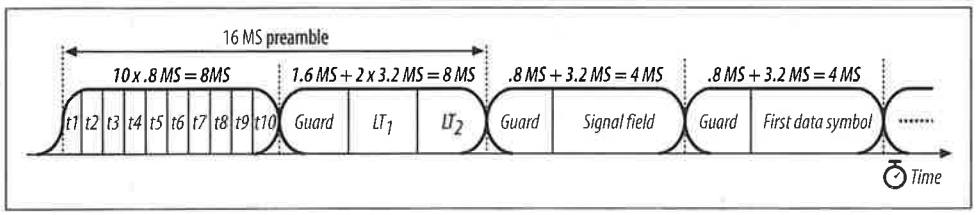


*Figure 11-12. Preamble and frame start*

## Preamble

As with all other common IEEE 802 networks, and certainly all 802.11 physical layers, the OFDM physical protocol unit begins with a preamble. It is composed of 12 OFDM symbols that synchronize various timers between the transmitter and the receiver. The first 10 symbols are a *short training sequence*, which the receiver uses to lock on to the signal, select an appropriate antenna if the receiver is using multiple antennas, and synchronize the large-scale timing relationships required to begin decoding the following symbols. The short training sequences are transmitted without a guard period. Two *long training sequences* follow the short training sequences. Long training sequences fine-tune the timing acquisition and are protected by a guard interval.

## Header

The PLCP header is transmitted in the Signal field of the physical protocol unit; it incorporates the Service field from the Data field of the physical protocol unit. As

shown in Figure 11-13, the Signal field incorporates the Rate field, a Length field, and a Tail field.

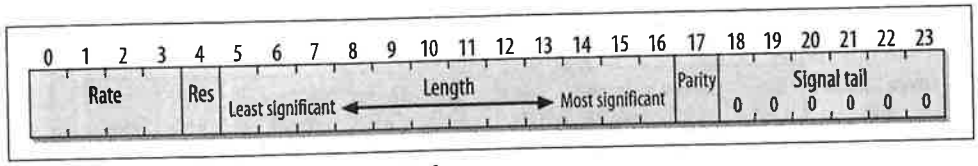


*Figure 11-13. Signal field of OFDM PLCP frame*

*Rate (4 bits)*
: Four bits encode the data rate. Table 11-2 shows the bits used to encode each of the data rates. See the section "OFDM PMD" for details on the encoding and modulation scheme used for each data rate.

*Table 11-2. Rate bits*

| Data rate (Mbps) | Bits (transmission order) |
|---|---|
| 6 | 1101 |
| 9 | 1111 |
| 12 | 0101 |
| 18 | 0111 |
| 24 | 1001 |
| 36 | 1011 |
| 48 | 0001 |
| 54 | 0011 |

*Length (12 bits)*
: Twelve bits encode the number of bytes in the embedded MAC frame. Like most fields, it is transmitted least-significant bit to most-significant bit. The length is processed by a convolution code to protect against bit errors.

*Parity (1 bit) and Reserved (1 bit)*
: Bit 4 is reserved for future use and must be set to 0. Bit 17 is an even parity bit for the first 16 Signal bits to protect against data corruption.

*Tail (6 bits)*
: The Signal field ends with six 0 tail bits used to unwind the convolution code. As such, they must by definition be processed by the convolution code.

*Service (16 bits)*
: The final field in the PLCP header is the 16-bit Service field. Unlike the other components of the PLCP header, it is transmitted in the Data field of the physical protocol unit at the data rate of the embedded MAC frame. The first eight bits are set to 0. As with the other physical layers, MAC frames are scrambled before transmission; the first six bits are set to 0 to initialize the scrambler. The

remaining nine bits are reserved and must set to 0 until they are adopted for future use.

### Data

The encoding scheme used for the data depends on the data rate. Before transmission, data is scrambled, as it is with the other physical layers. The Service field of the header is included in the Data field of the physical protocol unit because it initializes the scrambler.

### Trailer

The Data field of the physical protocol unit ends with a trailer. (The 802.11a specification does not call the ending fields a trailer, but it is a descriptive term.) It is composed of two fields:

*Tail (6 bits)*
: Like the tail bits in the PLCP header, the tail bits appended to the end of the MAC frame bring the convolution code smoothly to an end.

*Pad (variable)*
: As used by 802.11a, OFDM requires that fixed-size blocks of data bits be transferred. The Data field is padded so that its length is an integer multiple of the block size. The block size depends on the modulation and coding used by the data rate; it is discussed in the next section.

## OFDM PMD

The OFDM PHY uses a cocktail of different modulation schemes to achieve data rates ranging from 6 Mbps to 54 Mbps. In all cases, the physical layer uses a symbol rate of 250,000 symbols per second across 48 subchannels; the number of data bits per symbol varies. An OFDM symbol spans all 48 subchannels.

There are four rate tiers with the OFDM PHY: 6 and 9 Mbps, 12 and 18 Mbps, 24 and 36 Mbps, and 48 and 54 Mbps. Support is required for 6, 12, and 24 Mbps, which are lowest speeds in each of the first three tiers, and therefore the most robust in the presence of interference. The lowest tier uses binary phase shift keying (BPSK) to encode 1 bit per subchannel, or 48 bits per symbol. The convolution coding means that either half or one quarter of the bits are redundant bits used for error correction, so there are only 24 or 36 data bits per symbol. The next tier uses quadrature phase shift keying (QPSK) to encode 2 bits per subchannel, for a total of 96 bits per symbol. After subtracting overhead from the convolution code, the receiver is left with 48 or 72 data bits. The third and fourth tiers use generalized forms of BPSK and QPSK known as quadrature amplitude modulation (QAM). 16-QAM encodes 4 bits using 16 symbols, and 64-QAM encodes 6 bits using 64 symbols. The third tier uses 16-QAM along with the standard R=1/2 and R=3/4 convolution codes. To achieve

higher rates with 64-QAM, however, the convolution codes use R=2/3 and R=3/4. Table 11-3 summarizes the coding methods used by each data rate in the OFDM PHY.

*Table 11-3. Encoding details for different OFDM data rates*

| Speed (Mbps) | Modulation and coding rate (R) | Coded bits per carrier[a] | Coded bits per symbol | Data bits per symbol[b] |
|---|---|---|---|---|
| 6 | BPSK, R=1/2 | 1 | 48 | 24 |
| 9 | BPSK, R=3/4 | 1 | 48 | 36 |
| 12 | QPSK, R=1/2 | 2 | 96 | 48 |
| 18 | QPSK, R=3/4 | 2 | 96 | 72 |
| 24 | 16-QAM, R=1/2 | 4 | 192 | 96 |
| 36 | 16-QAM, R=3/4 | 4 | 192 | 144 |
| 48 | 64-QAM, R=2/3 | 6 | 288 | 192 |
| 54 | 64-QAM, R=3/4 | 6 | 288 | 216 |

[a]Coded bits per subchannel is a function of the modulation (BPSK, QPSK, 16-QAM, or 64-QAM).
[b]The data bits per symbol is a function of the rate of the convolution code.

## Clear Channel Assessment

The OFDM PHY specification leaves implementers a great deal of latitude in selecting techniques for noting a busy channel. Received signal strength thresholds determine whether the channel is in use, but the main guideline for 802.11a equipment is that it must meet certain performance standards. Implementations are free to use the Packet Length field from the PLCP header to augment clear channel assessment, but this is not required.

## An Example of OFDM Encoding

OFDM encoding, as you can no doubt see by now, is an intense, multistep process. One of the additions that 802.11a made to the original specification was Annex G, an encoding of Schiller's *Ode to Joy* for transmission over an 802.11a network.* Shortly after 802.11a was published, the IEEE 802.11 working group discovered several errors in the example and published a correction. If you are interested in learning about OFDM encoding in detail, you can refer to this example.

# Characteristics of the OFDM PHY

Parameters specific to the OFDM PHY are listed in Table 11-4. Like the physical layers presented in Chapter 10, the OFDM PHY also incorporates a number of parameters to

* Well, an English translation, anyway...

adjust for the delay in various processing stages in the electronics. As a final note, the extra radio bandwidth provided by the U-NII bands offers a great deal of throughput. There are eight overlapping channels available for the OFDM PHY, so it can offer up to 432 Mbps in an area where all eight channels are co-located.

*Table 11-4. OFDM PHY parameters*

| Parameter | Value | Notes |
|---|---|---|
| Maximum MAC frame length | 4,095 bytes | |
| Slot time | 9 µs | |
| SIFS time | 16 µs | The SIFS is used to derive the value of the other interframe spaces (DIFS, PIFS, and EIFS). |
| Contention window size | 15 to 1,023 slots | |
| Preamble duration | 20 µs | |
| PLCP header duration | 4 µs | |

Like the other physical layers, the OFDM PHY has a number of attributes that can be adjusted by a vendor to balance delays in various parts of the system. It includes variables for the latency through the MAC, the PLCP, and the transceiver, as well as variables to account for variations in the transceiver electronics.

CHAPTER 12

# Using 802.11 on Windows

Whether you've made it to this point by skipping Chapters 3–11, or whether you've read all the theory, we're now going to get our hands dirty and start installing equipment.

From the standpoint of practical system and network administration, working with 802.11 is similar to working with Ethernet. Installing 802.11 drivers is nearly identical to installing Ethernet drivers, and the network interfaces behave almost exactly like Ethernet interfaces. 802.11 interfaces cause an ARP cache to be brought into existence, and other software may even perceive the wireless interface as an Ethernet interface. Unlike many Ethernet drivers, however, 802.11 drivers can have a number of advanced knobs and features that reflect the additional management features presented in Chapter 7.

This chapter is not intended to be a definitive guide to Windows drivers for 802.11 network cards. There are two major development lines in Windows (9x versus NT and progeny), and adding additional software such as a VPN client can further complicate matters. (My advice is to install the wireless LAN card before any VPN client software.) There are a number of vendors, and, as you'd expect, the driver software varies from one vendor to the next. The examples show how to install a driver on Windows and explain the non-Ethernet driver features in some detail. I selected two 802.11 cards as examples: the Nokia C110/C111 and the Lucent ORiNOCO. While not particularly common, the Nokia card is interesting because it has a number of advanced features and exposes a number of the network parameters that were discussed in Chapters 2–10. The Lucent card (which is sold under a number of different labels) probably has the lion's share of the market, and it hides most of the exotic configuration parameters from the user. Most cards that are available fall somewhere between these two extremes.

# Nokia C110/C111

Nokia's 802.11b solution comes in two similar form factors. The C110 is a Type 2 PC Card with an integrated antenna; the C111 is basically the same, but with two external antenna connectors. The card ships with a CD to enable basic installation, and updated drivers are available from *http://forum.nokia.com/* after registering.

## Installation

Driver installation begins before inserting the card, so start by inserting the CD-ROM into the CD-ROM drive. After a splash screen, an installation program begins. Its main screen is shown in Figure 12-1. Select Installing Nokia C110/C111 to launch the installer.

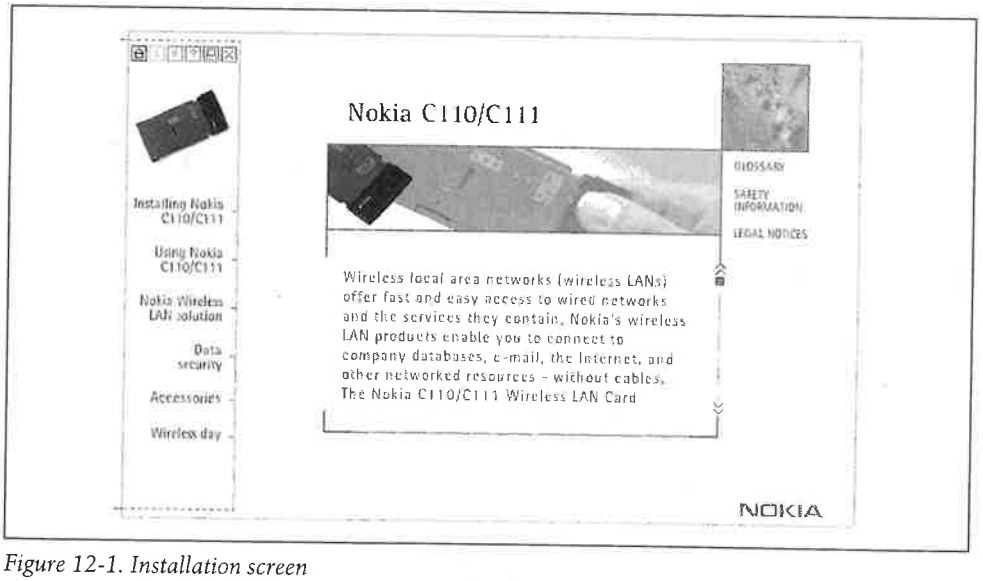


*Figure 12-1. Installation screen*

The next steps are very familiar. Selecting the install option launches InstallShield and brings up an admonition to close all other programs. Like all commercial software, the Nokia driver is licensed. Accept the license agreement to proceed. Next, the driver asks for the country in which the card is being used. This information is used to set the regulatory domain, which affects the radio channels that can be used. See Figure 12-2.

The setup program then asks where it should put the files that it installs. The default location is *C:\Program Files\Nokia C110*. Next, the driver asks which components should be installed. The software package is composed of three major components: drivers, help files, and administrator components. Typical users install only the first

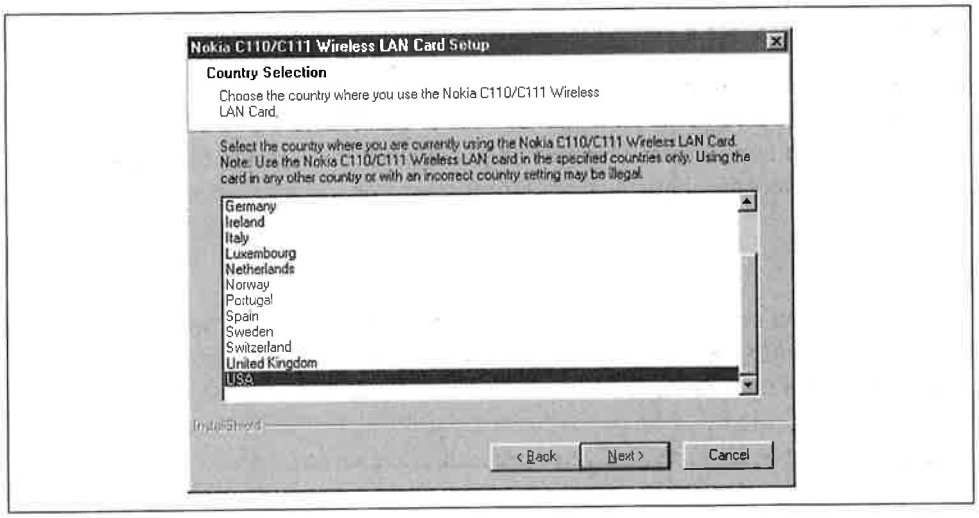


*Figure 12-2. Country selection*

two, but network administrators can use features in the software to streamline installation procedures for large numbers of users. A Typical installation consists of only the first two, and an Administrator installation uses all three. For good measure, a Custom installation allows any subset of the three components to be installed. In a Custom installation, the administrator component is not installed by default and must be selected explicitly.

At this point, the installation program has collected all the information necessary to install the driver. It copies files that were unpacked during the installation and makes registry changes to activate the new driver. After that completes, a dialog box appears and prompts you to insert the card into the computer to complete the installation. When the card is inserted into the PC Card slot, the message shown in Figure 12-3 appears, and the installation is complete.

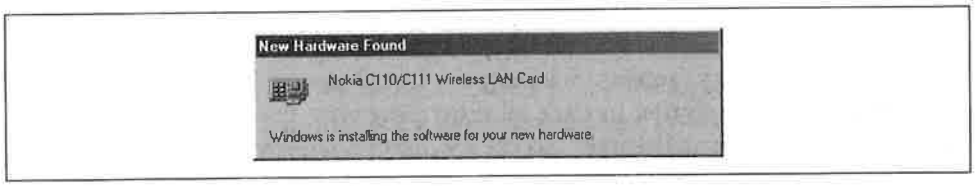


*Figure 12-3. Driver installation prompt*

The installer prompts you for the creation of a network profile before restarting the computer. Profiles are one of the card's advanced features; they are optional, but they make card management much simpler. It will be interesting to see whether other vendors pick up on this idea.

## Network Profiles

The Nokia card groups settings into *profiles*, which allow users to switch easily between networks. Immediately after installing the driver, the user is prompted to create a profile if none exists. Administrators may create customized driver installation disks or smart cards to distribute settings more easily. In addition to the run-of-the-mill network settings, the Nokia driver can control whether the system attempts to log in to a domain and a Microsoft workgroup. Profiles can also contain WEP keys as well as a number of 802.11 parameters. Figure 12-4 shows the initial Profile Wizard screen. Profiles are assigned text names; the name need not have anything to do with the SSIDs in use.

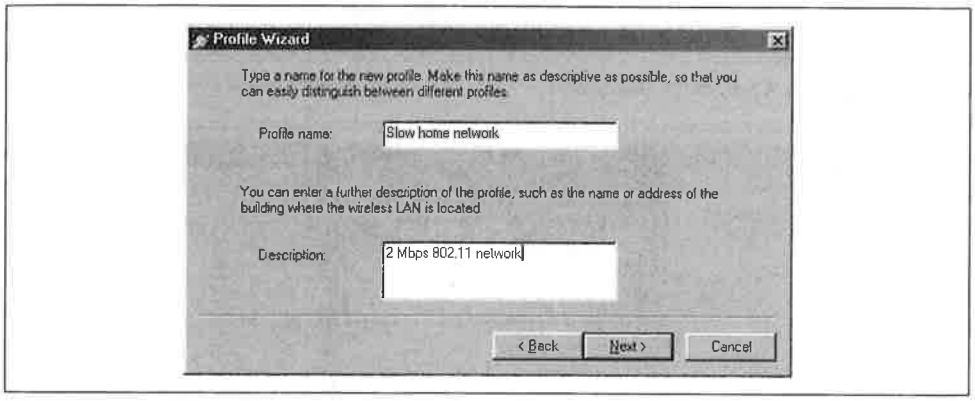


*Figure 12-4. Initial Profile Wizard screen*

After naming the profile and entering a detailed description, the user must then select the type of network in use. Infrastructure networks use access points, and ad hoc networks are independent BSSs.* After selecting the network operating mode, the user proceeds to a network selection window (Figure 12-5). Networks are distinguished by their service set IDs, which are called names for simplicity.

This window allows you to enter channel information. Unless you have an overwhelming reason to set the channel explicitly, leave the channel set to automatic, which means that the driver scans all channels when it is initialized. There are two ways to select the network name. One is to type the SSID for which the driver should search. To make it easier for basic users, the small unobtrusive button to the right of the network name field pops up a list of networks currently in range (Figure 12-6). This window is a nice touch; it shows you data rates and signal strengths to help you make an intelligent choice.

* Earlier Nokia products also had an operating mode called Instawave, which allowed direct station-to-station communication simultaneously with station-to-access point communication. Instawave was nonstandard and never found extensive use in production networks.

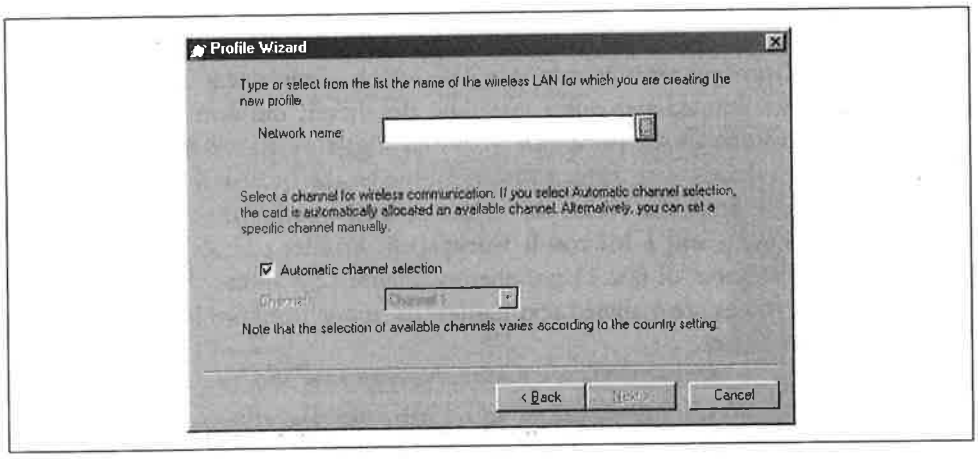


*Figure 12-5. Network parameter dialog box*

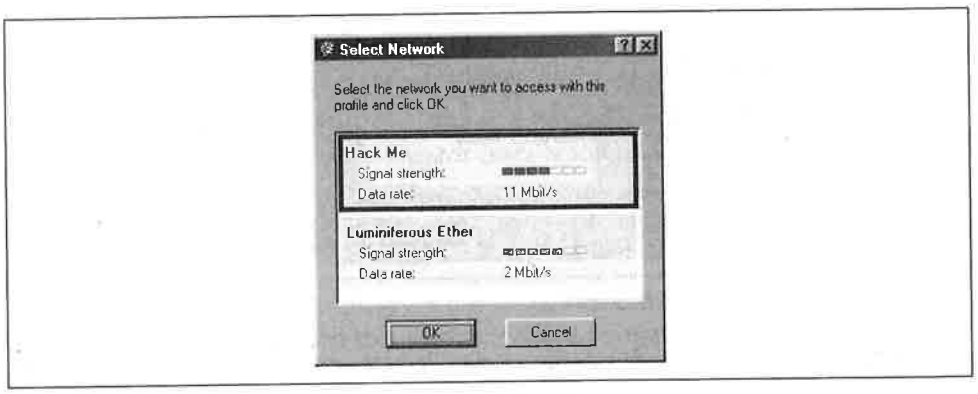


*Figure 12-6. Network selection box*

After selecting a network, the user is presented with the final screen for selecting basic network parameters (Figure 12-7). Most 802.11 networks use DHCP to assign IP addresses. If not, the profile can be modified later to specify an address explicitly. You can also specify Windows domain and workgroup names.

At this point, a network profile has been created. It appears on the main screen, as shown in Figure 12-8.

To select a profile, highlight it and click Apply. The selected profile gets a big green check mark to show that it has been selected. As part of choosing a profile, the driver maintains network configuration settings. These settings can be updated as the user changes profiles. When a new profile is selected, the dialog box in Figure 12-9 appears. My experience has been that you usually need to restart the system when you apply a new profile.

When the system comes back up, there will be a small default monitoring window, in addition to a taskbar icon. The small window, shown in Figure 12-10, displays the

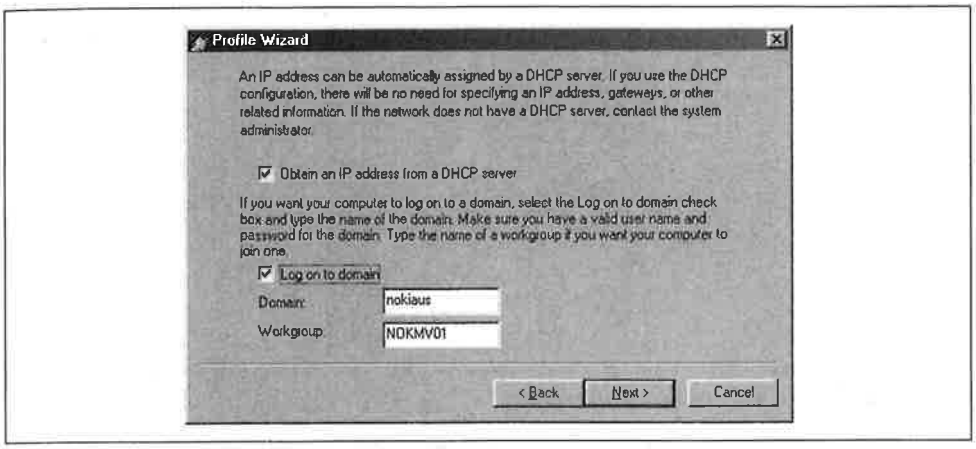


*Figure 12-7. Addressing and login options*

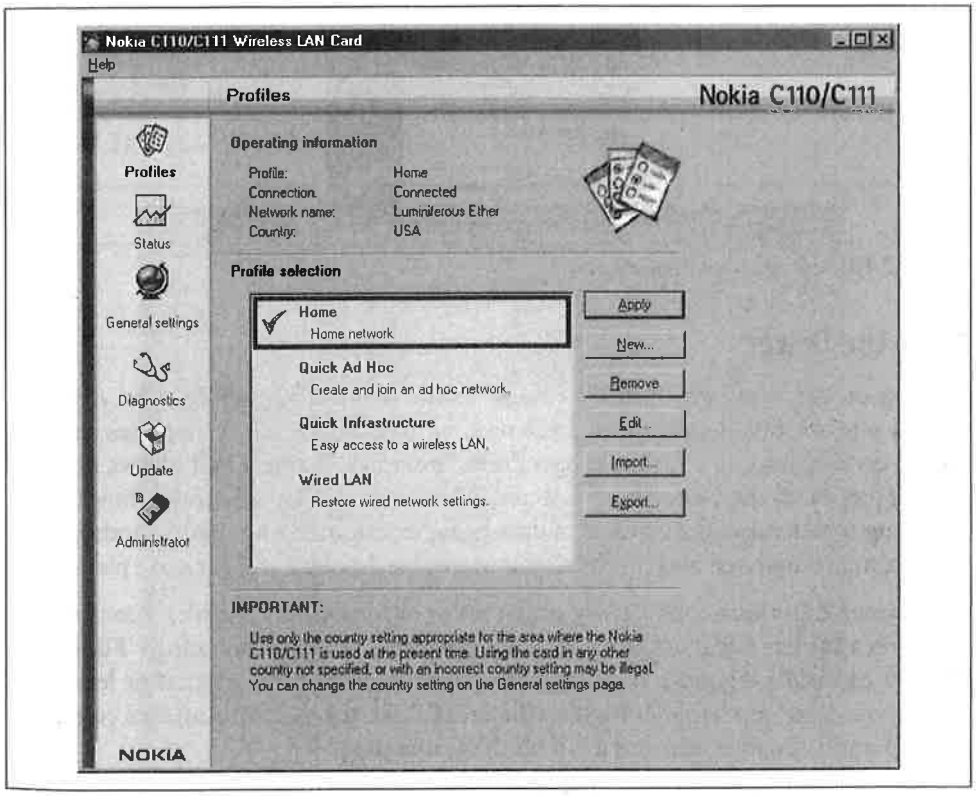


*Figure 12-8. Profile selection screen*

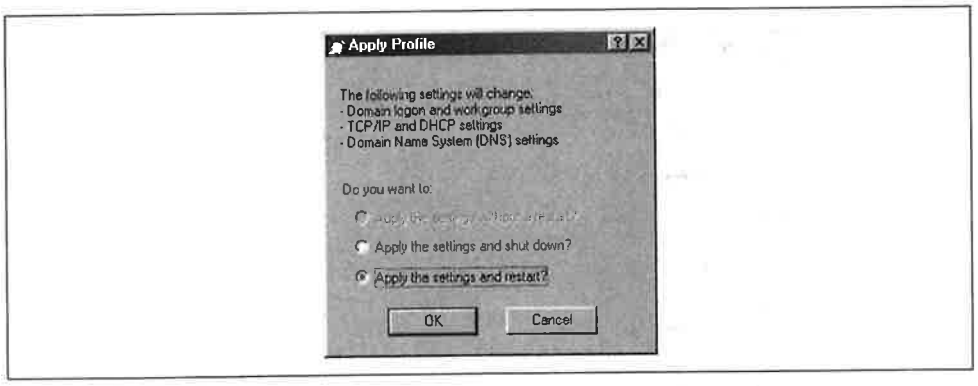


*Figure 12-9. Options when changing profiles*

network profile and a signal strength meter. It also provides a button next to the profile name for gaining access to the detailed configuration window.

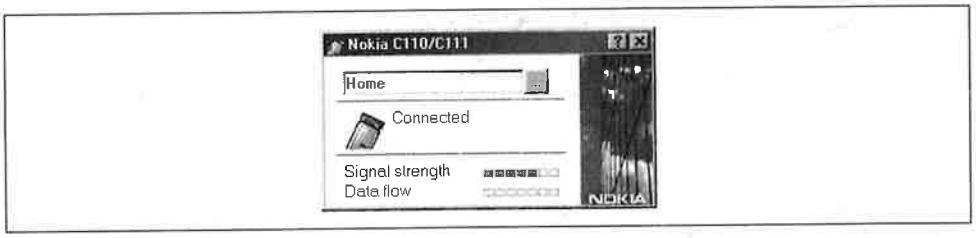


*Figure 12-10. Default monitoring window*

## Using the Driver

Clicking on the small gray button brings up the main driver screen. The driver is divided into six broad categories, each with an icon at the left. Categories may be further divided into tabs for more specific information. Figure 12-11 shows the Status category, with the General tab selected. The Status → General screen shows a signal strength meter and the amount of data being transmitted on the BSS. Both graphs are continually updated and can be useful in troubleshooting and network planning.

When troubleshooting connectivity problems or expanding a network, it can be useful to run a card as a scanner, simply to see the access points within range. For example, you can walk around a building with your laptop and ensure that at least two access points are reachable in high-traffic areas. Like the general statistics page, the access point listing is reached as a tab off the status page.

Figure 12-12 shows two access points. The 11-Mbps access point is a Nokia A032, which is capable of transmitting its IP address and workload. The mechanism used to accomplish this will be discussed in more detail in Chapter 16.

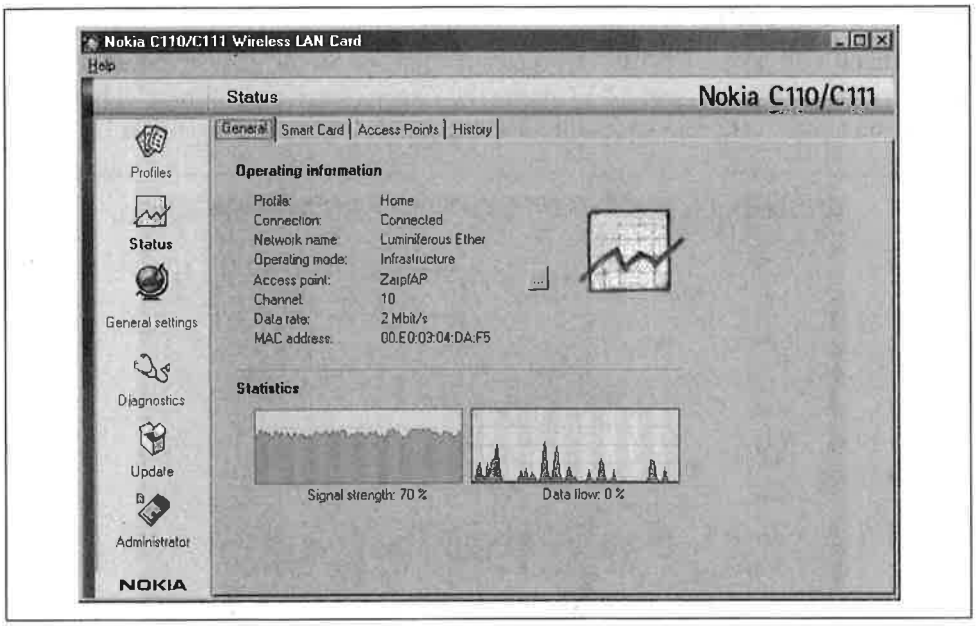


*Figure 12-11. Status → General driver screen*

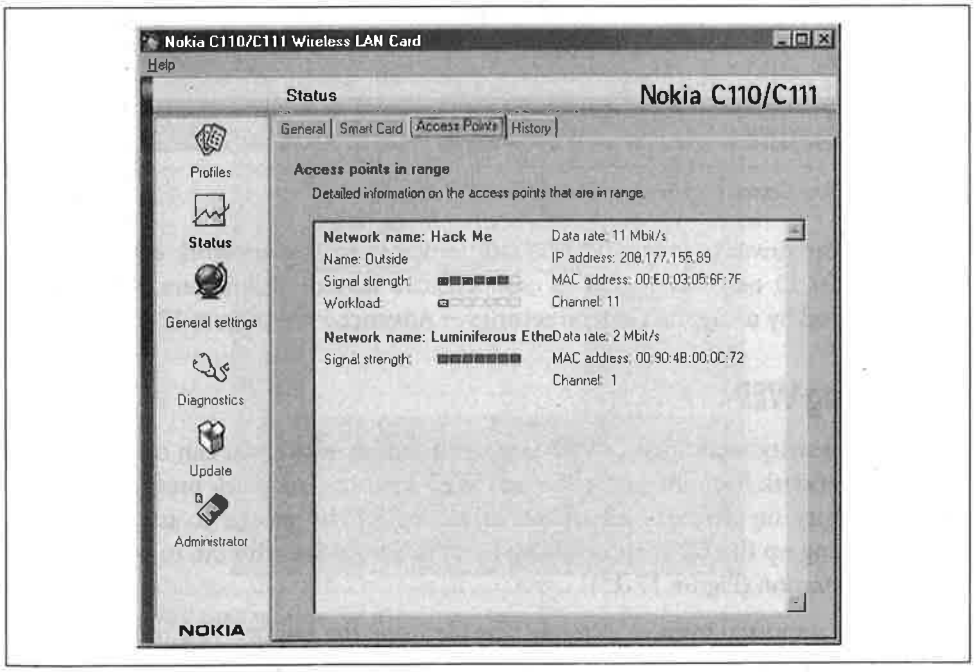


*Figure 12-12. General → Access point tab*

### Global driver configuration options

The General settings button can be used to gain access to the general driver settings used on a global basis. Figure 12-13 shows the General settings → General tab, which allows you to select the regulatory domain and enable low-power operation.

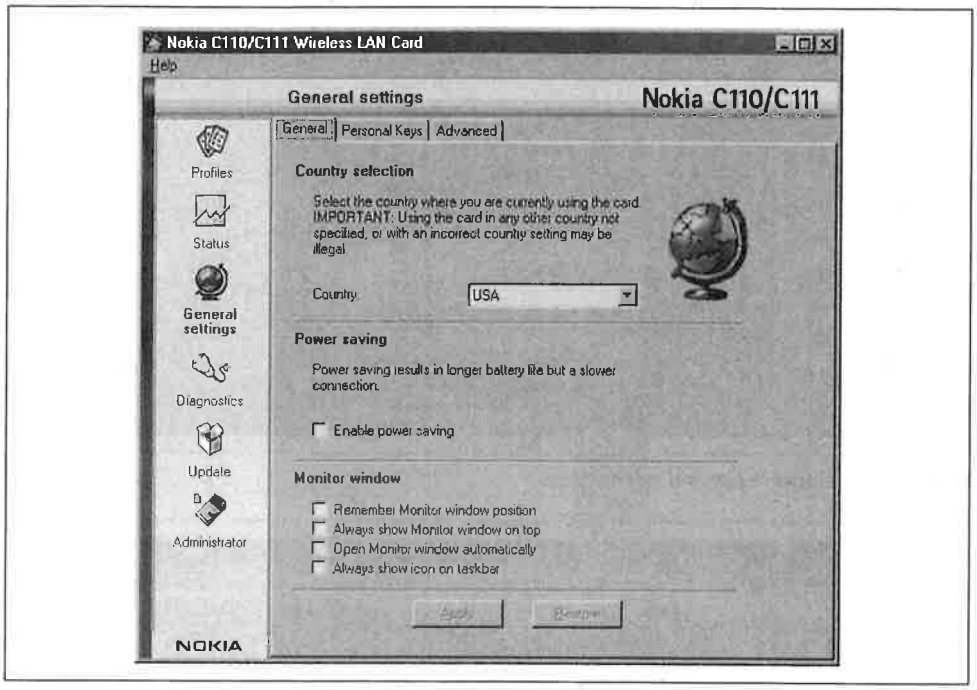


*Figure 12-13. The General settings → General tab*

By default, the driver manages TCP/IP properties and other network configuration and treats 802.11 network names as case-sensitive network identifiers. All of these can be changed by using the General settings → Advanced tab (Figure 12-14).

## Configuring WEP

Despite its security weaknesses, WEP is a significant measure you can take to secure an 802.11 network from intruders. To add WEP keys to a network profile, go to the profile category on the left side of the driver, select the profile to use, and click **Edit...** to bring up the Edit Profile dialog box. Select the Security tab to bring up the WEP configuration (Figure 12-15).

Keys may be imported from an external text file using the Import... button or may be added from scratch using the Add... button. Figure 12-16 shows the dialog box used to edit a shared WEP key.

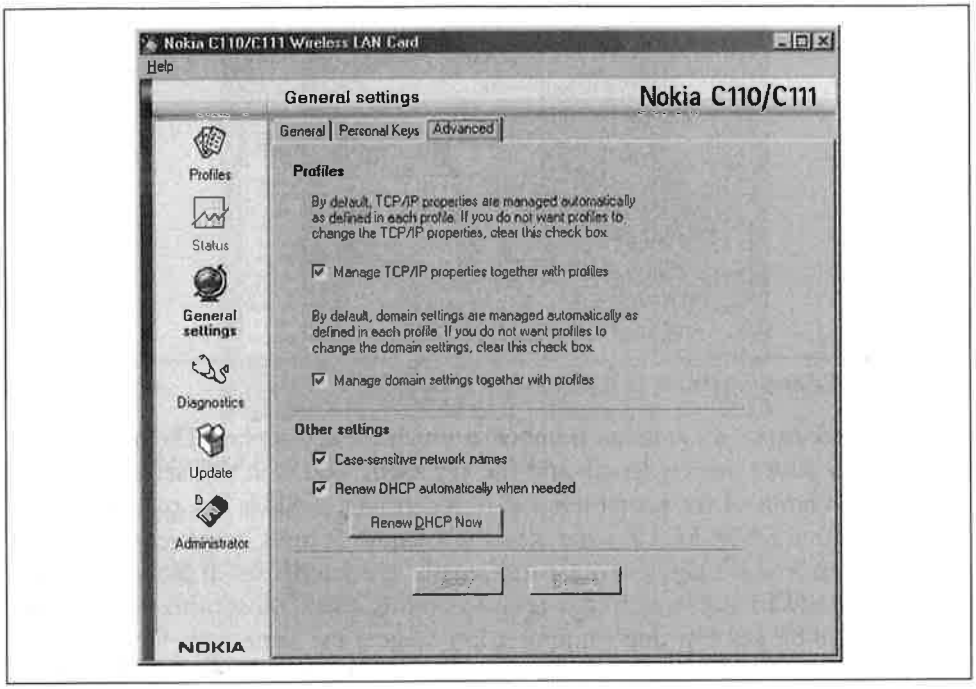


*Figure 12-14. General settings → Advanced tab*

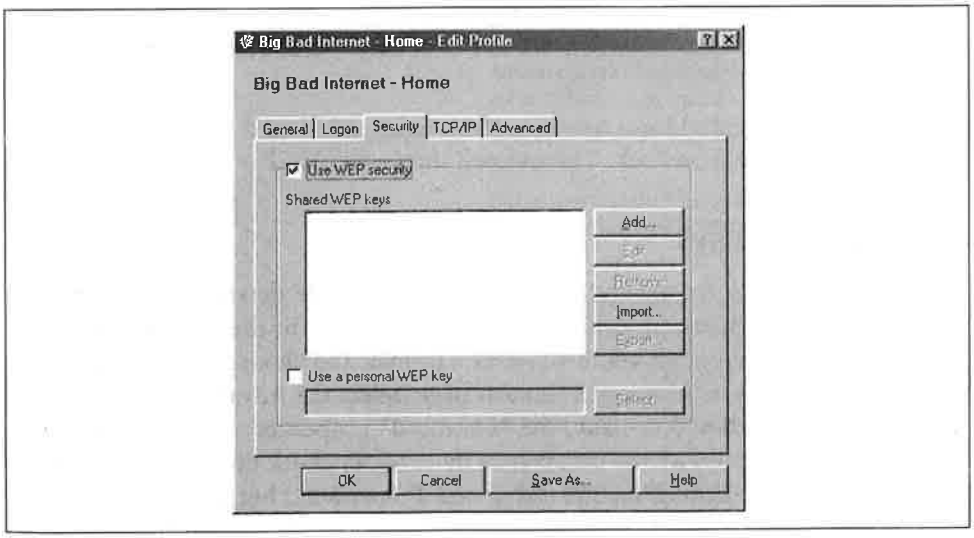


*Figure 12-15. WEP configuration*

802.11 permits four shared keys per SSID. Not all drivers support using all four, but Nokia's driver does. At the top of the dialog box there are fields for selecting the

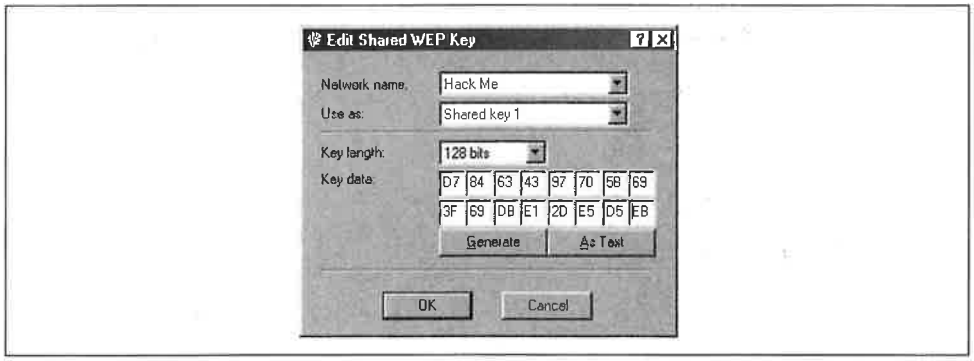


*Figure 12-16. Editing a shared WEP key*

SSID (network name) and the key number to which the key applies. The bottom part of the screen shows the key length and the key itself. Note that this screen specifies key length in terms of the actual number of secret bits; a 40-bit key corresponds to the standardized 64-bit RC4 key described in Chapter 5; most other vendors would refer to this as a 64-bit key. No other key lengths are standardized; Nokia also supports a 128-bit WEP key, which requires a 152-bit RC4 key. Most other vendors also support a 128-bit key but don't interpret key lengths the same way; for most vendors, 128 bits means a 128-bit RC4 key with 104 secret bits. The result is that Nokia cards won't interoperate with other vendors' products at the 128-bit key length.

For the first station on a network, the Generate button may be used to generate a random key. The As Text button allows you to cut and paste a hexadecimal string to use as the key.

When WEP is configured for a network, the wireless LAN card icon in the monitor window appears with a padlock. The padlock shows that WEP is in use.

## Advanced Properties

The Nokia driver allows network administrators to make detailed changes to the 802.11 parameters discussed in previous chapters. To get to the advanced settings, go to the main configuration window, select a profile, and choose **Edit...**, which displays the main profile editing page. On this page, select the Advanced tab to display the advanced properties. Normally, the "advanced" properties are configured automatically. If you want to set the parameters by hand, uncheck the Automatic configuration checkbox. Clicking on the Advanced Properties... button takes you to a window that lets you modify several of the basic 802.11 properties (Figure 12-17). Table 12-1 describes the available options and their defaults.

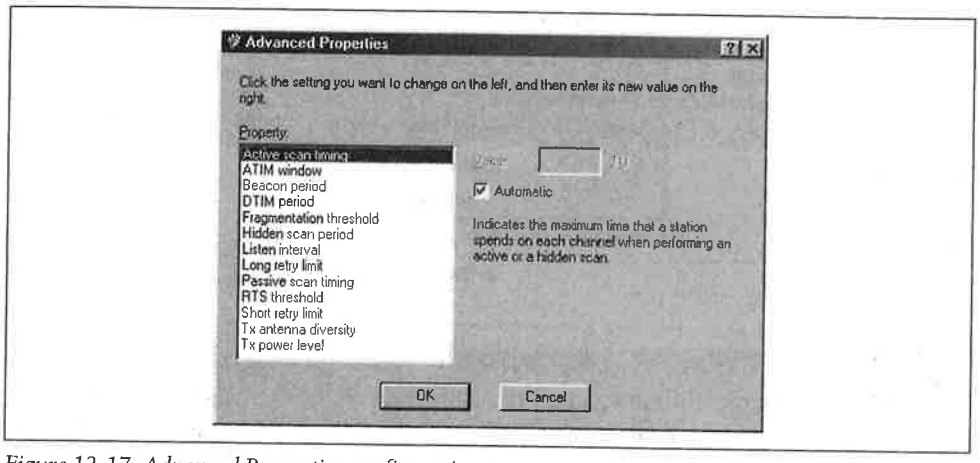


*Figure 12-17. Advanced Properties configuration screen*

*Table 12-1. Parameter defaults used by the Nokia driver*

| Property | Default ("automatic") value | Measurement units | Description | Reference |
|---|---|---|---|---|
| Active scan timing | 30 | TU | Number of time units spent monitoring each channel during an active scan. | Chapter 7;802.11 clause 11.1.3 |
| ATIM window | 0 | TU | Amount of time that stations in an ad hoc network must remain awake after a beacon. | Chapter 7;802.11 clause 11.2.2 |
| Beacon period | 100 | TU | Amount of time between target beacon transmissions for an ad hoc BSS . | Chapter 7;802.11 clause 11.1.2 |
| DTIM period | 10 | beacon periods | Number of beacon periods between DTIM messages. | Chapter 7;802.11 clause 11.2.1.3 |
| Fragmentation threshold | 2,346 | bytes | Packets larger than the threshold are fragmented at the MAC layer for transmission. | Chapter 3;802.11 clause 9.4 |
| Hidden scan period | 30 | seconds | | |
| Listen interval | 10 | beacon periods | Number of beacon periods between station waking up to listen to DTIMs for buffered traffic delivery. | Chapter 7;802.11 clause 11.2 |
| Long retry limit | 4 | attempts | Maximum number of attempts to transmit a frame bigger than the RTS threshold. | Chapter 3;802.11 clause 9.5.2.3 |
| Passive scan timing | 250 | TU | Amount of time spent listening for traffic on each radio channel during a passive scan. | Chapter 7;802.11 clause 11.1 |
| RTS threshold | 0 | bytes | Packets larger than the RTS are preceded by an RTS/CTS handshake. | Chapter 3;802.11 clause 9.2 |
| Short retry limit | 7 | attempts | Maximum number of attempts to transmit a frame shorter than the RTS threshold. | Chapter 3;802.11 clause 9.5.2.3 |

*Table 12-1. Parameter defaults used by the Nokia driver (continued)*

| Property | Default ("automatic") value | Measurement units | Description | Reference |
|---|---|---|---|---|
| Tx antenna diversity | 1 | N/A | 0–enabled<br>1–disabled | |
| Tx power level | 1 | N/A | 1–high power<br>2–low power | |

## Smart Cards

When 802.11 networks initially gained prominence, one of the biggest concerns was how to distribute configuration information to the mobile computers on the network. One of the more novel solutions was Nokia's smart cards. The C11x cards have an integrated smart-card reader. Administrators can write profiles out to smart cards using the Administrator menu in the main configuration screen and distribute smart cards to users. Smart cards can store the entire profile, including WEP keys and TCP/IP configuration, which reduces the possibility of user error—users don't have to type network parameters or keys correctly. Smart cards also present a tamper-resistant barrier to sensitive information (such as WEP keys) that might otherwise be stored in a file on the hard disk.

### Unlocking the smart card

Users with a smart card are presented with a challenge to unlock the card when the wireless card is placed in a PC Card slot. First, the dialog box of Figure 12-18 appears as the smart card is opened.

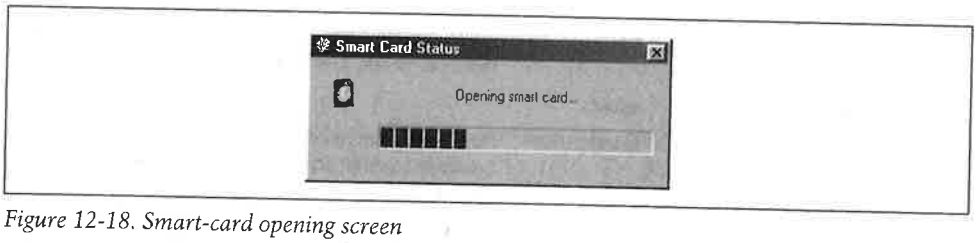


*Figure 12-18. Smart-card opening screen*

When the smart-card initialization completes, the user enters the PIN for the smart card, using the window in Figure 12-19.

After three unsuccessful attempts to open the smart card, it locks up and cannot be unlocked even by the correct PIN. Only the PIN Unlocking Key (PUK) distributed with the smart card, can be used to unlock the smart card after it locks. When the smart card is unlocked, any profiles that it stores can be used by the driver. Smart-card profiles cannot be edited directly and are identified with a special icon on the main configuration screen (Figure 12-20).

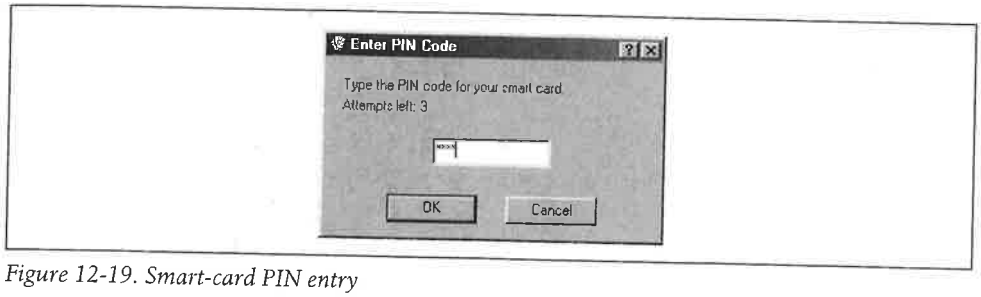


*Figure 12-19. Smart-card PIN entry*

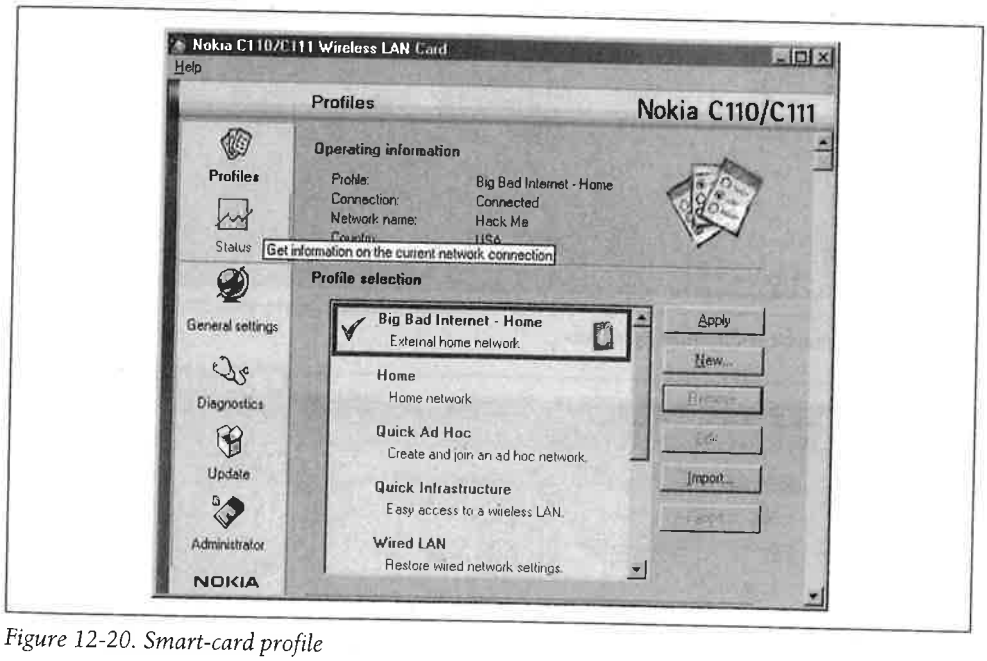


*Figure 12-20. Smart-card profile*

### Locking the smart card and changing the PIN

If desired, the smart card can be locked from the driver configuration panel. Select Status, then choose the Smart Card tab, and click on the **Lock...** button shown in Figure 12-21. You can also use this screen to change the PIN by clicking the Change **PIN Code...** button at the bottom.

### Moving profiles onto the smart card

Only driver installations that include the administrator routines can create or modify the data stored on a smart card. The **Administrator** button on the lefthand banner accesses administrative functions. Two main administrator functions are available: one to move profiles to smart cards and one to create installation disks with profiles readily available. See Figure 12-22.

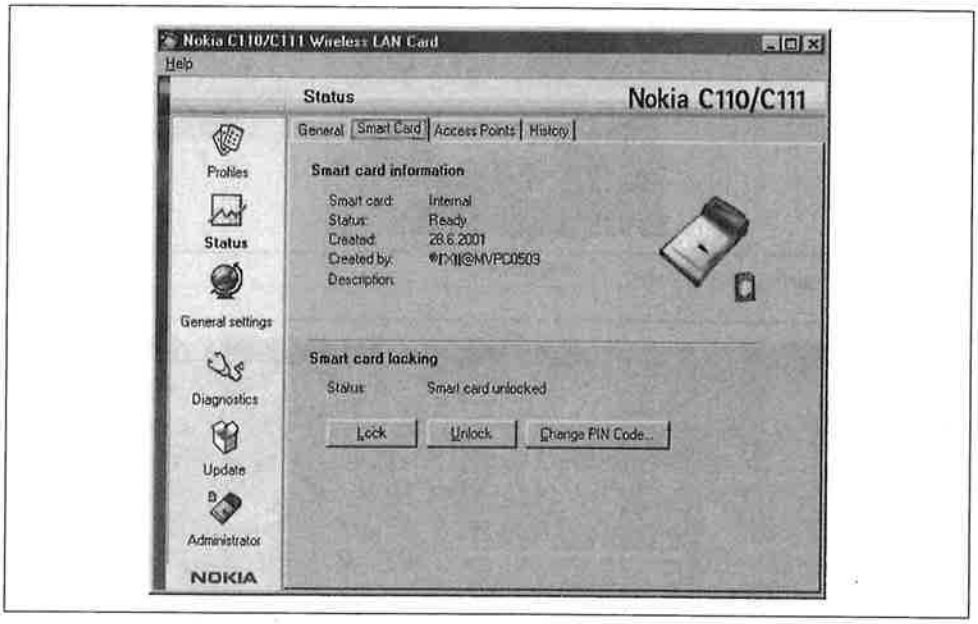


*Figure 12-21. Smart-card management tab*

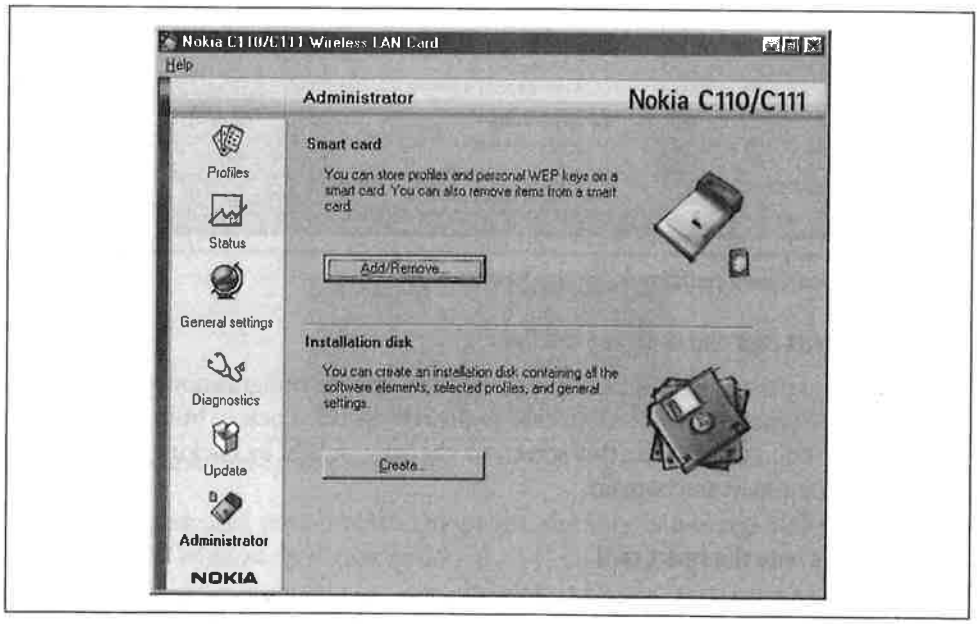


*Figure 12-22. Administrator functions of the C11x driver*

To access the smart-card functions, click on the **Add/Remove...** button at the top of the page. This brings you to Figure 12-23, which allows you to erase the smart card and move profiles from disk to smart card and vice versa.

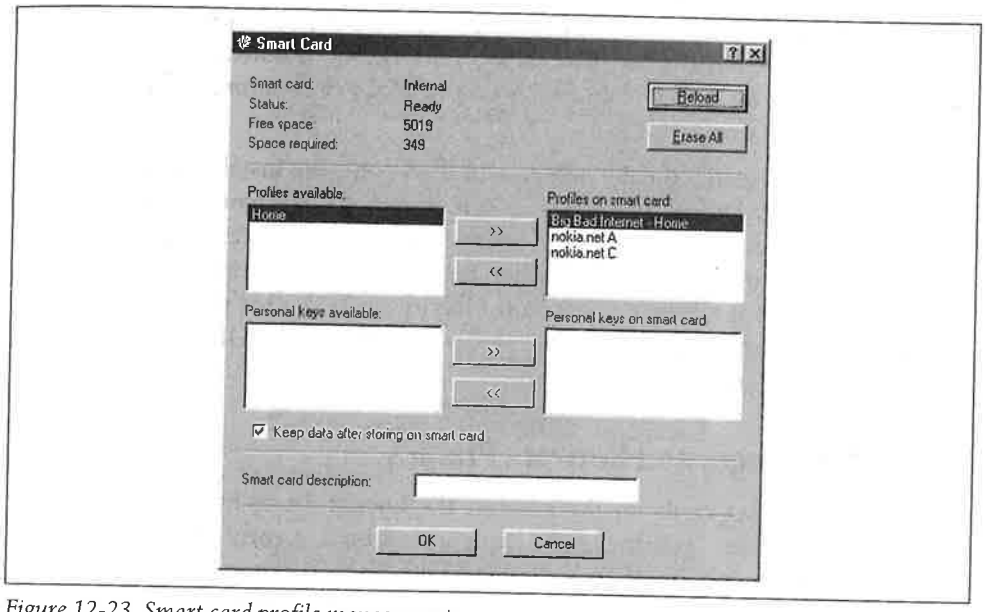


*Figure 12-23. Smart card profile management*

# Lucent ORiNOCO

With the acquisition of WaveLAN, Lucent Technologies became an overnight leader in the wireless LAN market space. WaveLAN has been involved in wireless LAN development since the early 1990s. Naturally, standardized products were not available until the late 1990s with the adoption of the initial 802.11 standard. After experiencing financial trouble throughout much of 2001, Lucent's reorganization efforts have split the company into several pieces. As this book was written, the reorganization had yet to affect the physical appearance of the product—the card I ordered still had Lucent branding.

Lucent sells two flavors of the card. The cards are identical, except for the size of the WEP key supported. ORiNOCO Silver cards support WEP keys with 40 secret bits (marketed as 64-bit cards), and ORiNOCO Gold cards support WEP keys with 104 secret bits (marketed as 128-bit cards).

## Installation

Lucent's card installation is more conventional. Begin by putting the card in an available PC Card slot and have the CD handy. When the card is inserted, Windows may identify it as a WaveLAN/IEEE card. Once 802.11 was ratified and products were brought to market, Lucent distinguished between the earlier proprietary cards and the 802.11-compliant cards by adding "IEEE" to the product name.

Drivers for the ORiNOCO cards are bundled with a CD-ROM. Windows 2000 ships with Lucent drivers, but the bundled drivers have caused problems in a number of installations. Fetch an update from *http://www.orinocowireless.com* before inserting the card for the first time.

Allow Windows to search for the drivers and then point the installation program at the CD-ROM drive. Different versions of Windows have different drivers, so select the directory corresponding to your version of Windows (e.g., *D:\DRIVERS\WIN98*). Once the driver is installed in the network stack, you must install the Client Manager, which is the user frontend to the driver. The client manager is distributed on a CD with the card, and updates are available from the ORiNOCO web site at *http://www.orinocowireless.com*.

## The Client Manager and Network Profiles

Like drivers for other cards on the market, the Lucent driver holds configuration information in profiles. Versions of the driver distributed in early 2001 were limited to four profiles, but fall 2001 revisions lifted that limit. This section covers the fall 2001 driver. After the driver and client manager are installed, you can create profiles to hold information about networks in the area. To get to the configuration, start the Client Manager. Figure 12-24 shows the main Client Manager screen.

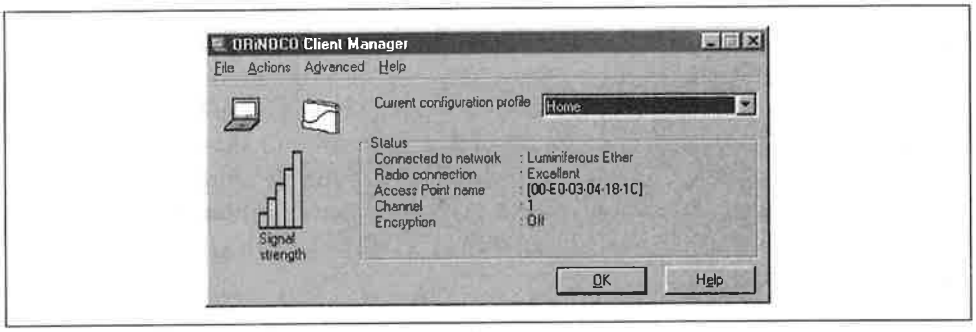


*Figure 12-24. Client Manager main screen*

Go to the **Actions** menu and select **Add/Edit Configuration Profile**. This brings up the main profile window, shown in Figure 12-25. Click on the **Add** button to begin creating the profile. The dialog box shown in Figure 12-26 appears. Name the profile on the lefthand side of the box and select the network type on the right. Access Point networks and Residential Gateways are infrastructure networks, and Peer-to-Peer networks are infrastructure networks.

Next, identify the network. As with other products, the network name is the SSID of the network. You can either type it into the network name field or use the Scan button to pull up a list of networks whose Beacons are currently being received.

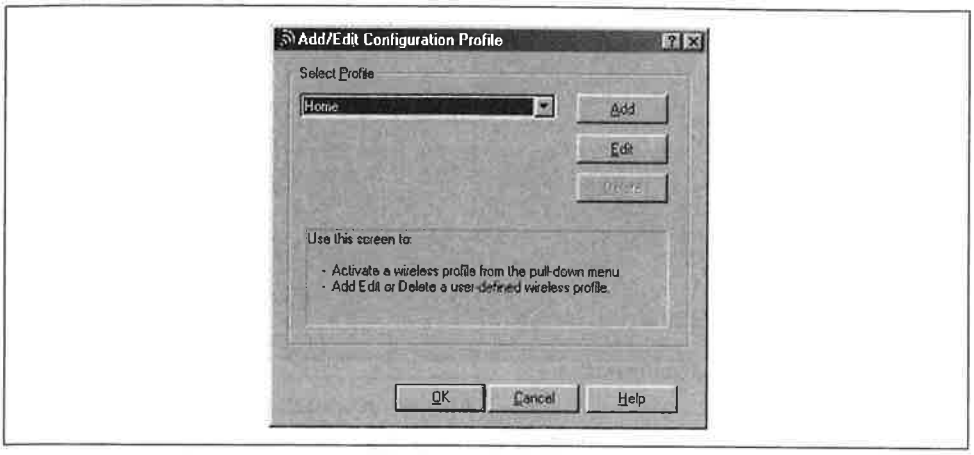


*Figure 12-25. Add/Edit Configuration Profile window*

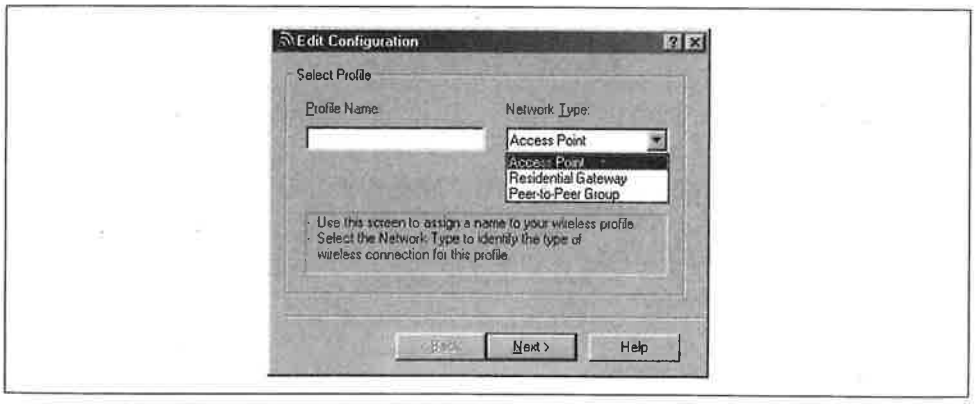


*Figure 12-26. Profile creation*

Figure 12-27 shows the network name configuration box, and Figure 12-28 shows the scan result box. Unfortunately, the scan result box gives no indication of signal strength.

Next, the user can configure WEP keys using the dialog box in Figure 12-29. Keys can be entered as alphanumeric strings and hashed into a bit string used as a key, or they can be entered directly. The algorithm used to generate a key from an alphanumeric string is not documented, which might lead to security questions. Much of the security of WEP resides in the key, and a simple key generator might compromise security by allowing dictionary attacks on the key.

Power management is controlled on a network-by-network basis and is configured after WEP using the dialog box in Figure 12-30.

The final item contained in a profile controls TCP/IP behavior and is set using the dialog box in Figure 12-31. The only option available with the Lucent driver is

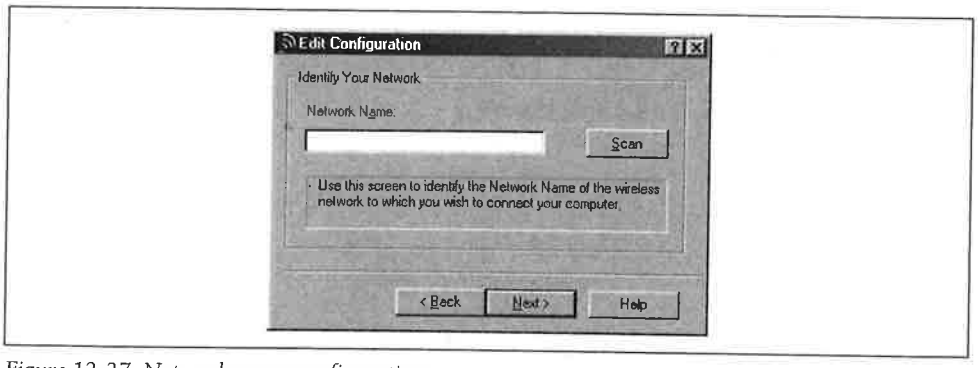


*Figure 12-27. Network name configuration*

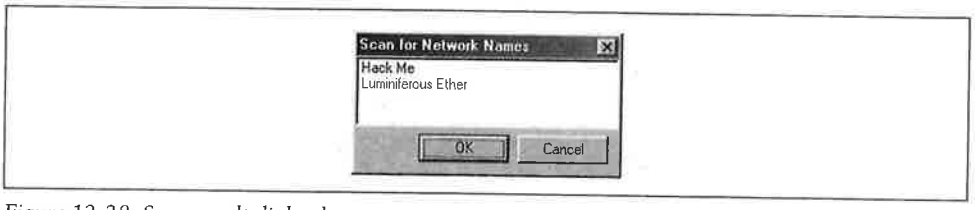


*Figure 12-28. Scan result dialog box*

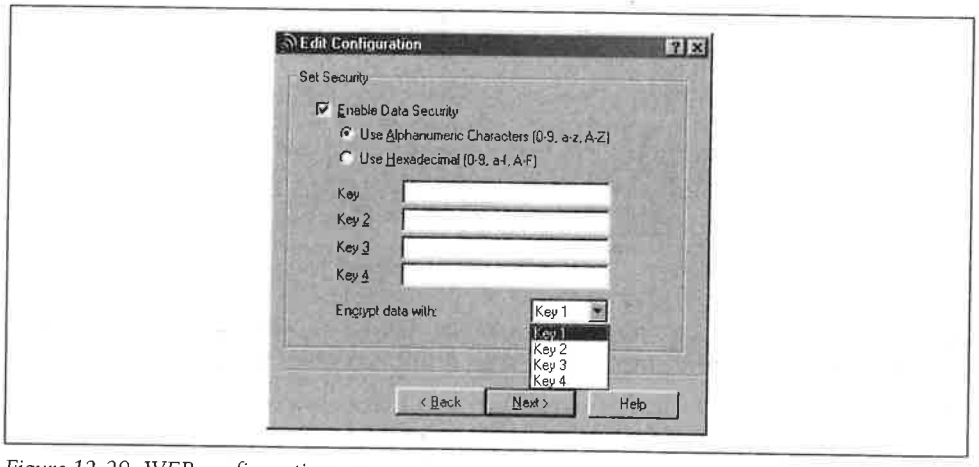


*Figure 12-29. WEP configuration*

whether a DHCP renewal will be issued when changing between profiles. No provisions are available for controlling Windows networking configuration.

## Using the Driver

Miscellaneous system administration tasks are performed through the Client Manager. The radio can be disabled, even when the card is active, by going to the File menu and choosing Disable Radio.

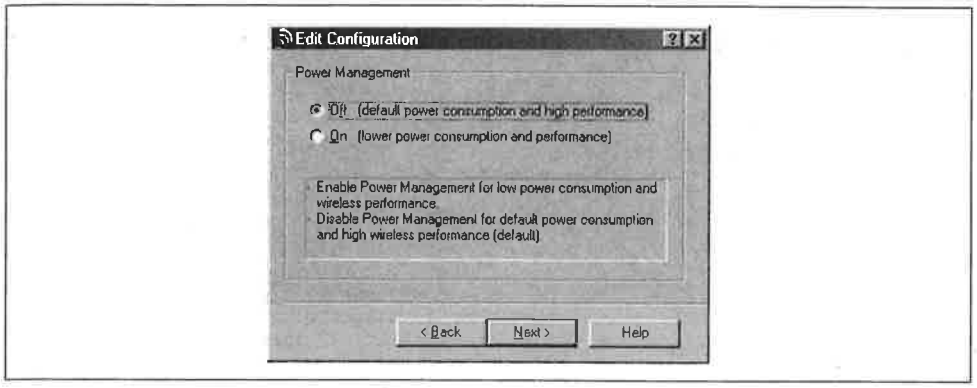


*Figure 12-30. Power management configuration*

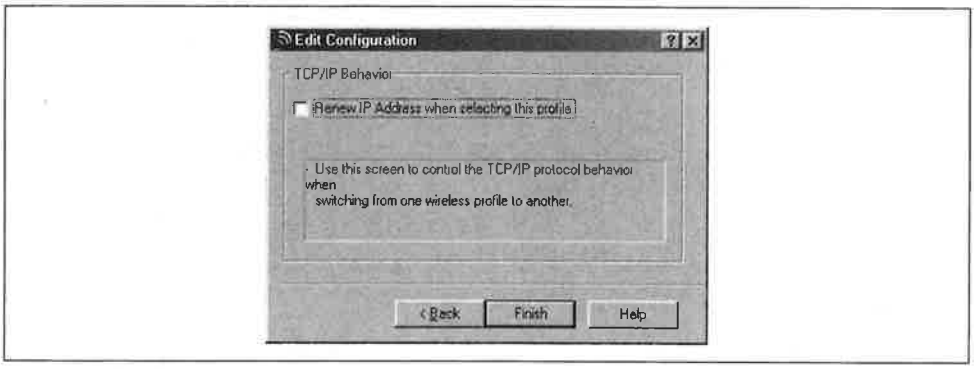


*Figure 12-31. Network control*

### Changing between profiles

One of the most common configuration tasks is changing between profiles. The operating profile can be changed through the drop-down box on the right side of the Client Manager or through the Action menu. Changing profiles does not require a system reboot, even if the new profile requires an IP address renewal. See Figure 12-32.

### Version information

Choosing Version Info from the Help menu brings up the version information. A version information window is shown in Figure 12-33. The driver and related software are implemented by several independent software pieces, all of which have their own version number. Most importantly, the driver for the card and the Client Manager user interface are separate and must be upgraded separately.

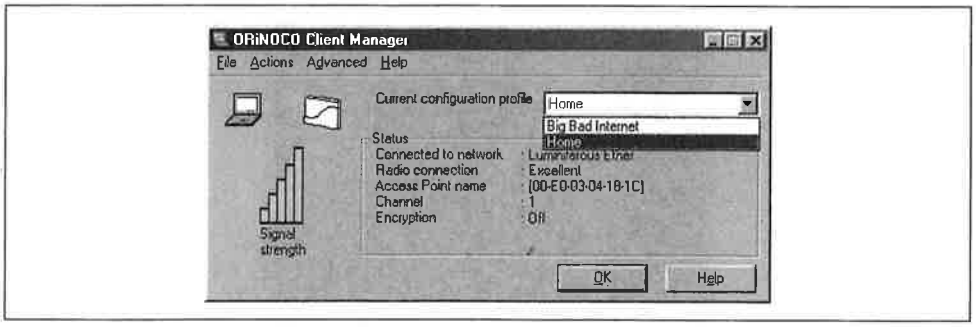


*Figure 12-32. Changing profiles*

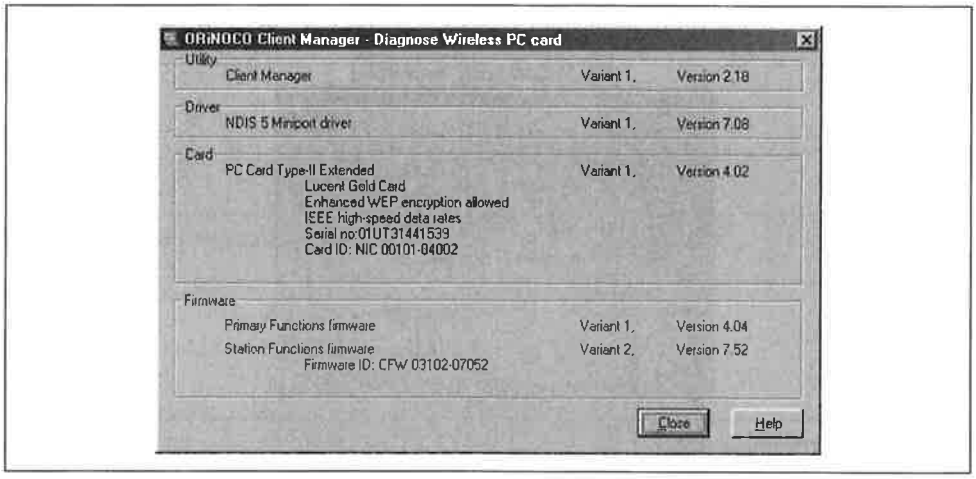


*Figure 12-33. Version information*

## Site Monitor

The Site Monitor can be accessed by going to the Advanced Menu and choosing Site Monitor, which displays network information for open networks in the area. Networks that require WEP authentication are left off the list, which limits the use of Site Monitor to older networks not using WEP or networks with intentionally loose security, such as neighborhood or community networks.

If you are currently associated with an open network, the Site Monitor begins by displaying the AP list, which is shown in Figure 12-34. By default, the AP list shows only the signal-to-noise ratio (SNR), but it can be set to display the signal and noise figures themselves. One field not shown in the figure is the AP name, which is available only when the access point is also a Lucent access point.

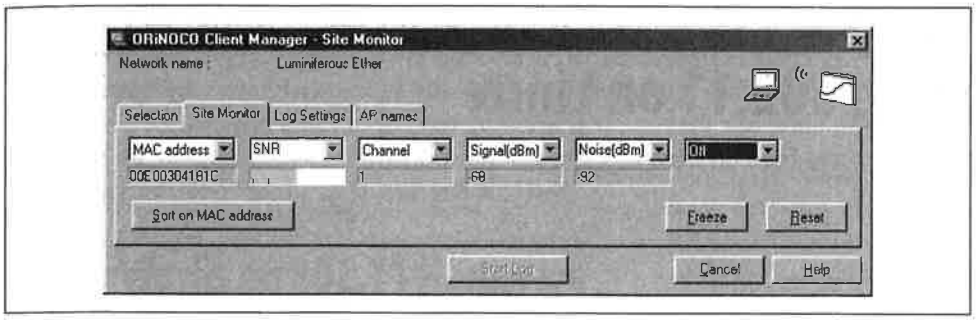


*Figure 12-34. Network details*

CHAPTER 13

# Using 802.11 on Linux

Of the operating systems currently in wide use and active development, Unix environments offer the flexibility and stability required by power users and network administrators. When new hardware hits the market, the technical staff responsible for making purchase recommendations often asks about Linux support because of the additional functionality that can frequently be gained from the environment.

By now, though, the successful 802.11 hardware vendors have come to understand that Linux support increases sales and customer satisfaction without imposing significant additional costs. Linux friendliness among vendors was not always the case. One common strategy in past years was to release a fully featured driver as a binary module and a limited-functionality open source driver. What this strategy failed to take into account was the wide diversity of Linux hardware platforms and the difficulty in compiling and troubleshooting a closed-source binary driver without low-level documentation. Binary-only drivers are usually painful to install and troubleshoot, while open source counterparts are developed rapidly and are well-supported by the user community. Vendors that tried the two-driver approach often watched in shock as the open source driver quickly evolved to match the functionality of the closed-source driver, thanks to the magic of rapid collaborative development. Linux users have rejected the closed-source approach for a variety of reasons, and the marketplace has rejected products that do not have open source drivers available. This could be just a coincidence, but that is unlikely.

In recognition of the pain of installing a binary driver, this chapter describes only the open source solutions for 802.11 networking on Linux. Installing and troubleshooting open source drivers is possible on a wide variety of supported hardware, and it is a well-known procedure that many people use each day. Linux drivers for Lucent-based cards are now part of the standard Personal Computer Memory Card Industry Association (PCMCIA) distribution, and I expect that *linux-wlan* will eventually become part of the standard operating-system distribution. As with Windows drivers, installing wireless cards on Linux creates Ethernet interfaces. Many Linux drivers expose an Ethernet interface through the kernel. (Frequently, wireless interfaces

even have the *eth* prefix!) Programs can use the Ethernet interface to send and receive data at the link layer, and the driver handles Ethernet-to-802.11 conversions. Many of the things you would expect to see with an Ethernet interface remain the same. ARP works identically, and the IP configuration is done with the same utilities provided by the operating-system distribution. *ifconfig* can even be used to monitor the interface status and see the data sent and received.

## A Few Words on 802.11 Hardware

As with other devices running under Linux, the more you know about the hardware, the better off you are. Only a handful of 802.11 chipset manufacturers exist. Most vendors use chipsets produced by Intersil (*http://www.intersil.com*, formerly known as Harris Semiconductor). Intersil's industry-leading position is the result of the success of its PRISM chipset. The initial PRISM, whose name is an acronym for Programmable Radio in the ISM band, was a common solution for vendors seeking a 2-Mbps DSSS 802.11 solution. When 802.11b was standardized in 1999, Intersil brought out the PRISM-2 chipset, which supported the 5.5-Mbps and 11-Mbps data rates.

Intersil's chipsets are used by Linksys, Nortel/Netgear, D-Link, and SMC for interface cards. Several new laptops with integrated 802.11 support are hitting the market, and many of these wireless-enabled laptops are powered by Intersil chipsets. Choosing between the supported Intersil-based cards is a personal trade-off between price, performance, range, and other factors.

Lucent's Hermes chipset is the major competitor to Intersil's PRISM. Unlike Intersil, Lucent also produces interface cards for the end user. Lucent's cards were first to market, so they are quite common. Several vendors seeking to bring a wireless solution to market chose OEM Lucent gear; the most notable example is Apple's AirPort, which is based on Lucent technology. OEM and rebranding relationships are quite common. Table 13-1 divides the major industry vendors by the radio chipset used in each vendor's cards, which should help you in choosing which driver to use. (In some cases, the OEM relationship is little more than a new sticker. Vendor B simply takes Vendor A's equipment and slaps a new logo on it.)

*Table 13-1. The silicon behind the brand*

| Lucent chipset-based cards | Intersil PRISM-based cards |
|---|---|
| Lucent Wavelan/IEEE and Orinoco | Linksys |
| Cabletron/Enterasys RoamAbout | SMC |
| Apple AirPort | Compaq (WL100 and WL200) |
| Compaq (WL110, WL210, and WL215) | D-Link |
| IBM High Rate Wireless LAN | Nokia |
| HP 802.11b Wireless LAN | Cisco (Aironet) |
| Dell TrueMobile 1150 (earlier versions were OEM Aironet cards) | |

In addition to the radio chipset, cards must have a MAC controller. Most cards on the market use an Intersil MAC controller. Several first-generation cards used an AMD Am930 MAC controller but have switched to the integrated MAC controller in the PRISM-2 chipset. Cisco's Aironet product line uses an Aironet-developed MAC controller with a PRISM-2 radio chipset.

The PC Card form factor is the dominant form factor, though it is not used exclusively. PC Cards can be used directly in portable computers and can be plugged into PCMCIA interfaces for other purposes. Many, but by no means all, access points use PC Card slots instead of fixed radio interfaces, which, conveniently, also allows for upgrades by swapping out the radio interface. Even PCI-based 802.11 solutions use PC Cards. Typically, a PCI solution consists of a PC Card plus a PCI carrier card with a PC Card interface.

# PCMCIA Support on Linux

Most add-on 802.11 solutions for laptop computers are based on the PCMCIA form factor. Adding 802.11 support to Linux requires an understanding of how the PCMCIA subsystem in Linux is put together and how it works to enable drivers for PCMCIA cards.

## PCMCIA Card Services Overview

Card Services grew out of an attempt to simplify system configuration. Rather than dedicating system resources to individual devices, the host system maintained a pool of resources for PC Cards and allocated resources as necessary. Figure 13-1 shows the procedure by which cards are configured on Linux.

When a card is inserted, the *cardmgr* process orchestrates the configuration of the device, as shown in Figure 13-1. The orchestration pulls together system resources, kernel components, and kernel driver modules through the configuration files stored in */etc/pcmcia*. Roughly speaking, the host takes the following steps:

1. A card is inserted into an empty PC Card socket, and *cardmgr* is notified of this event. In addition to any hardware operations (such as supplying power to the socket), *cardmgr* queries the *card information structure* (CIS) to determine the type of card inserted and the resources it needs. For more information on the CIS, see the sidebar "Card Information Structure."
2. *cardmgr* consults the card database stored in */etc/pcmcia/config* to determine which card was inserted. Part of the configuration is to associate cards with a *class*. For the purposes of configuring network cards, the important point to note is that items in the *network* class have additional network configuration operations performed on them later. The card is identified by the CIS data from step 1, and the class setting is set in the main system configuration file. At this point,

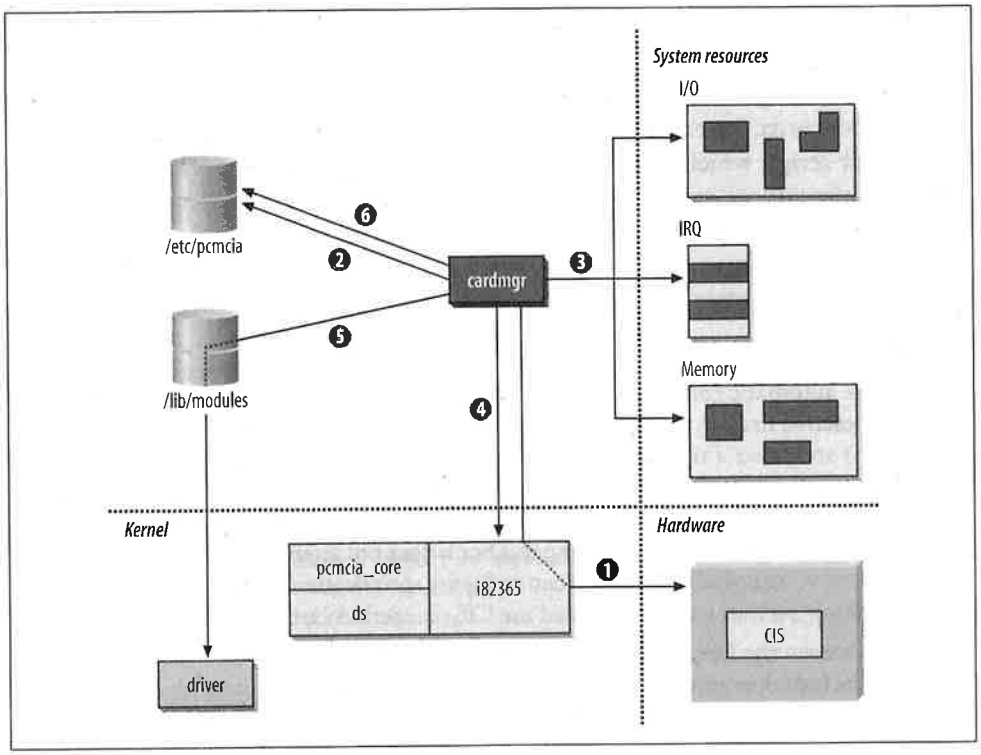


*Figure 13-1. Linux PCMCIA configuration system*

*cardmgr* beeps once. Successful identification results in a high-pitched beep; unsuccessful identifications are indicated by a beep of lower pitch.

3. *cardmgr* determines which resources are available to allocate to the card. Blocks of system resources are reserved for PCMCIA card use in the main configuration file, and *cardmgr* doles out resources as needed to cards. The number of I/O ports and the size of the memory window are obtained from the CIS.
4. Resources allocated by *cardmgr* are programmed into the PCMCIA controller, which is depicted in Figure 13-1 as interaction with the device driver. PCMCIA controllers implement *resource steering* to map resources required by the card onto available system resources. A card may ask for an interrupt, but the actual assigned interrupt is irrelevant. In operation, the card simply asks the PCMCIA controller to raise an interrupt, and the controller is responsible for looking up the interrupt assigned to the socket and firing the correct interrupt line.
5. Part of the configuration information obtained from the lookup in step 2 is the name of the device driver that should be loaded to use the newly inserted card. Drivers for PCMCIA cards are implemented as kernel modules. As part of the insertion process, the driver is informed of resources allocated in step 4. With

proper module dependencies, module stacking can be used to load multiple modules.

6. Further user-space configuration is performed based on the class of the device. Network cards, for example, have further configuration done by the */etc/pcmcia/network* script, which is configured by editing */etc/pcmcia/network.opts*. Successful configuration in this step generates a second high beep, and failure is reported with a low beep.

### Card Information Structure

To enable automatic configuration, every PC Card has a blob of data that enables the card to describe itself to the host system. The blob is called the *card information structure* (CIS) and takes a relatively straightforward link-list format. The building blocks of the CIS are called *tuples* because they have three components: a type code to identify the type of tuple, a length field, and a series of data bytes. Tuple formats range from trivial to highly complex, which is why this book does not attempt to classify them any further. Brave, stout-hearted readers can order the specification from the PCMCIA and use the *dump_cis* tool on Linux to read the CIS of inserted cards.

The CIS assists the host in automatic configuration by reporting information about itself to the host operating system. For example, network interface cards identify themselves as such, and the CIS enables the Card Services software to allocate the appropriate resources such as I/O ports and interrupt request (IRQ) lines. On Linux, the system administrator uses configuration files to match the CIS data to the driver.

## PCMCIA Card Services Installation

Installation of Card Services is documented in the PCMCIA-HOWTO. Both the software and documentation are available from *http://pcmcia-cs.sourceforge.net/*. Begin by unpacking the software and configuring it with *make config*.

Card Services asks for the current kernel source tree and picks relevant options out of it. In addition, it asks four questions. *Trusting* versions of the utilities allow operations by nonroot users, and are therefore set to no by default. *CardBus support* is included for newer cards, though no PRISM-2 cards currently use the CardBus interface. *PnP resource checking* is not required unless the computer hardware has plug-and-play hardware. Enabling this option can cause problems on computers that do not have PnP hardware and on several laptops. Finally, you must specify a module install directory. By default, modules are installed in */lib/modules/<version>/modules*, in which *<version>* is the version of the kernel source tree.

```
[root@bloodhound pcmcia-cs-3.1.28]# make config

    -------- Linux PCMCIA Configuration Script --------
```

```
The default responses for each question are correct for most users.
Consult the PCMCIA-HOWTO for additional info about each option.

Linux source directory [/usr/src/linux]:

The kernel source tree is version 2.2.19.
The current kernel build date is Sat Aug 25 07:11:52 2001.

Build 'trusting' versions of card utilities (y/n) [n]:
Include 32-bit (CardBus) card support (y/n) [y]:
Include PnP BIOS resource checking (y/n) [n]:
Module install directory [/lib/modules/2.2.19]:

Kernel configuration options:
    Symmetric multiprocessing support is disabled.
    PCI BIOS support is enabled.
    Power management (APM) support is enabled.
    SCSI support is disabled.
    IEEE 1394 (FireWire) support is disabled.
    Networking support is enabled.
     Radio network interface support is enabled.
     Token Ring device support is disabled.
     Fast switching is disabled.
     Frame Diverter is disabled.
    Module version checking is enabled.
    Kernel debugging support is disabled.
    /proc filesystem support is enabled.
    Maximum physical memory: 1GB

It looks like you have a System V init file setup.

X Windows include files found.
Forms library not installed.
    If you wish to build the 'cardinfo' control panel, you need the Forms
    library and the X Windows include files.  See the HOWTO for details.

Configuration successful.
```

After configuration, the installation can be accomplished with *make install*. Most configuration information is automatically detected from the environment without difficulty, though you need to set the PCIC environment variable to the correct type of PCMCIA controller. Two major types of controller are supported: the Databook TCIC-2 and compatibles, and the Intel i82365SL and compatibles. On RedHat and derived distributions, the variable is set in */etc/sysconfig/pcmcia*. Nearly all systems should set it to *i82365*.

## Troubleshooting Resource Conflicts

One of the revolutionary developments hyped by PCMCIA card vendors was that users were no longer directly responsible for maintaining low-level hardware configurations on IBM-compatible hardware. In many respects, this hype was drastically

overblown because users are still responsible for maintaining the resource pools used by PCMCIA Card Services to draw from for automatic configuration, and therefore they must still be familiar with the hardware configuration. Three major resources are managed by Card Services for users: IRQ lines, I/O ports, and direct memory access (DMA) channels. DMA channels are not required by network cards, however, and are not discussed in this section.

### IRQs

IRQs are used by devices that must use of the CPU periodically. Interfaces use IRQs so that when a buffer fills, the system CPU can be notified and drain the buffer. One limitation of the PC architecture is that it has only 15 available IRQs, and many are occupied by standard hardware. Table 13-2 shows common IRQ usage, which may help you determine which IRQs are available for PCMCIA cards. Disabling any extra components frees the IRQ. Table 13-2 also shows common IRQ settings on PC hardware. As a rule of thumb, IRQs 3, 5, and 10 are readily available on most machines.

*Table 13-2. Common IRQ settings*

| IRQ number | Common usage | Purpose |
|---|---|---|
| 0 | System timer | Fires 18 times per second to maintain coarse clocking. |
| 1 | Keyboard | Allows operating system to monitor keyboard strokes by user. |
| 2 | Cascade | Two interrupt controller chips are used; the second controls IRQs 8–15 and is wired into IRQ 2 on the primary. |
| 3 | Second serial port | The second and fourth serial ports (COM2 and COM4 under Windows) both use IRQ 3. If only one serial port is present, IRQ 3 may be used by expansion devices. |
| 4 | First/third serial port | The first and third serial ports (COM1 and COM3 under Windows) both use IRQ 4. Generally, it is not a good idea to use IRQ 4 for expansion devices because loss of the serial port also means that terminal-emulation software cannot be used. |
| 5 | Second parallel port | Most systems have only one parallel port, but IRQ 5 is also commonly used for sound cards. |
| 6 | Floppy controller | All systems have floppy disks, which can be especially important on portable computers. |
| 7 | First parallel port | The first parallel port can frequently be disabled on laptops without an issue, unless the parallel port is used extensively for printing. |
| 8 | RTC | The Real-Time Clock maintains finer-grained timers |
| 9 | Video (older systems) | Older systems required an IRQ for the video controller, and it was typically assigned to IRQ 9. Most video controllers are now on the PCI bus and do not require a dedicated IRQ. |
| 10 | | Usually available for expansion devices. |
| 11 | Usually PCI bus or SCSI controller | Generally not available for expansion devices. |
| 12 | Usually PS/2 mouse port | Generally not available. |

*Table 13-2. Common IRQ settings (continued)*

| IRQ number | Common usage | Purpose |
|---|---|---|
| 13 | FPU | The floating-point unit IRQ is used by the math coprocessor, even on systems with a CPU with an integrated math coprocessor such as the Pentium series. |
| 14 | Primary IDE | The first IDE channel is used by the main system hard disk, and thus IRQ 14 is almost never available on a portable system. |
| 15 | Secondary IDE | Portable systems typically place the CD-ROM on IRQ 15, making this IRQ unavailable for use. |

## I/O ports

I/O addresses are used for bidirectional communication between the system and a peripheral device. They tend to be somewhat poorly organized, and many devices have overlapping defaults. Each I/O port can be used to transfer a byte between the peripheral device and the CPU. Most devices require the ability to transfer multiple bytes at a time, so a block of ports is assigned to the device. The lowest port number is also called the base I/O address. A second parameter describes the size of the I/O window. Table 13-3 lists some of the common port assignments. Refer to your hardware vendor's documentation for details on additional devices such as IR ports, USB controllers, the PCMCIA controller, and any resources that may be required by motherboard components.

*Table 13-3. Common I/O ports*

| Device name | I/O range (size) |
|---|---|
| **Communication ports** | |
| First parallel port | 0x3bc–0x3bf (4) |
| First serial port | 0x3f8–0x3ff (8) |
| Second serial port | 0x2f8–0x2ff (8) |
| **Disk drives** | |
| Primary IDE | master: 0x1f0–0x1f7 (8) |
| | slave: 0x3f6–0x3f7 (2) |
| Secondary IDE | master: 0x170–0x177 (8) |
| | slave: 0x376 (1) |
| Floppy controller | 0x3f0–0x3f5 (6) |
| **Input devices** | |
| Keyboard | 0x060 (1) |
| | 0x064 (1) |
| **Multimedia/gaming** | |
| Sound card | 0x220–0x22f (16) |
| | FM Synth: 0x388–0x38b (4) |
| | MIDI: 0x330–0x331 (2) |

*Table 13-3. Common I/O ports (continued)*

| Device name | I/O range (size) |
|---|---|
| Joystick/Game port | 0x200–0x207 (8) |
| **System devices** | |
| Interrupt controllers | 0x020–0x021 (2) |
| | 0x0a0–0x0a1 (2) |
| DMA controllers | DMA channels 0-3: 0x000–0x00f (16) |
| | Page registers: 0x080–0x08f (16) |
| | DMA channels 4-7: 0x0c0–0x0df |
| CMOS/real time clock | 0x070–0x073 (4) |
| Speaker | 0x061 |
| Math coprocessor | 0x0f0–0x0ff (16) |

# linux-wlan-ng for Intersil-Based Cards

The most commonly used driver for PRISM-based cards is the *linux-wlan* open source driver developed by Absolute Value Systems. In a shining example for other vendors, Intersil has actively supported the *linux-wlan* project. Two subprojects exist. The original *linux-wlan* code supports cards based on the original PRISM chipset, which generally run at 2 Mbps, and the newer *linux-wlan-ng* codebase supports cards based on the PRISM-2 chipset running at 11 Mbps. *linux-wlan-ng* supports most cards based on the PRISM-2 chipset.* The examples in this section were written with a Linksys WPC11 as the example card, largely because of its affordability relative to other PRISM-2 cards.

At this point, most users will be interested in 11-Mbps cards supported by *linux-wlan-ng*. Both projects aim toward developing a complete 802.11 layer for Linux. Currently, both projects support simple encapsulation of 802.11 frames and RFC 1042 encapsulation. In practice, this poses a problem only with strange access points that support only 802.1h translation.

## Prerequisites

Before starting the installation of *linux-wlan-ng*, there are some housekeeping tasks to take care of. Compile-time configuration is taken care of in part by pulling some configuration information out of the kernel configuration and the PCMCIA utility configuration. Most distributions include the option of installing source code but do

* Nokia's C11x cards are based on the PRISM-2, but the addition of the smart-card reader changed the programming interface enough so that the C11x is not supported by *linux-wlan-ng*.

not include the configuration used to build the kernel or any of the additional software packages.

Officially, *linux-wlan* is developed against kernels later than 2.0.36 and PCMCIA Card Services (*pcmcia-cs*) later than 3.0.9, but *linux-wlan-ng* was largely developed against kernel 2.2 and PCMCIA services 3.1. Compiling *linux-wlan-ng* requires configured source trees for both the kernel and PCMCIA, so during the preinstallation process, it is a good idea to update both components to the latest versions available. Refer to the documentation bundled with *linux-wlan-ng* to get an idea of the software environments it has been tested against and upgrade to current versions. Kernel configuration and compilation has been extensively documented elsewhere; refer to the documentation for your distribution. Card Services software is now hosted at *http://pcmcia-cs.sourceforge.net*, and documentation is bundled with the source package. The examples in this section were written using kernel 2.2.19 and *pcmcia-cs* 3.1.28, though there is nothing specific about them.

### Kernel compilation

Because the configuration in the kernel source tree is used by the subsequent configuration of both PCMCIA Card Services and *linux-wlan*, it must match the currently running kernel. Many problems can be avoided, and countless hours can be saved if you begin the process by downloading clean kernel source and configuring from scratch. *linux-wlan* does not require any specific options for support other than general network support, but it is useful for other reasons to enable certain options. The ISC Dynamic Host Configuration Protocol (DHCP) client, for example, requires that both the packet socket (*CONFIG_PACKET*) and socket filtering (*CONFIG_FILTER*) be enabled. Some applications of the driver also are enabled by the kernel/user network link driver (*CONFIG_NETLINK*). Kernel configuration and compilation is extensively documented in both online and offline resources and may have been installed with your documentation. Kernel source code is available from *ftp://ftp.kernel.org* and countless mirror sites worldwide.

## Compiling and Installing linux-wlan-ng

The first step is to get the software from *http://www.linux-wlan.org*. Version numbering follows the Linux kernel convention, with odd minor numbers used for development versions and even minor numbers used for stable production versions. The software can be obtained by FTP from *ftp://ftp.linux-wlan.org/pub/linux-wlan-ng/*; the latest version as this book was written was 0.1.8-pre13.

### Compile-time configuration

Run *make config* to configure the software. A configuration script is run, and you need to supply the location of your configured kernel source and PCMCIA Card Services

source trees. A few minor questions follow; these guidelines may help you answer those questions:

- Building with debugging information compiled in is a good idea when you're troubleshooting. Debugging can be disabled easily, and the code is a minuscule performance hit when compiled in and not used.
- Kernel PCMCIA is the PCMCIA software included with the 2.4 kernel series. Users of kernel Version 2.4 need to choose whether to use Kernel PCMCIA or PCMCIA Card Services; users of earlier kernels must use PCMCIA Card Services.
- PLX-based adapters are PCI wireless adapters. Most users will use PCMCIA cards and can disable PLX support.
- Card Services drivers support PCMCIA cards and should be enabled.

A sample configuration looks something like this:

```
[gast@bloodhound linux-wlan-ng-0.1.8-pre13]$ make config
-------------- Linux WLAN Configuration Script ------------
The default responses are correct for most users.
Linux source directory [/usr/src/linux]:
The kernel source tree is version 2.2.19.
The current kernel build date is Sat Aug 25 07:11:52 2001.
pcmcia-cs source dir [/usr/src/pcmcia-cs-3.1.28]:
Alternate target install root directory on host []:
  Module install directory [/lib/modules/2.2.19]:
PCMCIA script directory [/etc/pcmcia]:
It looks like you have a System V init file setup.
Target Architecture? (i386, ppc, or alpha) [i386]:
Prefix for build host compiler? (rarely needed) []:
Compiling with a cross compiler? (y/n) [n]:
Build for debugging (see doc/config.debug) (y/n) [n]: y
Build for Kernel PCMCIA? (y/n) [n]:
Build PLX???? based PCI (_plx) adapter drivers? (y/n) [n]:
Build PCMCIA Card Services (_cs) drivers? (y/n) [y]:
Configuration successful.
```

### Building the software

Run *make all* to build the software. After some compilation messages, four main components are compiled:

*wlanctl-ng*
: The general administrative utility used to control the current running configuration and change the state of the hardware.

*wlancfg*
: Used to change values in the management information base and alter the configuration.

*p80211.o*

A generic 802.11 utility layer for Linux, implemented as a kernel module. It includes data structures required to handle buffers, build frames, translate between the supported frame types (RFC 1042-encapsulated, 802.1h, vanilla encapsulated, and the raw packet buffers used by the kernel), interact with the network devices in the kernel, and perform the functions placed in the MAC Layer Management Entity (MLME).

*prism2_cs.o*

The kernel module driver for PRISM-2–based cards. It implements the hardware interface to the Intersil HFA384x chipset used in supported cards, implements hardware interrupt service, allows user-space utilities to gather statistics and counters data from the hardware, and enables the management operations discussed in Chapter 7, such as scanning, joining, authentication, and association.

### Installing the software

After compilation, become *root* and run *make install* to install the software. By default, the two administration programs (*wlanctl-ng* and *wlancfg*) are placed in */sbin*, and the two modules are installed in the module directory tree that corresponds to your current kernel version. On kernel Version 2.2.19, for example, *p80211.o* would be installed in */lib/modules/2.2.19/net* and *prism2_cs.o* would be placed in */lib/modules/2.2.19/pcmcia*.

The installation also places some configuration files in the main PCMCIA Card Services directory, which is */etc/pcmcia* by default. *wlan-ng.conf* contains card definitions for hardware known to work with *linux-wlan-ng*. It is copied to the Card Services directory, and a directive is put at the end of the main card definition file, */etc/pcmcia/config*, to include the PRISM-2 card definitions. The configuration script for wireless interfaces, *wlan-ng*, is also placed in */etc/pcmcia*. *wlan-ng* pulls in configuration options from an auxiliary option file, *wlan-ng.opts*. If either *wlan-ng* or *wlan-ng.opts* exists, the older versions will be backed up and will receive the suffix *.O*.

## Administration of the 802.11 Interface

The main administrative tool in *linux-wlan-ng* is *wlanctl-ng*, a tool to control the internal state of the driver through *ioctl()* commands. As an example, consider the *wlanctl-ng* command used to join a network:

```
[root@bloodhound]# /sbin/wlanctl-ng wlan0 lnxreq_autojoin ssid="Luminferous Ether" \
authtype=opensystem
message=lnxreq_autojoin
  ssid=Luminferous Ether
  authtype=opensystem
  resultcode=success
```

Commands operate on an interface, which is supplied as the first argument. The command to perform is the second argument. In the join command, two additional arguments are required. One is the SSID, so the kernel driver can identify the Beacon frames corresponding to the desired network. SSIDs with spaces can be used, provided that quotation marks are used to delimit the SSID. The other argument is the authentication type. Shared-key authentication is required if the WEP key is set.

*wlanctl-ng* commands fall into three broad classes. Commands that begin with *dot11req_*, such as *dot11req_mibset*, work with data structures specified in 802.11. The second class of commands, which begin with *p2req_*, work with the PRISM-2 chipset. A third class, prefaced with *lnxreq_*, are commands specific to the Linux driver.

WEP keys are set by writing the MIB variables corresponding to the default key on the interface the key must be set for:

```
[root@bloodhound]# wlanctl-ng wlan0 dot11req_mibset \
mibattribute=dot11WEPDefaultKey3=01:02:03:04:05
message=dot11req_mibset
  mibattribute=dot11WEPDefaultKey3=01:02:03:04:05
  resultcode=success
```

Some commands are not implemented and return a result code of *not implemented*. One such command is *dot11req_scan*, which should run a generalized active scan.

One command that network administrators are likely to find near and dear is the command that enables promiscuous mode packet capture.* The driver command is *lnxreq_wlansniff*, and it takes two additional arguments. One is the channel to sniff on, and the final argument is the enable flag, which can be used to enable and disable the sniffing functionality.

```
[root@bloodhound]# wlanctl-ng wlan0 lnxreq_wlansniff channel=1 enable=true
message=lnxreq_wlansniff
  enable=true
  channel=1
  resultcode=success
```

For convenience, I have written shell functions around the *lnxreq_wlansniff* command to cut down on typing. Both shell functions take two arguments: the interface name and the channel. Some users may not use multiple 802.11 cards and may choose to have the shell function take just one argument instead. In any case, here are the functions I use, which you are free to customize:

```
wlan-promisc ()
{
  /sbin/wlanctl-ng $1 lnxreq_wlansniff channel=$2 enable=true
}
```

* Promiscuous mode packet capture was disabled in *linux-wlan*-0.1.7 and later, but patches to enable the functionality are available. See Chapter 16 for details.

```
wlan-normal ()
{
  /sbin/wlanctl-ng $1 lnxreq_wlansniff channel=$2 enable=false
}
```

To activate promiscuous packet capture on channel 5, I run *wlan-promisc wlan0 5*. When promiscuous capture is activated, it is not possible to use the network for communication. To restore network connectivity, simply run *wlan-normal wlan0 5*. With only one wireless interface, the shell functions can be simplified even further by hardcoding *wlan0* as the interface used by the function and taking only the channel number as an argument.

### IP addresing

Setting IP addresses is distribution-specific. On RedHat and derivative distributions, including Mandrake, configuration is stored in */etc/sysconfig/network-scripts*, with data for each interface shown in *ifcfg-interface*. *linux-wlan-ng* interfaces begin with *wlan*: *wlan0*, *wlan1*, and so forth. An example static IP configuration for the first wireless LAN interface would hold IP address and netmask information just as with any other interface:

```
[root@bloodhound network-scripts]# cat ifcfg-wlan0
DEVICE=wlan0
BOOTPROTO=static
IPADDR=192.168.200.125
NETMASK=255.255.255.0
NETWORK=192.168.200.0
BROADCAST=192.168.200.255
ONBOOT=no
```

Bringing up the interface is a matter of running *ifup* with the interface name as an argument as *root*. Most Linux kernels are built without bridging support and cannot by default be configured with two interfaces on the same logical IP subnet. This is often a danger for stations with wireless LAN interfaces because the wireless interface is sometimes used to replace an existing wired interface.

```
[root@bloodhound network-scripts]# ifup wlan0
```

Dynamic IP configuration with DHCP is allowed; refer to your distribution's documentation for details on enabling dynamic boot protocols. DHCP configuration depends on forming a successful association with an access point. If the association fails, the DHCP discovery packets result in "not associated" error messages in the log.

## Configuring linux-wlan-ng

Card definitions are stored in */etc/pcmcia/config*. At the end of the file, there is a line that reads *source ./*.conf*. This line enables additional PCMCIA driver packages to add card definitions. *linux-wlan-ng* adds the file *wlan-ng.conf*, which is a series of

driver associations. As an example, consider the definition of the Linksys WPC11 card:

```
card "Linksys WPC11 11Mbps 802.11b WLAN Card"
   version "Instant Wireless ", " Network PC CARD", "Version 01.02"
   bind "prism2_cs"
```

The first line is a text string used to identify the card to the human user, and the *bind* directive specifies which driver will be loaded to support the card. The *version* strings are used to match the CIS of the card, which can be viewed using the *dump_cis* utility:

```
[root@bloodhound]# dump_cis
Socket 0:
  dev_info
    NULL 0ns, 512b
  attr_dev_info
    SRAM 500ns, 1kb
  vers_1 5.0, "Instant Wireless ", " Network PC CARD", "Version 01.02",
    ""
  manfid 0x0156, 0x002
  funcid network_adapter
  lan_technology wireless
  lan_speed 1 mb/sec
  lan_speed 2 mb/sec
  lan_speed 5 mb/sec
  lan_speed 11 mb/sec
  lan_media 2.4_GHz
  lan_node_id 00 04 5a 0d 0c 70
  lan_connector Closed connector standard
  config base 0x03e0 mask 0x0001 last_index 0x01
  cftable_entry 0x01 [default]
    Vcc Vmin 4750mV Vmax 5250mV Iavg 300mA Ipeak 300mA
    Idown 10mA
    io 0x0000-0x003f [lines=6] [16bit]
    irq mask 0xffff [level] [pulse]

Socket 1:
  no CIS present
```

Note that the CIS specifies how large an I/O window is necessary. In this case, 64 (0x3F) bytes of I/O space are required by the card. When the card is inserted, the kernel configuration prints out the resources assigned to the driver in */var/log/kernel/info*:

```
Oct 26 21:03:38 bloodhound kernel: prism2_cs: index 0x01: Vcc 5.0, irq 10, io
0x0100-0x13f
```

In addition to the card associations in *wlan-ng.conf*, the configuration script *wlan-ng* and its option file *wlan-ng.opts* are installed in */etc/pcmcia*. *wlan-ng* is called whenever a PRISM-2 wireless card is inserted. It identifies the card, associates with an access point or starts an independent BSS if desired, and initializes the network interface using whatever hooks are provided by the Linux distribution. *wlan-ng.opts* is used to configure the desired SSID and WEP keys.

### Selecting a network to join

To configure the network to join, some parameters in *wlan-ng.opts* must be changed. One section of the script is labeled "STA START" and contains the network parameters AuthType and DesiredSSID:

```
#========STA START=====================================
AuthType="opensystem"
DesiredSSID="linux-wlan"
```

AuthType can be set to *opensystem* for a network not employing access control, and *sharedkey* for networks that use the shared-key authentication mechanism of WEP. The *DesiredSSID* parameter must be set to the SSID announced by the Beacon frames for the network that the station will join.

### Configuring WEP

Stations that use WEP must also configure WEP-related parameters in *wlan-ng.opts*:

```
#========WEP===========================================
dot11PrivacyInvoked=true
dot11WEPDefaultKeyID=2

dot11WEPDefaultKey0=01:02:03:04:05
dot11WEPDefaultKey1=01:02:03:04:05:06:07:08:09:0a:0b:0c:0d
dot11WEPDefaultKey2=
dot11WEPDefaultKey3=
```

Several settings matter only for access points; the settings that matter for stations are:

*dot11PrivacyInvoked*
: This must be set to `true` to enable WEP. If it is set to `false`, WEP is disabled.

*dot11WEPDefaultKeyID*
: This is set to 0, 1, 2, or 3, depending on which of the WEP keys are used on the network.

*Four dot11WEPDefaultKeyXs (in which X is 0, 1, 2, or 3)*
: These are used to program the WEP keys themselves. Keys are entered like MAC addresses, in which each byte is entered as a two-character hexadecimal number, and the bytes are separated by colons.

### Enabling debugging output

If the *linux-wlan-ng* module was built with debugging code enabled, the PCMCIA Card Services system can be set up to pass configuration options to the driver. Module options are configured in */etc/pcmcia/config.opts*, using the *opts* parameter to the *module* directive, like this:

```
module "prism2_cs" opts "prism2_debug=3"
```

Debugging can be enabled for both the protocol module (*p80211*) and the PRISM-2 hardware driver (*prism2_cs*). Debugging of the protocol module is enabled with the

*wlan_debug* option, and debugging of the driver is enabled with the *prism2_debug* option. Five debugging levels are defined, with higher levels increasing the log output; Table 13-4 describes them in detail. Debugging levels are cumulative, so level 3 includes all messages from level 2 plus additional information.

*Table 13-4. linux-wlan-ng debug levels*

| Level | Meaning |
|---|---|
| 1 | Error messages for rare exceptions. |
| 2 | More exceptions are logged. |
| 3 | Basic status output. |
| 4 | Additional status output. |
| 5 | Function entry and exit. |

After making changes to *config.opts*, the card manager must be restarted so that the options are passed to the driver when the card is inserted. The easiest way to restart *cardmgr* is to send it the HUP signal:

```
kill -HUP `ps aux | grep cardmgr | grep -v grep | awk '{ print $2 }'`
```

## Using linux-wlan-ng

Using wireless LAN interfaces driven by *linux-wlan-ng* is no different from using regular wired Ethernet interfaces. Configuration is identical to wired interfaces, and operations are similar. System administrators can, for the most part, treat 802.11 interfaces like Ethernet interfaces, but with higher latency. The higher latency is due partly to the lower bit rate of the wireless medium and partly to the requirement for fully acknowledged frame exchanges.

## Common Problems

Most common problems are really resource conflicts, but resource conflicts may manifest themselves as a variety of error messages. Tracking down resource conflicts is a large task described in more detail in the section "Troubleshooting Resource Conflicts." Resource conflict tracking is often a tedious process of listing out the resources commonly used by a system and attempting to identify the conflict. It can be particularly annoying on laptop hardware because of the number of built-in devices that may be sucking up resources without extensive notification. This section describes the common problems that users may face before the next section delves into the hairy business of resolving resource conflicts.

### Compilation difficulties

Most compilation difficulties can be traced to relying on an improper source tree for a dependent package. The most reliable way to install *linux-wlan-ng* is to first build

both a new kernel and new PCMCIA package from a clean source before attempting to build *linux-wlan-ng*. Although this takes time, it does guarantee that the compile-time configuration stored in the source tree matches the running version, and it will, therefore, eliminate most compilation problems.

### Cards are identified as "anonymous memory card"

Card Services on Linux attempts to identify cards based on configuration information stored on the card itself in the CIS on the PCMCIA card. If Card Services is unable to identify a card, as a last resort, it identifies a card as an anonymous memory card. This error message almost always means that there is a resource conflict; as a result of the conflict, Card Services software is unable to communicate with the card to read the identifying data structures on the card. Vary the resource exclusion ranges to resolve the conflict; see "Troubleshooting Resource Conflicts" earlier in this chapter.

### "Tx attempt prior to association, frame dropped" error message

This message is quite self-explanatory: a frame was queued for transmission before the station successfully associated with an access point. Several things might cause this error:

- If the desired SSID is not found, no association is made.
- Authentication is a precondition of association. If the authentication type is mismatched or the WEP key used for authentication is incorrect, the association fails.
- Resource conflicts can interfere with the sending and receiving of frames, which may cause transmissions (and therefore authentication or associations) to fail. Reconsider resource allocations and PCMCIA Card Services configuration.

### Odd behavior from the lights on the card

*linux-wlan-ng* does not try to control the lights on the card in a consistent way between different card vendors. Generally speaking, one light becomes solid when the station associates with an access point, and a second light blinks to indicate traffic. If your card behaves differently, don't panic. Check at the access point for an association; if an association exists, then the card is functioning normally but uses the lights in a different manner. Naturally, the truly brave may attempt to rewrite the driver, but for most of the world, it is easier simply to note the behavior of the lights.

### Dropped sessions

Associations may be dropped for a variety of reasons, many of which are not the fault of the client wireless LAN card. One frequent cause of this behavior is access

points that conserve resources by timing out idle associations. Nokia access points are notable for this behavior.

The original *linux-wlan* package for PRISM-1–based cards did not reassociate. *linux-wlan-ng* attempts to reassociate when an association times out, but it may not always be successful. If timeouts must always be avoided, run a script on the wireless station to send traffic periodically to prevent the association from timing out.

#### "MAC initialization failure, result=xxx"

The first major task for the PRISM-2 driver is to initialize the MAC controller chip on the card. The error code is one major clue. Positive result codes indicate that the firmware is at fault, and negative error codes are due to driver faults. Frequently, the error is due to a timeout period expiring; retry the operation to ensure that the error is caused by the system and not by a timeout expiration. Firmware problems are rare and generally seen only when the firmware loaded on the card was bad or the firmware load was incomplete. Driver problems are usually due to resource conflicts, especially with I/O ports. In some cases, a laptop with 5-volt PCMCIA slots may experience problems with a 3-volt card, though the failure usually happens later in the initialization process. I saw this message on an IBM ThinkPad T21 when the card was inserted, and the system was on battery power. If the T21 was plugged into AC power, however, the driver would load and configure without a hitch.

## Agere (Lucent) Orinoco

Wireless networking is far older than the 802.11 standard. The first commercial wireless network hardware was manufactured by the WaveLAN division of AT&T and gained market acceptance in the early 1990s. When Lucent was spun off from AT&T in the late 1990s, the WaveLAN division, like most communications product manufacturing at AT&T, was made a part of Lucent.

Early WaveLAN hardware was a completely proprietary system. After 802.11 was finally standardized in 1997, new hardware that complied with the standard was sold under the WaveLAN brand. To distinguish the standards-compliant cards from the proprietary cards, the former were called "WaveLAN IEEE" cards, while the latter were simply "WaveLAN" cards.

As the market for 802.11 hardware continued to develop, Lucent decided to rename the WaveLAN division under another brand name. The new name, Orinoco, comes from the third largest river system in the world. During the rainy season in South America, the Orinoco swells with fresh rainfall flowing in from over 200 tributaries. At its peak, the river grows to over 10 miles wide and more than 300 feet deep. Nearly 1,000 miles of the Orinoco's 1,300 miles are navigable; it is no wonder that the river's name is derived from the native words for "a place to paddle."

Lucent's initial strategy for support on open source platforms was to offer a choice of drivers. Two drivers were available. A closed-source proprietary binary driver, *wavelan2_cs*, provided full functionality largely equivalent to the drivers available for other platforms. A second, less functional open source driver, *wvlan_cs*, was made available under the GPL. For a variety of reasons, the proprietary driver was shunned in favor of the open source driver, and a devoted group of programmers enhanced the open source driver until it was equivalent to the proprietary driver.

Two open source drivers are now available for Lucent cards on Linux, depending on the kernel version employed. Kernel Versions 2.0 and 2.2 use the older *wvlan_cs* driver bundled with the *pcmcia-cs* package; kernel Version 2.4 uses the *orinoco_cs* driver in the kernel. The older *wvlan_cs* driver was based on a low-level library provided by Lucent, which was difficult to maintain and was a recurring source of bugs. Rather than continuing along an evolutionary dead-end, the maintainers of *wvlan_cs* learned from the *wlan-ng* driver and rewrote the lower layers of *wvlan_cs*. The result was *orinoco_cs*, a much more stable and robust driver. (*orinoco_cs* was originally known as *dldwd_cs*, which stood for David's Less Dodgy WaveLAN Driver!) *wvlan_cs* is no longer being maintained, and new installations should use *orinoco_cs* instead. Some Linux distributions may even ship with *orinoco_cs*.

## Kernel 2.0/2.2: wvlan_cs

In kernel Version 2.2 and earlier, support for WaveLAN adapters is provided by the *wvlan_cs* driver bundled with PCMCIA Card Services for Linux.

### Compiling and installing

Some wireless LAN support code is activated in the kernel by enabling the Wireless LAN extensions (*CONFIG_NET_RADIO*) kernel configuration option. *CONFIG_NET_RADIO*-enabled kernels collect wireless statistics and expose additional data structures used by the WaveLAN driver.

The driver for PCMCIA WaveLAN cards is part of the PCMCIA Card Services package. Previous sections of this chapter described how to compile and install the PCMCIA Card Services package and how PCMCIA cards are dynamically configured on Linux systems. After building a kernel with the wireless LAN extensions, you must rebuild the PCMCIA utilities as well.

The standard wireless tools are also required. They can be downloaded from *http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html*. Building the wireless tools is a straightforward task. After unzipping the file into a temporary directory, just run *make* in the root directory of the software distribution, and it will build the wireless tools.

### PCMCIA configuration

Although the Orinoco card is rebadged on the outside, the CIS still identifies the card as a Lucent WaveLAN/IEEE, and the software is the same as it has always been for the WaveLAN/IEEE:

```
[root@bloodhound]# dump_cis
Socket 0:
  dev_info
    NULL 0ns, 512b
  attr_dev_info
    SRAM 500ns, 1kb
  vers_1 5.0, "Lucent Technologies", "WaveLAN/IEEE", "Version 01.01", ""
  manfid 0x0156, 0x002
  funcid network_adapter
  lan_technology wireless
  lan_speed 1 mb/sec
  lan_speed 2 mb/sec
  lan_speed 5 mb/sec
  lan_speed 11 mb/sec
  lan_media 2.4_GHz
  lan_node_id 00 02 2d 38 82 e4
  lan_connector Closed connector standard
  config base 0x03e0 mask 0x0001 last_index 0x01
  cftable_entry 0x01 [default]
    Vcc Vmin 4750mV Vmax 5250mV Iavg 300mA Ipeak 300mA
    Idown 10mA
    io 0x0000-0x003f [lines=6] [16bit]
    irq mask 0xffff [level] [pulse]

Socket 1:
  no CIS present
```

When the card is inserted, the */etc/pcmcia/wireless* script is run, using the configuration options in */etc/pcmcia/wireless.opts*. The wireless script is a frontend to the *iwconfig* program. Editing fields in *wireless.opts* sets the arguments to *iwconfig*. Therefore, knowing how *iwconfig* works allows you to set the appropriate fields in *wireless.opts*.

### iwconfig

The main command-line tool for managing a WaveLAN wireless interface is *iwconfig*. When run with no parameters other than the interface name, *iwconfig* displays extended information about the radio interface, provided the kernel was built with radio extensions:

```
[root@bloodhound]# iwconfig wvlan0
wvlan0    IEEE 802.11-DS  ESSID:"Luminiferous Ether"  Nickname:"HERMES I"
          Mode:Managed  Frequency:2.457GHz  Access Point:00:E0:03:04:18:1C
          Bit Rate:2Mb/s   Tx-Power=15 dBm   Sensitivity:1/3
          RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
```

```
Link Quality:46/92 Signal level:-51 dBm  Noise level:-94 dBm
Rx invalid nwid:0  invalid crypt:0  invalid misc:0
```

The nickname is an internal name used by the driver. By default, it is set to "HERMES I" after the name of the Lucent chipset used in Lucent's wireless LAN cards.

### Setting the network name

The basic task required to join a network is to select the appropriate network name, or SSID. *iwconfig* uses the *essid* parameter to set the desired network name. If the network name includes a space, it must be enclosed in quotation marks:

```
[root@bloodhound]# iwconfig wvlan0 essid "Luminiferous Ether"
```

### Setting the network channel

The network-operating frequency can be selected in two ways. The *freq* parameter can take an operating frequency directly, or the *channel* parameter can be used with the appropriate channel number, and the driver will derive the frequency from the channel number. The following two commands are equivalent:

```
[root@bloodhound]# iwconfig wvlan0 freq 2.432G
[root@bloodhound]# iwconfig wvlan0 channel 4
```

### Setting the network mode and associating with an access point

Most 802.11 stations are in either ad hoc networks or infrastructure networks. The *iwconfig* nomenclature for these two modes is *Ad-hoc* and *Managed*. Select between them by using the *mode* parameter:

```
[root@bloodhound]# iwconfig wvlan0 mode Ad-hoc
[root@bloodhound]# iwconfig wvlan0 mode Managed
```

For stations in an infrastructure network, the *ap* parameter may be used to request an association with the specified MAC address. However, the station is not required to remain associated with the specified access point and may choose to roam to a different access point if the signal strength drops too much:

```
[root@bloodhound]# iwconfig wvlan0 ap 01:02:03:04:05:06
```

### Setting the data rate

Most cards support multiple bit rates. *iwconfig* allows the administrator to choose between them by using the *rate* parameter. Bit rates can be specified after the *rate* parameter, or the keyword *auto* can be used to specify that the card should fall back

* I use the term SSID in this book to refer to a network name. Some drivers, including the WaveLAN drivers, use ESSID instead. The distinction is that an ESSID is a network name assigned to an extended service set, not any old service set.

to lower bit rates on poor-quality channels. If *auto* is combined with a bit rate, the driver may use any rate lower than the specified rate:

```
[root@bloodhound]# iwconfig wvlan0 rate 11M auto
```

### Configuring WEP

The key parameter controls the WEP function of the driver. Keys can be entered as hexadecimal strings as in the PRISM driver. Enter the string four digits at a time with dashes between the two-byte groups. (Although it is not mentioned in the documentation, I can enter keys in the colon-separated MAC address format as well.)

```
[root@bloodhound]# iwconfig wvlan0 key 0123-4567-89
[root@bloodhound]# iwconfig wvlan0 key 01:23:45:67:89
```

Multiple keys can be entered using a bracketed index number:

```
[root@bloodhound]# iwconfig wvlan0 key 0123-4567-89
[root@bloodhound]# iwconfig wvlan0 key 9876-5432-01 [2]
[root@bloodhound]# iwconfig wvlan0 key 5432-1678-90 [3]
```

Longer keys can be entered simply by using more bytes. If the key length is longer than 40 bits, the key is assumed to be a 104-bit key.

```
[root@bloodhound]# iwconfig wvlan0 key 0011-2233-4455-6677-8899-0011-22 [4]
```

Once multiple keys have been entered, select one by entering the index number without a key value:

```
[root@bloodhound]# iwconfig wvlan0 key [2]
```

Activate WEP processing using `key on` and disable WEP using `key off`. These can be combined with an index number to select a new WEP key:

```
[root@bloodhound]# iwconfig wvlan0 key [3] on
[root@bloodhound]# iwconfig wvlan0 key off
```

Finally, two types of WEP processing can be done. An *open* system accepts data frames sent in the clear, and a *restricted* system discards cleartext data frames. Both of these parameters can be combined with an index number:

```
[root@bloodhound]# iwconfig wvlan0 key [4] open
[root@bloodhound]# iwconfig wvlan0 key [3] restricted
```

The *key* parameter may also be accessed with the *encryption* parameter, which may be abbreviated as *enc*.

### Tuning 802.11 parameters

*iwconfig* allows you to tune the RTS and fragmentation thresholds. The RTS threshold in the *wvlan_cs* driver is 2,347, which effectively disables RTS clearing. In an environment likely to have hidden nodes, it can be set using the *rts_threshold* parameter with *iwconfig*. *rts_threshold* can be abbreviated as *rts*.

```
[root@bloodhound]# iwconfig wvlan0 rts 500
```

The default value of the fragmentation threshold is 2,346. In noisy environments, it may be worth lowering the fragmentation threshold to reduce the amount of data, which must be retransmitted when frames are lost to corruption on the wireless medium. Set the parameter by using the *fragmentation_threshold* argument to *iwconfig*. It may be set anywhere from 256 to 2,356, but it may take on only even values. *fragmentation_threshold* may be abbreviated as *frag*.

```
[root@bloodhound]# iwconfig wvlan0 frag 500
```

802.11 stations maintain several retry counters. When frames are retransmitted "too many" times or wait for transmission for "too long," they are discarded. Two retry counters are maintained. The long retry counter, set by the *retry* parameter, is the number of times transmission is attempted for a frame longer than the RTS threshold. The short retry counter, set by the *retry min* parameter, is the number of times transmission will be attempted for a frame shorter than the RTS threshold. Unlike many drivers, *iwconfig* also allows for configuration of the maximum frame lifetime with the *retry lifetime* parameter. To specify a value in milliseconds or microseconds, append "m" or "u" to the value:

```
[root@bloodhound]# iwconfig wvlan0 retry 4
[root@bloodhound]# iwconfig wvlan0 retry min 7
[root@bloodhound]# iwconfig wvlan0 retry lifetime 400m
```

### wvlan_cs driver parameters

Several options can be passed to the *wvlan_cs* module by *cardmgr* when it is loaded into the kernel. These options are most easily set in */etc/pcmcia/config.opts*. See Table 13-5.

*Table 13-5. wvlan_cs driver parameters*

| Parameter | Value type | Description |
|---|---|---|
| *irq_list* | Comma-separated integer list | Specifies interrupts that may be used by driver. |
| *port_type* | integer, range 1–3 | Sets network type to infrastructure (1), wireless distribution system (2), or ad hoc network (3). |
| *station_name* | string | Sets station name; defaults to card setting. |
| *network_name* | string | Configures name for ad hoc network or name of target infrastructure network. |
| *channel* | integer, range 0–14 | Channel number/operating frequency for ad-hoc networks; not used for infrastructure networks. Default is 3. |
| *ap_density* | integer, range 1–3 | Sets threshold for roaming based on a low-density (1), medium-density (2), or high-density (3) installation. The default is a low-density installation, which minimizes roaming activity. |
| *medium_reservation* | integer, range 0–2,347 | Sets RTS/CTS threshold. Default is 2,347. |
| *frag_threshold* | even integer, range 256–2,346 | Sets fragmentation threshold. Default is 2,346. |
| *transmit_rate* | integer, range 1–7 | Each integer has a meaning; default is *auto select high speed* (3), which allows for fallback to lower speed. |

*Table 13-5. wvlan_cs driver parameters (continued)*

| Parameter | Value type | Description |
|---|---|---|
| *eth* | integer, range 0–1 | If set to 1, all devices are *ethN*. If set to 0, all devices are *wvlanN*. By default, this is set to 1, so all devices are *ethN*. |
| *mtu* | integer, range 256–2,296 | Maximum transfer unit; default is 1,500. |

### Troubleshooting

Naturally, all the PCMCIA troubleshooting notes from the previous section apply.

## Kernel 2.4: orinoco_cs

In addition to the WaveLAN cards and any OEM versions of WaveLAN cards, *orinoco_cs* contains basic support for some PRISM-2–based cards and Symbol cards that use the same MAC chipset. *orinoco_cs* has been part of the Linux kernel distribution since kernel Version 2.4.3.

### Compiling and installing

Some wireless LAN support code is activated in the kernel by enabling the Wireless LAN extensions (*CONFIG_NET_RADIO*) kernel configuration option. *CONFIG_NET_RADIO*-enabled kernels collect wireless statistics and expose additional data structures used by the *orinoco_cs* driver. To compile the *orinoco_cs* driver itself, recompile the kernel with Hermes support (controlled by the *CONFIG_PCMCIA_HERMES* variable).

The standard wireless tools are also required. They can be downloaded from *http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html*. As with the previous driver, installation is a straightforward matter of running *make* in the right spot.

### PCMCIA configuration

Most distributions ship with the *wvlan_cs* driver still enabled. To change the driver used by the distribution, it is sufficient to change the module binding the PCMCIA configuration. The author of *orinoco_cs* supplies a file, *hermes.conf*, which contains card definitions for the cards supported by *orinoco_cs*. Because *hermes.conf* ends in *.conf*, it is sourced by the line at the end of */etc/pcmcia/config* that reads all *.conf* files. However, to avoid binding conflicts, you must comment out all the lines that bind the older *wvlan_cs* driver to newly inserted cards. Alternatively, it is sufficient to edit the definition of your wireless card to bind the *orinoco_cs* driver after grabbing identification information from the output of *dump_cis*:

```
# in hermes.conf
#
card "Lucent Technologies Wavelan/IEEE"
   version "Lucent Technologies", "WaveLAN/IEEE"
   bind "orinoco_cs"
```

```
# from standard /etc/pcmcia/config
#
# card "Lucent Technologies WaveLAN/IEEE"
#    version "Lucent Technologies", "WaveLAN/IEEE"
#    bind "wvlan_cs"
```

### Configuring the orinoco_cs interface

Configuration of *orinoco_cs* is identical to the configuration of *wvlan_cs*. When the card is inserted, the */etc/pcmcia/wireless* script is run, using the configuration options in */etc/pcmcia/wireless.opts*. The wireless script is a frontend to the *iwconfig* program. Editing fields in *wireless.opts* sets the arguments to *iwconfig*. For details on configuring the options to *iwconfig*, see the previous section on the *wvlan_cs* driver.

CHAPTER 14

# Using 802.11 Access Points

In even the simplest 802.11 network, proper configuration of the access points is essential. Without properly configured network interfaces, no traffic will be bridged on to the wired network.

Access points can be divided into two groups. Home gateways and small office products are targeted at price-conscious users and do not offer much in the way of functionality beyond simple connectivity. Business-grade products are more expensive, but they possess functionality that is key to working with larger networks containing multiple access points. This chapter takes a look at two access points targeted at the high-end commercial user: the ORiNOCO AP-1000 and the Nokia A032. Extrapolating low-end product administration from the basic set of tasks is straightforward.

The major tasks facing a network administrator are to connect the access point to the wired network, configure the network interfaces, enable security features, and perform any configuration adjustments necessary to tune the wireless network to the area in which it is deployed. This chapter assumes that power and wired network connections are straightforward and have been taken care of.

## General Functions of an Access Point

All access points provide a similar set of features to network users because the standard feature set is specified by 802.11. Configuration of these features, of course, is vendor-specific, but the products are fairly similar to each other.

Access points are bridges between the wireless world and the wired world. As bridges, then, all access points have features that one would expect to see on a network bridge. They have at least two network interfaces: a wireless interface that understands the details of 802.11 and a second interface to connect to wired networks. To take advantage of the installed base and expertise, the wired interface is almost always an Ethernet port. Many products also have a WAN port. Sometimes the WAN port is a serial port that can be connected to a modem for use with a dial-up ISP account. It

is also common to find DSL interfaces in access points. Still, other access points add a second Ethernet port for connection to a DSL modem or cable modem. Some access points have multiple wireless interfaces so network managers can increase hot-spot capacity by using two interfaces on nonoverlapping channels. Many access points also offer the option of using external antennas to further boost range and allow for fine-tuning of antenna placement. Bridges have some buffer memory to hold frames as they are transferred between the two interfaces, and they store MAC address associations for each port in a set of internal tables.

Bridging tables are, of course, highly implementation-specific, and there is no guarantee of similarities across the industry. Generally speaking, though, inexpensive consumer devices are fundamentally designed with the assumption that they will be the sole access point for a network, while the more expensive commercial devices include more advanced features to make large deployments and rollouts easier. Commercial-grade devices are also designed to work cooperatively; the most common feature is a vendor-proprietary Inter-Access Point Protocol (IAPP). An IAPP allows wireless stations to move from access point to access point without interrupting link-layer connectivity. At this point, no standard for an IAPP exists, though the 802.11 working group is addressing this shortcoming. (Some cheaper products may support an IAPP, however.) Network management is generally much more sophisticated on commercial-grade products to enable network engineers to manage the tens or hundreds of devices used to create a large-scale coverage area.

All but the cheapest access points have a TCP/IP network interface. TCP/IP interfaces are intended for remote management and typically accept only basic configurations; anything beyond an IP address, netmask, and a single static default route is atypical. Depending on the level of sophistication of the hardware and software, varying levels of low-level interface configuration are possible. Naturally, all products allow the configuration of the network name. Other low-level parameters, however, may or may not be configurable. For access points that use PCMCIA network interfaces, the configuration of the wireless interface may depend on the firmware present on the PCMCIA interface card.

Depending on the market for which an access point is developed, it may offer services to its wireless clients. The most popular service is DHCP; wireless stations may be assigned addresses automatically upon association. Many access points can also perform network address translation (NAT), especially the "home gateway"–type products that can connect to a modem and dial up an ISP.

Security has been a sore point for wireless network managers since before the advent of 802.11's success. Access points have a privileged position with respect to security concerns because they are the gateways to the wired network and are ideally positioned to implement security policies. As detailed elsewhere in this book, the tools that access points provide to enforce security polices are sorely lacking. Naturally, WEP implementations are fully configurable with new keys and can be set to either

open-system or shared-key authentication, but few other standardized access control tools exist. (Naturally, several vendors have implemented proprietary approaches.) The major access control method implemented by access points is *MAC address filtering*. With address filtering, network administrators can give each access point a list of the MAC addresses of clients that should be allowed to access the network. Many products that offer remote network management provide some tools to filter management access, but few of the filters are based on anything other than easily forged source IP addresses.

Management interfaces often leave something to be desired. Configuration of access points tends to be challenging because access points must be manufactured cheaply, and low-cost devices tend not to have the processing power to run an easy-to-use configuration engine. Most vendors use lightweight operating systems running on low-powered hardware, but one of the trade-offs of using a lightweight operating system is that it does not provide the programming environment necessary to build rich functionality. Early access points offered both a cryptic command-line interface and a web-based management interface. This is still a common model. It is not unheard of for a vendor to supply proprietary management software for just one operating-system platform. (Most offenders in this category are management applications confined to run on Microsoft Windows.) Typically, host software connects over a serial port, either RS-232 or USB, and runs a proprietary management protocol to change configuration variables. Moreover, it is only recently that vendors have even begun to address seriously the concerns of large-scale access point management.

Debugging and troubleshooting tools are as advanced as management tools, which unfortunately means that they often leave network administrators mired in inconclusive or irrelevant information. Ideally, products should maintain detailed logs of activities, but it is common to find vague logs of results that give very little insight into failures. Counters can be helpful, but only if the right counters are accurately maintained. Tools such as *ping* and *traceroute* are common, but network analyzers and packet capture tools are not.

## Types of Access Points

Broadly speaking, there are two classes of access points in the marketplace. A low-cost tier is sold widely through retail channels directly to the end user. These low-cost devices are specialized computing platforms with only limited memory and storage. The higher-cost tier incorporates additional features required to support large deployments; frequently, these devices have additional memory and storage and resemble small general-purpose computing platforms in their design.

### For the home: residential gateways

The low-cost tier is composed of devices often called *residential gateways*. Residential gateways are designed to be as low-cost as possible, so only the basic features

required for the typical small or home office are included. Residential gateways generally share the following characteristics:

- Most devices include a DHCP server to make plug-and-play configuration easier.
- They are often deployed by users with one routable IP address, so NAT implementations are common.
- Depending on the type of customer the residential gateway is aimed at, the WAN interface is a modem, a serial port, or even DSL. (Some residential gateway products may use an Ethernet port as the "WAN" connection to a cable modem or DSL modem.)
- They are often built as a single integrated unit, complete with a built-in antenna. If suitable coverage cannot be found, it is necessary to relocate the entire unit.
- Many products now claim to have an *IPSec pass-through* feature to allow the use of IPSec through NAT, which works with varying degrees of success depending on the IPSec VPN solution chosen.
- Configuration of residential gateways often relies on a Windows program that is installed and uses a proprietary protocol over serial or USB to configure the device.
- They are often sold directly to the end user and are designed to be aesthetically pleasing. Unfortunately for many end users, the improved visual design prevents the stacking of residential gateways with other network equipment.

As this book was written, residential gateways typically cost $150 to $300. Examples of the residential gateway class of device are the 3Com Home Wireless Gateway, the Apple AirPort, the D-Link DWL-1000AP access point, the Intel Wireless Gateway, the Linksys WAP11, and the the Orinoco RG-1100.

### For the office: enterprise (corporate) gateways

*Enterprise gateways*, which often go by many other names that imply the buyer values features over cost, provide everything residential gateways do, plus additional features useful for larger-scale environments. Enterprise gateways generally share the following characteristics:

- The area over which mobility is required is much larger and requires several access points working in concert. Enterprise products support an IAPP so that a group of access points can be used to provide mobility through large areas. All IAPPs are proprietary at the time this book was written, but efforts to standardize the IAPP and enable roaming with devices from different vendors are underway.
- Upgrades are much easier with enterprise products. The wireless interfaces themselves are often PCMCIA cards, which makes the upgrade path much easier. As an example, many enterprise gateways do not require "forklift" upgrades to move to 802.11a. Instead, the wireless interface can be replaced

with an 802.11a-compatible interface, and a software upgrade provides a driver for the new card and the software features necessary for 802.11a.

- Enterprise-class products are designed to make deployment as easy as possible. Many high-end products can draw power over the unused pins in the Ethernet cable by complying with draft versions of the IEEE 802.3af standard.
- Frequently, site survey tools come bundled with enterprise-class products so network managers can plan large deployments by directly assessing coverage quality.
- Wireless interfaces on enterprise gateways usually allow for the possibility of using external antennas, and a wide selection of antennas is available to provide coverage in specific types of areas. External antennas may come standard with some enterprise products. Transmission power can be adjusted on many devices to enlarge or shrink the coverage area.
- Security developments appear on high-end products first. Enterprise gateways were the first to implement address filtering, and they are the test-bed for new security features implemented in software. Some vendors have enhanced the association process with proprietary key exchanges. 802.1x is beginning to appear in these products as well.
- Reflecting the administrative demands, configuration of enterprise-class devices is done with easily scripted command-line interfaces or SNMP, and monitoring and management capabilities are far more extensive than in residential gateways.

Naturally, these additional capabilities do not come without a price. As this book was written, enterprise gateways typically cost $800 to $1,100. Prices for enterprise-class products are not subject to the same downward pressure as residential gateways. Generally, enterprise-class products are made with much more generic hardware than residential gateways. Software upgrades are more common and can continue to add value through the life of the product. Examples of the enterprise gateway class of device are the 3Com AirConnect, the Cisco Aironet 350 Series Access Point, the Intel PRO/Wireless LAN Access Point, the Nokia A032, the Orinoco AP-1000 and AP-2000, and the Proxim RangeLAN2. This chapter describes the management of the Nokia A032 and Orinoco AP-1000 as examples. As you'll see, the mechanisms for configuring and managing these access points are different, but the information you need to supply and the configuration parameters that you can control are fundamentally similar. Even if you don't use Orinoco (a.k.a. Lucent, a.k.a. WaveLAN, a.k.a. Agere) or Nokia products, this chapter will show you what you need to do to configure any commercial access point.

## Selecting Access Points

When choosing an access point, you should take a number of factors into account. With the emergence of 802.11 as the main vendor-neutral standard, standards compliance is generally not a big factor. Most 802.11 equipment has gone through

compatibility testing with the Wireless Ethernet Compatibility Alliance (WECA); equipment that has received WECA's "Wi-Fi" (short for "wireless fidelity") certification has proven successful interoperability and standards compliance. Originally, the Wi-Fi program was only for 802.11b equipment, though a successor certification program called Wi-Fi5 is in development for 802.11a gear. Wi-Fi certification is almost certainly something to look for because it is practically a guarantee of interoperability.

802.11 is a complex standard, however, with several optional features. External antennas are often useful for creating a dense coverage blanket over an area. Not all access points can connect to external antennas; it may be an extra-cost option. Even if an access point has an external antenna connector, there is no guarantee that you'll be able to find a wide range of antennas available for use. 802.11 only requires that any connectors for external antennas have a standard impedance of 50 ohms. The actual physical connector may be proprietary to the vendor. If external antennas are important for your deployment plans, make sure that a wide range of antennas is available, whether through the 802.11 vendor or another source.

Security has been an area of notable innovation. The standard specifies one method, WEP, with a short key length. Some vendors offer longer key length versions of WEP, which are nonstandard but generally interoperable. Longer key length WEP implementations may add cost, however. More importantly, though, WEP has been conclusively demonstrated to be severely flawed. Some attempts at fixing WEP have been made by vendors, but they are fundamentally constrained by the design. As a result, there are a number of proprietary prestandard security mechanisms based on 802.1x (which also appears to be flawed). Some products have even begun to support 802.1x, with a few enhancements for wireless use. Before you commit to an evolutionary dead end, ensure that any solution can be easily upgraded to incorporate the standard security framework being developed by the 802.11i task group.

If roaming is important, a single-vendor solution is mandatory. Most vendors ship products with a protocol that enables roaming between access points, but products from different vendors are not guaranteed to interoperate. In the absence of a standard, there cannot be compliance testing. While this is being addressed within the 802.11 working group, until a standard is finalized, only a single-vendor solution can provide roaming between access points.* Until a standardized protocol hits the market, you may want to investigate the situations in which the vendor claims to enable roaming. Can stations move between access points even when WEP is enabled? How quickly can stations move between access points? It may also be worthwhile to

* While this is generally true, there are some exceptions due to OEM relationships. The Apple AirPort has been reported as allowing roaming connections to and from Lucent access points, but the AirPort is essentially a rebadged Lucent access point with a far superior aesthetic appearance.

obtain a guarantee from the vendor that any access points you purchase now can be easily upgraded to the forthcoming standard.

802.11 includes a number of power-saving functions in the standard. Most are optional. If your deployment is based heavily on battery-powered devices, it may be worth evaluating which power-saving features are included with particular devices. It may also be worth experimenting with devices to see just how much longer batteries last with the power-saving functions enabled.

In some deployments, getting power to the access points can be a major headache. To blanket an area with the coverage required for a large implementation, access points often need to be placed in an area where power is not easily accessible. Long antenna runs can degrade signal quality unacceptably, so it is much better to bring power to the location. Installing new electrical conduits is often quite expensive. Work must be performed by licensed electricians, and building codes may impose additional restrictions. Some products can supply power over the unused pins in the Ethernet cable. Network wire is not subject to the same restrictions as electrical cable and can be installed by network administrators.

Device management is an important consideration. Wireless networks are a new service, and network staff will need to plan, evaluate, purchase, deploy, and maintain the additional hardware. Large deployments may have tens or hundreds of access points, which can easily make network management a headache without good tools. Does the vendor offer an access point manager to configure large numbers of devices in parallel? Can management of the access points be incorporated into your existing network management infrastructure using tools that you already have deployed? Are the management tools secure enough? Many products can be managed only with clear-text protocols, which may be just an annoyance or a major violation of a security policy. Experience with other network devices has shown that software upgrades are a frequent occurrence. How is the software upgraded, and how much functionality can upgrades add? Can new protocol features be added with firmware updates?

Depending on the size of the deployment, it may be possible to evaluate equipment before buying. See if you can get a feel for the range of each access point and test with a variety of common cards. Capacity on an 802.11 network is ultimately limited by the radio link, but you will want to make sure that there are no other capacity restrictions. Does the access point provide the processing power to run the wireless side at maximum capacity with WEP enabled? Some products incorporate cryptographic processors to help with the load of WEP, but many do not. Products that depend on a central processor to run WEP may run out of capacity if they are upgraded to the faster 802.11a standard. Products upgraded to 802.11a will also suffer if they do not have Fast Ethernet ports. Try to set up a test network and get a feel for the configuration required to integrate the access points with the rest of your network gear.

As with many other purchasing decisions, of course, there are a number of "soft" factors that may not be easily quantifiable. Warranties, a relationship with the vendor, and the quality of the technical support may all influence the purchasing decision. Soft factors are not technical nor easily quantifiable, however, so I will not discuss them.

### Are Access Points Really Necessary?

Access points are not required for a wireless network. Wireless stations can be used in independent networks, which do not require an access point. Building a Unix box that routes between an Ethernet network and a wireless network is not difficult, and hardware can often be reused from the scrap pile. Why, then, would anybody use an access point?

Now that residential gateways have fallen well below the $200 mark, building a Unix router is no longer a cost-effective option for single–access point networks. Once you consider what staff time is worth, building a Unix router is a pretty silly use of staff time. Access point hardware has some advantages over redeployed general-purpose platforms, too. Access points are small devices with no moving parts. As a result, they do not consume a great deal of electrical power and do not generate much heat. There is one notable exception to this rule, though. Apple offers a "software base station" that transforms any desktop machine into a bridging access point. With a few mouse clicks and very little effort, a desktop computer can become a base station.

Unix-based routers have never been effective in larger deployments because of the lack of mobility support. Effective roaming requires transparent *bridged* access, not routed access, to the link layer at different physical locations. However, roaming with 802.11 is possible only when access points can communicate with each other to track the movement of a wireless station. In the future, it is likely that an open source Unix distribution will have the features necessary for an access point: low-level access to fundamental 802.11 parameters on the card, Ethernet bridging, and an IAPP. Until then, though, there is no substitute for commercial products.

## ORiNOCO (Lucent) AP-1000 Access Point

The AP-1000 is the mid-range Orinoco product. The low end of the product line consists of products for the home market; the high-end products (for example, the AP-2000) has features such as enhanced security and upgrade ability to 802.11a.

In addition to purchasing the AP-1000 base unit, you must purchase wireless interfaces separately. Unlike many other products, the AP-1000 has two slots for PCMCIA wireless interface cards. (A less expensive version of the AP-1000, the AP-500, has only a single slot.)

## Unix-Based Access Points

One of the most basic preconditions for making a Unix-based access point is enabling access point functions in the wireless interface card. One of the major hurdles is rewriting the 802.11 headers. All traffic in an infrastructure network flows through the access point. Access points must rewrite the transmitter and receiver addresses in the 802.11 headers. Other management functions may be required as well. For example, 802.11 includes a number of power-saving mechanisms for infrastructure networks in the specification, but they can be used only on networks with access points that implement them.

There is also a nontechnical hurdle. Many vendors have actively supported the development of open source Unix drivers for their cards. After all, vendors make money selling hardware, and it is a good thing for them to sell cards for all client systems, even those that run open source Unix. Access points are a different story, however. Vendors have not been as forthcoming with the interface used to put cards into the access point mode. Access points are quite lucrative, and providing a driver interface in the access point mode in the card could potentially cannibalize access point sales.

At one point, the only way to get an Intersil-based card to act as an access point interface was to purchase the reference design from Intersil. (The reference design shipped with firmware that had access point functionality, and that firmware was not sold separately.) Intersil's shipping station firmware does, however, include something called a "Host AP Mode." In the Host AP Mode, the PRISM-2 chipset automatically takes care of "menial" tasks, such as transmitting Beacon frames and acknowledging incoming transmissions. Jouni Malinen has developed a driver to use the Host AP Mode with Linux kernel 2.4. In conjunction with the Ethernet bridging implementation in the kernel, this driver can be used to build an access point. It is available from *http://www.epitest.fi/Prism2/*.

With the present state of driver software, it is possible to build a Unix-based router. (I mean "router" pedantically, as "layer 3 network device.") One interface would connect to a wired network as it always has, and a second wireless interface could be run in IBSS mode. Ross Finlayson has established a community network at a coffee house in Mountain View, California using a FreeBSD-based router. The project's home page is at *http://www.live.com/danastreet/*, and there is a page devoted specifically to the router itself at *http://www.live.com/wireless/unix-base-station.html*.

The Lucent access points in this class do not include an internal DHCP server, but they are compatible with external DHCP servers. This is a reasonable assumption—if you are adding wireless capability to an existing network, it's almost certain that you already have a DHCP server on your network. Lucent's products for home use incorporate a DHCP server.

## The Management Interface

The AP-1000 is managed almost exclusively with SNMP through a Windows-based SNMP tool designed specifically for the AP-1000. Command-line support was added in the Fall 2001 software release (Version 3.83). Only basic configuration can be done over the serial port. Three main items may be configured with the serial port: the IP network interface (address, mask, and router, or the use of DHCP), the wireless interface (network name, channel, and WEP settings), and the SNMP "password" (community string). The use of SNMP as a management protocol is a strange choice. A great deal of the configuration information on the AP-1000 should remain secret, but SNMP does not provide any facilities for protecting new settings as they travel across the wire. Even worse, the community string is easily recovered with a packet sniffer. Once the community string is recovered, a malicious attacker could change the settings on your access point.

The default configuration is to boot with the DHCP client enabled, so any DHCP server on the network can assign it an address. If you like, you can set up your DHCP server to assign a fixed address to the access point by associating the desired address with the Ethernet address printed on the AP-1000.

## Introduction to the Orinoco AP Manager

The AP Manager software runs only on Windows (95/98/ME/NT/2000/XP). It is essentially a frontend to SNMP software, so you can build your own frontend on any SNMP-capable workstation.

The main AP Manager window is shown in Figure 14-1. When the software starts, it sends UDP probes to port 192 on the local network to locate existing access points. After an AP boots, it answers the requests, and the AP Manager displays a list organized by IP address:

```
[gast@bloodhound]$ tcpdump ip host 192.168.200.222
tcpdump: listening on eth0
19:33:18.190921 192.168.200.222.2159 > 255.255.255.255.192: udp 116
19:33:18.191196 192.168.200.100.192 > 192.168.200.222.2159: udp 116 (DF)
```

After locating access points, the AP Manager software monitors each AP it finds for reachability using ICMP echo requests.

### Applying configuration changes

Many configuration changes require a restart. First, the changes are saved to the access point using SNMP Version 1 *SET* requests. Then the access point is restarted so it can reboot with the new configuration. Any change to the access point typically requires a reboot.

The access point's operating system is based on software from KarlNet, so it is not a surprise that the SNMP protocol operations direct the access point to set a large

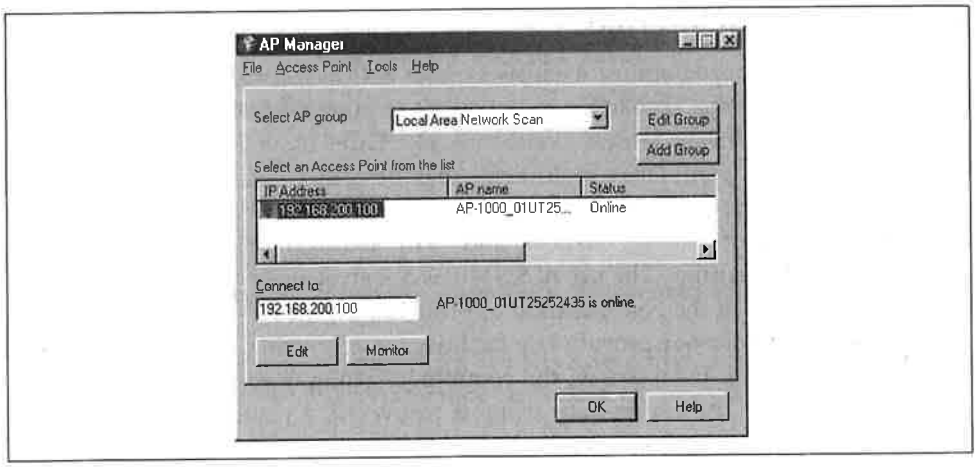


*Figure 14-1. AP list*

number of variables using KarlNet's SNMP enterprise number (762). Decoding the SNMP traffic and understanding it are two different matters, however. Just because it is easy to find where objects in the SNMP MIB are being set does not make it easy to find out what they do.

## Basic Management Tasks

Running the access point is simple. Straight out of the box, the AP-1000 uses DHCP to obtain the address for the management interface.

### Viewing the network configuration

Edit the access point from the main window and select the Access Point IP tab. Specify an IP address, network mask, and default gateway. Somewhat unusually, the AP-1000 also allows you to set the default TTL. See Figure 14-2.

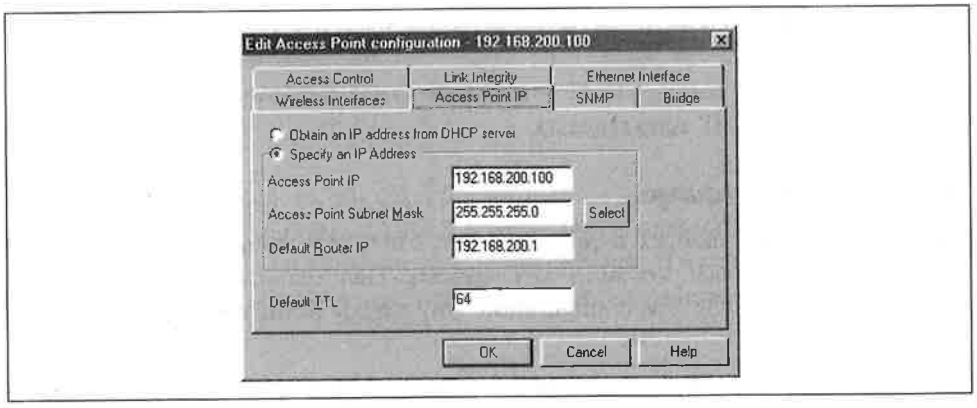


*Figure 14-2. Access point IP interface*

One nice thing about the AP Manager is that assigning the subnet mask is especially easy. The Select button raises a subnet mask panel that shows the mask in dotted decimal, hexadecimal, and mask-length notations. See Figure 14-3.

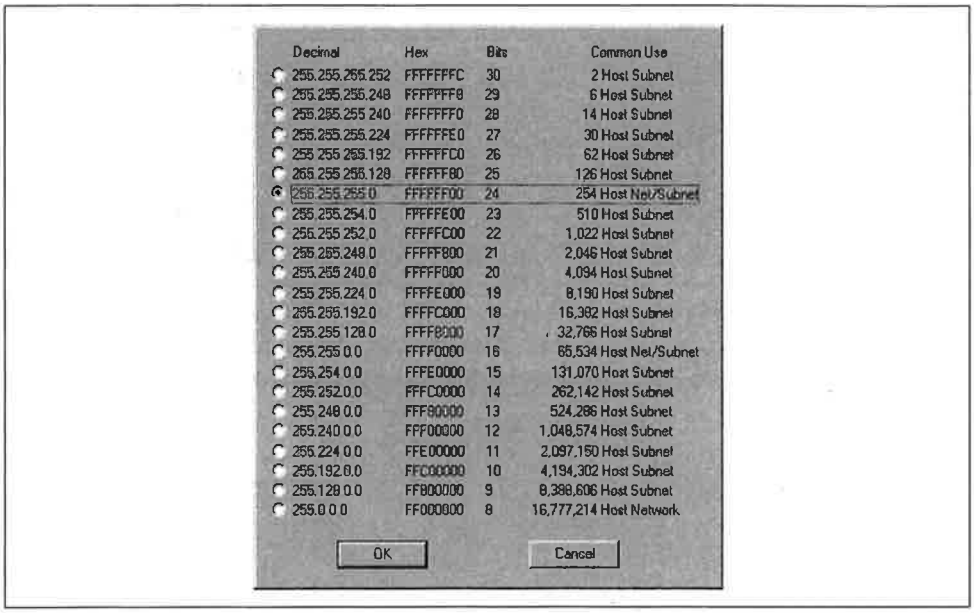


*Figure 14-3. Selecting a subnet mask*

### Configuring the radio network

Configuration of the radio network is done on the Wireless Interfaces tab, shown in Figure 14-4. Each PCMCIA slot may have a wireless interface. Two configuration buttons are present for each interface. The Security button configures WEP, and the Advanced button configures the channel and radio parameters. The only parameter set on the main page is the network name (referred to in the specification as the ESSID).

The Advanced wireless interface configuration dialog box is shown in Figure 14-5. The channel, which is displayed both as a channel number and as a channel frequency, can be changed in the top dialog box. When you select the operating channel, you must be aware of any regulatory restrictions in your location.

Settings for the DTIM interval and RTS/CTS thresholds should be left as their default values unless they need to be changed to address a performance problem. Chapter 17 addresses changing 802.11 parameters in response to problems.

The multicast data rate is set with its own button in the advanced setup. Multicast messages are transmitted to all stations simultaneously, so it is vital that any multicast transmissions are made at a rate intelligible to all associated wireless stations.

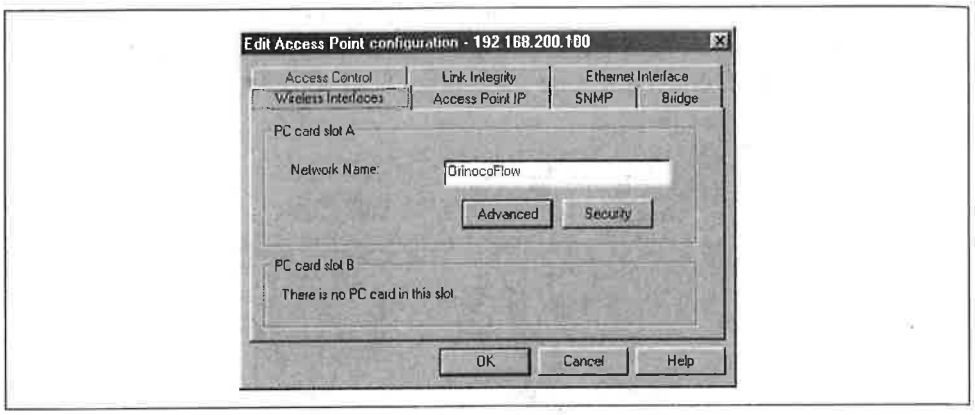


*Figure 14-4. Wireless Interfaces tab*

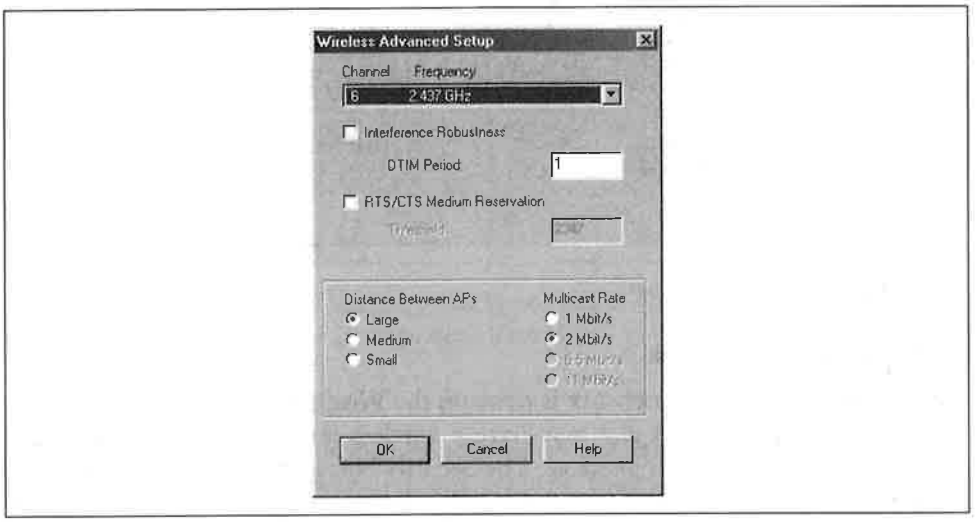


*Figure 14-5. Advanced wireless interface configuration*

The default value for the multicast rate is 2 Mbps because all commercial 802.11 direct-sequence cards are capable of 2-Mbps operation. You may change this to a higher rate, but you must ensure that network coverage is good and all stations are equipped with 802.11b cards compatible with the higher data rates.

The vendor cautions that the Interference Robustness setting should be used rarely with specific directions from an expert. There is a further warning that if the setting is needlessly enabled, connectivity and roaming may suffer. The documentation suggests that the Interference Robustness setting is used to work around interference close to the base station or in the path from the client to the base station. Based on their description, in many cases, it would be easier to redesign the wireless network than to suffer through what appears to be a nonstandard option.

The Distance Between APs option claims to tune the access point configurations to match the deployment. "Large" networks have very little overlap between BSSs and stations close to the edge of the BSS coverage area. "Medium" networks have 20% overlap, and "small" networks have an overlap area of 50% or more. It is not clear, however, which parameters are changed in response to setting this option.

### Configuring the wired network

Configuring the wired network is done with the Ethernet Interface tab, shown in Figure 14-6. The interface is an RJ-45 jack, and the Ethernet transceiver is capable of full-duplex operation if it is connected to an Ethernet switch. By default, the port is set to automagically negotiate speed and duplex. Not all autonegotiation implementations are created equal, though, so you may need to pick a setting for compatible operation. One way to tell if the Ethernet interface is not negotiating properly is to monitor the number of collisions, especially late collisions, on the switch port. Late collisions are abnormal at any level and are a common indicator of mismatched duplex settings.

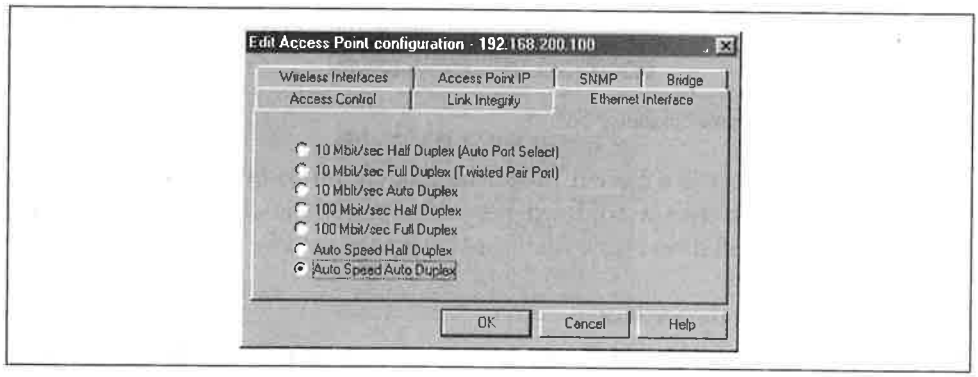


*Figure 14-6. Ethernet interface configuration*

## Security Configuration

As with most 802.11 access points, security configuration is limited to configuring WEP, MAC address filtering, and limiting management access based on source IP address.

### Configuring WEP

To enter WEP keys, click the Security button on the Wireless Interfaces tab of the main access point configuration screen. The security configuration is essentially WEP configuration; clicking the Security button brings up the WEP configuration dialog box shown in Figure 14-7. The AP-1000 supports both "64-bit WEP" (40

secret bits) and "128-bit WEP" (104 secret bits), depending on the wireless interface card used. "Silver" Orinoco cards support only the shorter length, and "Gold" cards support both. Keys can be entered either as alphanumeric strings or hexadecimal strings. Figure 14-7 shows a key being entered with a leading *0x* to denote that the key is being entered as a hexadecimal number.

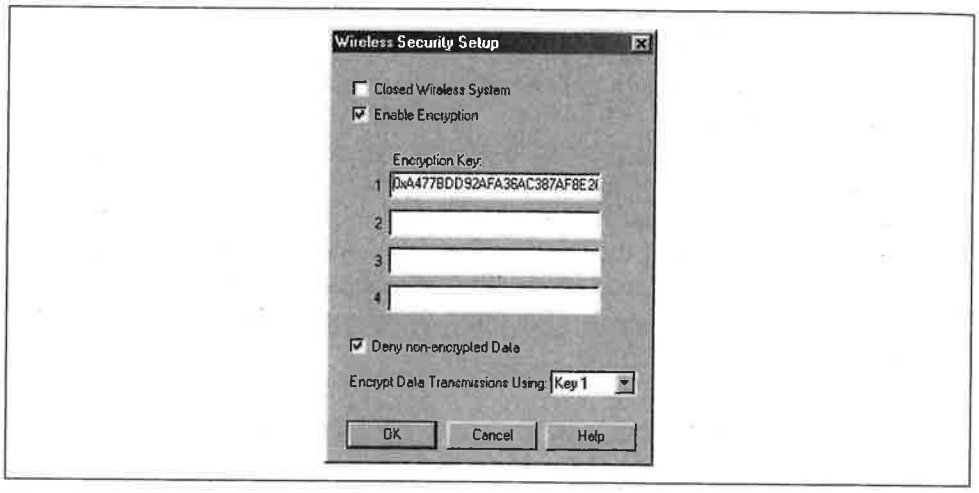


*Figure 14-7. WEP configuration dialog box*

*Closed Wireless System* is a Lucent proprietary extension to 802.11 that drops any frames from stations with a wildcard network name. It is compatible only with Orinoco cards and will drop traffic from cards manufactured by other vendors.

### Filtering management connections

Management is done over SNMP and is controlled through the SNMP tab on the main access point configuration screen. The SNMP configuration box is shown in Figure 14-8. Read Password and Read/Write Password are community strings; both are set to public as the default. Traps are sent to the specified trap host; a setting of 0.0.0.0 disables traps.

Management access can be restricted using the SNMP IP Access List, which may contain up to five entries. (It's also based on source IP address filtering, which is a common but dubious security mechanism.) To add an entry to the list, click on the Add button to the right of the access list display. Doing so brings up the entry creation and edit screen, shown in Figure 14-9. Each entry is composed of an IP address and mask, and any source IP address matching one of the entries is permitted to access SNMP data. You can use the final field to restrict the interface on which SNMP processing takes place; 1 is used for the Ethernet interface, 2 and 3 for the two wireless interfaces, and X for any interface.

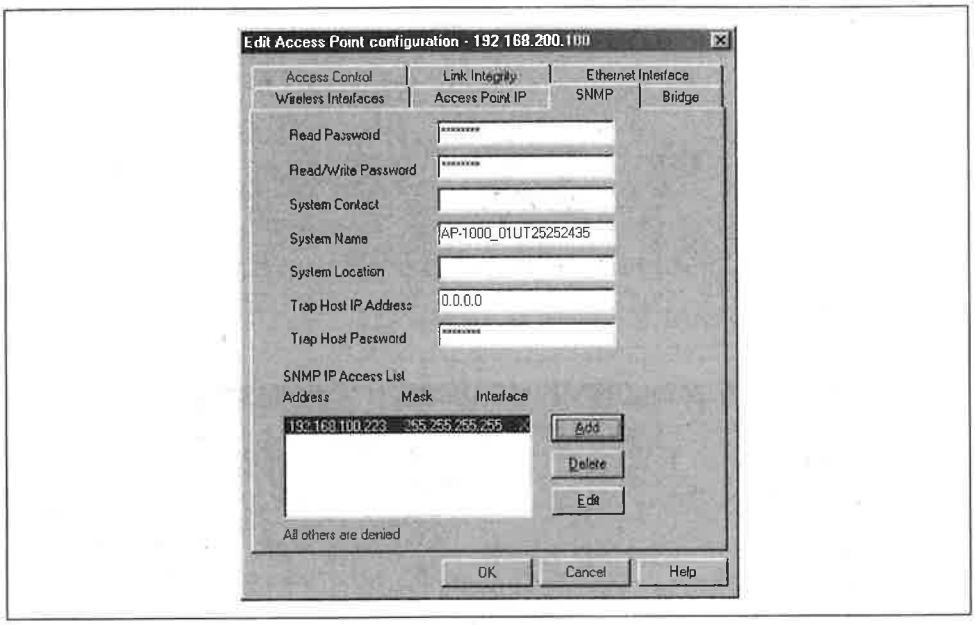


*Figure 14-8. SNMP tab*

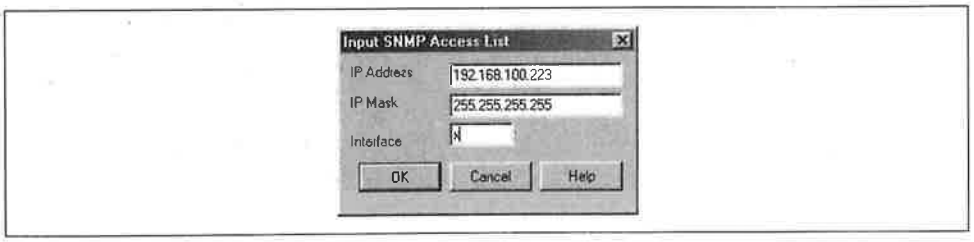


*Figure 14-9. SNMP access filter dialog box*

### Restricting associations

Address filtering is configured under the Access Control tab, shown in Figure 14-10. AP Authentication is a local table stored inside the access point of MAC addresses that are allowed to connect. MAC addresses can also be stored in a RADIUS server. Adding addresses to the list of AP-authenticated addresses requires a system restart.

## Monitoring Wireless Stations

Monitoring can also be performed using the AP Manager. Instead of editing the access point, monitor it. The main monitoring window is shown in Figure 14-11. By clicking on the various tabs across the top, you can see the bridge table, ARP table, various network statistics, and the list of associated stations.

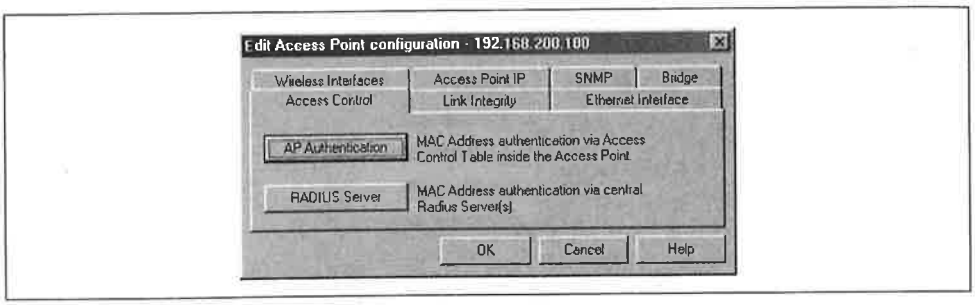


*Figure 14-10. Access control tab*

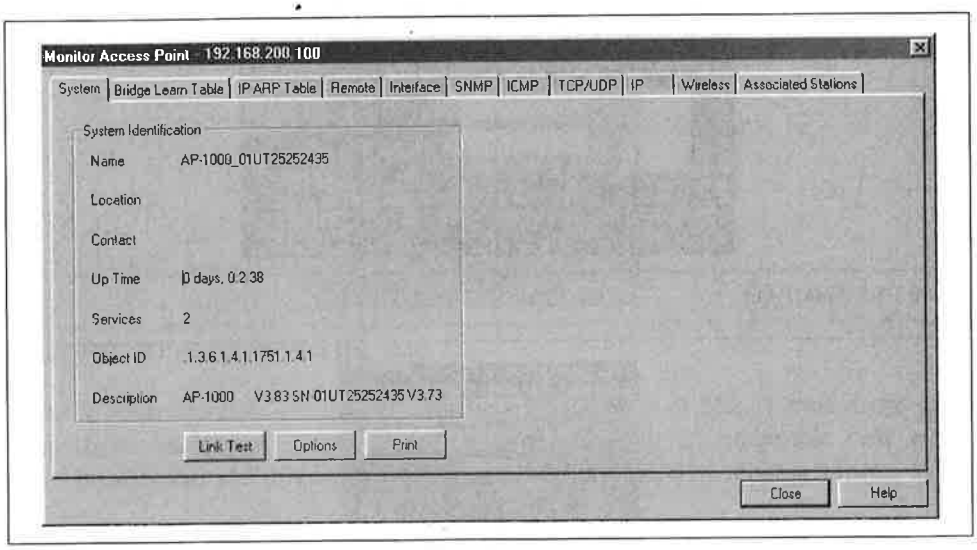


*Figure 14-11. Access point monitoring*

### Monitoring associated stations

A common management task is to show the list of associated stations. Obtaining the list in the AP Manager is straightforward. Simply click on the Associated Stations tab at the top of the monitoring window, and the list of stations is displayed, as in Figure 14-12.

Naturally, associated stations must be given entries in the system bridge table. To view the bridging table, click on the Bridge Learn Table tab at the top. Figure 14-13 shows a simple example. Interface numbers have the same meaning as they do for SNMP access list purposes. Only one of the stations shown in Figure 14-13 is a wireless station. Beneath the display of this window is an implementation of the bridge group MIB.

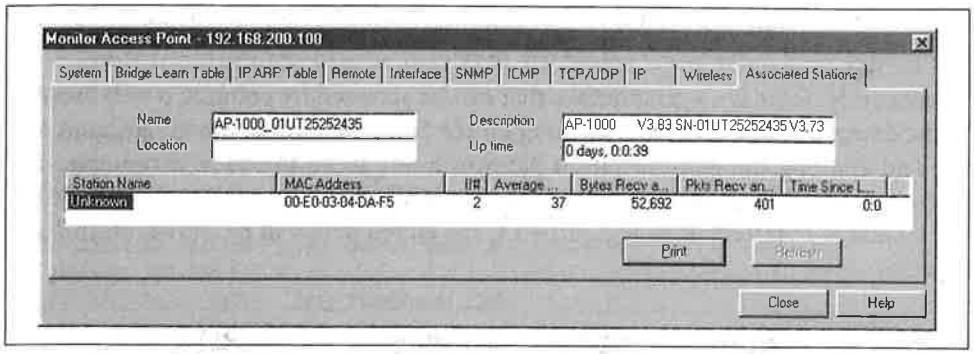


*Figure 14-12. Associated stations*

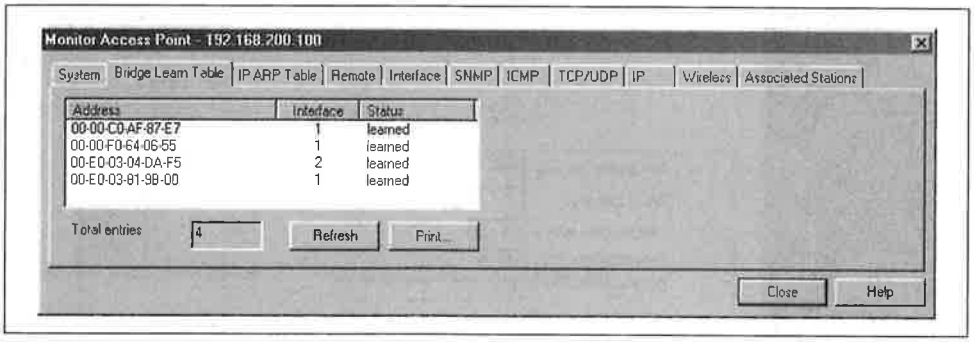


*Figure 14-13. Bridge table*

# Nokia A032 Access Point

The A032 access point is the only access point in Nokia's current product line. It replaces the A020 and A021 access points. Unlike some products on the market, it comes with an interface card in the box and even ships with an external omnidirectional antenna. I found the external antenna to be problematic. Shortly after I started using it, the interface card stopped working, and I was unable to get a replacement.

## A Tale of Two Management Interfaces

Like many network devices, the A032 has a console port that can be accessed using a null-modem cable. Configure the terminal emulator to the standard settings (9,600 bps, no parity, 8 data bits, and 1 stop bit) to access the console port. If a network address has already been configured, you can telnet to the device to gain access to the command-line interface.

The command-line interface lacks a great deal of the sophistication of the command line on more powerful devices. It has only limited help and does not assist with command

completion. Configuration is based on assigning values to parameters with the *set* command; its general format is `set` *`parameter value`*.

Alternatively, there is a web interface that can be accessed by pointing a web browser at the device's IP address. After clicking on the Setup link in the upper left-hand corner and entering the administration password, you'll see the basic setup screen in Figure 14-14. Changes to the device configuration frequently require a restart. Before the system is restarted, as in Figure 14-14, the edited data will be shown on the web screen.

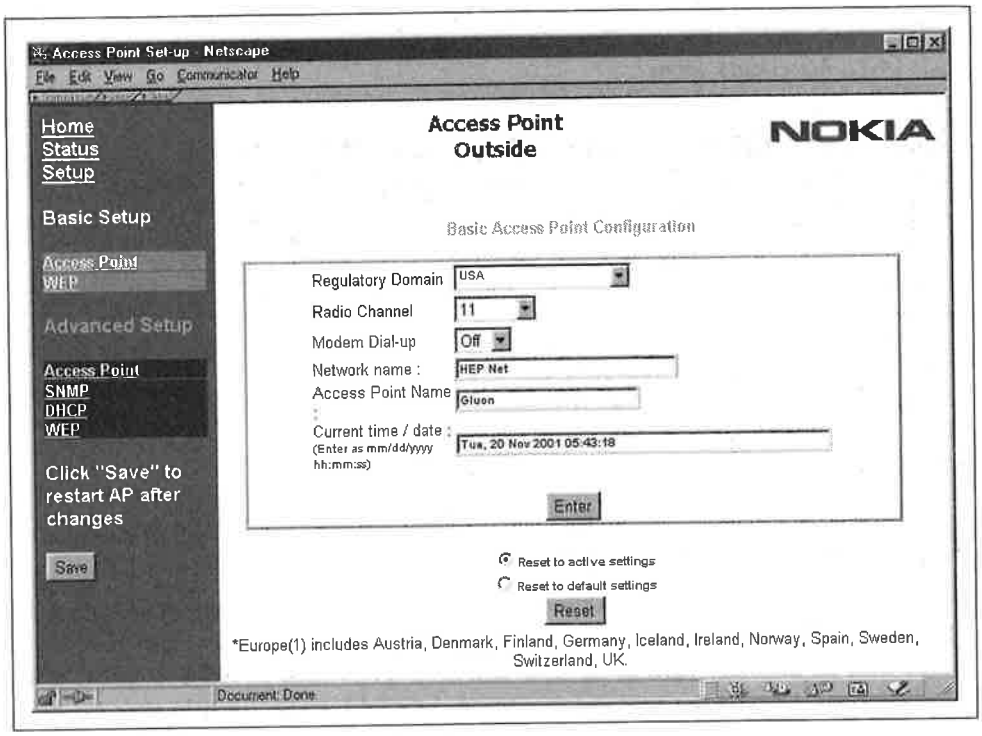


*Figure 14-14. Web-based management interface*

Web management is not for those concerned about security. Due to the limited computational power and storage space of the A032, it does not implement a secure web server, so the password is transmitted in the clear over the link layer. Of course, the link layer may be protected using WEP, but that is hardly reassuring.

## Basic Management Tasks

To enable remote management and run the access point in the most basic way, you should configure the IP network interface for remote access and configure the wireless network interface with the necessary parameters. Bridging is implicitly enabled and needs no explicit configuration.

### Viewing the network configuration

The current configuration can be displayed by typing the *config* command at the command line:

```
CMD:config
=========Current Configuration =========
net_name: Photon        AP_name: Outside
channel:   5   domain: USA       lan: 10baseT
IP_address: 192.168.155.90  subnet_mask: 255.255.255.0
gateway: 192.168.155.1
RTS_threshold: 2301    Frag_threshold: 2346
short_retry: 15  long_retry: 15
sifs_time: default   protocols: all
telnet: On:23  web: On:80
lock: off
manager: Specific       admission: All
basic rates: 1000 2000 kbits/s
wep-key range: min 128 max 128
community_get: public   community_set: private
Specific Managers (manager_IP):
    192.168.155.91 <accept requests> <no traps>
    <empty>
    <empty>
    <empty>
Radio NID: 00E003056F7F  Management NID: 00E003802F88
```

### Configuring the radio network

The most basic parameter for the radio network is the channel to be used. Several channels are defined by the standard, but not all are allowed in each geographical area where 802.11 equipment is sold. To ensure that a legal channel is selected, the A032 also includes a way to set the regulatory domain, which prevents inappropriate radio channels from being selected. This information can be set in the web interface from the basic setup screen shown in Figure 14-14.

The ESSID, or network name, is configured with the *net_name* parameter on the command line. Spaces are allowed if the network name is enclosed in quotation marks. The network name must be shared by every access point within the BSS.

```
CMD:set domain usa
Configuration has been updated
CMD:set channel 11
Configuration will be updated on next restart
CMD:set net_name "HEP Net"
Configuration will be updated on next restart
CMD:set ap_name Gluon
Configuration will be updated on next restart
```

Older Nokia access points had two modes of operation. One mode was in accord with 802.11 standards, and the second was a proprietary mode that has been eliminated in current shipping versions.

Several 802.11 parameters can be set only at the command line:

*Short retry counter (short_retry)*
: By default, the short retry is set to 15, though it can be set as high as 31.

*Long retry counter (long_retry)*
: By default, the long retry is also set to 15, though it can also be set as high as 31.

*RTS threshold (rts_threshold)*
: By default, the RTS threshold is set to 2,301 bytes. It can be set as low as 1 byte or as high as 2,500 bytes.

*Fragmentation threshold (frag_threshold)*
: By default, the fragmentation threshold is 2,346 bytes as prescribed by the 802.11 standard. It may be set as low as 257 or as high as 2,346.

```
CMD:set short_retry 29
Configuration will be updated on next restart
CMD:set long_retry 7
Configuration will be updated on next restart
CMD:set rts_threshold 512
Configuration will be updated on next restart
CMD:set frag_threshold 1024
Configuration will be updated on next restart
```

The A032 also allows adjustment of the SIFS time, though this should not be necessary in practice because the interface is set to comply with the 802.11 standard.

### Configuring the wired network, bridging, and management parameters

The predecessor to the A032 had both a 10baseT (twisted-pair RJ-45) and a 10base2 ("Thinnet") interface. In recognition of the dominance of twisted-pair physical connections, the A032 comes with only an RJ-45 port.

For management purposes, an IP address and default route can be assigned to the A032. The address may be used as a default gateway by associated wireless stations, but in practice, it often makes more sense to use the IP address simply for device management purposes. The IP interface is configured with the *ip_address*, *subnet_mask*, and *gateway* parameters:

```
CMD:set ip_address 10.1.2.100
Configuration will be updated on next restart
CMD:set subnet_mask 255.255.255.0
Configuration will be updated on next restart
CMD:set gateway 10.1.2.1
Configuration will be updated on next restart
```

One additional setting can be applied to the wired network. The A032 can be configured to bridge only TCP/IP frames. Many other link-layer protocols rely heavily on multicast frames, and if the A032 is connected to such a network, the available transmission time on the wireless network will dwindle as the multicasts are forwarded:

```
CMD:set protocols tcpip
Configuration will be updated on next restart
```

The command-line interface is normally accessible over Telnet connections on port 23, and the web interface is available on port 80. Both of these services can be run on arbitrary ports or disabled to prevent management. To disable a service, simply use the value off with the appropriate parameter:

```
CMD:set telnet 23
Configuration will be updated on next restart
CMD:set web off
Configuration will be updated on next restart
```

All of these parameters can be configured through the web interface by using the advanced access point setup link on the lefthand bar. The advanced access point page is shown in Figure 14-15.

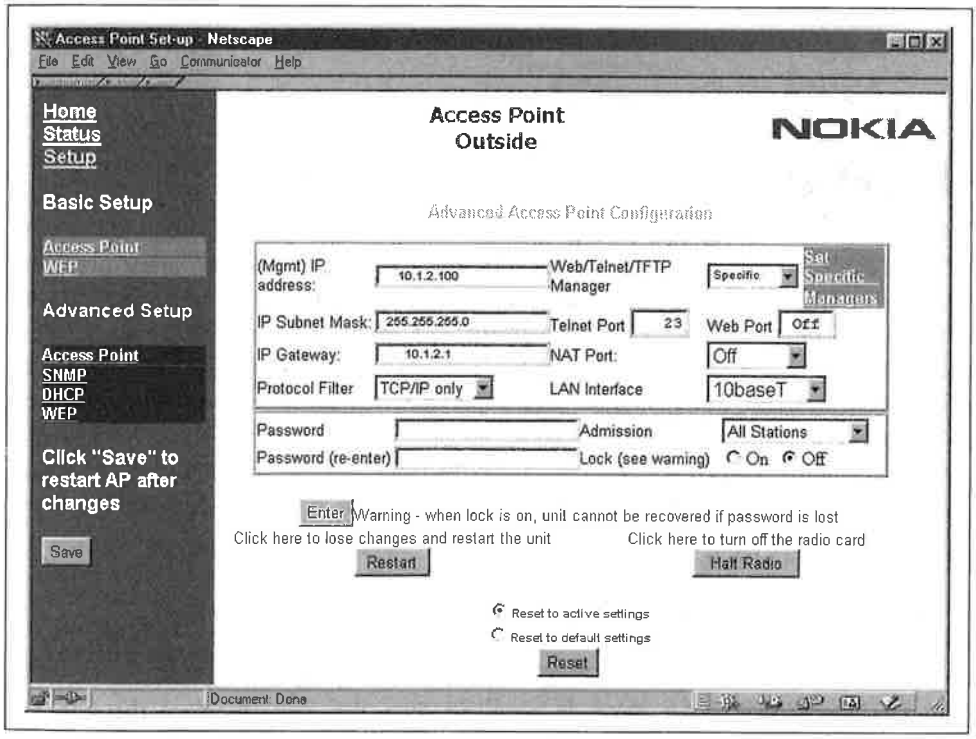


*Figure 14-15. Network interface configuration*

### Viewing the next restart configuration

Configuration changes to the A032 do not take effect immediately. When the system starts up, it reads the configuration from nonvolatile memory. Many configuration commands can be used with + appended to show the settings that will be effective on the next restart. For example, *config+* can be used to show the network settings that will be effective on the next restart:

```
CMD:config+
=====Next Configuration (effective after restart)======
```

```
net_name: HEP Net      AP_name: Gluon
channel:  11   domain: USA      lan: 10baseT
IP_address: 10.1.2.100  subnet_mask: 255.255.255.0
gateway: 10.1.2.1
RTS_threshold: 512    Frag_threshold: 1024
short_retry: 29  long_retry: 7
sifs_time: default   protocols: TCPIP
telnet: On:23  web: Off
lock: off
manager: Specific      admission: All
basic rates: 1000 2000 kbits/s
wep-key range: min 128 max 128
community_get: public   community_set: private
Specific Managers (manager_IP):
    <empty>
    <empty>
    <empty>
    <empty>
Radio NID: 00E003056F7F  Management NID: 00E003802F88
```

## Configuring DHCP

DHCP is useful with access points because it minimizes client configuration. The A032 includes a DHCP server that is suitable for small networks; larger networks may wish to take advantage of more generic solutions. The DHCP server can be allocated a pool of addresses. The administrator specifies both the DHCP base address (*dhcp_base*) and the size of the pool (*dhcp_pool*). The A032 DHCP server can also be configured to hand out a default gateway and DNS server information with the *dhcp_gateway* and *dhcp_dns* parameters:

```
CMD:set dhcp_base 10.1.2.200
Configuration will be set at next restart
CMD:set dhcp_pool 50
Configuration will be set at next restart
CMD:set dhcp_gateway 10.1.2.1
Configuration will be set at next restart
CMD:set dhcp_dns 192.168.100.5
Configuration will be set at next restart
```

The DHCP server configuration is accessed over the web interface through the DHCP link in the advanced setup section. The parameters that can be set over the web interface should be evident from Figure 14-16.

## Security Configuration

The A032 supplies three main security tools for the network administrator. All are based on 802.11 and therefore inherit all the security limitations of 802.11. The three tools are WEP, filtering of management connections, and filtering of associated stations.

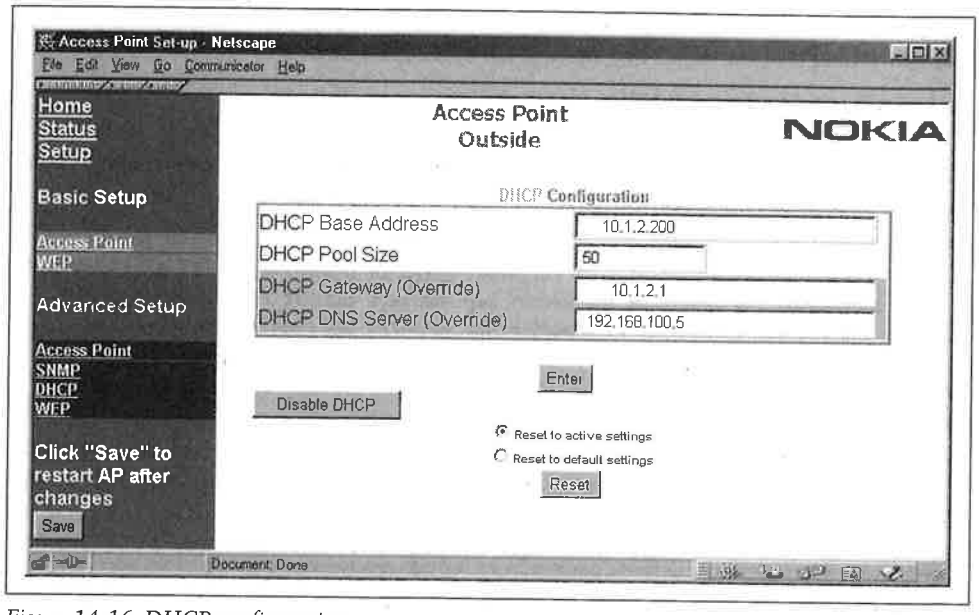


*Figure 14-16. DHCP configuration*

### Configuring WEP

WEP keys can be stored locally on the A032 or fetched from a RADIUS server. Local keys are entered by the administrator and stored on the device. They are configured by setting the *wep_key* parameter to *set*. Unlike many other parameters, *wep_key* takes two values: the key number and its value. To erase a key, the value *n* can be used. As mandated by the specification, four keys can be entered; one must be set active by using the *wep_key_active* parameter:

```
CMD:set wep_key 1 BF8D2E9D0D6F669F4A36DA254FA651AC
Configuration will be set at next restart
CMD:set wep_key 2 n
Configuration will be set at next restart
CMD:set wep_key_active 1
Configuration will be set at next restart
```

The A032 provides two WEP-related settings to help control associations. One is the *WEP mode*. Normally, the WEP mode is set to *wep*, which means that stations can use either a shared WEP key or a key from RADIUS to authenticate the station to the access point. The mode can also be set to *open*, for no authentication, *personal*, to require a key from RADIUS, or *wifi*, for compatibility with equipment that attempts open authentication but transmits using a shared key for encryption.

```
CMD:set wep_mode wep
Configuration will be updated on next restart
```

Additionally, the administrator can specify a required key length so only longer keys are permitted. The *wep_key_range* parameter takes two arguments, the minimum

and maximum key lengths, specified in secret key bits. To allow all users to connect, allow anything from a 40-bit key up to a 128-bit secret key:

```
CMD:set wep_key_range 40 128
Configuration will be set at next restart
```

Alternatively, the key range can be set to *normal* to allow only 40-secret-bit keys or *high* to allow only 128-secret-bit keys:

```
CMD:set wep_key_range high
Configuration will be set at next restartAs noted in Chapter 5, Nokia quotes all WEP
key lengths as the length of the secret key. Nokia's 128-bit keys have 128 secret
bits and are not compatible with most other vendors' implementations.
```

As with all other command-line work, the settings described in this section can be made through the Advanced WEP configuration page. Figure 14-7 shows this page, set to use the local (device) key database. RADIUS keys are discussed in the next section.

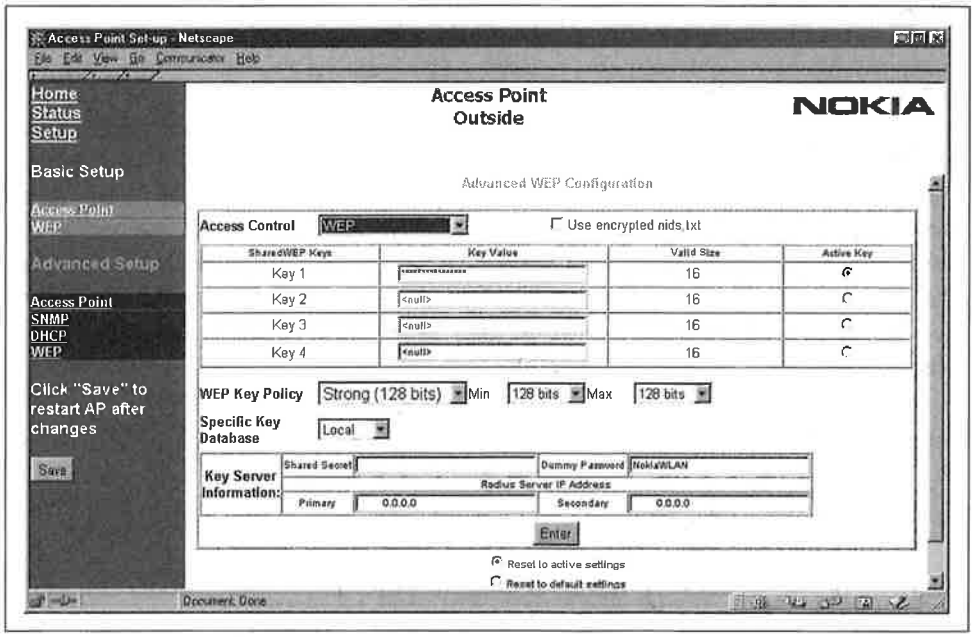


*Figure 14-17. Advanced WEP configuration*

## Personal WEP and RADIUS

Personal WEP is something of a misnomer because it does not map keys to users. It maps keys to MAC addresses. Each MAC address must be configured as a user account on the specified RADIUS server. When the A032 receives an association request from a station, it queries the RADIUS server for the WEP key stored in the

RADIUS database for the MAC address in the association request. After receiving the WEP key, the association proceeds normally.

Keys are stored in the RADIUS server under the MAC addresses that are allowed. A RADIUS "account" for each card must be created. When the A032 receives an association request, it sends an authentication request to the server consisting of the client's MAC address and the dummy password in Figure 14-17. The RADIUS server returns an acceptance message if there is an "account" for the MAC address that includes the client's WEP key in its response.

### Filtering management connections

It is conceivable that an access point may need to be configured to allow management connections only from selected stations. If access is restricted using the *set manager specific* command, the A032 checks each incoming management connection against the list and validates its access privileges. Stations can be added to the list using the *set manager_ip* command, which takes three arguments. The first is the IP address of the manager. The second and third arguments are for accepting requests and sending SNMP traps; a setting of 1 is used to enable the function, and a setting of 0 is used to disable it. To allow management connections from 10.1.2.88, use the following commands:

```
CMD:set manager_ip 1 10.1.2.88 1 0
Configuration will be set at next restart
CMD:set manager specific
Configuration will be updated on next restart
```

Specific manager configurations can be updated from the "Set Specific Managers" link in the Advanced Access Point Set-up, as shown in Figure 14-18.

### Naming allowed stations

For ease of management of a number of features on the A032, MAC addresses can be associated with text entries called *network identifiers* (NIDs). Naming stations is done only at the command line using the *nid* command, which takes one of three arguments, all of which are self-explanatory:

```
CMD:nid add 00005e000145 Matthew
CMD:nid list
----NID----- ------Name------ M
00005e000145 Matthew         ,NH
CMD:nid delete 00005e000145 Matthew
CMD:nid list
<list empty>
```

NID mappings can be retrieved from or sent to the access point using TFTP. Simply fetch or send the file named *nids.txt* from the access point's IP address. If management filtering is on, the TFTP client must be on the manager list. In the file itself, the last column in the NID list is used to note whether the entry is a station (N) or a

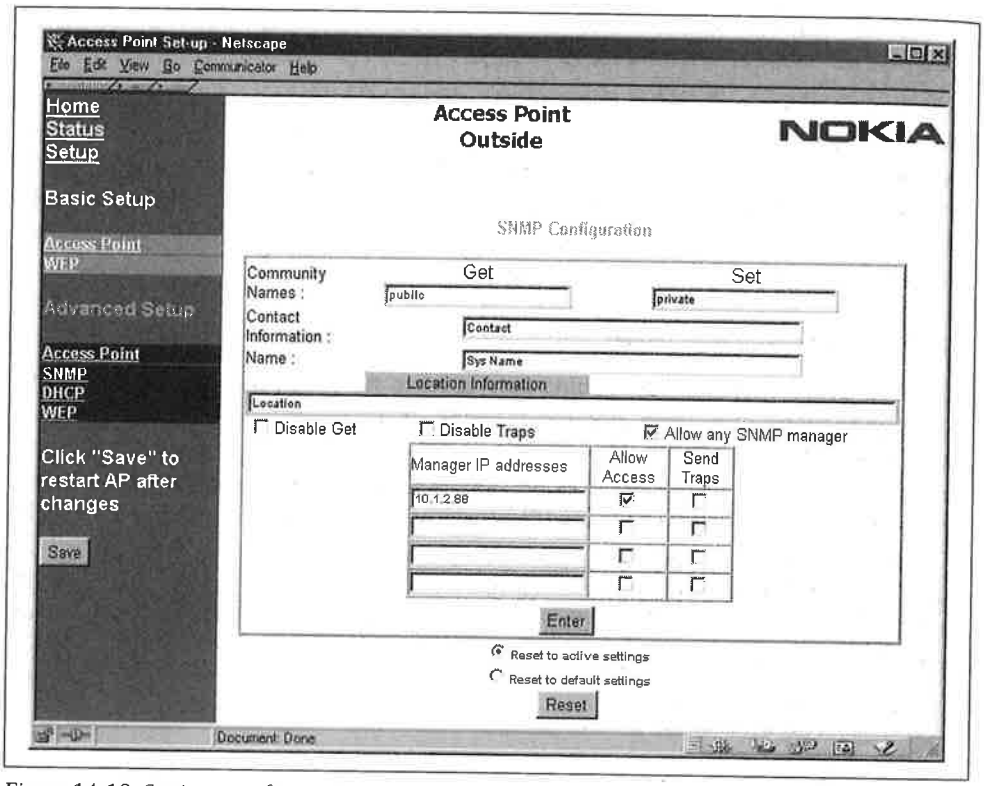


*Figure 14-18. Setting specific managers*

bridge (B). Bridge entries are relevant only when the A032 is deployed as a wireless bridge.

```
[gast@bloodhound]$ tftp a032
tftp> get nids.txt
Received 140 bytes in 0.0 seconds
tftp> quit
[gast@bloodhound]$ cat nids.txt
/ NID/Bridge list for AP(Outside) on Sat, 24 Nov 2001 23:12:39
49f7e6ca2f478073a1dcdf2cfeba08cc
00005e000145 Matthew          ,NH,
```

Up to 200 name/MAC pairs can be stored. No facility is provided by Nokia to synchronize the lists. System administrators must develop scripts to keep a single copy of *nids.txt* up to date and push it out to access points using TFTP.

### Restricting associations

MAC address filtering is performed using the NIDs. Associations can be restricted to "named" stations. If a station does not have an entry in the NID list, it is not allowed to associate:

```
CMD:set admission named
Configuration will be updated on next restart
```

Address filtering is enabled with the web interface by restricting admission to named stations, as shown in Figure 14-19.

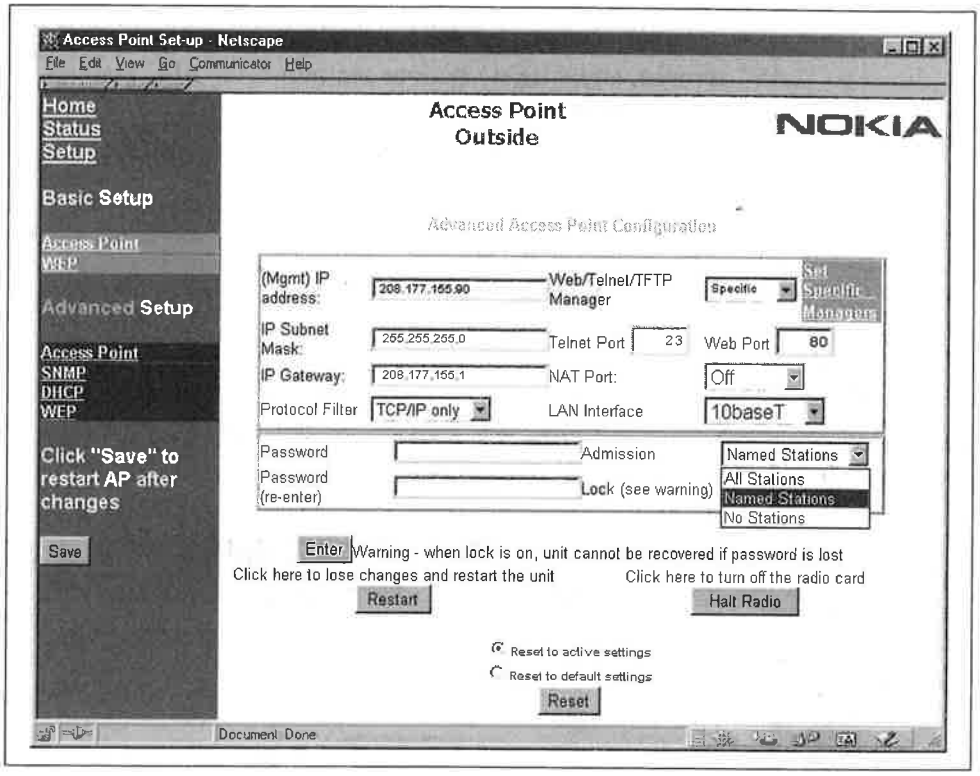


*Figure 14-19. Enabling address filtering for associations*

## Monitoring Wireless Stations

802.11 is a link-layer protocol, and monitoring the link-layer associations is important to troubleshooting. The status menu in the web interface allows the administrator to see all the associated wireless stations, all the known wireless stations, and all the known MAC addresses.

All the currently associated stations are listed with *show a*. Stations with a name-to-MAC mapping are displayed by name:

```
CMD:show a
Total bridge entries     :6
Number Wireless Stations :1
   MAC Address   State       Channel Power IP Address
c110             Associated   11      On    10.1.2.91
```

The difference between "all associated stations" and "all wireless stations" is that the latter includes any wireless stations learned about from another access point. All wireless stations are displayed with *show s*.

All known stations are printed with *show g*. Stations connected to the Ethernet port are shown as "LAN" stations, and wireless stations are shown as "AIR" stations. A few MAC addresses are reserved for internal use.

```
CMD:show g
Total bridge entries     :6
Number Wireless Stations :1
   Net ID        Interface  IP Address
0000C06F37EE        LAN     208.177.155.88
001067003F86        LAN     208.177.155.1
c110                AIR     208.177.155.91
00E003056F7F       Internal
00E003000000       Internal
00E003802F88       Internal  208.177.155.90
```

*show g* also provides the content for the "Network Summary" in the web interface, as shown in Figure 14-20. Internal stations are not displayed in the web interface.

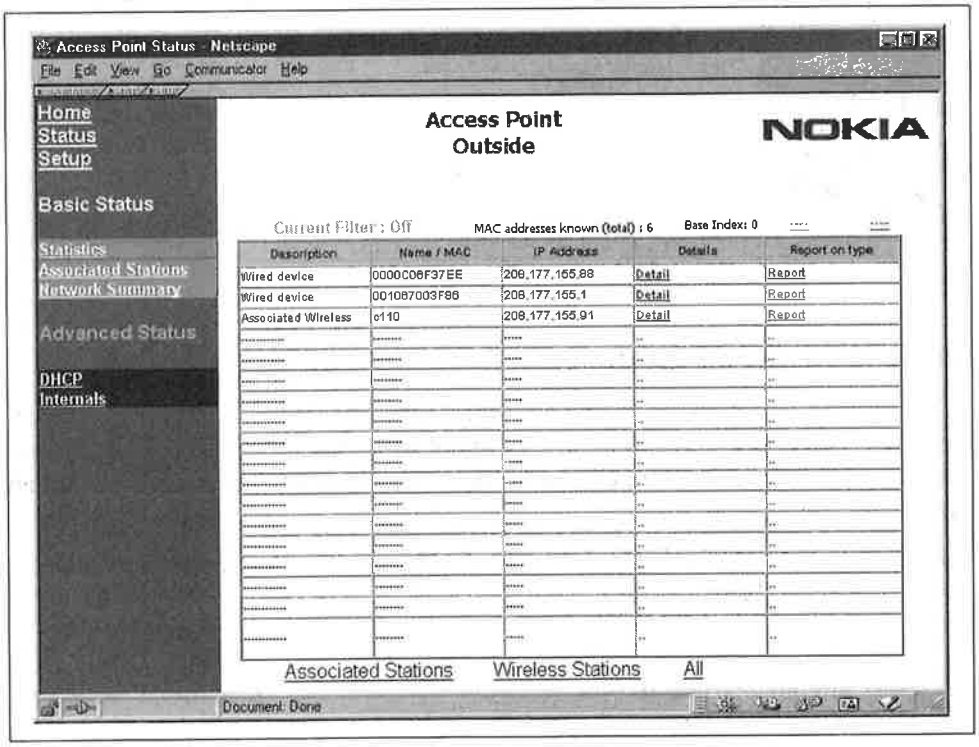


*Figure 14-20. Network summary page*

## Monitoring System Resources and Load

Counters can be monitored at the command line using the *stats* command. Statistics for both the wired and wireless interfaces are maintained, and they can be viewed by using *stats lan* and *stats air*:

```
CMD:stats lan
Stats last cleared : Sat, 24 Nov 2001 23:56:18
Accumulation time (secs) :       661

LAN Statistics:          Last 10s  Cumulative
tx lan frames      :       4358      23747
tx data bytes      :     371680    2039080
rx lan frames(all):        5952      36074
rx frames accepted:        5948      30980
rx data bytes      :     513936    2717364
rx discard frames :           0        311
CMD:stats air
Stats last cleared : Sat, 24 Nov 2001 23:56:18
Accumulation time (secs) :       663

Air Statistics:        Last 10s  Cumulative
tx frames :              5972      31657
tx bytes :             630273    3529370
tx fail  :                  0         62
rx frames(all) :         4360      23882
rx frames(mgmnt):           0          3
rx frames(data):         4360      23879
rx data bytes  :       441773    2433732
discard frames :            0        116
```

Software versions and basic system information are shown on the "Advanced Internals" status page in the web interface, reproduced in Figure 14-21. The bar graph at the bottom gives some indication of the system load. The buffer load shows the percentage of buffer memory used. Under normal situations, it should be 60% or less. System loading is the processor utilization. Radio usage is self-evident. Obviously, the access point shown in Figure 14-21 is under a heavy load.

## Troubleshooting Tools

Troubleshooting is not the A032's strong point. The only network troubleshooting tool supplied with the A032 firmware is the lowly *ping*, and it works as expected. *traceroute* is not included.

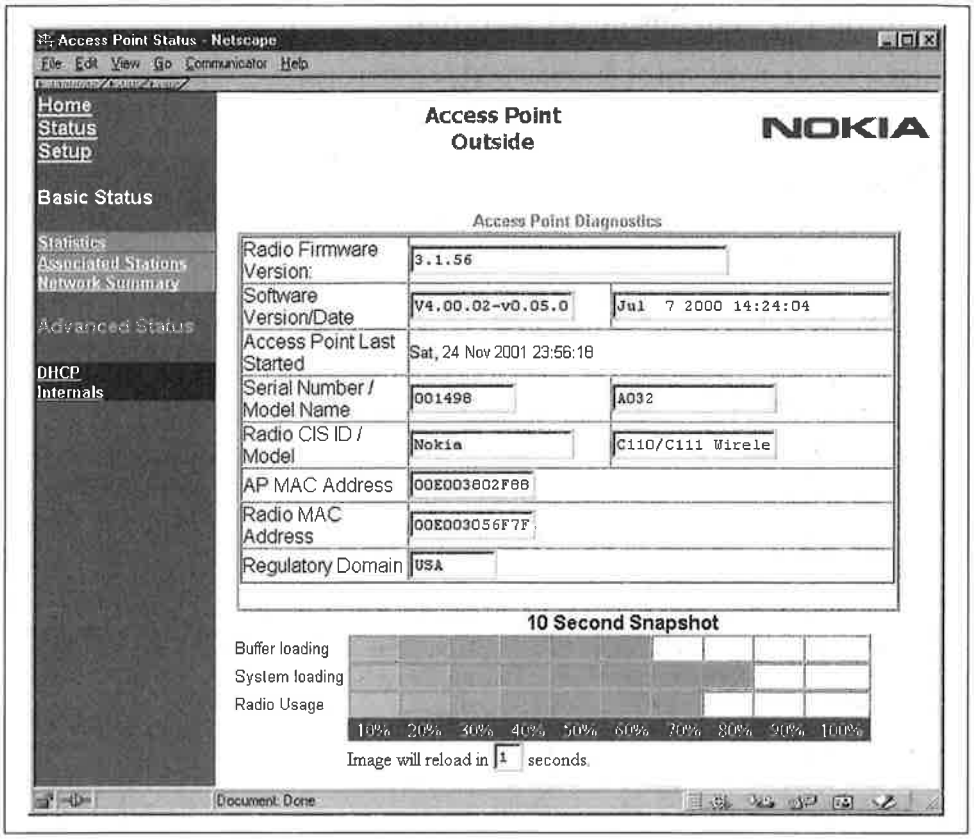


*Figure 14-21. Advanced Internals status page*