The DISH® Network: An Operator's Manual

A detailed guide to how DISH programming gets from provider to customer, written for both technical and non-technical readers.

Edward B. Tomme

Technical or not
Your background matters little
Your knowledge will grow.

Huge program content
Is compressed and encrypted
So more can be sent.

Our satellites sit
In fixed spots quite far away
Transceiving signals.

Whispers from the sky
Grow strong and are directed
Deep inside a house.

Complex yet friendly
Boxes find, unlock, and send
Channels to the screen.

Preface

When I joined DISH in late 2012 part of my job was to become the technical subject matter expert on everything DISH does. As I began my studies, I immediately looked for a document that could guide me to the understanding I needed. I quickly discovered there wasn't one. Everything I found was either dispersed across the Net (both Inter- and Intra-), was contained within the corporate memory of select individuals, or hadn't really been looked at in depth by DISH as it was the purview of other companies such as EchoStar. I even asked where our company technical library was and discovered one did not exist.

I began collecting documents and books to help get me up to speed, but these were still islands of information that were being pumped into my personal corporate knowledge, helping me but not necessarily others. I realized that while my particular job required me to be aware of the gamut of DISH technical operations, there were bound to be others who could benefit from at least parts of this hard-gained knowledge. With that thought in mind, I decided to write this guide to move the understanding of the DISH technical process from perishable, piecemeal, individualized knowledge to a systematic, accessible reference for everyone from engineers to corporate leaders.

I've used the word "technical" several times already, and my background is as a physicist. However, before you decide this document will be over your head, please take a few minutes to look at the beginnings of each chapter. I taught undergraduate physics at the United States Air Force Academy and was recognized in student end-of-course critiques as having an unusual talent for making highly technical concepts accessible to intelligent non-specialists.

Leveraging that talent, I've designed this document with a diverse audience in mind. The first parts of each chapter are overviews of the basic concepts required to understand the process, but without a lot of technical jargon. Collectively, these overviews should give a sufficient understanding of what we do at DISH for those of you who aren't normally technically inclined. The remainder of each chapter will treat the topic in much more depth, a level at which technical specialists will likely need to know.

I hope you find this information useful, and welcome your feedback on ways to improve it.

-- Ed Tomme, Englewood, Colorado, June, 2013

Table of Contents

List of Abbreviations, Acronyms, and Symbols

λ	Wavelength (lower case Greek letter *lambda*)
θ	Angle (lower case Greek letter *theta*)
π	The constant relating linear and angular dimensions of a circle (lower case Greek letter *pi*)
2-D	Two-dimensional
2PSK	Phase shift keying of order two
3-D	Three-dimensional
8PSK	Phase shift keying of order eight
A	Amplitude
AM	Amplitude modulation
b	Bit
B	Byte
B&W	Black and white
Bd	Baud
bps	Bits per second
Bps	Bytes per second
c	Speed of light
CAT	Conditional access table
CD	Compact disc
CONUS	Continental United States
d	distance
D	Diameter
dB	Decibel
DBS	Direct broadcast satellite
DiSEqC	Digital satellite equipment control
DPSK	Differential phase shift keying
DMA	Designated market area
DNS	Domain name system, domain name server
DRM	Digital rights management
DSL	Digital subscriber line
DVB	Digital video broadcasting
DVR	Digital video recorder
ECM	Entitlement control message
EIRP	Effective isotropic radiated power
EIT	Event information table
EM	Electromagnetic
EMM	Entitlement management message
EPG	Electronic program guide
ESA	Electronically steered array

f	Frequency
F_g	Force of gravity
FCC	Federal Communications Commission
FM	Frequency modulation
FSS	Fixed service satellite
G_t	Transmit antenna gain
G_r	Receive antenna gain
GEO	Geostationary (or geosynchronous) orbit
H_2O	Di-hydrogen oxide (water)
HD	High definition
HDMI	High definition multimedia interface
hr	Hour
Hz	Hertz (one cycle per second)
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet protocol
IR	Infrared
ISS	International Space Station
ITU	International Telecommunications Union
k	Arbitrary constant
L_{atm}	Loss due to atmospheric absorption
L_{rain}	Loss due to rain scattering
LEO	Low-Earth orbit
LHCP	Left-hand, circular polarization
LNB	Low-noise block down-converter
LNBF	Low-noise block down-converter with feed horn
LPCM	Linear pulse code modulation
m	Meter
MAPI	Messaging application programming interface
MoCA	Multimedia over Coax Alliance
MPEG	Moving Picture Experts Group
NIT	Network information table
N_s	Solar Noise
O_2	Oxygen molecule
OTA	Over the air
P_t	Transmitter power
PAT	Program association table
PCM	Pulse code modulation
PID	Packet identifier
PMT	Program map table
PSK	Phase shift keying

PTAT	Prime Time Anytime
QPSK	Quadrature phase shift keying
RCA	Radio Corporation of America; common name of a connector type used in audio/video devices
RF	Radio frequency
RGB	Red/green/blue
RHCP	Right-hand circular polarization
s	Second
S	Speed
SD	Standard definition
SDT	Service description table
SI	*Le Systèm Internationale d'unités*—the International System of Units; essentially, the metric system.
SNR	Signal-to-noise ratio
S/PDIF	Sony/Phillips digital interconnect format
S-Video	Separate video
T	Period or temperature
TV	Television
UHF	Ultra-high frequency
UV	Ultraviolet
VCR	Videocassette recorder
VHF	Very high frequency
YCbCr	Digital video format separating out luminance (Y), from two chrominances (Cb and Cr)
YPbPr	Analog version of YCbCr

List of Figures

Unless otherwise noted, all figures were developed by the author. Where they were adapted from other sources, an endnote will indicate that source.

SI Prefixes

Le Systèm Internationale d'Unités—the International System of Units (SI)—defines a set of prefixes that currently span 48 orders of magnitude. They can modify every base and derived SI unit to provide consistent, power-of-ten multiples of those units for easy abbreviation of very long numbers. Note that with the exception[1] of kilo-, all of the negative power-of-ten prefixes are lower case and all of the positive power-of-ten prefix abbreviations are upper case.

Selected SI prefixes relevant to this document are:

Power of Ten	Meaning	Prefix and Abbreviation
10^{-9}	One billionth	Nano- (n)
10^{-6}	One millionth	Micro- (μ--lower case Greek letter *mu*.)
10^{-3}	One-thousandth	Milli- (m)
10^{-2}	One-hundredth	Centi- (c)
10^{3}	One thousand	Kilo- (k)
10^{6}	One million	Mega- (M)
10^{9}	One billion	Giga- (G)
10^{12}	One trillion	Tera- (T)
10^{15}	One quadrillion	Peta (P)

Overview

While DISH is at its most fundamental a service organization, providing entertainment and information to millions of Americans, its foundation is undeniably technical. Electronics-based receivers, electromagnetic signals, and satellites dominate the way we deliver our service to our customers.

There is a great need for many DISH employees to understand the technology through which we operate. Call center operators' knowledge needs to be more than checklist-deep so they can intelligently and efficiently solve customer problems and needs. Installers need to understand why they point their dishes in the directions they do and why they use certain kinds of coax cable in certain situations, among other things. Engineers need to understand the advantages of MPEG-4 over MPEG-2 and the differences between 8PSK and QPSK. And executives need to have a grasp of the entire content delivery chain so they can make good business decisions based on sound technical rationale.

However, the level of understanding required by each of these groups varies markedly. Engineers making technical evaluations of hardware obviously need to know much more about the inner workings of a receiver, say, than do executives or installers.

How This Document Is Organized

To that end, this document is written on two levels. Each chapter begins with a section called *The Basics*. Those sections are written specifically for intelligent non-specialists and summarize the much more technically rigorous content in the remainder of the chapter. Even for the technical specialist, *The Basics* will provide a roadmap of where the chapter is headed and what they should be able to take away from it. Taken together, the collective *The Basics* sections will give a good lay understanding of the technical difficulties associated with delivery of DISH content to our customers.

Following *The Basics* in each chapter is a section called *Technical Discussion*. It is here that, should the topic be of interest, you will find much more information about the chapters' themes, including equations and technical diagrams where applicable, but even those sections won't be written on a level that takes an advanced technical degree to understand. These sections are designed to provide more detail than the basics while still being at a readable level.

That covers the intra-chapter organization of this document. But what about the overall flow? One logical way to ensure the entire technical process is covered is to start when content (entertainment and information) is provided to DISH as a signal stream and follow that signal until it emerges as pictures on a customer's television set or computer monitor. A review of the table of contents will reveal that is exactly the flow this document takes. We will start with a discussion of how content enters our system and then follow it from content provider to satellite uplink center and see it be retransmitted within satellites orbiting 22,000 miles above the Earth. We'll follow the signal down to the individual small rooftop dishes on customer facilities, and then into those facilities through cables and receivers until it becomes a television program on a screen.

Finally, we will present information about the receivers and satellites DISH uses in chart form in several appendices. An extensive bibliography and list of references are presented at the end of the document to point you to more thorough sources. The *Technical Discussion* sections of this document contain endnotes throughout showing where the information came from and to provide parenthetical information that could inhibit the chapter's smooth flow.

Math the Easy Way

So yes, this is the story of the technical side of DISH. And technical usually means math, doesn't it? Not always! We will do our absolute best to describe all of these concepts using words and pictures, but sometimes that just won't be enough and we'll have to use an equation. But fret not, those of you who haven't had much math since high school! We're going to do math the easy way. We'll describe the equation and all the different letters and symbols, the variables and constants, in it. We'll then describe what we want the end result to be, and will describe what we have to do to the variables to get that to happen.

Let's look at an example. The equation for the area of a circle is

$$A = \pi r^2$$

In this equation, the *variable A* stands for the area of the circle. The symbol π (the lower-case Greek letter *pi*) is a *constant* so we can't change it at all (it has the approximate value 3.14). The *variable r* stands for the radius of the circle, the distance from the center of the circle to any point on its edge. So, if our goal is to make the area of the circle larger, then the left side of the equation (the *A*) needs to get larger, which means the right side of the equation needs to get larger as well. There are only two terms on the right side: π and r. The constant π can't

change, so we're left with the option of changing the value of r. Since the left side got bigger, the right side must get bigger so r must get bigger. That's all there is to it! To increase the area of a circle, the only thing we can do is to increase its radius. Notice that we can all but ignore anything that's called a constant—they're in the equation, but we can't change their values to make our results change. Yes, it's a simple example, but we'll only be doing really simple math.

For a slightly more complicated example, let's look at an equation with a fraction in it. The equation for speed is

$$S = \frac{d}{t}$$

In this equation, *S* stands for speed, *d* stands for distance, and *t* stands for time—all three are variables. If you don't believe this equation, you can stick units in to see if it works: speed can be in miles per hour, distance can be in miles, and time can be in hours. Using those units, the equation becomes

$$\frac{Miles}{Hour} = \frac{Miles}{Hour}$$

which definitely balances.

Now that we believe the equation, let's see how we can increase our speed. If you make the *top number* in a fraction *bigger*, the *fraction gets bigger* (try it: which is larger: 1/3 or 2/3?) If you make the *bottom number* in a fraction *bigger*, the *fraction gets smaller* (which is larger? 1/10 or 1/100?).

With that review, how do we make the speed get bigger? We have two variables on the right side to play with this time: d and t. Since d is in the top of the fraction, we'd make it bigger to make the speed bigger. Since t is in the bottom, we'd make it smaller to make the speed bigger. So to make the speed bigger, we can either make *d* bigger, make *t* smaller, or we could use a combination of both of those actions. Put in a couple of numbers yourself to make sure you can understand the concept. We promise the math you'll find in this document—even in the so-called technical discussions—will be no harder than these example!

A Short Course on Signal Flow from Provider to Customer

While each individual chapter is summarized in an easily understandable manner by its own section called *The Basics*, the section you are now reading is a similarly constructed summary of the entire document, a sort of road map to where we'll be going along the way to an overall understanding of the DISH network. Figure 1

shows a broad overview of how the content we buy gets to our customers. You'll see other versions of this picture at the start of each of our chapters, highlighting the specific portion of the signal path being discussed. Since this section is an overview of the entire process, the entire signal path is shown here.

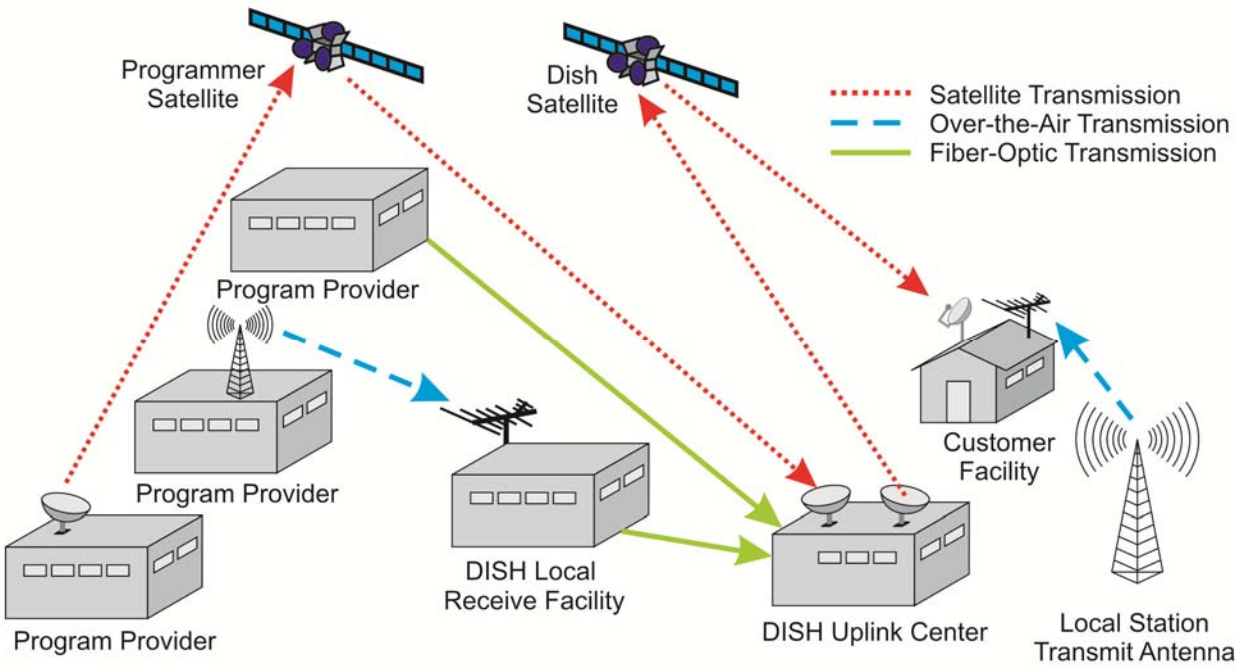

Figure 1: The Signal Flow Roadmap--From Provider to Consumer.

Before we at DISH have content to provide to our customers, we must acquire it from program providers. As we receive it, we convert it to a standardized, compressed, encrypted data stream. Content acquisition, MPEG compression, encryption, and data streams are thus the subject of our first chapter, *Prepping the Signal*.

While many of the acquisition, conversion, and encryption processes described in that chapter are actually done in our uplink centers, we'll actually use *The Uplink Center* chapter to concentrate on the physics of how our data stream is sent up to DISH satellites far above the Earth. There, we'll look at how physical and political constraints on the electromagnetic spectrum force us to use certain frequencies for our transmissions. We'll also explain why our uplink antennas are so large, and discuss the code that we use to turn the ones and zeros that represent our television programs into an electromagnetic wave that can be understood from thousands of miles away.

Next, we'll look at some more hardware in the *DISH Satellites* chapter to understand why they have to be so large, what they do to our data stream, and

why their orbits have to be where they are. We'll use some more basic physics and electrical engineering concepts to explain these topics.

Following the signal back down from space, we'll use *The Dish on Your Roof* chapter to explain why your rooftop dish looks like it does and some of the reasons your signal might not always be as strong as you'd like it to be. We'll also look at the very important piece of hardware that sticks out in front of the dish, as well as some of the other electronic components that help move the signal along before it gets into your house.

Our final chapter, *Inside Our Customer's Home: Receivers*, will look at some more DISH hardware, including the cables and receivers inside your house. We'll discuss how the encrypted, compressed, standardized data stream is decrypted and decompressed back into something that will make sense to a television set and, ultimately, to our customers in the form of the programming they want to watch.

Chapter Summary

As you can see from this overview, we've got a lot on our plates. However, stick with us and we're sure you'll soon be a much better informed operator of the great company we call DISH.

One last thing: since this book is written for both technical and non-technical readers, we've decided to do something unusual to make the non-tech reader feel more at ease. At the start of each chapter we'll include a haiku* that summarizes what the chapter is supposed to be about. Now, we know the tech readers are at this very moment rolling their eyes and saying to themselves, "Seriously?" But we'll bet that our non-tech readers will end up saying, "So *that's* what he meant." Here's the first one for this introductory chapter:

Technical or not
Your background matters little
Your knowledge will grow.

* A traditional Japanese poetry style consisting of three lines of five, seven, and five syllables.

1. Prepping the Signal

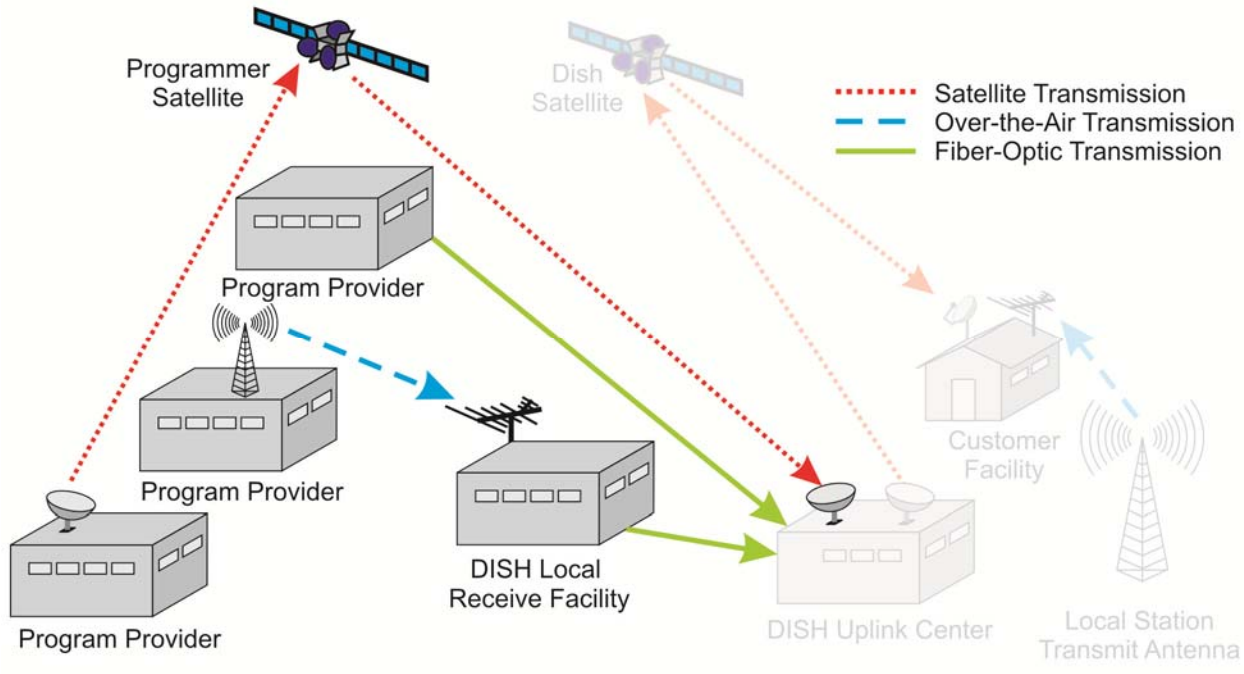

Figure 2: The Signal Flow Roadmap—Content Acquisition.

Huge program content
Is compressed and encrypted
So more can be sent.

The Basics

It probably goes without saying, but DISH does not produce the content we provide to our customers. We are value-added middle-men who buy content from other companies and then deliver it to the end user. So how do we get the content and what form is it in when we retransmit it up to our satellites? Answering those two questions is the goal of this chapter.

Content Acquisition. We buy content from a number of providers ranging from broadcast television networks, to media consortia controlling multiple non-network channels, to local broadcasters in various designated market areas (DMA) around the country, to other sources. That content comes to us in three ways. As promised in the *Overview* chapter, Figure 2 graphically shows these

methods and gives us our first view of how the overall road map of signal flow from provider to consumer will be displayed chapter-by-chapter. First, providers can beam signals up to one of the fixed service satellites (FSS) the programmers own or lease, where the signal is retransmitted back down to one of our two primary uplink stations in Gilbert, Arizona and Cheyenne, Wyoming. Those uplink stations also are connected to the Internet through very high-capacity fiber-optic cables, and other national content providers send us programming through those conduits. Finally, we receive some of our content over-the-air in exactly the same way as you'd receive your local channels from an old-style antenna. Most local channels pay for fiber connections to our facilities, but a few of them still rely on us to receive their broadcasts via antenna.

The first two methods of content acquisition are, by their nature, digital. That is, the information is contained in a coded series of ones and zeroes. In contrast, some of the over-the-air broadcasts can be analog, where a continuously varying electromagnetic wave carries the information through changes in signal strength (amplitude) and/or signal frequency. Immediately upon reception of over-the-air signals at our uplink centers, that analog signal is converted into a digital signal so it can be processed more efficiently by our equipment. We have a very small facility in most of the DMAs that basically consists of a television antenna, a computer, and an Internet connection. The computer takes the incoming signals from all of the local channels DISH carries for that market, converts them to the digital form we can use, and feeds them into the Internet where they're directed to one of our two primary or four much smaller secondary uplink centers. The over-the-air method is almost exclusively used to receive local channels.

MPEG. The digital coding standard we use to store and transmit content throughout our network is called MPEG, which might be familiar to you through MPEG video content on the web or through audio MP3s, which are an audio-specific form of the much more general MPEG format. MPEG is a way to not only encode media into ones and zeroes, but a way to very efficiently compress the signal so we can squeeze more of it through what may be thought of as limited capacity pipes from our uplink antennas to our customers' satellite dishes. The more the signal is compressed, the more of it we can push through the same pipes. But compression comes at the price of degraded picture quality; the trade-off between quality and quantity is one of the major business decisions DISH executives deal with when deciding what capital equipment purchases we will make in the future.

DISH currently uses two different versions of MPEG to compress our signals: MPEG-2 and the newer MPEG-4. MPEG-4 is about twice as efficient in compressing signals, meaning that a show that might take six billion ones and zeros with MPEG-2 would only take three billion with MPEG-4. While we won't go into the details of how these signals are made so much smaller until this chapter's *Technical Discussion*, it's not hard to see why MPEG-4 is our preferred method since we can send about twice as much content to our customers through the same pipes. That increased content can come in a combination of additional channels or higher quality video, making our service that much more valuable to subscribers.

So if MPEG-4 is that much better, why don't we use it for all of our signals? MPEG is just an agreed-upon standard for how the ones and zeros must look to any receiver. No matter how much the signal is compressed to save space (and increase the speed at which we can transmit it), the MPEG standard guarantees that any compliant receiver on the far end will be able to figure out how to put that compressed picture back together. DISH still has a large number of receivers in our customers' facilities that *only* have decoders able to speak MPEG-2. The standard used by MPEG-4 doesn't mean anything to them at all. Those receivers are typically older, standard-definition models that are becoming obsolete. However, until every single one of those receivers has been retired, we still must provide MPEG-2 signals for them. It's one of our more difficult business decisions on whether to continue supporting the millions of older receivers that only can decode the MPEG-2 standard or to spend the money to replace all those receivers that, to those customers who still have them, work perfectly well. Upsetting satisfied customers, even to improve their technical experience, always is a risk.

Encryption. There's another digital code we need to discuss, and that's the encryption code we put on our signal so it can only be viewed by authorized customers. Basically, every receiver's smart card is continually updated with codes that, first, let it know what programming it's allowed to see based on the packages and premium channels our customers have purchased and, second, tell it how to decrypt the signals it's getting from our satellites. Implicit in that explanation is the fact that during the MPEG compression process we scramble the bits, the ones and zeros that make up the signal, according to a very secret process. So the signal is not only MPEG-coded, but "DISH coded" as well.

Scrambling the signal is only part of the story. We've also got to ensure that our customers receive *all* the content they're paying for (and don't receive stuff for

free, except for our DISH Perks, of course). This other side of the encryption coin is being able to link the database our customer service agents use to sign customers up for packages and premium channels with the receivers in their homes, so the receivers can decode the right channels but no more. That process involves knowing which customer has which receiver, making sure the encryption keys associated with each and every channel go to every receiver, and making sure that the receivers only know how to use the keys for the channels their owners have purchased. A lot of that technology is very closely held, so we can't go into it in a lot more detail here.

Multiplexing. Finally, these coded signals are combined by a process known as multiplexing. Each of our seven satellites only has about 30 transponders—communications equipment that can receive and rebroadcast our signals. Those 200-ish transponders have to be able to carry the many hundreds of national channels and many more hundreds of local channels. There simply aren't enough transponders for each one of them to be reserved for a single channel, and they're very expensive to build and then orbit on satellites tens of thousands of miles above the Earth. Fortunately, multiplexing allows us to simultaneously use a single transponder to broadcast many, many channels. Multiplexers break apart each channel's signal, assigns a number identifying which channel it belongs to, and sends it up to the satellite. In this way, they can send multiple channels through the same "pipe", reassembling them on the other end via their identification numbers.

Summary. This section has been a rather abbreviated introduction to some very fundamental things we do to the DISH signal, from receiving it to preparing it for transmission up to our satellites. Some of the major points you should understand are:

- We buy content from providers, who send it to us via satellite transmissions, or over the Internet via high-capacity fiber optic cables
- We capture content from some local channels via over-the-air broadcasts
- We use either the MPEG-2 or the MPEG-4 standard to reduce the amount of data required to store programs, with MPEG-4 being about twice as efficient as MPEG-2
- We encrypt all of our transmissions to prevent unauthorized viewing

- We combine multiple program channels into a single transmission stream through a process called multiplexing in order to most efficiently use the satellite transmission frequencies we've been assigned.

If you're only reading The Basics, you can continue your study on page 38. However, we invite you to try out the first *Technical Discussion* below to see if it more thoroughly scratches your itch for knowledge on how the DISH network works.

Technical Discussion

This being the first of our Technical Discussions, it might be worth reminding you of their purpose. While the word *technical* might seem a bit scarier than *basics*, we've tried not to change the level of difficulty in these sections. The point is for us to be able to treat each of the topics we've already discussed at more than the executive-summary level. *The Basics* are all about breadth while the *Technical Discussions* are about depth—but without getting so deep you'll need to break out a pocket protector and a slide rule.

Content Acquisition Methods

For the most part, the description of how we acquire our programming in *The Basics* section above is about as detailed as we'll need to get. We receive signals from content providers such as HBO, the major broadcast networks, and local channels either through high-speed fiber connections, downlinks from providers' satellites, or, for some local channels only, from over-the-air reception of broadcast transmissions. In every case, the signals are either already coded into an MPEG-standard data stream (we'll discuss that process in detail later in this chapter), or we immediately convert it into an MPEG. Regardless of where we actually receive the signals, they all end up in computers at one of our uplink centers.

DISH has a number of uplink centers for our television business located around the country. While we'll refer to them as *DISH centers*, like some of "our" satellites, they're actually owned and operated by our sister company, EchoStar. Figure 3 shows the location of our two primary and four secondary uplink centers. As we'll see later in the *DISH Satellites* chapter, we use seven satellites to link these uplink centers with our customers: three in what is called the Western Arc, three in the Eastern Arc, and one for international programming. While there is

Figure 3: Uplink Center Locations.[2]

some overlap, our Cheyenne center primarily handles communication to the Western Arc and Gilbert handles the Eastern Arc.

While we control the signals we *transmit* on those seven satellites, many of the signals we *receive* come from providers' satellites in the geostationary belt, a term we'll discuss in the *DISH Satellites* chapter. However, for the purposes of this discussion, you just need to know the satellites are all arranged in a ring around Earth about 22,000 miles above the equator. In order to receive those signals, we have to have antennas.

Most of the large satellite antennas you've likely seen look like the 40-foot diameter circular parabolic antennas at our Cheyenne uplink center in Frame I of Figure 4 We'll go into detail on why we call them parabolic dishes in *The Uplink Center* chapter, but we need to point out here that our reception antennas for the signals from our providers' many satellites are not your typical circular, parabolic dishes. Instead, at Gilbert and Cheyenne we use what's called a torus antenna, as shown in Frame II. That antenna is over 100 feet across.

The word *torus* means *doughnut shaped*, as shown in Frame III

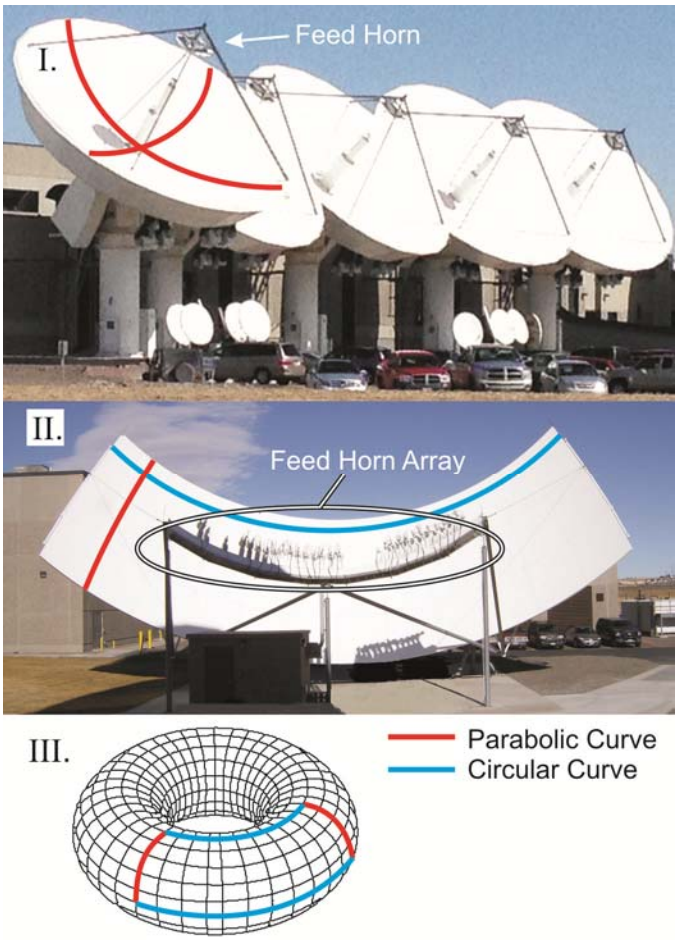

Figure 4: Torus Reception Antenna.[3]

(note that we're seeing the *inside* of the doughnut in the picture of the antenna in Frame II but the *outside* of what the antenna would look like with the outline in Frame III). However, the torus we use for this antenna is special: in one direction its curve takes the shape of a parabola[4] (indicated by the red curves in all three frames) while in the other direction it's circular (blue curves). The circular curve allows us to receive signals from more than one satellite in the geostationary belt, in contrast to the parabolic dish which is very directional and is designed to receive signals from only one satellite. With the torus antenna, signals from individual satellites are directed to one (and only one) of the feed horns in the array. The parabolic curve helps to collect and focus the energy from each of those satellites onto the proper feed horn. Again, we'll discuss the shapes and functions of the antennas in much more detail later, but the torus antenna is different enough that it's worth pointing out here.

So, regardless of the way we receive programming from our providers—via fiber, over the air broadcasts, and from satellites—it all ends up in an MPEG data stream at one of our uplink centers.

What is MPEG? [5]

MPEG is a word that's bandied about within many parts of our company, but what does it really mean and why do we place so much emphasis on it?

MPEG stands for Moving Picture Experts Group, the organization that designed their eponymous coding scheme. What MPEG is, though, is a standard way to compress pictures and audio format so they don't take so much data bandwidth to transmit. A note on terminology: while we'll go into the subject of data bandwidth in detail in *The Uplink Center* chapter later, what we need to understand at this point is the use of MPEG compression significantly reduces the amount of information we need to transmit. And since we pay for (and eventually have to charge our customers for) every one or zero we send, that savings can be substantial. Additionally, in many cases the cost of data links goes up with the required data transmission rates, so reducing that required rate can produce significant cost savings for DISH. For that reason, we'll primarily discuss the *size* of our programs in terms of *required storage space* instead of speed of transmission during this chapter. For those of you who already understand data bandwidth, this decision may seem a bit stilted, but we'll build up to the data bandwidth concept in later chapters.

As shown in Figure 5, a high-definition (HD) movie can contain over a *million billion bytes* (a byte is a string of eight ones and/or zeros), or in other words over a terabyte of information. (There is a list of prefixes and abbreviations for numbers using millions, billions, and similar quantities on p. xxi. It might be worth a quick glance at this time.) Since at the time of this writing the largest hard drive DISH offers on our digital video recorders will only hold about 2 TB (yes, the *T* prefix in front of the abbreviation for *byte* is in that same table that you might want to look at…), at that rate we couldn't store more than *one* full-length movie having this much information content! Not to mention the exorbitant cost of sending 155 MB per second of information *for each HD channel* we provide to our customers. With MPEG, instead of a movie taking up a terabyte of storage space, it really takes up about 5 GB, or 1/200th the space, and DISH can transmit a single HD channel using only between about 1/10 and 1½ MB per second. A lot smaller and faster, huh?

Factor	*Value*	*Result*
Screen width Screen height	1920 1080	2,073,600 pixels per frame
Colors per pixel	3 ●●●	6,220,800 colors per frame
Bits per color	8	49,766,400 bits per frame
Bytes per bit	8	6,220,800 bytes per frame
Frames per second	25	155,520,000 bytes per second
Seconds per minute	60	9,331,200,000 bytes per minute
Minutes per hour	60	559,872,000,000 bytes per hour
Hours per movie	2	**1,119,774,000,000 bytes per movie**

Figure 5: Information Content in an HD Movie.

YCbCr Down-Sampling. So how does MPEG achieve such huge reductions in file size? Well, first of all, some of that size reduction isn't related to MPEG. It's been done since the dawn of color television broadcasts. The movie size quoted above was for a Blu-Ray-quality picture, where the color depth was eight bits per color for each of the three colors that are used to make all the other colors. In that example, 24-bit color using the entire RGB (red/green/blue) signal was stored. That's exceptionally good color, and we don't typically ever see that on any

broadcast television signals. In fact, 24-bit color gives us the possibility of producing 2^{24}, (pronounced *two-to-the-twenty-fourth power*) or almost 17 million, different colors. Estimates of the number of distinct colors the human eye can distinguish range from one to ten million,[6] so 24-bit color is way more than enough.

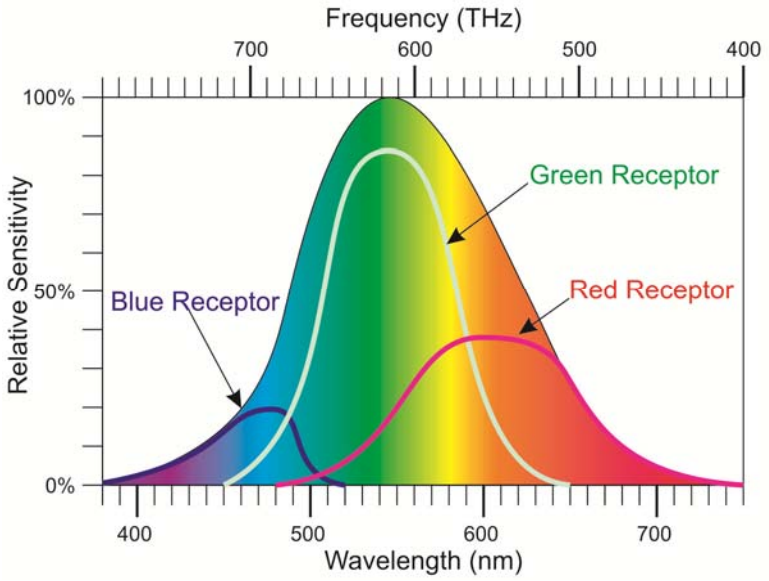

Figure 6: Color Response of the Human Eye.[7]

It turns out that there's a whole lot of physiology behind video compression algorithms, which make sense because the first place we could find unnecessary data to eliminate from our transmissions is in places humans can't or don't perceive it. As shown in Figure 6, the human eye's sensitivity to light varies with color. Don't worry about the scales at the top and bottom for now—we'll talk about frequency and wavelength at length in the next chapter. Just note that the eye's maximum sensitivity occurs in the green region of the spectrum and falls off sharply at the red and blue ends of the visible range. Also shown in the figure are the sensitivities of the three types of color receptors we have inside our eyes. Note that the blue receptor isn't nearly as sensitive as the green (its curve doesn't go up as much) and that the red receptor covers a much wider region of the spectrum than the other sensors.

When looking for fine detail, human eyes are much more sensitive to changes in brightness than they are to differences in color, and as we just saw, they are more sensitive to some colors than others. Instead of capturing and storing the full RGB, the most commonly used method of color image capture uses a weighted average of the RGB signals to store a grey-scale image—called *luminance*[8]—similar to what you'd see in a black and white (B&W) picture. The weighting values reflect the ability of the eye to see those colors, so the B&W image reflects what the eye would actually see in terms of brightness/darkness. As you'd expect from the eye sensitivity figure, the weighting factor for green (71.5%) is higher than the factor for red (21.3%), which is higher than that for blue (7.2%).[9] Once

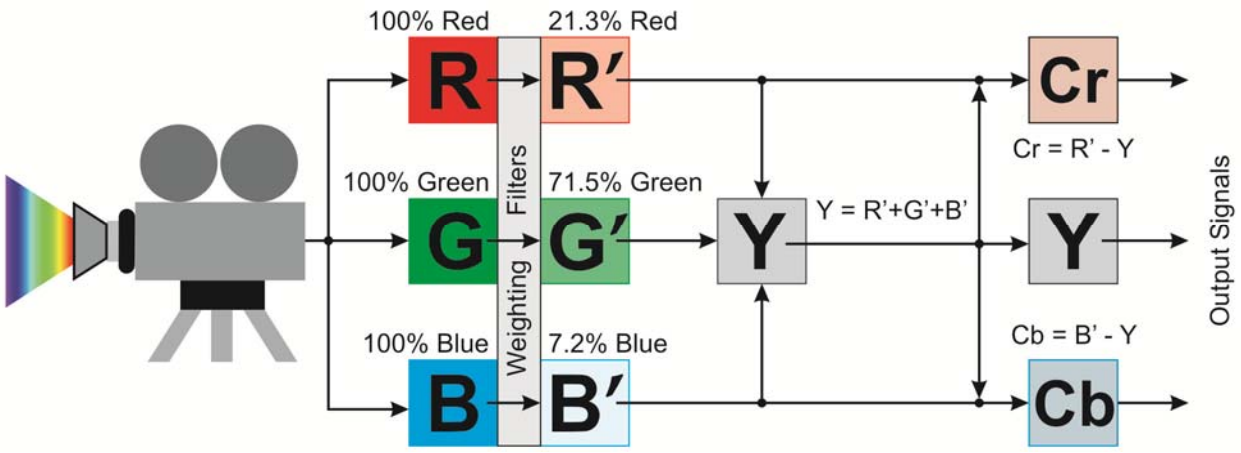

Figure 7: Signal Compression Using Color Difference Signals.

the luminance signal has been created, it is subtracted from the original weighted red and blue signals to produce red and blue *chrominance* signals, signals that carry the *color difference* from B&W, as illustrated in Figure 7. These signals are typically referred to as Cr (red) and Cb (blue), while the luminance signal is designated Y. Even though the green signal is not directly transmitted in the YCbCr system, it can be reconstructed from these three components.[10]

So that's a lot of information about changing light into signals, but how does that save us data storage space? The key is recalling that the human eye is not as sensitive to changes in color as it is to changes in luminance. That means we don't have to send as much data for color as we do for brightness. Figure 8 shows how this saves us storage space. In Frame I, we show a representative grid of

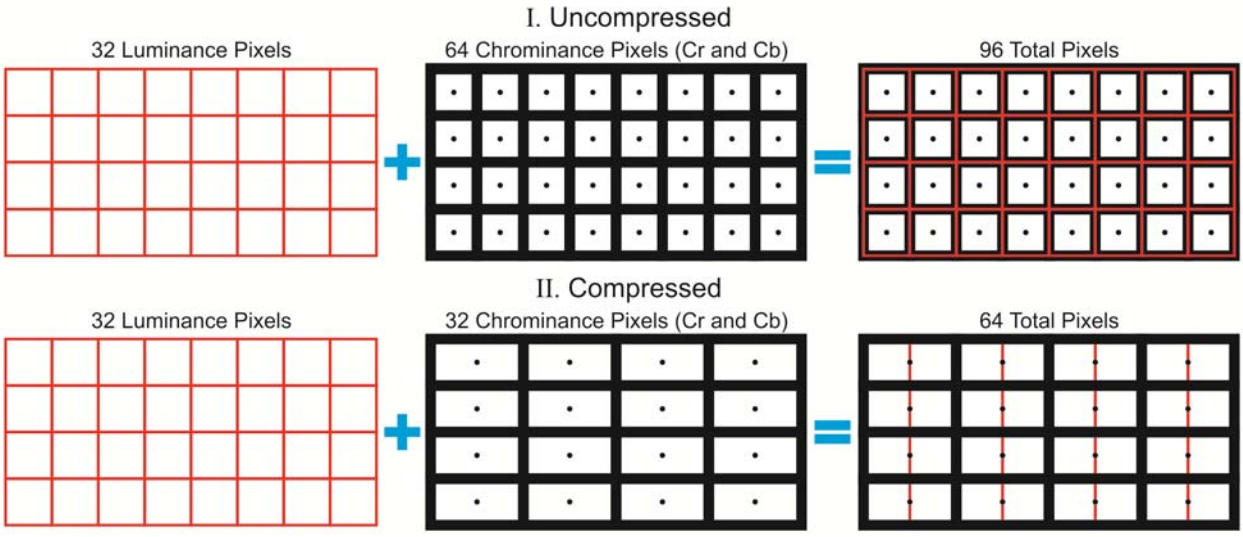

Figure 8: Chrominance Sub-sampling to Save Storage Space.

eight by four pixels on a TV screen (a **pixel** is a *picture element*, or the smallest portion of a picture than can be individually changed). The left, red-coded pixels represent the B&W data we would send for the luminance portion of the picture. The middle, black-coded pixels represent the color difference signal we'd send to recreate the color information for the signal. Note that in the figure only one set of pixels is shown in the middle section of each frame, but there would actually be two sets, one each for the Cr and Cb information. As indicated in the figure, adding all of these pixel data requirements together means we'd need to send data for 96 total pixels. This situation, where every luminance pixel is complemented by both a red and a blue chrominance pixel, is completely uncompressed—all the data gathered by the camera is transmitted.

In Frame II, however, we've employed physiology-based compression to reduce the amount of pixel information we send. Since the eye is very sensitive to brightness, we still send information on the same 32 luminance pixel. However, we now take the chrominance information for each two adjacent pixels and average them, sending only the average value in one chrominance pixel to cover two luminance pixels. That means instead of information for 32 pixels for both Cr and Cb, we only send the information for 16 pixels each—a total of 64 total luminance and chrominance pixels for which we need information. That's a ⅓ saving in storage requirements just based on the eye's physiology! And the eye really won't detect the difference between these two images at a normal viewing distance (i.e., where the eye can't distinguish individual pixels). The specific compression scheme shown here is just one of many that are typically used,[11] but it's quite representative of how this sort of video data compression is obtained.

But we promised we'd talk about MPEG in this section, and we really haven't gotten to that yet. Patience, grasshopper. We're about to explore that topic in depth. We just needed to examine some basic compression techniques before moving on to more advanced ones. And, in fact, MPEG actually does take advantage of this three-channel luminance/chrominance system, but that compression technique predated MPEG by decades.

MPEG Basics. MPEG is a method of coding images (and sound) with an expensive and complicated encoder so that it can be decoded by a relatively simple and inexpensive decoder on the far end. That system makes a lot of sense for our business model, since few encoders are required compared to the huge number of decoders in all of our receivers. However, *the MPEG standard does not define the encoders or decoders*. It only deals with the digital stream of ones

and zeroes that pass *between* those two pieces of hardware. A bitstream that conforms to the MPEG standard is called an MPEG-compliant bitstream. *How* a video image is converted into this bitstream and *how* the bitstream is reconverted back into video is up to individual hardware and software developers. The reason for this approach is to enable future developers to employ ever-more-sophisticated techniques to increase the performance (speed and/or quality) of the encoders and decoders—things that may be proprietary to specific companies—while ensuring compatibility with existing decoders.

Long-term bitstream compatibility is a huge issue for many companies including DISH. We'll look at this issue in more detail in the *Inside Our Customer's Home: Receivers* chapter, but some of our receivers are only compliant with an older version of MPEG. That fact puts severe financial and operational constraints on not only our installers, but on our satellite constellation as well.

By mentioning "an older version of MPEG," we've revealed that the Experts Group's "standard" solution isn't really written in stone. The MPEG standard has evolved over time as technology has improved, but the group tries to make the standard as long-lived as possible to avoid exactly the kind of financial problems they've caused DISH. One thing they do to help ensure continuity, though, is to require the decoders capable of reading a newer version of MPEG to also be able to read all older versions as well—for example, the MPEG-4-compliant bitstream includes all the features of MPEG-2. While there are other standards besides MPEG-2 and MPEG-4, it's those we'll concentrate on in this document, since those are the two on which DISH receivers are currently based.

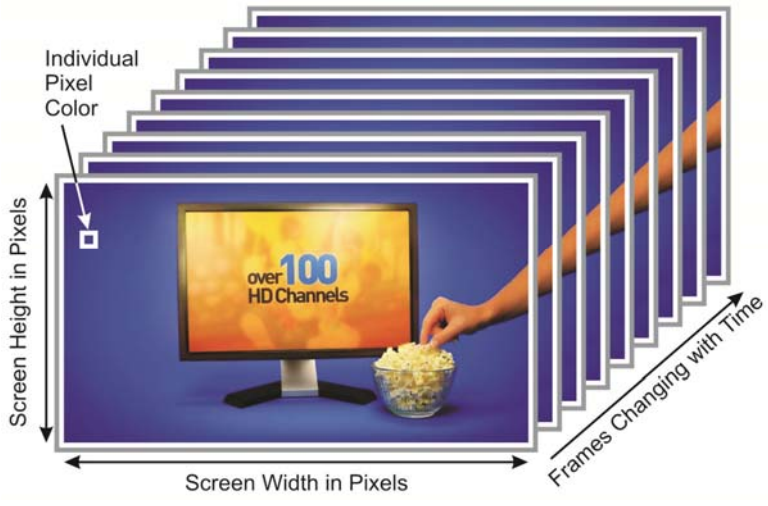

Figure 9: Compression Dimensions in a Video Stream.

As shown in Figure 9, there are four basic dimensions to any video signal: the *width* and *height* of the screen as measured in pixels, the *coloring* of each individual pixel, and the changes to the pixels over *time*. Data related to each of these dimensions can be compressed, however some of those compressions are easier to do than others.

Let's look at how we can do some compression on each of these dimensions.

Run-Length Compression. We've already looked at compressing the individual pixel coloration in our discussion of YCbCr down-sampling, where we used the fact that the eye doesn't need color information as much as it does brightness. That technique reduced our data requirements by a third or more. We can also compress the data needed to color each individual picture by another method: noting similarities from pixel to pixel *within an individual frame.* (This within-a-frame distinction is important, as the both the MPEG-2 and more advanced MPEG-4 systems can look for similarities *between* frames; we'll discuss that later.)

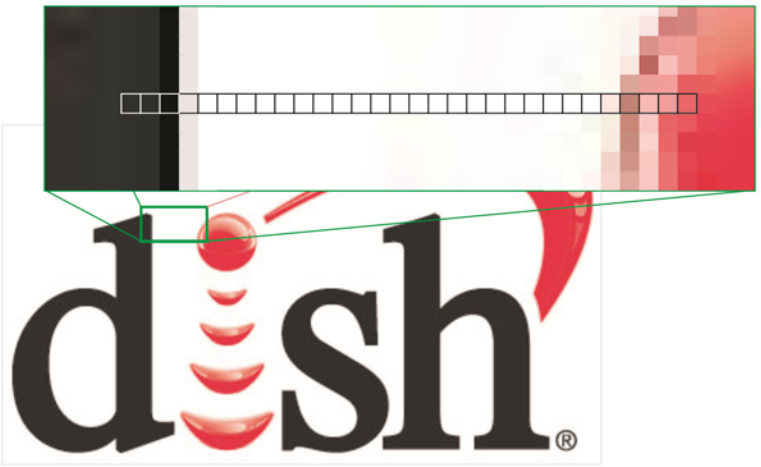

Figure 10: Reducing the Data Sent for Repetitive Pixels.

Take a look at Figure 10. It shows a relatively high quality rendering of the DISH logo, but then blows up a section of it in the upper inset to show that the digital picture is really made up of pixels. A line of 30 pixels has been highlighted for this illustration. Starting from the left, there are three black pixels, a grey pixel, then 21 white pixels, and finally a series of five slightly different red-shades. In an uncompressed world, we'd send this as eight bits time three colors for each of the pixels, for a total of $8 \times 3 \times 30 = 720$ bits.

However, what if we had a way to count same-color pixels? We could then send code that said pixel 1 is black (using the three eight-bit numbers that indicate the combination of red, green, and blue that combine to make black), ditto, ditto, pixel 4 is grey, pixel 5 is white, ditto 1, ditto 2,…,ditto 20, pixel 26 is reddish, and so on. Instead of sending the pixel color codes for the repeated colors, we'd just send code that said "black, repeat twice, grey, white, repeat 20 times, reddish…" There will obviously be some overhead data needed (data not related directly to colors) to let the computer know how many times to repeat, so maybe it's not worth it to use the shortcut for the three black pixels. However, the 21 white ones look like they're just begging for this compression technique. Now, we will send 8×3 bits for the *three black* pixels, the *grey* pixel, one of the *white* pixels,

and each of the *five reddish* ones, plus the small overhead to say "repeat 21 times." That comes out to ten colors $(3 + 1 + 1 + 5)$ needing $8 \times 3 \times 10 = 240$ bits plus change. Quite the data savings over the original 720, isn't it? And that's the gist of compression of repetitive regions. Just say "ditto" in the code and you don't have to actually send the information.

Moving Object Recognition. Pixel-by-pixel compression is a very rudimentary form of compression. Repetition can be exploited even more effectively when looking at a series of frames over time. Figure 11 shows how this can save huge amounts of data. Let's say we have a fixed camera location focused on a lovely building. A bright red Ferrari speeds through the camera's field of view. Frame I shows the initial location of the car and Frame II shows the second location of the car. Notice that the information content of the two frames is almost exactly the same. Frame III shows the portion of the image that has changed between Frames I and II.

One of the ways that MPEG is able to achieve such great compression ratios is that it looks for similarities between frames so it doesn't have to transmit the same information more than once. Then, it only has to transmit the changed information. We'll see how this works in a few pages, but imagine if the above frames were from an HDTV show. Each uncompressed frame contains about 6.2 MB of information. In this example, the car and the "shadow" of the car shown in Frame III take up less than 1/10th of the frame, so sending only the changes yields a savings of 5.6 MB for one frame alone. Even more amazingly, MPEG encoders can actually determine what part of the frame was the part that

Figure 11: Moving Foreground Compression.

moved, and can tell the decoder, “take the red Ferrari and move it 500 pixels to the left and 20 pixels down.” Of course, it’s not in English like that, but the end result is the same. Then, it only has to transmit the car’s “shadow” since the information about the car has already been sent.

MPEG can also work with more complicated camera set-ups like pans and zooms. The encoder can determine the direction of motion of the camera, find similarities between the backgrounds, and again only send the difference between the frames. Plus, it can look ahead a few (MPEG-2) or many (MPEG-4) frames and find the things common to all of them, determining where any moving objects are headed, and projecting their motion via mathematics instead of data transfer—a much more efficient way to do business. Using our Ferrari example, Figure 12 shows how the encoder determines an axis along which the motion appears to flow. Notice that this axis will *not* point in the same direction as the time flow axis, otherwise the object would appear to be stationary. Frame I shows the starting and ending positions of the car, with the motion flow axis indicated as linking the rear axle of the car in both positions. Frame II shows how the car can then be recreated in exactly the right position for intermediate frames just by moving it along the motion flow axis.

The way these recreations actually work in practice is that the MPEG encoder scans the reference picture (the first frame in our original Ferrari example, Figure 11.) It then looks at subsequent frames to determine the motion flow axis (or axes, for scenes with more than one moving object or where the background and/or distinct objects are moving, as could be the case during a pan or zoom). The encoder then *predicts*, based on these motion axes, what the picture should look like for the next few frames. Those

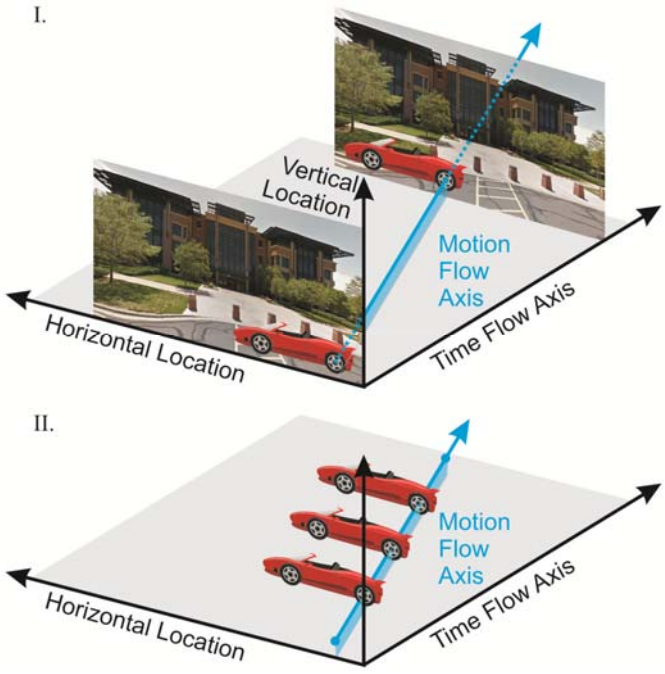

Figure 12: The Motion Flow Axis.

Figure 13: Flat and Warped Images.[12]

predictions are just mathematical shifts of the picture, which takes up a *lot* less memory than sending the actual picture. The sending-math-is-simpler concept is similar to what we saw with the repetitive pixel discussion related to Figure 10 (p. 19).

Warping. Another way MPEG can save storage space between frames for shapes it recognizes is to use a technique known as warping. By knowing what an object looks like flat, MPEG encoders can "drape" the colors of the flat object onto an object moving in the real world. For example, the two objects shown in Figure 13, the Colorado flag and the brightly colored car, are seen in their top representations as flat images. As the flag is seen to wave in the wind, the MPEG data stream would only need to transmit the pixels for the flag's colors once, and then would just need to determine the shape of the flag in subsequent frames to be able to drape those colors onto the waving flag. Similarly, once it recognizes the side-view of the car, it can follow the distortions of the car's shape as it makes a turn to the left, as shown in the lower part of the image. The portion of that image enclosed in the red outline could be mathematically warped from the already-transmitted side-view, while the remainder of the image would need to be transmitted as it comes into view during the turn (since it is not contained in the original view of the car).

The MPEG Coding Process. So, how do all of these techniques like warping and moving object recognition actually get put into practice? The encoder uses three kinds of frames. The first kind is a complete, self-sufficient image which we'll call the *intra*-coded image or **I-frame**. It's a frame with compression based solely on what can be done with respect to redundancies *within that frame alone*; run-length compression is an example of this type of intra-frame compression. The other two types of coding are related to *inter*-coded frames, frames that exploit *redundancies between different frames*.

The first type of inter-coded frame is a predicted image or **P-frame**. That kind of frame is built from the *difference* between the I-frame and a subsequent frame. The encoder holds the I-frame in memory, compares it to subsequent frames, predicts the motion, converts those predictions to mathematical shifts/warps (which take up significantly less memory than the actual image) and stores those predictions. It then uses those predictions internally to generate a completely new image and then compares that predicted image to the actual image. It then notes any differences between the predicted image and the actual image and stores *those* differences. The only data sent for the P-frame is the math and the difference information. That's where the data savings are realized because, as we've seen, sending the math takes up a whole lot less space than full image data.

Figure 14 illustrates this process, with the process flow shown in Frame I. The encoder captures the I-frame, which in this figure is shown in Frame II. It stores the frame in memory then compresses this picture using only *intra*-frame techniques. It then sends the compressed image to the decoder. Next, the encoder captures and stores the next frame in the video stream, Frame III in our example. It then uses *inter*-frame techniques to predict any motion between the frames, storing the math it uses for this prediction. Since predictions aren't perfect, it

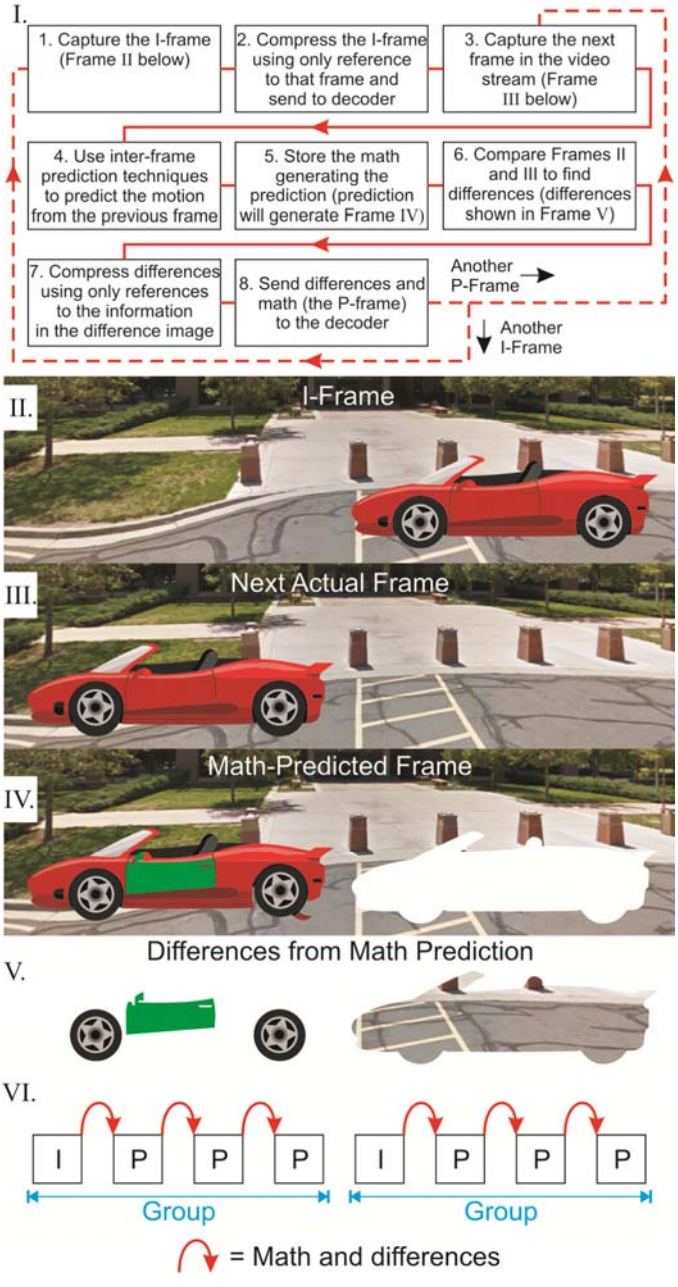

Figure 14: Generating Predicted Frames.

then generates what it thinks would be the subsequent frame based on this math. In our example, that predicted frame is shown in Frame IV.

The encoder then looks for any differences between the captured frame and what it predicted. In our example, the differences are that the car's door has turned green (a gross exaggeration of an error that wouldn't likely occur in a real encoder, used here for the purposes of illustration), the wheels have turned a bit (the encoder probably would have caught that, too, but it's more subtle), and the bit of the road that was hidden behind the car is now in view (there's no way any encoder could know that information). Those differences are shown in Frame V.

The encoder then compresses the differences using *intra*-frame techniques and only sends the math and the compressed difference information to the decoder. The decoder takes the I-frame, uncompresses it, and displays it. It then takes the I-frame, performs the math and applies the uncompressed difference information to generate the P-frame, and displays that frame.

As shown in Frame VI and by the two dotted line paths in Frame I, this process of generating P-frames can go on for several frames before another full I-frame is generated. There is a limit, though, to how many P-frames can be used. Think about what would happen were the I-frame to become corrupt during transmission. All frames in the group based upon this I-frame would also be corrupt. For that reason, encoders generate I-frames quite regularly.

You may have noticed that one of the largest items in the difference image from Figure 14 was the portion of the picture that was hidden from view behind the car in the I-frame. The revelation of new image information from behind moving objects can be a significant source of extra data to transmit between predicted pictures. One way to significantly reduce the amount of data transmitted to fill in these gaps is to use *bi-directional prediction*, that is, prediction that uses both previous frames and future frames to generate intermediate frames. Frames generated from bi-directional prediction, **B-frames**, are the second type of inter-coded frames, along with the P-frames we've already discussed.

Frame I of Figure 15 shows a series of four frames from a simulated video sequence. Can you see that the entire background information is contained in the first and last frames of this sequence? In order to avoid having to generate and transmit the information about the background between all four frames, the encoder takes the first frame, stores it, then compresses and sends it. It then stores the second, third, and fourth frames (it could be more frames based on the

sophistication of the encoder, but we'll be able to illustrate the process using only four). It then does the prediction process we described above for a P-frame, but between the *first* and the *fourth* frames only.

The encoder then transmits the P-frame (corresponding to the fourth video frame), inserting a small amount of code that essentially tells the decoder, "Hey! Don't display this frame right away. I'll be sending two more that go in between this one and the last one in just a second." It then performs a predictive math/difference process on the second frame using *both* the first and the fourth frames. Notice that we don't have to send any information about background differences with this B-frame, since the entire background is known from the combination of the first and fourth frames. In this example, the only difference information that would have to be transmitted would be the rotation of the tires, which differs in each frame.

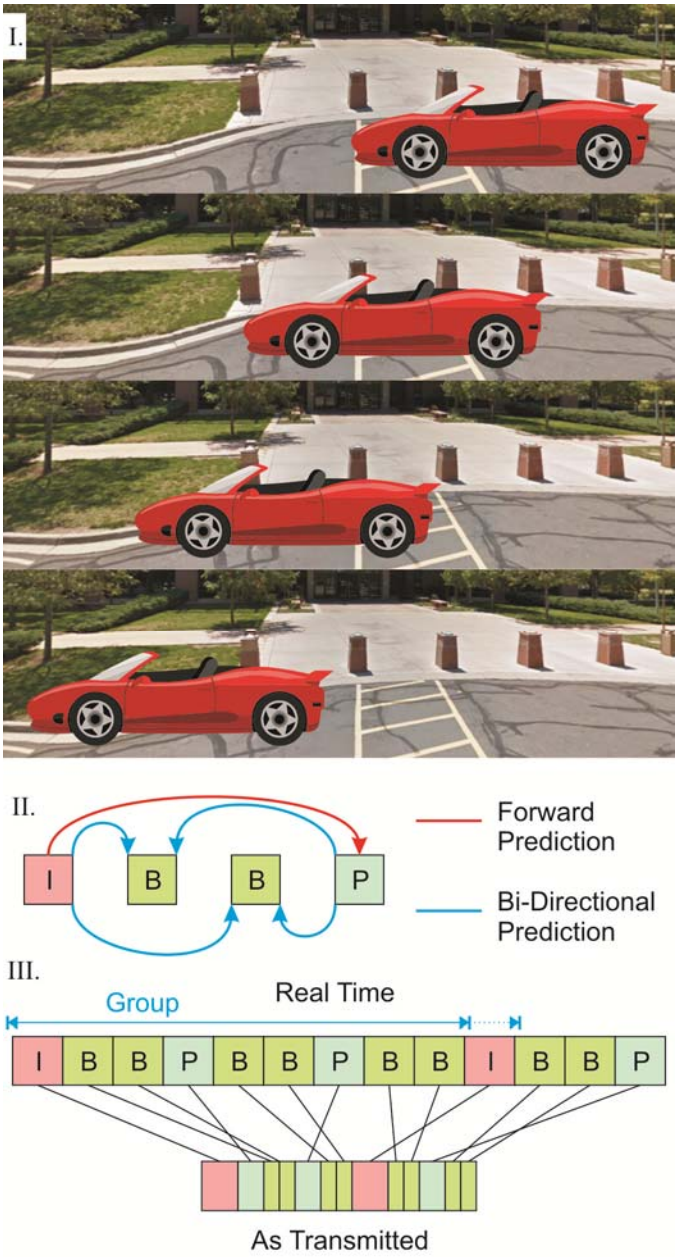

Figure 15: Generating Bi-Directionally Predicted Frames.[13]

Frame II shows how the first and fourth frames (the I- and P-frames, respectively) are used to generate the second and third frames (B-frames). Information from the actual I-frame image and the actual P-frame image are used to generate the math and difference information for the compressed P-frame, only projecting the I-frame into the future (forward prediction). Then, information about the actual I-

frame image, the compressed P-frame, and the actual B-frame images are used to generate the math and difference information about the compressed B-frames (bi-directional compression).

Frame III illustrates how the frames are transmitted to the decoder out of order. The top row of boxes shows the frames as they hit the encoder in real time. The I-frame on the left comes first, followed by two B-frames, etc. However, when they're transmitted, both the I-frame and the P-frame are necessary to reconstruct the two B-frames, so the P-frame is transmitted before the B-frames. Also notice that the transmitted stream is compressed (it takes up less horizontal distance in this representation), with I-frames taking more transmission space than P-frames, which in turn take more space than highly-compressed B-frames. (The actual amount of compression shown in the figure is for illustration only to show these relative sizings—real compression amounts will not be the same as shown.) Finally, notice that the group of pictures dependent upon the left I-frame extends all the way to the next I-frame. However, the group isn't completely independent of the next group since the last two B-frames in the group depend upon the I-frame in the next group. That dependence is graphically illustrated by the fact that the second I-frame is transmitted before the last two B-frames of the first group.

While we've shown two B-frames in between I- and P-frames in this illustration, MPEG-2 can only generate one B-frame between the two. MPEG-4 can generate one, two, or more B-frames between I- and P-frames, depending upon the complexity of the encoder. Recalling that B-frames take significantly less memory space, those additional B-frames can lead to a huge compression advantage.

Inter-frame compression techniques like these give the best results—i.e., the most compression—for a channel like Bloomberg, where the only thing that changes much is the scroll at the bottom of the screen and the lips on the talking head's face. As can be imagined from the left frame in Figure 16, it's got to be pretty easy to predict what a large part of

Figure 16: Compressionally Passive vs. Active Programming.[14]

the screen on Bloomberg will look like for fairly long periods of time. The technique definitely doesn't work well for a basketball game on a parquet floor, where the camera pans to keep the action in view and both the foreground players and background court are made up of constantly-moving, complicated pixel groups, such as the right frame in the figure. Such sporting events do take advantage of frame-to-frame compression, but don't achieve anywhere near the gains that more placid channels do. A close look at how DISH allocates channels on our transponders will show that we don't typically put active channels like ESPN together with other sports networks, instead balancing them with more passive channels so the data requirements are better spread out between transponders. We'll look at such channel allocations later when we discuss receivers in the *Inside Our Customer's Home: Receivers* chapter.

Audio Compression. Of course, our video would need a tinny piano player banging away in the background at every customer's home if it weren't for the audio stream that accompanies the pictures. We don't do silent movies! Audio is encoded similarly to video, and similar physiological considerations go into making the compression as undetectable to the ear as the video compressions are to the eye.

The ear is every bit as complex an instrument as the eye, and a great deal of research has gone into exactly how that organ takes pressure waves in the air, turns them into mechanical movements of some fairly complex structures inside the ear, and finally into the sounds we perceive in our brains. As shown in Figure 17, the external ear acts somewhat like an antenna, collecting and directing sound waves down the ear canal. There, the eardrum responds to the changes in pressure in the sound wave and moves in and out, its movements causing the hammer to move with it. The hammer moves the anvil, which in turn moves the stirrup, which vibrates a special spot on the cochlea which we'll discuss in a few paragraphs.

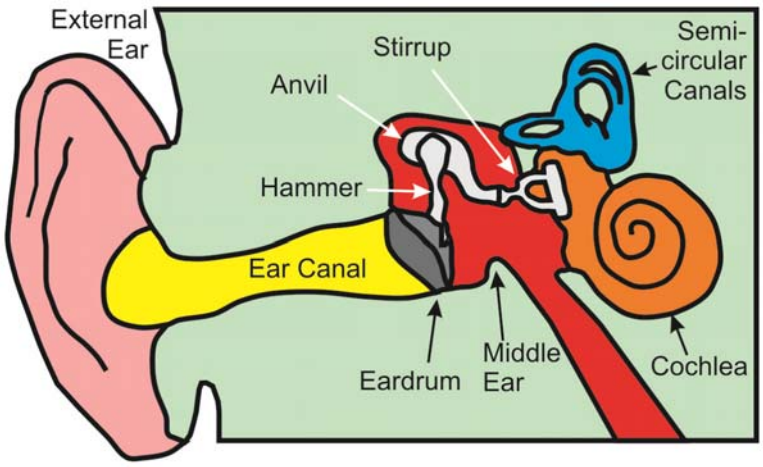

Figure 17: Structure of the Ear.[15]

Humans have what we like to call 20/20 hearing, but

the definition of the 20s is very different from what you're more used to with vision. In this case, the first 20 means twenty hertz, or low pitches with twenty vibrations of a sound wave per second. The second 20 means twenty kilohertz, or very high pitches with twenty thousand vibrations of a sound wave per second. That's the typical range of frequencies our ears can hear—hence, the 20/20 mnemonic.

Knowing the range of frequencies we can hear helps a bit with audio compression. We obviously don't need to send information about signals out of that range since they would just be wasted data. However, the necessary frequency range isn't the only important thing to know about our hearing. Knowing how discriminating the ear is would also be a great bit of information to use in our compression schemes.

That special spot on the cochlea we mentioned earlier on which the stirrup vibrates is attached to a long, thin membrane that winds around inside the entire length of the spiral cochlea. This *basilar membrane* is tapered very precisely so that certain sound frequencies only cause the part of the membrane with just the right thickness to vibrate. This quality is illustrated in Frame I of Figure 18. The membrane is very thin near the eardrum and the bones of the middle ear, thickening the further away it gets. (This membrane is actually coiled up inside the cochlea, but is shown in this figure stretched out in a straight line for clarity.) Frame II illustrates how the membrane vibrates in response to both loud and soft

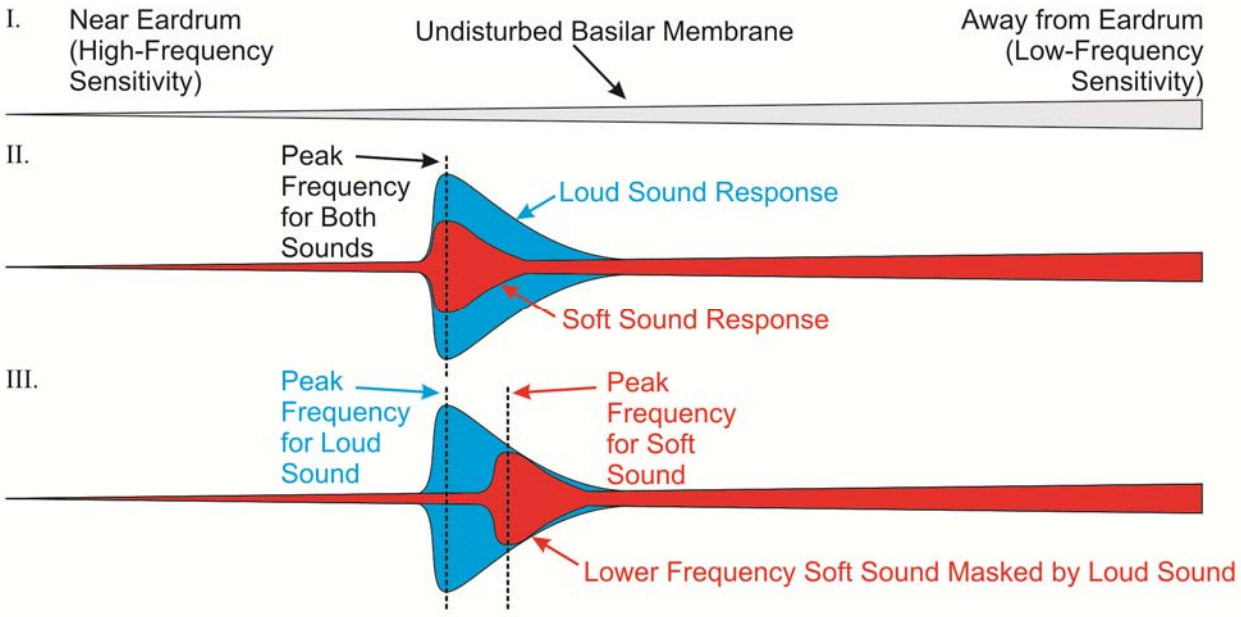

Figure 18: Sound Masking in the Inner Ear.[16]

sounds, with the colored outlines showing the regions and maximum heights of vibrations of the membrane when it responds to sounds of certain frequencies. The heights of the vibrations are exaggerated in this figure for clarity, but the shapes of the responses are reasonably accurate. This frame shows that a loud noise of one frequency drowns out a soft one at the same frequency. No huge news flash there.

To this point in our discussion of audio, all of this information is just physiological trivia with respect to MPEG. However, take a look at Frame III. Now we have two sounds, a loud one and a soft one, but they peak at *different* frequencies. Notice that the loud noise at the higher frequency drowns out the softer noise at the lower frequency. Now *that* is something we can use to compress our audio data stream. If we know the response of this membrane to different volumes and different frequencies, we can use that knowledge to reduce the amount of information we have to send. If, for example, we have the situation in Frame III, there's no point in sending information about the softer signal since it will never be heard by the ear.

There are a multitude of additional techniques for compressing MPEG audio files, all based on auditory physiology. However, we'll end our discussion of this topic after only giving the single example above. Why, you may ask, since we devoted multiple pages to MPEG video compression, are we giving short shrift to audio? It's a matter of importance. A typical audio file uses up about eight *thousand* bytes every second, while a video file of the same length uses about five *million* bytes. The audio files are less than 1% the size of video files, so we don't gain that much by working hard to compress them.

One final bit of information about audio files before we move on: when you watch a TV show, you get both the audio track and the video track, with the implication that they're all part of the same file. That's not true at all. DISH sends all of our audio channels up separately from the video channels so they can be combined at the customer's convenience based on language or other needs. Making sure the right audio ends up with the right video is just another bit of the magic performed by our receivers, and we'll discuss how that works in the *Inside Our Customer's Home: Receivers* chapter.

***The Business Case for MPEG Encoders and Decoders*.** Of course, all of these tricks for both audio and video compression require very complicated, very expensive encoders to perform the computations, but they still output an MPEG-

standard bitstream that can be *interpreted by relatively simple (and inexpensive) decoders*. The more time an encoder has to work, the better a job at finding redundancies in the frames it can do. That's why a live video broadcast takes up more storage space than the same broadcast that we've stored and prepared for later transmission such as internet video on demand—the live stream is being encoded on the fly, and our customers typically won't accept more than a few seconds delay for the encoder to work its magic.

And, again, the MPEG system works well for the DISH business model because we can spend a whole lot of money on a small number of very good, very capable encoders and not nearly so much money on the decoders in each of the millions of receivers in our customers' homes and businesses. Some of our encoders are great at on-the-fly compression, looking only a few frames in advance and compressing as much as they can in a very short time. We use these for the live broadcasts that form the bulk of our current business. We also buy different encoders for our pay-per-view and on-demand offerings, since that content doesn't go out live. That gives us a lot more time for the encoders to squeeze the images down to the absolute minimum bits possible. That's why our *Blockbuster at Home* movies can show up very rapidly over the Internet—they've been compressed just as much as possible and don't take up very much space at all.

But regardless of which kind of encoding we use—on-the-fly or plenty-of-time—the relatively inexpensive decoders on our receivers don't care. The encoders put out a *standard* MPEG code that tells the decoders what they need to do to put the compressed pictures back together again so our customers' eyes can make sense of them. These decoders can be cheap because the stream is standardized, regardless of the money that went into the encoders' capabilities.

MPEG-4 vs. MPEG-2. So why is MPEG-4 so much better than MPEG-2? The fact that its number is higher seems to imply that it's newer, and in fact it is. MPEG-2 was first published in 1995 and its latest update was in 2007. MPEG-4 was first published four years later in 1999 and new parts are still being developed. There are already 28 annexes to the original standard and more are in the works.

Many of the compression techniques we discussed above are either not present in MPEG-2, or are only there in rudimentary forms. One example is the *adaptive motion regions* we illustrated by moving the Ferrari around. MPEG-4 can really recognize regions shaped like a Ferrari—the regions it recognizes adapt to the shape of the objects that are moving. MPEG-2, on the other hand, uses fixed,

4x4-pixel macro blocks for motion sensing, and that's not a very large area. It can tell that specific blocks have moved from frame to frame, but the data savings are nowhere near as large because we still have to transmit information about every one of those macro blocks. Much like the blind men and the elephant, MPEG-2 doesn't know what object it's manipulating. In many cases, MPEG-4, knows that the trunk and the tail and the huge legs belong to the same object. Figure 19 shows a few more of the many differences between the two flavors of MPEG, and by now it should be obvious that MPEG-4 is the much more sophisticated version.

Feature	MPEG-2	MPEG-4
Run-Length Encoding	X	X
YCrCb Chroma Down-Sampling	X	X
Advanced Compression Techniques		X
Warping		X
Adaptive Motion Regions		X
Two-Way (Past and Future) Predictive Reference Frames	±1	±Many
Relative Bitrate Required	1	< ½

Figure 19: Comparison of a Few MPEG-2 and MPEG-4 Features.

As you can imagine, MPEG compression is much more complicated than we've hit on here, and many scholarly books and articles have been written on the subject. We could throw out phrases like discrete-cosine transfers, spatial and temporal frequency analysis, and such, but that's well beyond the scope of what we're trying to accomplish here. If you're interested, they're all discussed in detail in the references from the bibliography. However, the takeaway for MPEG is that there are a number of ways to reduce the amount of ones and zeroes a television program requires for storage. MPEG-4 does that job much more effectively than MPEG-2, but its decoders are more expensive and complicated. It's a business decision on whether to spend money on millions of decoders or additional satellites, but satellites are so very expensive that in most cases the move to the more expensive decoder is definitely the way to go.

Encryption [17]

There's an additional step that goes on between MPEG and coding for transmission. That step is designed to make it very difficult for unauthorized users to see our transmissions without paying for them. DISH isn't in the business of providing free TV! You may rightly object that encryption is something that happens only at the uplink center, and should therefore be included in *The Uplink Center* chapter. You're very correct, but you'll soon see that the rest of the content of that chapter is so completely unrelated to encryption (and MPEG encoding) that it's more appropriate to discuss it here. So, we'll breach the walls of the uplink center in this chapter on content acquisition just a little bit. One

Figure 20: The Three Levels of Protection Surrounding DISH Content.

caveat, though. Encryption technology and processes are closely held, so we'll only treat this subject at a very high level. Going into more detail would likely be a case of "if we told you we'd have to kill you!" One final thing: while we're treating encryption as a separate process from MPEG compression, those processes actually run concurrently and there are common scrambling algorithms built into the MPEG encoders at our uplink centers and complementary common descrambling algorithms built into every DISH receiver's MPEG decoder.

Just as DISH's sister company actually owns the uplink centers where our content is compressed into MPEG streams and sent up to satellites, we do not do the encryption of our content ourselves. Instead, a joint venture between EchoStar and the Swiss company Nagra—NagraStar—designs and implements encryption for us. They're the company that produces and encodes the smart cards that are required by every receiver DISH customers use.

As illustrated in Figure 20, there are three levels of protection that ensure only those customers entitled to receive specific channels are actually able to see them: subscriber management, subscriber authorization, and data scrambling. Access to each of these levels is strictly limited, with fewer and fewer people having access as you get closer to the content. Thousands of call center agents and other DISH employees have access to the subscriber management system, while only a very, very few trusted folks know the details of how our data encryption system is implemented.

The first level, subscriber management, is actually owned and managed by DISH in our DISH Promo tool updated by customer service agents at our call centers. That database contains, among other things, the packages to which every customer subscribes and their payment statuses—an example is shown in Figure 21. It also contains the serial number of the smart card(s) inside each customer's receiver(s).

DISH Promo passes this information to the subscriber authorization system at NagraStar.

Service Code Name	Service Type	#	Added
America's Top 200	Basic	1	2/8/13
DMA Green Bay, WI	Locals	1	2/8/13
Fox Sports North	RS Net	1	3/4/13
Big Ten	RS Net	1	3/4/13
Starz	Premium	1	3/4/13

Figure 21: Example DISH Promo Section Showing Authorized Packages.

The second level of encryption management, the subscriber authorization system, uses the DISH Promo information to determine which card should allow which channels. This notification is done through an Entitlement Management Message, or EMM. EMMs are sent to receivers through the same process used to distribute programming—to our satellites and back down. They go out regularly to every receiver in our network to keep them activated. They're also sent out to receivers that have had their service levels changed in DISH Promo (channels added, soft disconnects, etc.) as soon as a "hit" is sent by the customer service agent or a change is made by billing software. Since we cannot target a specific receiver through this signal path, *every* receiver sees *every* EMM, but the receivers ignore all but the ones coded for them. The EMM authorizations are then held in the receiver's smart card to enable them to decode control words—discussed in the next paragraph—for each authorized channel. The authorization messages are transmitted in encrypted form and decrypted by the receivers.

The second thing the subscriber authorization system does is to generate control words unique to *each channel*. These control words are the keys required to unlock channels. They change every 15-20 *seconds* and are delivered to the receivers via Entitlement Control Messages, or ECMs. Every receiver gets the same set of control words via the single ECM. The control words are encrypted in a manner similar to that of the EMMs. The ECMs are read by the smart card in the receiver and only those control words for channels authorized by the EMM can be decrypted. Since the EMMs and ECMs are transmitted via broadcast to the smart cards, they're easy for potential thieves to see. That's why it's important for us to be able to easily replace these cards in the event pirates were ever successful at breaking into our system.

As an implied but integral part of this subscriber authorization system is the set of tables that tells the receivers where to look for specific video channels and their associated audio streams. As you can imagine, creation and maintenance of these tables is a huge task. It's associated with the encryption process because, as you just learned, every video or audio channel has a stream of control words

that apply only to it. With what you know to this point, that means several hundred channels of programming, each with at least one and sometimes several accompanying audio tracks (second language tracks, tracks for the home and visiting teams for some games that share a video feed, etc.). Compound that with the fact that we've got seven satellites broadcasting these channels, with some of the channels passing through more than one satellite (for East Coast or West Coast viewing, for example). And that's just on a normal day. The tables have to be revised any time there's a problem such as a specific satellite transponder being out of service. So, there's a lot that goes on when you select a specific channel on your receiver: finding the right channel/satellite/audio stream/control word. But we'll discuss that at length in the *Inside Our Customer's Home: Receivers* chapter later.

The final level of encryption management is the actual scrambling of the ones and zeros in the MPEG stream. That's done with an industry-standard process—the DVB Common Scrambling Algorithm.[18]. For example, if the code to show a certain part of a picture in uncompressed format is 11111, it might come out of the scrambler as 00101, 10010, or even 1100101, where extra bits might be inserted on occasion just to make things more interesting for those who are trying to steal our services. Regardless of the exact encryption process being used for data encryption, it's different from the process used to encrypt the EMM and ECM transmissions, again, to make piracy more difficult.

It's obviously important that every one of the receivers knows the encryption process being used at the time so they can invert it and retrieve the original MPEG-coded content. But remember they'll also need access to the channel's control word being used at that moment and that only comes if their receiver has been authorized to decrypt the control word. The control words are the keys that allow the decryption process to operate successfully on the encrypted data stream.

Pay-per-view broadcasts are a little different. In addition to these encryption techniques, the smart card has a built-in credit limit based on business practices sent to it by DISH Promo and the subscriber authorization system. That way, the receiver doesn't have to ask "mother-may-I" every time a customer wants to purchase programming. It only checks in with mom, the DISH central computer, when the credit limit has been exceeded or when a customer wants to make a large purchase that may be affected by such items as the customer's credit score, credit limit, history of non-payments, etc. When necessary, that communication

passes over a low-tech modem, which is why you have to have your receiver connected with either a phone line or broadband if you want to order pay-per-view programming without calling an agent.

As with MPEG, there's obviously a lot more to encryption than we've discussed here but this should give you a pretty good overview of the general process. You need to understand some more concepts before we go any deeper, so we'll save that discussion for the *Inside Our Customer's Home: Receivers* chapter. However as a bottom line for this introduction to encryption, in contrast to MPEG much of this process is very closely held, as we do our dead level best to ensure that nefarious individuals aren't able to steal from our pockets.

Data Transport—Combining Multiple Programs on a Single Channel

So, we now have an MPEG-standard data stream that's been encrypted, but how do we efficiently send this information into the rest of the network? When you consider that a single passive show like Bloomberg only consumes as little as 500 thousand bits per second (kbps) in high-definition, a sports show like a basketball game uses as much as 12 million bits per second (Mbps) in HD, and a single DISH HD transponder can handle over 41 Mbps, it's obvious that in the interest of efficiency we have to somehow send more than one signal at a time to the transponders.

The way we combine these signals is to use an electronic device called a *multiplexer*. The multiplexer takes data streams from multiple sources, breaks the streams up in to discrete packets of information, and sends them to the transmission line packet by packet. Frame I of Figure 22 shows this graphically.

However, in a real multiplexer the data packets aren't colored but are instead identified with a small amount of data called a *header* that not only identifies which stream the packet belongs to but what number packet in that individual stream it is. (The header actually contains a lot more information, but that's beyond what you need to understand at this level.) As shown in Frame II, the header is combined with the program information, called the *payload*, to form the complete packet. Packets all must contain the same total information between the header and the payload.

In Frame I, there are a lot of green packets and very few blue packets. The number of packets represents the relative data rates of the different programs, so blue might be Bloomberg and green might represent ESPN. The multiplexer knows the average relative data rates of each of the programs and plucks off the

right number of packets from each channel so no one channel's packets are depleted too quickly or not quickly enough. The math for this fixed-data-rate situation is relatively straightforward. When a packet is selected by the multiplexer, it is sent down the transport stream. You can see in the figure that there are a lot more green boxes in the transport stream than blue ones, as it should be.

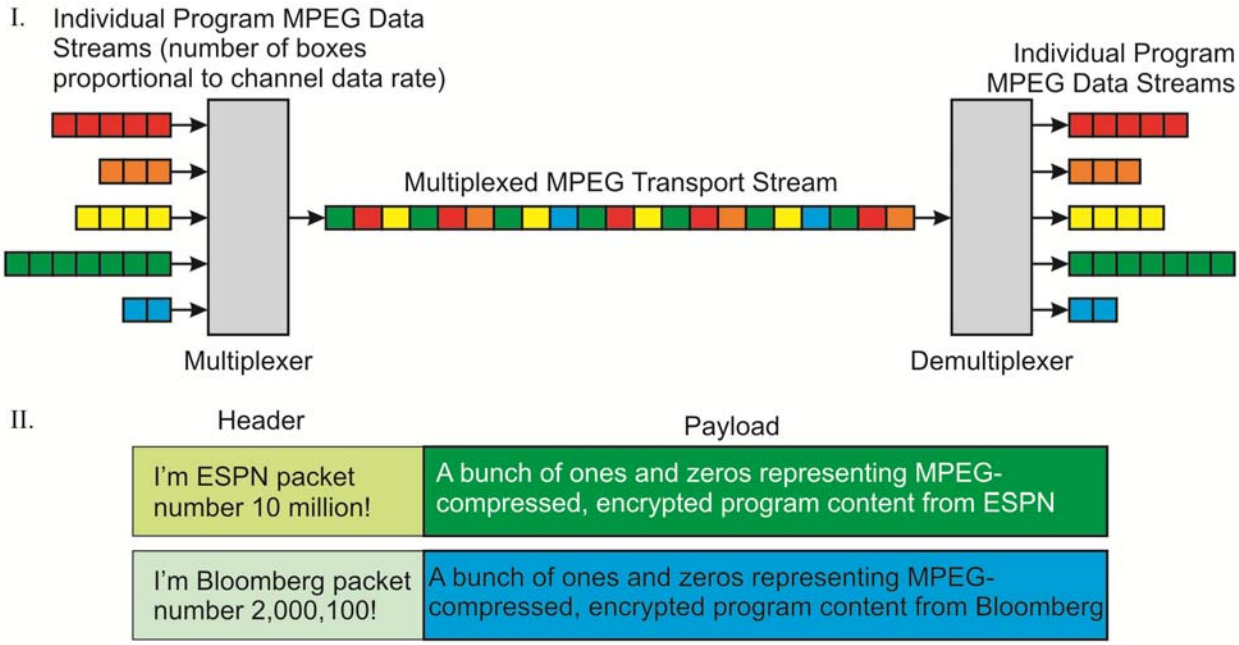

Figure 22: Multiplexing and Demultiplexing Program Data Streams.

We'll talk about receivers at length in the *Inside Our Customer's Home: Receivers* chapter, but a small part of the receiver is best discussed here. That part is the demultiplexer. Its function is to reverse what the multiplexer did. It reads the headers for each of the incoming packets, discards them, and routes the associated payloads to the appropriate outgoing data streams in the right order.

While that simple mux/demux (multiplex/demultiplex) system works well for incoming data streams with known, fixed bit rates, it doesn't meet our needs here at DISH. Again, we're very limited on the amount of information we can send up to and back down from our satellites so we want to be just as efficient as possible. Let's look at an example of how we can be even more efficient.

Let's say that we're watching that busy basketball game on ESPN. Suddenly, we go to a break in the action and the station shows a stationary graphic about the shooting percentages of both teams so far in the game. That content doesn't take up nearly as much storage space as the actual game does. Now, coincidentally,

over on Bloomberg, they've gone to a commercial break, and that commercial has a lot of very rapidly changing images. That content takes up a whole lot more storage space than Bloomberg usually does. If we were to keep static data rates for both channels, we'd be wasting some of the space we normally reserve for fast-moving ESPN and might not be able to transmit the commercial on Bloomberg because it's got too much content.

The solution we use is variable data rates. When ESPN is moving quickly, we devote a lot of our storage space to it. When it slows down, we take some away and use it elsewhere. The problem is that our simple multiplexer doesn't know how to deal with these variable speed data streams. To the rescue comes the *statistical multiplexer*, or stat-mux. It's much more complicated inside than the simpler mux/demux system. It has storage bins for each of the channels and can sense when they're starting to fill up. It's smart enough to know to send more packets from channels whose bins are filling rapidly. Its biggest limitation is that it can only send so many packets out to the transport stream in a set amount of time—a limitation on the overall bit rate—so we have to be smart about how we allocate the data rates for the incoming channels. We don't want to overwhelm the stat-mux. The way we do this is to always ensure we send *less* than the maximum number of packets the stat-mux can handle.

The associated smart demultiplexer on the other end still expects its data to arrive at a constant rate, though, and that constant rate is the maximum amount the stat-mux can ever deliver. That means that when we're sending less than the maximum amount, the smart stat-mux knows to send packets with headers that say "Hey, I'm empty. Throw me away," and to generate associated payloads that are the right size but don't really contain any meaningful information. With statistical multiplexing, we're able to squeeze about eight HD broadcasts onto a single bitstream, which is pretty efficient.

Multiplexing is obviously a lot more difficult in practice than this basic theory may lead you to believe, but at this level if you can now visualize all of our programs being dissected on one end, sent up to the satellites, and then reassembled by the receiver on the other end, you've got the concepts we want you to have. We'll go into more detail on this later in the *Inside Our Customer's Home: Receivers* chapter after you've been exposed to a few more advanced concepts.

Chapter Summary

A common theme throughout this document is that DISH is in business to make money. We do this by trying to be as efficient and effective as we can whether we're talking about reducing call times in our call centers, reducing the number of trucks we roll for in-home service, or making sure that the receivers we get back from our customers are in top-notch shape before we send them back out to the field.

Just like our other business units such as call centers and receiver remanufacturing, the way we deal with our signals needs to be as efficient as possible as well. We're limited by the amount of data we can send up into space, and getting more capacity the brute-force way would involve launching more billion-dollar satellites. In this chapter we've looked at a number of ways we're working to squeeze as much performance out of those satellites as we possibly can by making our signals carry just as much information in as little time as possible.

We've talked about the three ways we receive content from providers: via satellite downlink, over-the-air, and via fiber. We then discussed how we take that content and compress it via MPEG so that it doesn't take up nearly so much space. We talked about the need to make sure only people who are paying for our service receive it and how we encrypt our signal toward that end. Finally, we showed that by combining more than one channel onto a single transponder signal, we could more effectively use those signals to get more information to our customers. Now that you've seen the technical details, you might better appreciate how the haiku at the beginning of this chapter (and in the footer on every page) sums up these concepts quite concisely.

We've now come to the end of our first major chapter and have followed the signal from our program providers to the point where it's ready to exit the uplink center and be beamed into space. We're about to embark on perhaps the longest link in the path our signal takes on its way from provider to customer: the trip into space. Join us in the next chapter to see how that's done.

2. The Uplink Center

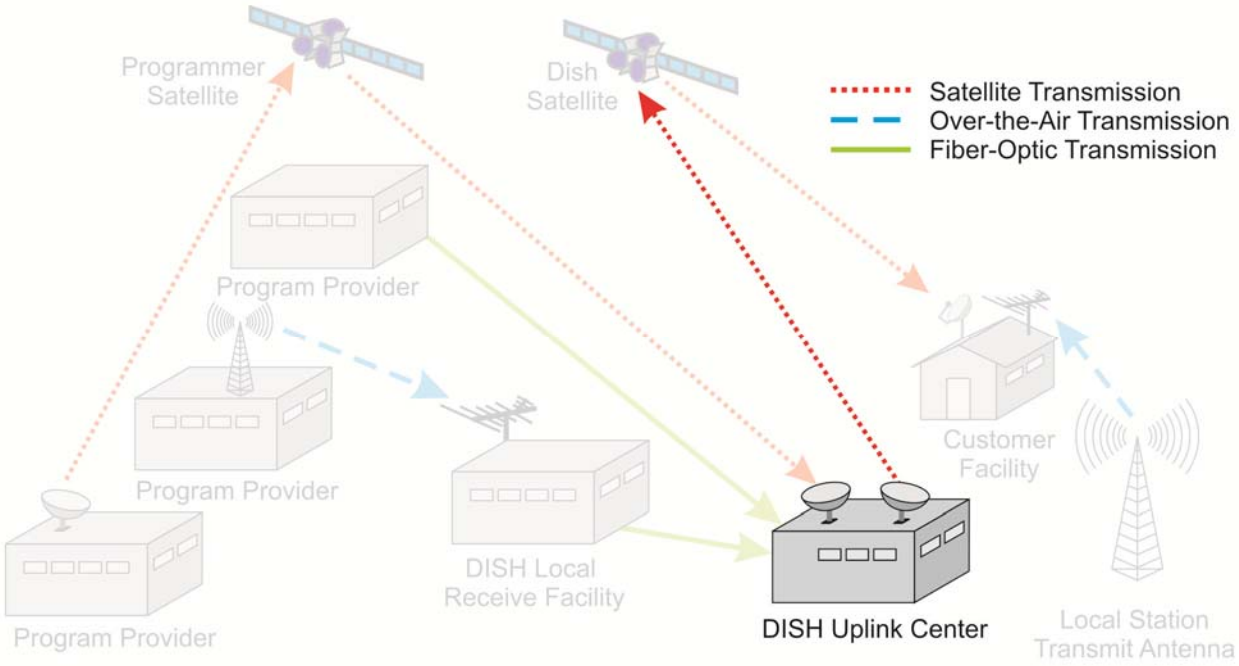

Figure 23: The Signal Flow Roadmap—Uplink.

Large antennas fire
Focused beams to space bearing
Coded messages.

The Basics

(Continued from p. 7) Just as a caveat up front, this chapter will be somewhat lengthy and more technically oriented than some of the others. The reason is that a lot of groundwork will be laid here we'll use later. Many of the problems we'll encounter with our signal on the way up to the satellite will be very similar to what happens on the way back down. Much of the knowledge about encoding, waves, and antennas we gain here will translate directly when we talk about downlink later. Consider this to be fair warning that this chapter is pretty important.

It also may seem strange to call this *The Uplink Center* chapter, since we've already discussed a lot of what goes on in the uplink center—MPEG decoding and coding, encryption, and the like. However, everything we've discussed previously was just to prepare our signal to be transmitted into space—to be *uplinked*. This chapter will discuss the how and why of the things we do to actually transform that series of ones and zeros in our computers into something that has a chance of making it in a recognizable form to satellites orbiting the Earth.

Electromagnetic Waves. Inside the uplink center, the signal is transmitted over wires. Outside, we use radio waves. Now, radio is just a form of electromagnetic radiation—exactly the same thing as the light our eyes see except that it has a different wavelength. I'm sure you're familiar with waves in the form of ripples on a pond or a puddle—they are a series of peaks and valleys in the water's surface that travel outwards from a disturbance, say, a rock you threw in the pond. As shown in Figure 24, the distance between any two adjacent peaks (or adjacent valleys) is called the wavelength.

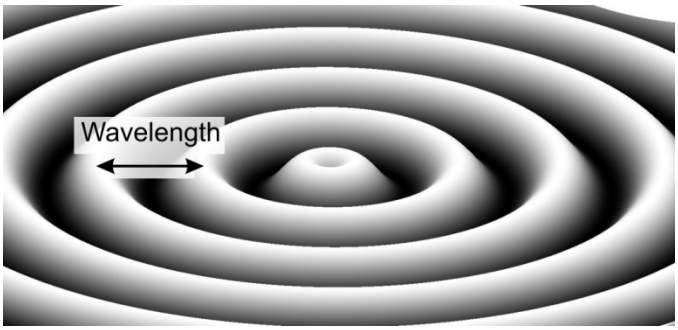

Figure 24: Ripples on a Pond.

The wavelength of visible light is very, *very* small—over 200 times smaller than the width of a human hair. The wavelength of radio can vary between about a tenth of an inch and many miles long. That's quite the range! Figure 25 shows where radio and visible light fit into the electromagnetic spectrum. Because the wavelengths of gamma rays and radio waves are so very different, we had to squeeze the wavelength scale on the right side of this diagram and stretch out the left side so all the regions would show.

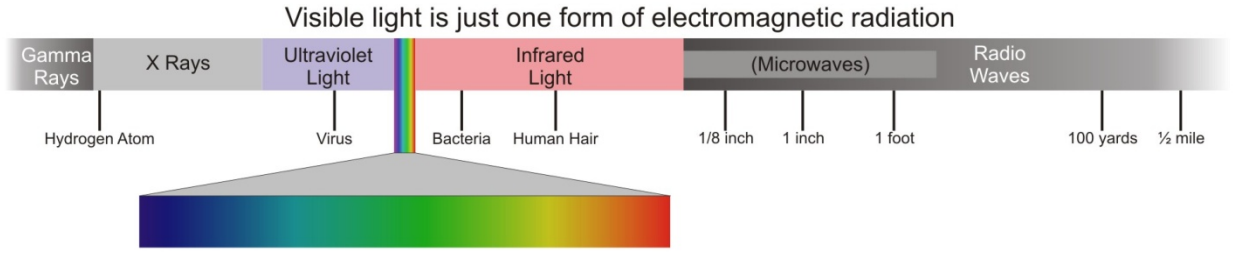

Figure 25: Visible Light and the Electromagnetic Spectrum.

The radio waves DISH uses are a little longer than half an inch. We didn't just arbitrarily choose to use this wavelength, but it's a really good length for several reasons, some physical and some political. Let's talk about the physical constraints first.

We think of the Earth's atmosphere as being transparent. It has to be: we see the Sun and the stars, don't we? Well, it's definitely transparent to visible light, but that's because our eyes have evolved to take advantage of a small window of transparency to a specific section of the electromagnetic spectrum. The atmosphere isn't nearly so transparent to every wavelength, which is a good thing because some forms of radiation like X-rays and gamma rays can be quite harmful to life!

The only other place the atmosphere is pretty transparent is in the portion of the spectrum we call radio waves. In fact, it's even more transparent there than at visible frequencies; visible light is very heavily blocked by clouds but most radio waves pass through clouds almost unimpeded. So the physics of the atmosphere limits our choice of how to get our signal from the ground into space because we've got to use a portion of the spectrum that's transparent just about all of the time, day or night, rain or shine. Thus, we use radio waves.

The second major constraint on our choice of wavelength is dominated by politics. Lots of people want to use the radio wave portion of the spectrum because it *is* so transparent. They have all sorts of uses for it, from radio broadcasts (obviously!) to television to police radar to satellite communication—far too many uses to list them all here. There are many, many people and groups that want to use radio frequencies. If all of them were to proceed willy-nilly, there would be a huge interference problem. To help sort out competing claims for this limited resource, government agencies around the world have agreed to license the use of the radio portion of the electromagnetic spectrum. In the United States, the applicable agency is the Federal Communications Commission, and they're the ones who have licensed DISH to use wavelength bands of 0.663 inches to 0.682 inches. Now, that's not much of a range, especially when you consider that the FCC licenses the wavelength range from four *hundredths* of an inch to a little over 31 *miles*. But as we mentioned, a very large number of users need access to that transparent portion of the spectrum, so it's got to be divided up pretty finely. Thus, we use this portion of the spectrum for two reasons: the atmosphere is relatively transparent there and we hold FCC license to it. Now that

we know what wavelengths we can use and why, let's look at some of the things those wavelengths force us to do with our equipment.

Antenna Physics. Some of the most obvious things you'll notice about an uplink center are the numerous, very large dish-shaped antennas there. Those dishes are actually made in the shape of a parabola, because the special properties of that shape allow the radio waves we generate to come out in a tightly focused beam. Having a tight focus is a good thing because we want as much of the energy in that beam as possible to hit our satellites, which are tens of thousands of miles away. In fact, they are so far away that hitting them with our beam is like shooting at a quarter over 30 miles away.

The reason the parabolic antennas we use to communicate with our satellites are so large is related to the wavelength we've been assigned and the very small apparent size of our satellites as viewed from the uplink center. The tightness of focus of the radio beam depends on two factors: the wavelength we're using and the size of the dish. To make a beam tighter, we can use a smaller wavelength, use a larger dish, or both. Since our wavelength has already been assigned to us, the only thing we have available to change is our dish size. The concept is shown in Figure 26—notice the decreasing angle between the aiming direction and the beam spread direction as the dishes get bigger. The need for such a tightly focused beam is the reason why our dishes are so large.

Up until now, we've discussed the spectrum in terms of wavelength because it's easy to build a picture of the distance between the crests of a wave on a pond in your mind. The way the FCC actually divvies up the spectrum is by frequency.

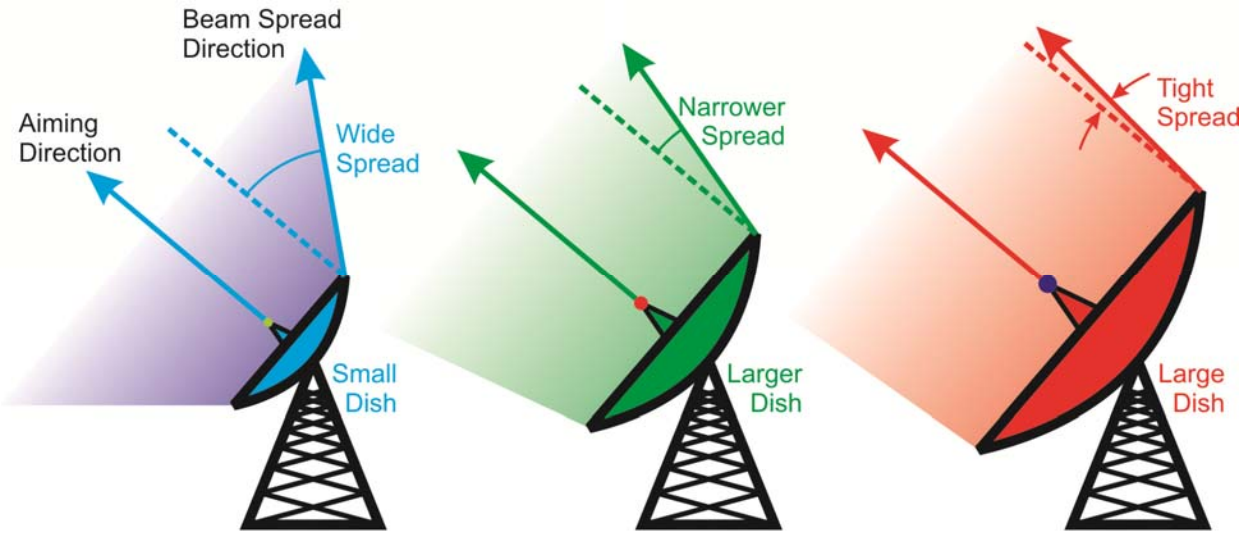

Figure 26: A Bigger Dish Means a Tighter Beam.

Frequency is just the number of times a wave wiggles up and down in a second, and that value is closely related to the wavelength since electromagnetic waves *all* travel at the speed of light. If wavelength goes up, frequency goes down (and vice versa). Frequency and wavelength are just two sides of the same coin—they describe a wave from different points of view.

The frequency band of the spectrum corresponding to the 0.663-inch to 0.682-inch wavelengths we discussed above runs from 17.8 to 17.3 gigahertz (GHz), where a hertz is one complete wiggle of the wave and the prefix *giga-* means one billion. That means the waves DISH uses to transmit our signal wiggle up and down about 17 billion times a second. The exact numbers aren't important at this level, but you may hear them thrown around the office so we thought we'd let you know what they mean.

Bandwidth and Signal Coding. The reason we're switching terminology from wavelength to frequency here is that we need to discuss the bread and butter of our business: *bandwidth*. The term bandwidth actually has meaning in two different contexts. The one that makes us money is its meaning related to the rate at which we can transfer information from our uplink center to our customers' televisions, computers, and wireless devices. That bandwidth is known as **data bandwidth**, and is measured in bits per second, where a bit is one of the *one* or *zero* symbols that stores information in computers. Anything we can do to make the data bandwidth higher helps to make us money, because it means we can give our customers higher quality picture and sound, more channels, or both.

There are a number of ways we can increase our data bandwidth. You've already seen two of them when we discussed MPEG compression and MPEG transport earlier. The point of compression was to minimize the amount of ones and zeros DISH has to send while still giving the customer outstanding picture and sound quality.

Another way we can increase our bandwidth is to transmit the data more rapidly. Take a look at Figure 27. It shows three ways of moving those ones and zeros around using a flashlight. At the left side using the red line, we only transmit one bit (an individual one or a zero) at a time. A zero is coded as the flashlight being off and a one is when the flashlight is on. In order for the people transmitting and receiving the signals to stay in synch, one and only one state (on or off) of the flashlight is allowed each second. Were it not so, how would we know whether the light remaining on, say, for four seconds was a single long "one" or a "one-

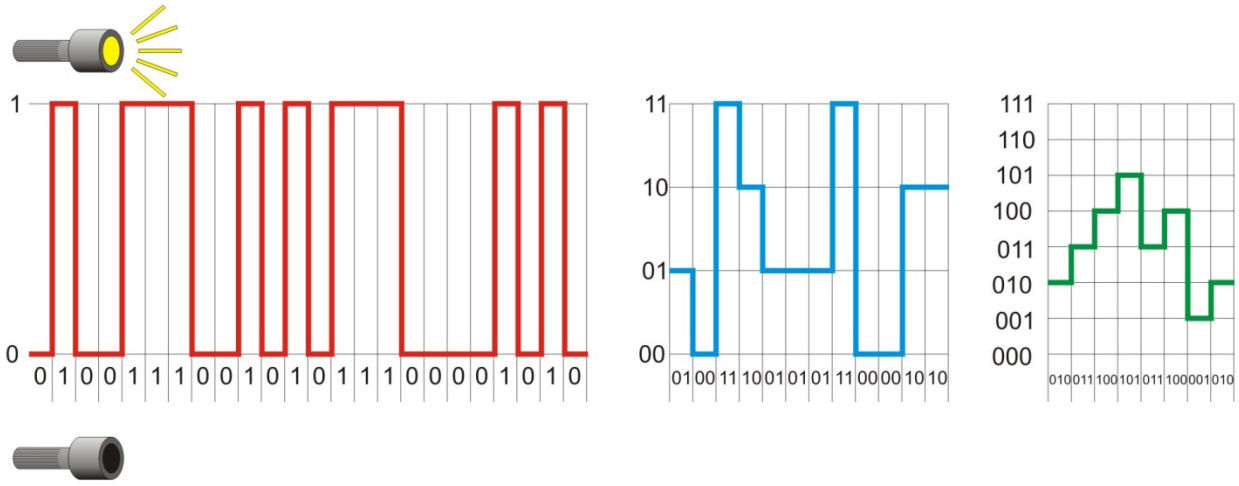

Figure 27: Flashlight Codes.

one-one-one"? Below the signals, you can see the ones and zeros the on-off signals represent: 0 1 0 0 1 1 1 0 0 1 0 1 0 1 1 1 0 0 0 0 1 0 1 0.

Now, what if we had a flashlight that could not only be on or off, but could be set to one of four settings: off, low, medium, and high? We could now transmit four states instead of just two. Conveniently, there are only four combinations of ones and zeros taken two at a time: 00, 01, 10, and 11. The middle, blue section of the figure shows the same set of ones and zeros as we listed at the end of the previous paragraph, but taken two at a time. Notice that to transmit the sequence of ones and zeros using this four-intensity-level scheme with our flashlight took half as much time, but the sequence is the same: 01 00 11 10 01 01 01 11 00 00 10 10.

Finally, let's imagine a flashlight that had eight settings: off, very low, low, medium low, medium, medium high, high, and very high. Again, there are conveniently only eight ways to group ones and zeros taken three at a time: 000, 001, 010, 011, 100, 101, 110, and 111. The right, green section of the figure shows the same sequence of ones and zeros grouped this way, and it takes only a third as much time to transmit as the original red signal. This method of data transmission is definitely more time efficient than the red binary code!

While on-and-off coding schemes might work in some situations, it's not appropriate for our goal of transmitting data to satellites. Here's why. Let's imagine that we start to move away from the flashlight. At first it's still easy to tell the difference between the different levels, even for the eight-level code. As we get further and further, the different levels start to blend until it's hard just to tell when the flashlight is even on, much less what level of *on* it's trying to display.

This is an example of why we don't use a code for sending signals to our satellites that depends on the strength of the signal.

Varying the strength of the transmitted signal is called amplitude modulation, and what we've shown is somewhat similar to what's used for digital AM radio. We *shift* the amplitude to convey the data. What we use at DISH to convey our data, however, is shifts in the *phase* of a signal, a method known as **phase shift keying**. Phase isn't as easy to draw or explain as amplitude, so we won't go into it in *The Basics*. We just want to expose you to the phrase and to say that it's similar to the flashlight codes you now understand, but just varies a different property of waves. (We'll discuss phase shift keying extensively in this chapter's *Technical Discussion*.) Just as there were numerous ways we could shift amplitude with the various levels of flashlight intensity, there are several flavors of phase shift keying as well. *2PSK*, phase-shift keying of order two, is the simple two-level system like the red curve in Figure 27. 2PSK is rarely used anymore since it's pretty slow compared to **QPSK** (standing for *quadrature phase shift keying*, a system using four levels similar to the blue curve in the figure), and even less efficient than **8PSK** (*phase shift keying of order eight*) which uses—you guessed it—eight levels like the green curve above.

By using 8PSK, we're able to triple the data bandwidth, making us lots of money. We also still use QPSK to transmit some of our standard definition channels. Those channels don't have as much data content, so we can get away with less compression. Also, some of our older standard definition receivers only know how to decode QPSK.

So we've now discussed three ways to increase our data bandwidth: MPEG compression to reduce the amount of data that needs to be transmitted, combining more than one TV channel on each signal we send up (multiplexing), and QPSK/8PSK to transmit that data more rapidly. The final way we'll discuss here to maximize our data bandwidth is to use our allocated spectrum just as effectively as possible. We do that using two techniques: spectral bandwidth minimization and polarization.

We send up 32 different signals to our satellites, and those signals carry every single one of the hundreds of channels we deliver to our customers. Why 32? Math, of course. Different kinds of signals require different minimum and maximum frequencies. For example, your ear can only hear sounds from low pitches at 20 hertz (twenty wiggles of the sound waves in the air every second) to very high pitches at about 20 thousand hertz (the older we are, the lower this

second number becomes as our high pitch hearing deteriorates). The **spectral bandwidth**—measured in hertz and describing the difference between the maximum and minimum possible frequencies in the signal—of your ear's response is therefore about 19,980 hertz. To accurately reproduce music digitally, CD makers typically require a spectral bandwidth of about 40 *thousand* hertz—a number twice as high as our ears require because of rules related to sampling a wave and converting it to digital format that we won't cover here. A high-definition television signal including its audio tracks requires a little over 4 *million* hertz of bandwidth, and as you learned in the *Prepping the Signal* chapter, we cram about eight HD broadcasts onto one MPEG bitstream. That comes out to about 30 million hertz per signal.

Figure 28: Cramming Transponder Channels into Our Bandwidth Allocation.

As shown in Figure 28, If we take that 30 million hertz and try to see how many of them would fit into our FCC frequency bandwidth allocation of 500 million hertz (17.8 -17.3 gigahertz), it conveniently comes out to right about 16, half of the 32 signals we send up. We'll discuss that factor of two in a minute, so be patient. In fact, if we had 34 signals (half of which would be 17), there wouldn't be enough room in our FCC allocation for them to all fit without overlapping, and they not allowed to overlap at all because they would interfere with each other. By smartly using our *spectral* bandwidth by squeezing in as many right-sized transponder bands as we can, we're helping to maximize our overall *data* bandwidth in our given FCC allocation because more bands equals more data delivery capacity.

Finally, let's get to that factor of two we talked about earlier. You may have heard that DISH uses right- and left-handed circularly polarized waves for our signals. Polarization is a fancy way of describing the direction that waves wiggle up and down (or right and left, or…). We'll go into a lot more detail in this chapter's *Technical Discussion*, but for now that's all you really need to know. It's possible to use two mutually-exclusive polarization states to send signals in the same frequency band without them interfering with each other. That's where the factor of two comes in because right- and left-hand circular polarizations are a pair of those mutually-exclusive polarization states. Instead of just transmitting the 16 bands that fit into our allocation, we can double that number through the magic of two polarizations, as shown in Figure 29.

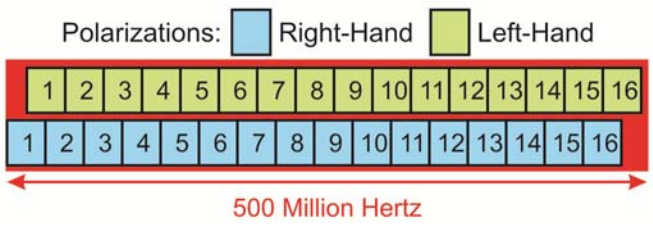

Figure 29: Differently Polarized Signals Don't Interfere.

We've covered a lot in this section of *The Basics*, but it's information that's fundamental to understanding how DISH operates. You should now know that

- Radio waves are just another form of light, but with much longer wavelengths
- DISH is authorized to use a small band of radio frequencies by the FCC, and that the FCC stringently regulates the radio spectrum because Earth's atmosphere is so transparent to that frequency band
- We use parabolic dish antenna because they allow us to send out a focused beam of radio waves, and the bigger the dish (at a given wavelength), the tighter the focus
- We use very large dishes because our satellites are very far away and we need to put just as much of our beam on them as we can
- We try to maximize our data bandwidth—the rate at which we can transfer information—because it allows us to send more channels and higher quality video and audio, both of which help us make more money as a company
- We maximize our data bandwidth by
    - MPEG Compression (as discussed in the *Prepping the Signal* chapter)
    - Putting more than one channel on a signal stream (also as discussed in the *Prepping the Signal* chapter)
    - Using QPSK and 8PSK coding to send more than one bit at a time
    - Effectively using our spectral bandwidth by squeezing in just as many signals as possible within our FCC allocation
    - Using right- and left-handed circular polarization to double the number of signals we can transmit within our allocation.

That wraps up this overview of *The Uplink Center*. If you're only reading *The Basics*, you can continue your study on page 97.

Technical Discussion

As you just read in *The Basics*, this chapter will take the signal we acquired, encoded, and encrypted and will beam it up to our satellites. Along the way, there are a number of things that we do to the signal and a number of pieces of

hardware we use to direct that signal we need to understand in more detail. We'll first take a look at waves and the electromagnetic spectrum so we can see why we use the frequency bands we do. We'll then look at some of the physical constraints such as diffraction that drive our transmission antennas to be the size and power they are. Finally, we'll examine the process we apply to our signals before we send them into space: phase shift keying.

The Electromagnetic Spectrum

One of the biggest constraints on the DISH business is spectrum. Not only do we have to purchase the right to use it, we have to make sure we get a part that's actually going to be of use to us. This section will go into quite some detail about why DISH uses the portion of the spectrum we do.

We've talked in the introduction about radio waves, but radio is just a portion of the electromagnetic spectrum. The light waves your eyes can see are also electromagnetic waves. As shown in Figure 30, they are just a small part of the electromagnetic spectrum and just differ from radio in where they lie in that spectrum. All of the rest of the physics of light is exactly the same as radio waves. The fading of the spectrum at the right and left ends of the figure is intentional. It's designed to show that electromagnetic radiation can exist with higher and lower frequencies, but they're just not shown here.

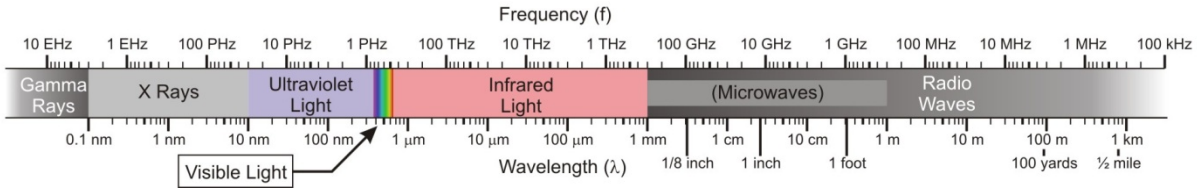

Figure 30: The Electromagnetic Spectrum.

Don't worry too much about the unit abbreviations in the figure—the ones we'll be using are the fairly common ones: *mm*, *cm*, and *m* for millimeter, centimeter, and meter, respectively, for length and *kHz*, *MHz*, and *GHz* for kilohertz, megahertz, and gigahertz, respectively, for frequency. A kilohertz describes a wave that wiggles up and down 1,000 times per second. Megahertz and gigahertz waves wiggle up and down a million and a billion times per second, respectively.

Note that in Figure 30, both wavelength and frequency are shown. Depending on what they majored in, technical people may think of electromagnetic waves in terms of frequency, wavelength, or other less-common things like energy or wave number. All those terms refer to exactly the same concept, just using different words.

Waves. A brief review of the terms that describe a wave is shown in Figure 31. In Frame I, imagine you're looking at a snapshot in time of a cross-section of the ripples on a pond. Notice that the horizontal axis is labeled distance, which is what you'd expect when you're looking at a snapshot and measuring horizontal and vertical positions with a ruler. The light blue curve is in the shape of a sine wave because we've chosen the origin of the axes to be where the curve crosses the distance axis on its way up (the curve would be a cosine wave if the axes were shifted so that the curve crossed the displacement axis at its maximum positive displacement). We could have chosen to place the axes at any point, but the one we chose just makes this description a little easier.

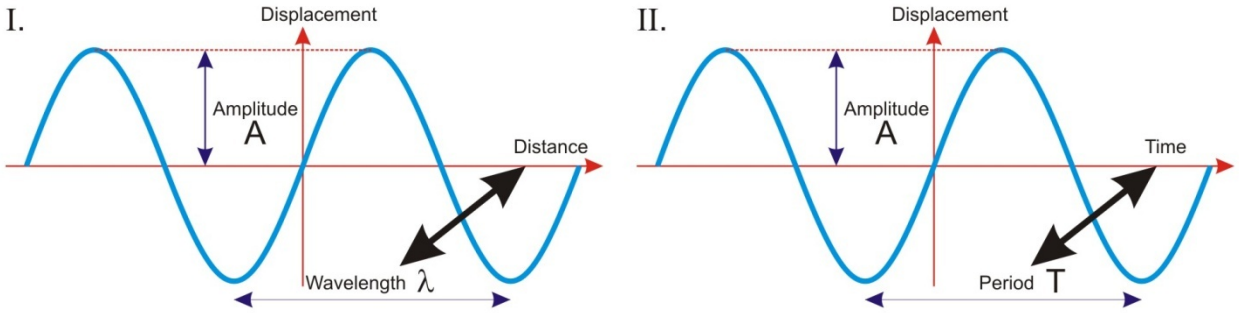

Figure 31: Parts of a Wave.

There are two terms describing waves that we can now define from this figure: **amplitude**, ***A***, and **wavelength**, λ (the Greek lower case letter *lambda*). Amplitude is simply the maximum height of the ripple from the normal, undisturbed level of the pond. Notice that the amplitude in the upward direction is equal to the amplitude in the downward direction. Wavelength is equally easy to see. It's defined as the distance between any two similar points on the sine curve. We've chosen to show it from trough to trough on this curve, but it's the same distance regardless of which two similar points we choose.

Frame II looks remarkably similar to Frame I, but it's actually a completely different setup. Instead of a single point in time, imagine you've decided to look at a single point in space on the surface of the pond and, using a stopwatch, noted the height of the water at that point as time passed. Notice that the horizontal axis in this frame is now labeled time, again, what you'd expect if your measurements were taken with a stopwatch. As you'd also expect, the amplitude of the wave doesn't change from the one we noted in Frame I just because you're watching it change with time. However, another feature of the wave is now apparent: the **period**, ***T***. The period is defined as the time it takes for the wave to oscillate from one crest to another (or any other set of similar points on the

wave). We've tried to emphasize the difference between these two figures with the heavy arrows showing that if the horizontal axis is distance you'll measure a wavelength but if it's time you'll measure a period.

But in Figure 30, we showed wavelength and frequency, not period. What's up with that? If you think about it, the period is just the number of seconds it takes for the wave to go through a complete cycle, measured in seconds per cycle. If we turn that upside down, we get the number of cycles per second, which is the definition of the hertz (Hz). Thus, **frequency**, ***f***, is related to the period through the equation $f = 1/T$. If you were standing there with your stopwatch, frequency would describe how many times the wave wiggled up and down each second. Notice that in Figure 30 the waves wiggle up and down millions (mega), billions (giga), trillions (tera), and more times per second.

One last tidbit about wave definitions and we'll move on. Frames I and II of Figure 31 are actually tied together by the speed of the wave. Much like you can find the speed of a car by measuring how long it takes you to drive a certain distance (speed equals distance per time), the wavelength and frequency of all forms of electromagnetic radiation are related to the speed of light, *c*, through the equation $c = \lambda f$, which in words just says speed equals distance per time. That fixed relationship (*c* is a constant, which means that changing either λ or f requires a corresponding change in the other variable to make the equation still balance) means that we can simultaneously show both wavelength and frequency scales on any figure dealing with waves.

The Electromagnetic Spectrum and the Atmosphere. Now that we've reminded ourselves of what waves are and how to describe them technically, let's move on to a concrete example. Since DISH uses radio waves to send our signals to and from satellites, we'll look at the radio wave portion of the spectrum in a bit more detail. It's been broken down into a number of bands and we're sure you've heard of at least a few of them.

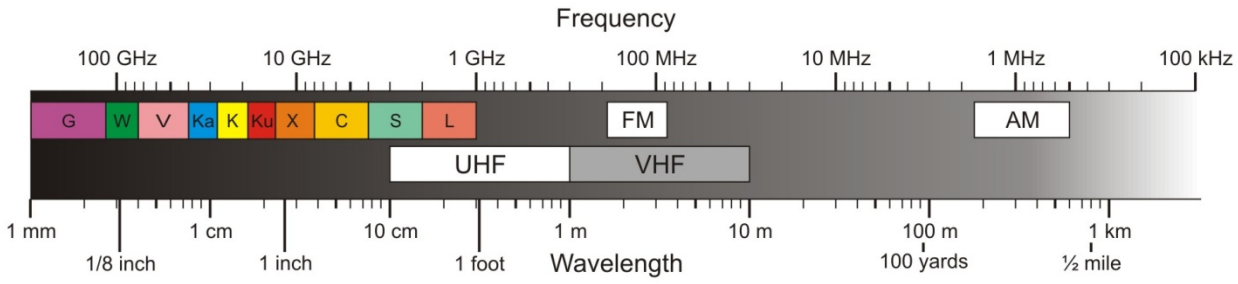

Figure 32: Selected Bands in the Radio Wave Portion of the Electromagnetic Spectrum.

Figure 32 displays some of the Institute of Electrical and Electronics Engineers (IEEE) radio frequency bands,[19] as well as the more familiar AM, FM, VHF, and UHF bands (not IEEE-defined) for reference. The letters designating the bands are in a somewhat random order originally designed to enhance secrecy during World War II, so don't expect their order to make much sense. When DISH first started as a company, we sold those large backyard dishes you may remember, and those dishes were designed to receive C-band signals. When we made the transition to selling signals over the smaller dishes we now use, we also transitioned to the shorter wavelengths and higher frequencies of the Ku band (pronounced *kay-you* band). C-band dishes range from 5' to 12' in diameter.[20] In contrast, DISH Ku-band antennas are about 2'-3' in diameter. We made that change for a number of reasons we'll explain shortly, and we'll also discuss the reasons for the different sizes of dish for the different bands. Example C- and Ku-band antennas are shown in Figure 33. Make sure you note where the Ku band is in the following figures. It's color-coded red or red-and-white hashed wherever it's shown.

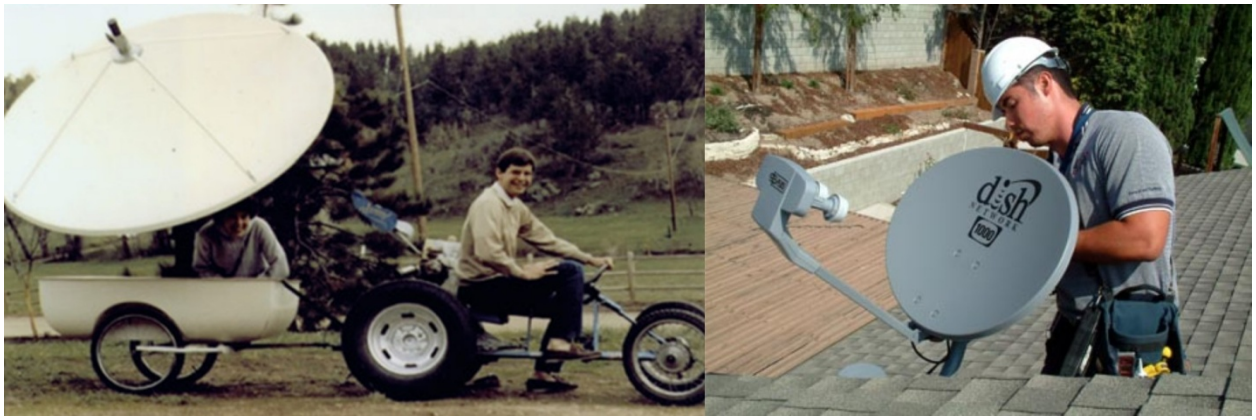

Figure 33: C-band and Ku band DISH antennas.

Let's put these bands back into the electromagnetic spectrum chart and try to understand why we use them instead of, say, infrared, x-rays, or other portions of the spectrum to communicate with our satellites.

Figure 34 shows the radio bands and the rest of the electromagnetic spectrum at the top, but it also starts to relate these spectral definitions to a very important physical phenomenon in the real world: how transparent is the sky? You may think the sky is pretty transparent, since we can see the Sun most days and the stars most nights. However, our eyes have evolved to take advantage of a small window of transparency. As you can see from the blue-shaded region of the figure, the vast majority of the electromagnetic spectrum is almost completely blocked by the atmosphere, which is a good thing for us since higher-frequency

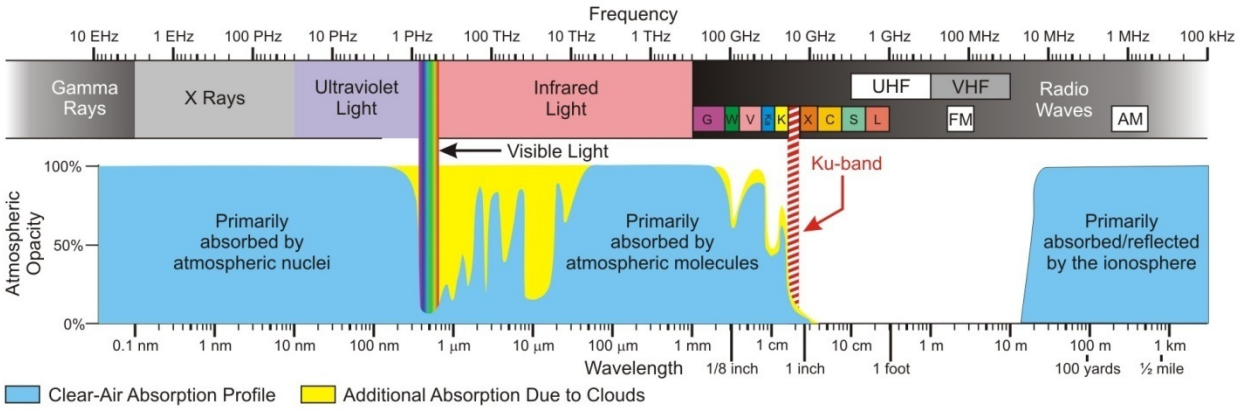

Figure 34: Atmospheric Transparency.[21]

waves like X-rays and gamma rays would quickly be fatal if they made it to the Earth's surface.

The lower portion of Figure 34 shows to what extent the various regions of the electromagnetic spectrum are absorbed by the Earth's atmosphere. Starting at the left side, the light blue region extending all the way up to 100% shows that gamma rays, X-rays, and most ultraviolet (UV) light are almost completely absorbed by the atmosphere. The blue region falls off very rapidly as it approaches frequencies corresponding to the violet end of the visible spectrum. This deep hole in the absorption profile, where the atmosphere only absorbs a small fraction of radiation, is called the *optical window*. It's here our eyes evolved to operate. As we move further to the right and frequencies continue to decrease, absorption shoots up again, though not to 100%. There are several other windows in the infrared (IR) region where we have learned to exploit the spectrum for such things as medium-wave infrared seekers on military targeting systems, for example.

The far infrared and microwave end of the radio region is again almost completely blocked by the atmosphere, but suddenly, after a few ups and downs, absorption drops off to almost zero for a very large region, known as the *radio window*. (Referring back to Figure 30, p. 48, you can see that the microwave region is just the high-frequency section of the radio spectrum.) This transparency makes the radio window a very useful commodity. Notice that the Ku band we use at DISH is just at the left edge of this window.

To the right of the radio window, absorption goes back up sharply again for long-wave radio frequencies, but this increase is a little deceptive. You can see the AM radio band lies in this range. Why would we ever design our radios to use a band that's almost completely blocked? The answer is a little surprising. The *entire*

atmosphere isn't opaque to most of these long-wave radio frequencies. The regions of high absorption for the IR and microwave regions and at frequencies higher than UV are all due to absorption by atoms and molecules throughout the atmosphere. The further the waves travel in the atmosphere, the more they're absorbed.

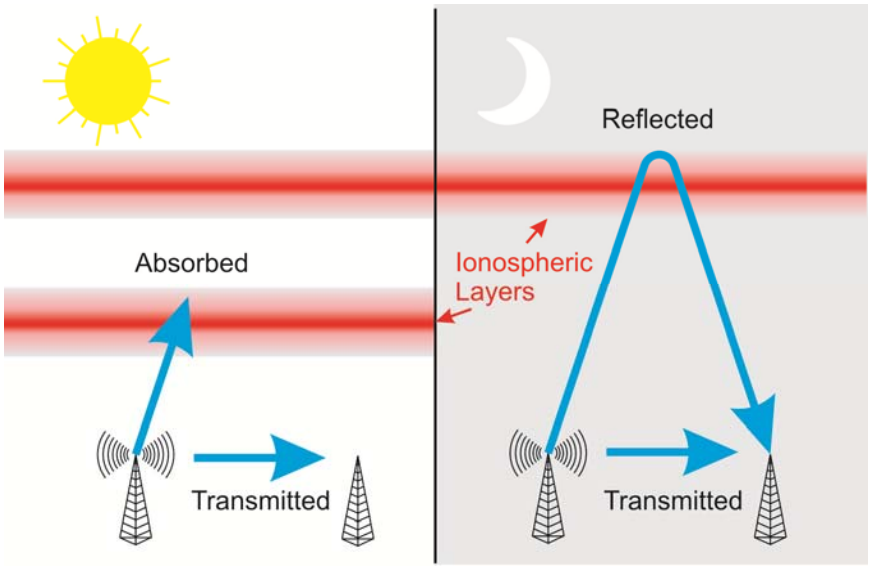

Figure 35: Ionospheric Absorption and Reflection of Radio Waves.

In contrast, the long-radio-wave absorption region is mostly due to absorption or reflection by atoms (high-temperature electrons and ions, actually) between about 50 and 400 miles above the Earth in the ionosphere. As shown in Figure 35, below that level, the atmosphere is relatively transparent to these waves. The lower, more absorbent level of the ionosphere all but disappears after sunset, allowing radio waves to hit the upper, more reflective layers. We can hear far away AM stations on the radio at night because the waves are actually reflecting off the ionosphere.

Let's look at absorption in the radio spectrum near the Ku band a little closer. Figure 36 shows a much more detailed absorption curve for the G through X bands. Actually, it shows two curves. While the curve in Figure 34 was for radiation coming straight down (or up) from space, these curves are for radiation traveling horizontally. Since the atmosphere is thicker at lower altitudes, you'd expect more absorption down low. And that's what you see, since the upper curve (more absorption) is for sea level and the lower one for about 30,000 feet up. Notice that the scale of the vertical axis is logarithmic, that is, a unit of measurement on the scale is ten times larger than the one immediately below it. (Prove this to yourself by noticing that there is the same amount of distance between 0.001 and 0.01 as there is between 0.01 and 0.1.) The reason we use log plots like this is to display information that has a huge scale of change. If we used a linear plot, the detail of the curve near the bottom would be completely washed

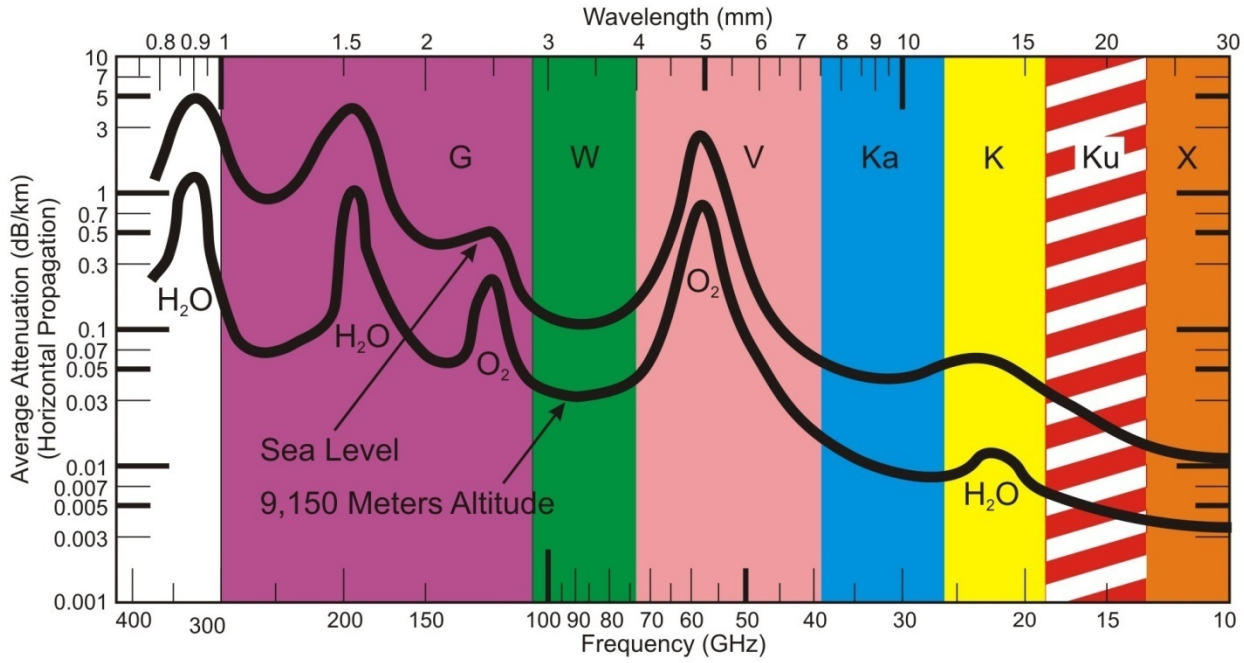

Figure 36: Atmospheric Absorption Due to Oxygen and Water Vapor.[22]

out—it would actually look like it was zero from the Ka band onward to the right. You may have noticed that the horizontal scale on all of the spectrum figures we've shown you earlier had a similar squeezing of the larger numbers—for the same reason.

So why are there so many sharp peaks of absorption, and why are they labeled with the symbols for water and molecular oxygen, H_2O and O_2? To understand this, let's look at a very simple but very powerful model of molecules. Frame I of Figure 37 shows two masses attached together by a coiled spring. The masses represent atoms and the spring is the chemical bond between them. From the physics class you probably had in high school or college, you may remember that a single mass on a not-too-stiff spring will oscillate up and down for quite some time before the oscillations are damped out. The two atoms we've modeled as masses act in a very similar manner.

Now, systems like these have *resonant frequencies*, that is, frequencies at which they really prefer to oscillate. Almost every one of you has probably experienced a resonance like this when your car's tire has been out of balance. Out of balance means one part of the tire is heavier than the rest. In Frame II, you can see that there's another two-masses-on-a-spring problem where one of the masses is the wheel/tire, the spring is the shock absorber, and the other mass (not shown) is the car itself. When your tire is out of balance, it gives the shock absorber an up

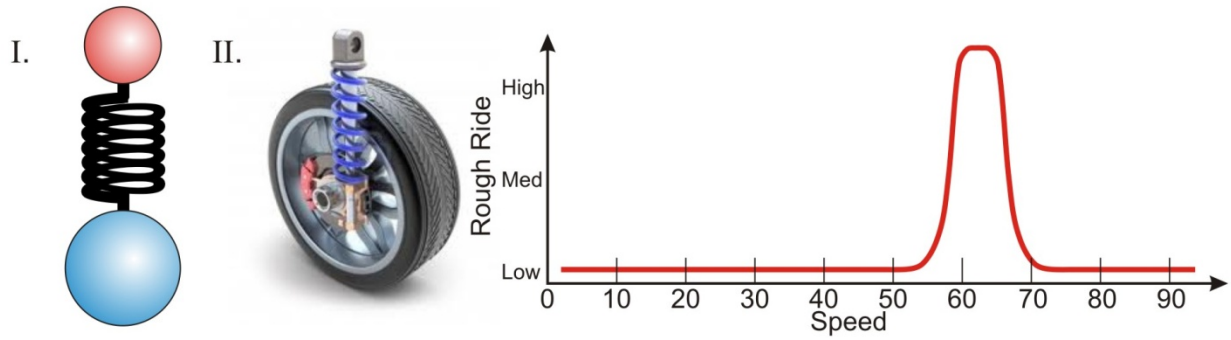

Figure 37: Molecular Model of Absorption Using Resonating Springs and Masses.[23]

force when the heavy part of the tire is up and a down force when the heavy part is down. You don't really notice the force much, though, until you get to highway speeds, when it really starts to affect the ride significantly. Were you to exceed the speed limit by a bit (*don't try this at home—professional drivers on a closed course and all those sorts of warnings*), you'd notice that the bumpiness dropped back down to where it was not very noticeable again. You can imagine how this resonance feels if you imagine yourself at the speed shown along the bottom of the plot. Starting at zero, the "rough ride" index, shown on the vertical axis, is pretty low. As you move further right on the plot, it stays very low until you reach a little faster than 50. There, the ride gets really rough until a little past 70, where it smoothes out again.

Near a system's resonant frequency the system is able to extract a huge amount of the energy that's driving it and convert it into mechanical energy. This works both for out of balance tires and for atoms being stimulated by electromagnetic radiation. While the spring mechanism in the tire/car system is pretty simple, the "springs" holding atoms together are very complex and have many, many ways to resonate. They can twist, they can move in and out, and they are hooked together in more than one way. Instead of one resonance, there are a large number of them for every molecule.

Return to Figure 36 and look at the strong V-band resonance for O_2. The reason for this feature is that one of the springs holding the two atoms in the oxygen molecule together has a resonant frequency of about 50 or 60 GHz. When radio waves (an electro-magnetic driving force) interact with the electrically-charged atoms (remember the electrons and protons inside the atom are charged particles), it causes the molecule to start to vibrate. A plot of the shape of that resonance is very similar to the one we saw for the shock absorber system—frequencies significantly above and below the resonance frequency don't drive

vibration in the atom much at all, while frequencies near the resonance are heavily absorbed, going into making the molecules vibrate. Notice that some resonances are stronger than others, as well, with the upward hump showing the K-band water resonance being one such weaker example in this figure.

Now that you understand resonances and absorption, notice how the bands in Figure 36 were seemingly chosen to avoid absorption resonances—do you think someone knew a little physics when they drew these up? The G, V, and K bands surround regions where a lot of absorption occurs, while the bands in between them are relatively absorption-free. *Caveat emptor* when someone approaches you with a great deal on ground-to-space spectrum in the V-band!

You may now heave a sigh of relief because we've come to the end of a rather detailed section on electromagnetic absorption. But why do we care? Take another look at Figure 34 (p. 52) and remember that we use the Ku band. Notice that we're right at the high-frequency edge of the radio window. Obviously, we need our signals to make it out to space, so we had to choose a region that wasn't eaten up with absorption. But why this band? The answer to that question partially lies in the subject of antennas. We'll discuss that in just a bit, but first we need to understand another constraint on the choice of frequency—this one not rooted in physics at all.

Political Considerations. We've mentioned numerous times that the electromagnetic spectrum is a valuable commodity, and like all such things many people want access to it. To resolve all the conflicting claims on spectrum, the world's nations have set up an organization, the International Telecommunications Union[24] (ITU), to allocate access to the radio spectrum. Established in 1865 as the International Telegraph Union, it's expanded far beyond that original scope and its work now covers the entire information and communications technology sector, including setting standards for everything from the Internet to 3-D television. The ITU is run by the United Nations and is responsible not only for spectrum allocation but for licensing the orbits of satellites—more about that in the *DISH Satellites* chapter. Working with 193 member nations and over 700 non-national organizations, the ITU divides up the spectrum to ensure equitable, interference-free spectrum use across the globe.

Within the United States, the Federal Communications Commission[25] (FCC) performs the spectrum allocation role, among other functions. To demonstrate how busy they are with this job, examine the small extract of the FCC Frequency Allocation Chart shown in Figure 38. Frequencies in GHz are shown across the top

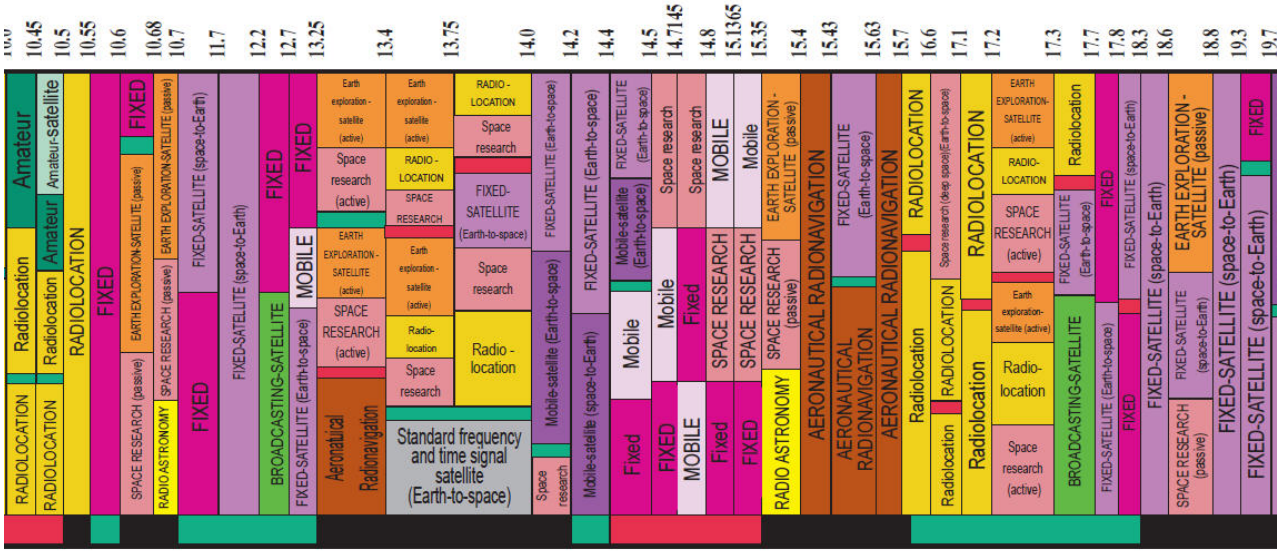

Figure 38: The 10-20 GHz Portion of the FCC Frequency Allocation Chart.[26]

of the chart. Each one of the colored bars shows a different use of a spectral band. This sample shows the entire Ku band (12-18 GHz), with some of the X band to the left and some of the K band to the right.

The colored bars at the bottom of the figure indicate whether the section of spectrum is government exclusive (red), government/non-government shared (black), or non-government exclusive (green). DISH uses the 12.2-12.7 GHz band to downlink our signals and the 17.3-17.8 GHz band for uplink. The entire FCC chart covers the frequency range from 6 kHz (50 km or *31 miles* wavelength) to 300 GHz (1 mm or 1/32" wavelength), a huge range which runs from about an inch off the right of the plot in Figure 34 (p. 52) to the left side of the G band—virtually all of the transparent radio window plus frequencies that will propagate below the ionosphere. Were the entire chart reproduced at the scale shown above, it would be almost *16 feet long*! It's just as cluttered for that whole range, too. And remember, that chart is just for the United States. Other countries have similarly busy charts. We told you a lot of people wanted to use the radio spectrum!

So, it should be apparent by now that there are two major constraints on the choice of frequencies DISH uses to communicate with our satellites, one based in physics and one based in politics. That just about does it for this chapter's discussion of the electromagnetic spectrum. There is one more small electromagnetic topic we'll hit later in the chapter—polarization—but we need to understand a few more concepts before it will make sense

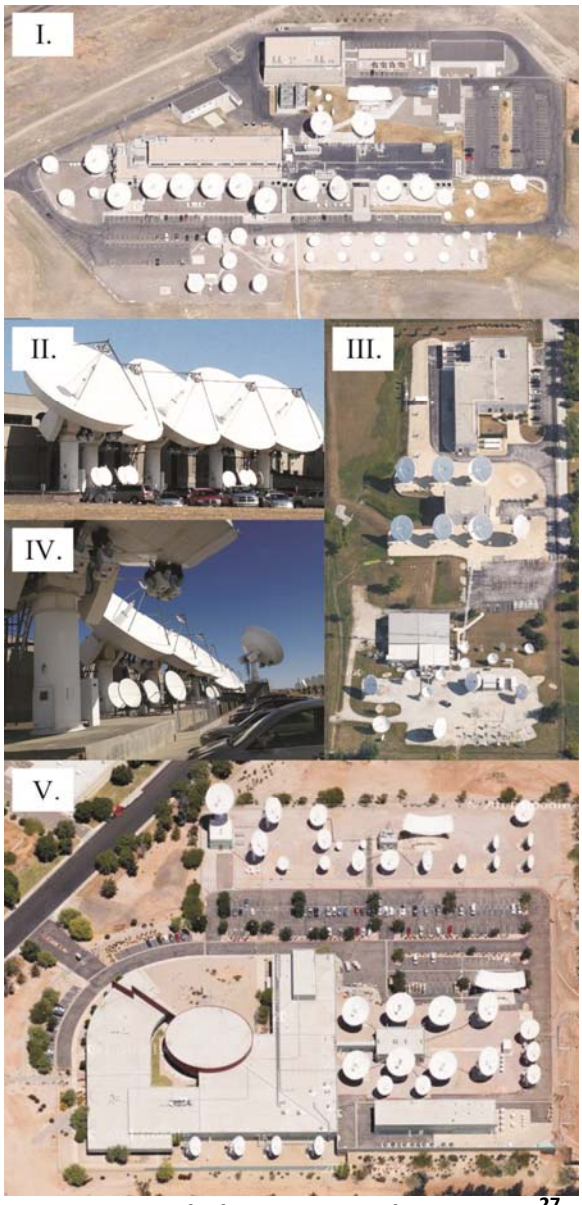

Figure 39: Uplink Centers and Antennas.[27]

Antennas and Diffraction

Let's move on to our next topic that relates to the physics of the uplink center. Perhaps the most distinctive characteristic of these centers is the variety of very large dish antennas you'll find near them. These antennas are how we actually get the DISH signals up to our satellites. Did you ever wonder why they're so big? Or why they're shaped the way they are? This section will explain the answer to those questions.

Satellite dishes, whether huge like at the DISH uplink centers (shown in Figure 39) or small like on your roof, are all made in the shape of a parabola. Well, pretty close to parabolas, but we won't get to that topic until we discuss beam shaping in the *DISH Satellites* chapter. As shown in Figure 3, p 12, our two main uplink centers, run by our sister company, EchoStar, are located in Cheyenne, Wyoming (Frames I, II, and IV) and Gilbert, Arizona (Frame V). An example of one of the smaller, secondary uplink centers at Monee, Illinois is shown in Frame III.

Figure 40 shows a typical dish with the parabolic shape and its focus labeled. From your high school geometry class, you may remember that a parabola is a curve that moves up as the square of the distance it moves across ($y = x^2$, in mathematical terms). We'll discuss a little more about that shape and how it relates to the focus in a bit, but first we need to talk a little more about waves. Instead of electromagnetic theory,

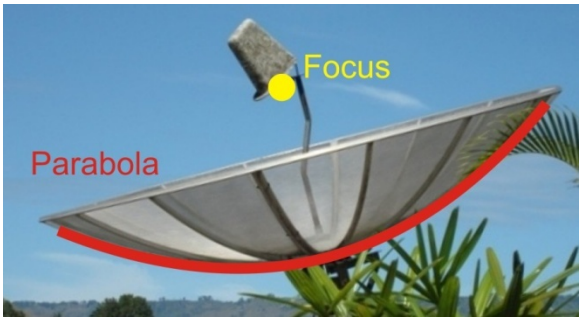

Figure 40: A Parabolic Antenna Dish.[28]

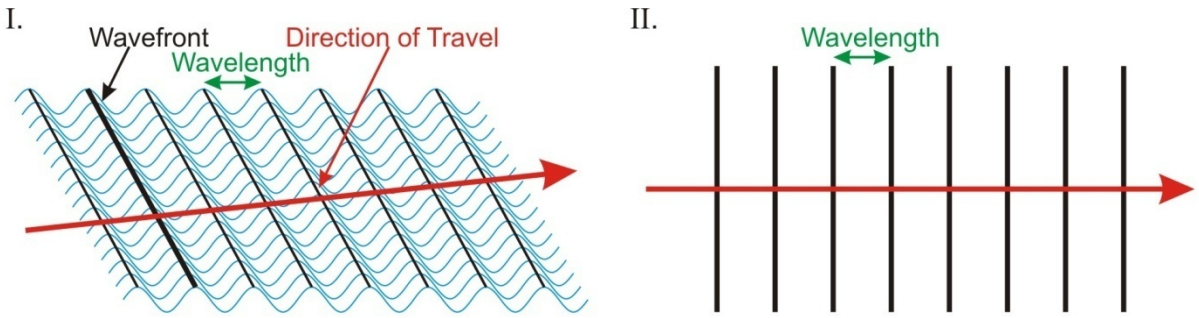

Figure 41: Waves and Wavefronts.

though, this time it will be just geometry.

Wavefronts. In order to show what effect using a parabolic dish has on focusing our signal beam, we'll be discussing a little about how waves travel. We could draw all the peaks and valleys of the waves, as is shown by the blue curves in Frame I of Figure 41. But that would take a lot of time and would confuse what we're trying to show in most cases. Instead, if you look at the solid black lines in that frame, you can see that they're drawn connecting the peaks of the waves. Every line of peaks has a black line on it. These lines are called **wavefronts**. The distance between the black lines is, of course, the wavelength. There's also a red line drawn perpendicular to the wavefronts that indicates the direction of travel of the wave. The direction of travel is always perpendicular to the wavefronts. So, instead of drawing the waves, from now on we'll just draw the wavefronts, and in most drawings will also indicate the direction of travel of the waves, as is shown in Frame II.

Now, the wavefronts don't always have to be in straight lines. Take a look at Frame I of Figure 42. It shows the wave pattern you'd expect if you repeatedly and regularly tapped your finger on the same spot on the surface of a pond.

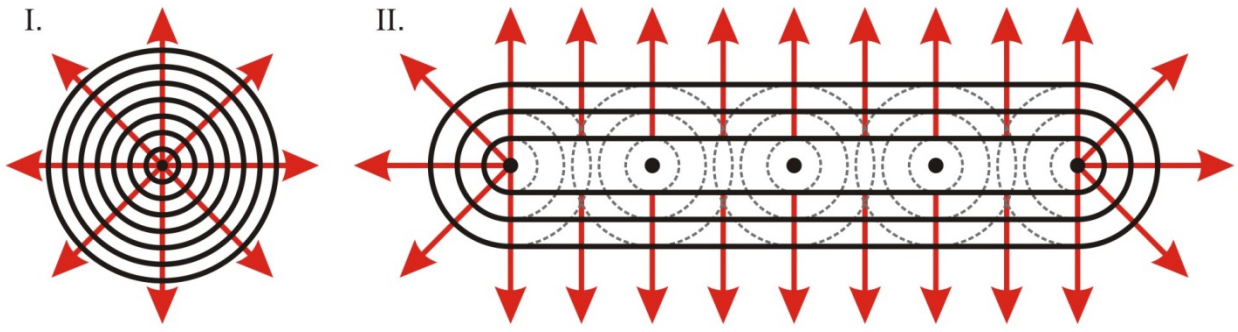

Figure 42: Circular Wavefronts.

Notice that to simulate any wave, we just have to take the wavefront we know exists at any time, break it down into a bunch of tapping points, and then link up the tops and bottoms of the resulting circles to create the next set of wavefronts. This simple geometric way of constructing waves results in a remarkably exact representation of how waves move in the real world.

Compare it to Figure 24 (p. 40) and you'll see what we mean. Circular waves would travel out from where you were tapping. Notice that the direction-of-travel lines are still perpendicular to the wavefronts, even though the wavefronts aren't all parallel this time.

In Frame II, we're showing what would happen if you repeatedly dipped a long, thin object, say, the edge of a piece of plywood, into the pond. For clarity, we've only indicated five points from where the circular waves move out from this tapping, but in reality there are an infinite number of them. Notice that we can draw a straight line of wavefronts that link up the very tops and very bottoms of each of the dashed circles. If we had drawn more tapping points, the tops and bottoms of their dashed circles would be in the same lines. The wavefronts behave like you'd expect from having played in water: lines of waves move away from the plywood upward and downward in the drawing, and these lines are joined by curved wavefronts on either end.

Reflection. As an example of wavefronts, let's look at how an electromagnetic wave is reflected from a surface. In Figure 43, the red wavefronts and red direction of travel arrow show a wave coming into a surface. The little circles on the black surface represent electrons in the atoms that make up the surface. Obviously, not all of them are shown! The ones that are highlighted in red correspond to electrons that coincide with the locations of wavefronts from the incoming wave. Remember that these are electromagnetic waves, so their wiggling electric fields cause the electrons in the surface to wiggle at the same frequency. As the electrons wiggle, they generate their own outward-traveling waves, as shown by the blue concentric circles. If we connect the farthest points on each of these circular wavefronts from each electron, it's easy to see that they all line up in straight lines, representing an outbound wave. The direction of outbound travel is indicated by the blue arrow. Notice that this very simple model accurately reproduces a fact that we all learned in school about reflection, that is, the inbound angle is exactly equal to the outbound angle.

Now reflection from a plane is pretty easy to understand. What we'll move on to, though, is just a little more complicated but it uses the same principles. We'll see

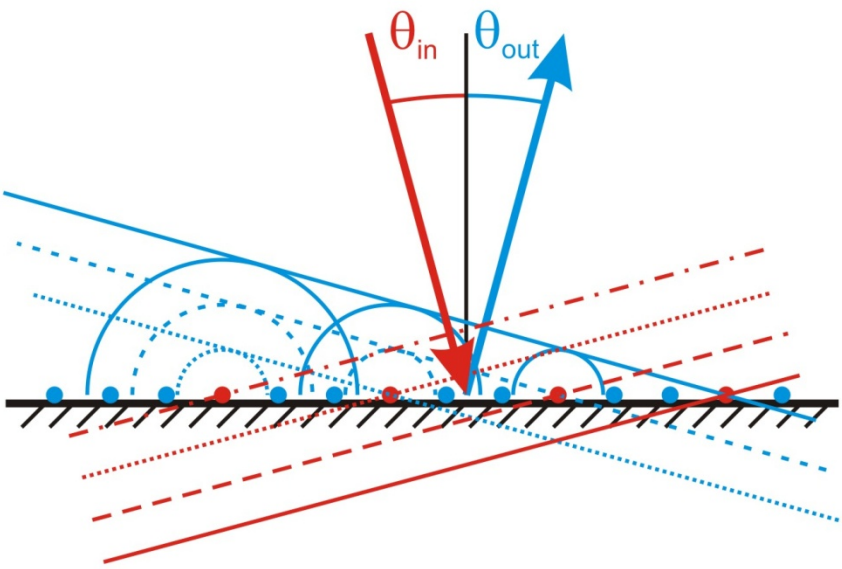

Figure 43: Reflection from a Flat Surface—Electron Oscillation Model.

how the shape of a parabolic dish ends up producing a beam that we can use to communicate with our satellites. In Figure 40 (p. 58) we presented the concept of the parabola and the focus. The whole point of the dish is to take the circular waves (spherical, if viewed in three dimensions) coming from the focus and turn them into lines of wavefronts (plane waves, in 3-D). The parabola is exactly the shape that can make this happen.

Take a look at Figure 44. Yes, it's very busy, but we'll go over it piece by piece. It shows the focus as a black dot near the center of the figure and the parabola as a blue curve. There are a set of heavy, dashed circular wavefronts moving down and away from the focus. These represent our signal coming out of the feed horn. When the wavefronts impact the parabola, they are reflected just as they were in the flat surface example. For convenience in making measurements later, we've only indicated points in this figure on the parabola where the wave from the focus impacts at exactly a multiple of a wavelength. Remember, there are many, many more points that could be treated in exactly the same way.

Let's look at the wave reflected from the green point on the left side of the parabola. The incoming wave from the focus elicits circular wavefronts from the green point, as shown by the thin green dashed lines. They move away from the green point back in the general direction of the focus. All of the other colored points similarly generate reflected, circular wavefronts, with the ones from the left half of the parabola being indicated with dashed lines and the ones from the right half being dotted.

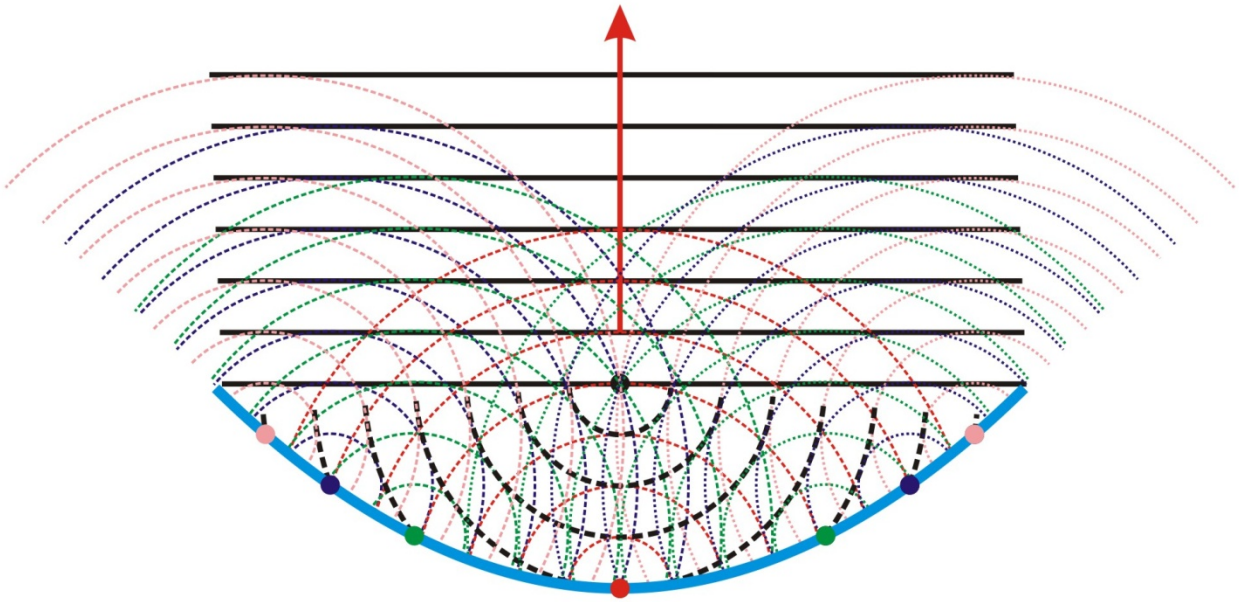

Figure 44: Reflection from a Parabolic Dish.

Now look at the collective effect of all these wavefronts. Starting at the top edges of the parabola, the thick, solid black lines join the highest points on each of the colored circles. In fact, if you count all the wavefronts, you'll discover that there are exactly the same number of wavelengths between the focus and the black lines. For example, following the wave through the green dot, there are five black wavefronts and three green wavefronts, for a total of eight, to the lowest black solid line. Moving through the pink dot, there are seven black wavefronts and one pink wavefront, again, for a total of eight, to the same line. Also notice that we could have drawn heavy black wavefront lines inside the parabola (measure down one wavelength below the lowest black line and you'll see where the line could have been drawn). We didn't draw these lines because the diagram was busy enough already.

So, just using this simple geometric method of evaluating reflection, we can see that *the function of the parabolic dish is to take the circular (spherical) waves coming from the focal point and convert them into a beam of linear (plane) wavefronts*.

Diffraction. Let's move beyond reflection to another property of waves. The basic plane surface reflection example in Figure 43 (p. 61) shows us that the model works well because it agrees with what we have personally observed. But waves are tricky things. They don't always act like many people would think, based on the intuition they get from playing in the bathtub. Let's look at a

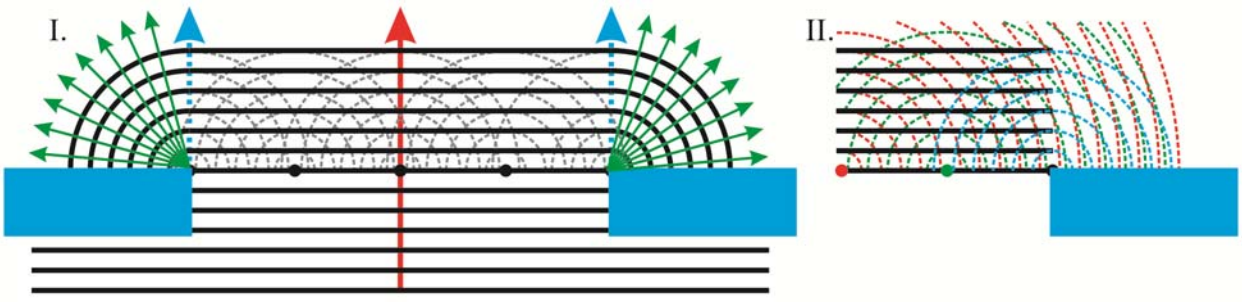

Figure 45: Diffraction.

specific example: a line of waves encountering an obstacle with a gap in it as is shown in Frame I of Figure 45. The red line shows a line of wavefronts headed upward until they encounter the blue obstacle. Many people would expect that the wave would be blocked by the obstacle (it is) and then pass straight through, staying between the dotted blue arrows in the figure (it doesn't). The reason it doesn't can be seen pretty clearly from our discussion that every wavefront can be modeled as a set of many, many points where progressive sets of circular waves come from. By connecting these wavefronts, a new, similar wave can be constructed. Notice how the part of the figure from the five black dots upward looks exactly like the top half of Frame II in Figure 42 (p. 59). Instead of the wave being completely shadowed by the obstacle, *some of it bends around the corner*! Very non-intuitive, isn't it? This phenomenon is called **diffraction**, and it is a huge limit to how well we can focus a beam. The green arrows show that the direction of travel of these diffracted waves is no longer directly up the page.

The reason we don't observe diffraction on a daily basis is that in order for it to be really apparent, the size of the gap in the obstacle has to be on the order of the size of the wavelength. Since light, the waves we're most used to seeing (though we don't normally recognize them as waves), has a wavelength that's about 200 times smaller than a human hair, we don't normally have the opportunity to see it passing through a gap of that size. However, the Net is a great place and we can find all sorts of examples of light diffraction. Case in point: Figure 46. Frame I shows the light pattern of a laser after it's passed through a very narrow slit. The very bright spot in the center of the image is the part of the light that passes straight through—essentially the part between the dotted blue lines in Figure 45, but spread out a bit. The progressively dimmer bright spots are the most apparent result of diffraction.

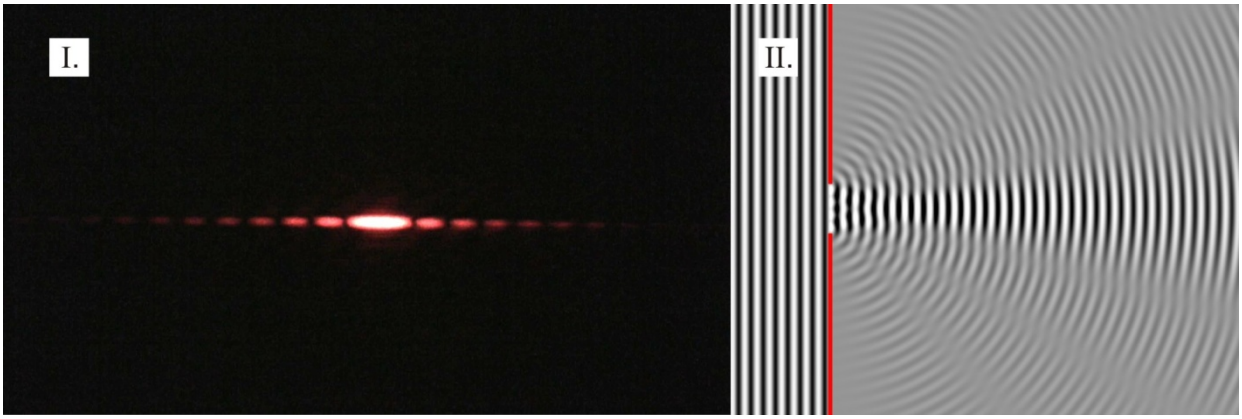

Figure 46: Single Slit Diffraction Examples.[29]

So, why are there alternating brighter spots separated by dark regions? Take a look at Frame II of Figure 45. It shows the wavefront details at the right side of the obstacle from Frame I. The wavefronts that pass directly through the gap can be connected in a simple manner by heavy solid black lines. However, to the right of those black lines the way to connect the wavefronts isn't nearly as obvious to the casual observer. It can be determined using some trigonometry, but that's beyond the scope of this text. If you're interested, just look up *diffraction equation* on the Net and you'll be inundated with sites hoping to explain it to you. The end result, however, are the bright and dark spots on the screen.

Frame II of Figure 46 shows an overhead view of this situation. Note the areas where the wavefronts are pretty dim—those correspond to the dark spots. There's a no-kidding great video of a ripple tank generating a diffraction pattern

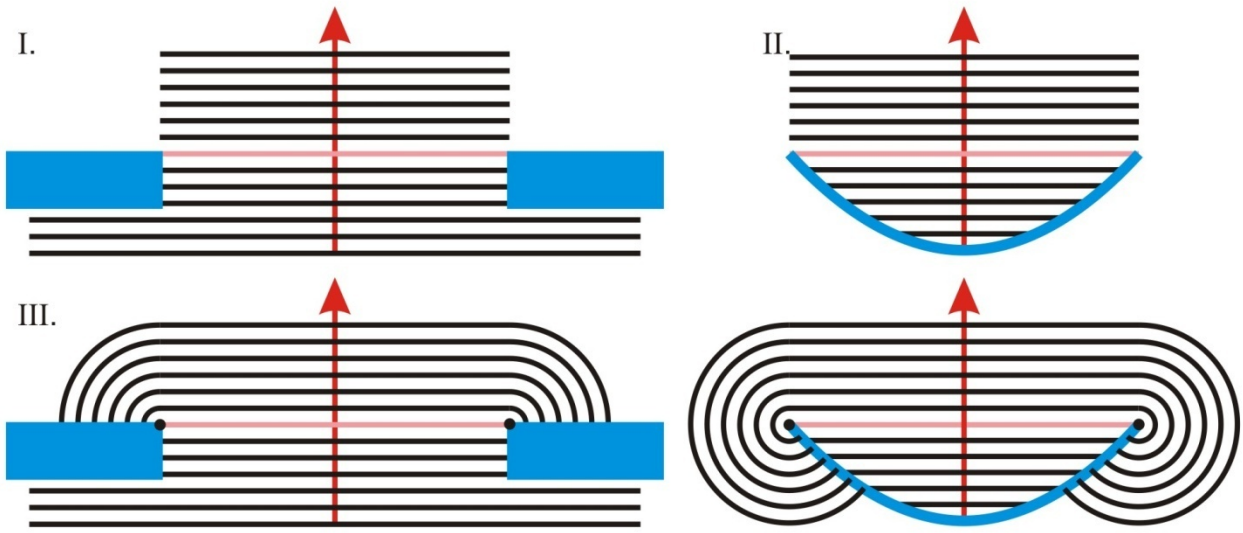

Figure 47: Parabolic Dishes = Diffracting Gaps.

like that in Frame II at the MIT Department of Physics' website.[30] It's well worth the time to view.

So what does all of this diffraction stuff have to do with our parabolic dish? Hopefully Figure 47 will answer that question. Notice the first wavefront that matters to diffraction is the pink one in Frame I, because that's the one where the circular wavefronts first start to spread out from the corners of the gap. There's *absolutely no difference* between that wavefront and the first parallel wavefront emerging from the dish in Frame II. Since Figure 45 conclusively showed that the wavefront in Frame I diffracts, the wavefront in Frame II must as well. Both diffractions are shown in Frame III. Notice that some waves even end up traveling *backwards* around the parabola! We'll see shortly that they'll end up contributing to a set of bright and dark spots *behind* the dish.

Beamwidth and Gain. So now that we understand that the somewhat esoteric concept of diffraction affects the beams that come out of our transmit dishes at the uplink center, let's see how that relates to the real world. The technical name for the bright spots in the Frame I of Figure 46 is a **sidelobe**, while the bright spot at the center is called the **main lobe**. We've got two goals for the main lobe: we want just as much of the wave's power in it as possible, since the power going into the sidelobes is just wasted—it won't make it to the satellite since it's not traveling in the right direction. Secondly, we want to adjust the width of the main lobe as appropriate for the real-world task. In the case of our uplink antennas, we want the main lobe to be just as narrow as we can get it because the satellite, even though it's about 100' across, only appears to be two-hundred *millionths* of a degree wide from the Earth (yes, they're *really* far away—but more about that in the *DISH Satellites* chapter).

So how do we make the beam narrow? The beamwidth is approximately given by the equation $\theta = k * \lambda/D$, where θ is the beamwidth; *k* is just a constant that depends upon the shape of the antenna and the definition of beamwidth we're using (there are several); λ is the wavelength of the electromagnetic radiation being focused; and *D* is the diameter of the parabolic dish. In words, what this equation says is that in order to make the beamwidth smaller (making the left side of the equation smaller), we can either move to a smaller wavelength (make the top of the divisor on the right side of the equation smaller), make a bigger diameter dish (make the bottom of the divisor larger), or both. By making the beamwidth smaller, we're able to put more of the power we generate into a specific spot.

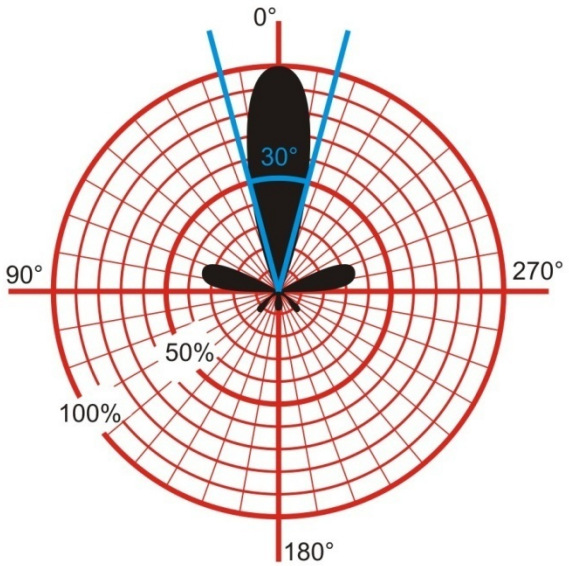

Figure 48: Beamwidth.

Figure 48 shows a plot of the main lobe and the sidelobes of a hypothetical radio antenna (i.e., we just made this picture up to show general features; however, it's reasonably representative of a real antenna). Straight ahead (the **boresight** direction of the antenna, labeled zero degrees) is at the top of the diagram and straight behind is at the bottom. The center of the circles on the diagram corresponds to 0% of the maximum power of the beam and the outer ring represents the maximum power, 100%. The black object in the diagram represents the various lobes of the beam. Notice that the vast majority of the power in this beam is in the **main lobe** along the boresight line. There are two somewhat significant **sidelobes** pointing about 75° left and right of the main lobe, some smaller sidelobes about 135° left and right, and even a small **backlobe** pointing directly behind the antenna.

The beamwidth, for our purposes, is measured from the point where the main lobe hits the 50% power level. You may hear this called the 3 dB (*three dee-bee)* point. They mean the same thing, but dB is engineer-speak. dB stands for decibel, but we don't need to go into that at this level. In this example, the beamwidth is about 30°, 15° to the left of boresight and 15° to the right, as indicated by the blue lines.

We've been showing waves and parabolas and such as flat drawings. In the real world, however, they're three-dimensional figures. There are many times that 3--D drawings are too complicated to show the basic concepts we're trying to convey. However, now that you've seen a 2-D drawing of what a beam looks like, we'll take a look at what it looks like in all its three-dimensional glory. Figure 49 shows a beam similar to the one we drew above but in all of its three-dimensional glory. The one we drew above was just that—a representative drawing designed to show essential features like beamwidth, the main lobe, sidelobes, and the backlobe. This one, however, is actually calculated from given antenna parameters. Notice that this one doesn't have its sidelobes suppressed nearly as much as the beam in Figure 48 did, and that its main lobe is much broader. Also

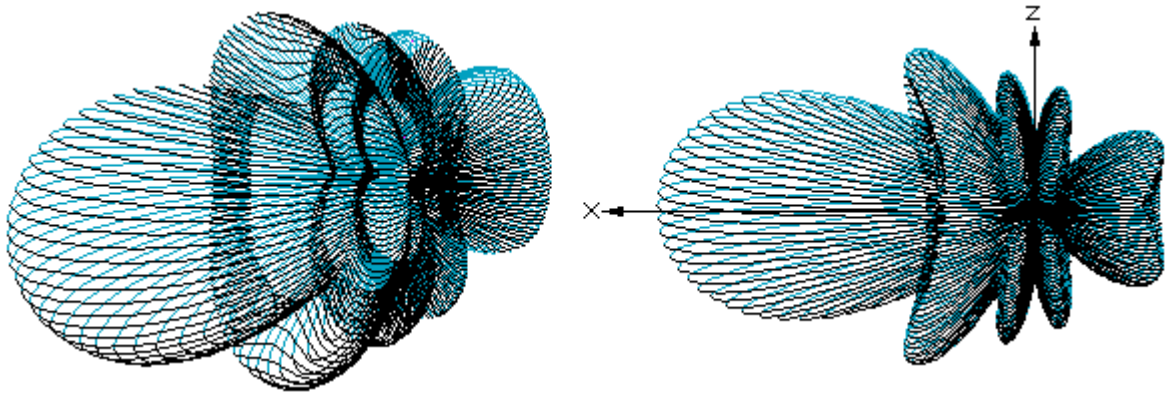

Figure 49: Three-Dimensional Antenna Radiation Pattern.[31]

notice the significant lobes pointing toward the rear of the antenna. It contains all of the essential features of an antenna beam we discussed above in simplified form, but also shows the complexity of a real antenna.

In addition to the beamwidth and lobe terms, you may also hear the term **gain** used in conjunction with antenna discussions. While there's a lot of math that goes into figuring out what the gain is, all it really means is how effectively are we at putting just as much of the power we're generating into the main lobe, and making that main lobe as narrow as possible. A gain of one means that we've basically got all of our power going into a sphere, equally transmitting in every direction. The higher the number for gain, the better for our application of trying to hit a very small, very far away satellite. Take a look at the black shaded area in the beamwidth figure, and try to imagine the same *area* except mashed down into a circle. Our best guestimate for this specific picture results in a circle that extends out to about the 30 or 40% ring. In that case, the gain would be 100%/40%, or 2.5.[32] What would the diagram look like were the gain higher? If the sidelobes were smaller and the main lobe narrower, the total area of the shaded region would decrease. The largest ring would still be at 100%, but now the circle we'd mash that shaded area into would be smaller. With the bottom of the gain fraction smaller (100% divided by a number smaller than 40%), the gain has to get larger.

So, let's apply what we've learned about beamwidth to some real numbers. Remember the approximate equation for beamwidth was $\theta = k * \lambda/D$. For a basic circular aperture, the constant *k* is 1.02. The wavelength of the DISH uplink signal is about 0.67 inches. The largest transmit antenna at the DISH uplink center in Cheyenne, Wyoming, is about 43 feet in diameter, or 480 inches (in order to use equations like these, the units all have to be the same; since we used inches for our wavelength, we need to use inches for the dish diameter as well).

Substituting these numbers into the beamwidth equation, we get a beamwidth of 0.001324, but that number is in radians (sorry about that—you don't need to understand radians except to know they're the way scientists and some engineers use to measure angles). There are 2π radians in a full circle of 360°, so converting to degrees (multiply by 360 and divide by 2π), we get a half-power beamwidth of 0.075°, which is a pretty tight beam.

Tight is a pretty vague word, though. How tight is tight in this case? To give you a visual comparison of what this beamwidth looks like, let's compare it to some things you have (and might have) seen. The Moon has an angular width of about half a degree and Jupiter (at its closest approach to the Earth) has an angular width of about 0.014°. As shown in Figure 50, that means the DISH beam appears to be about a sixth as wide as the Moon but about five times the size of Jupiter. As narrow as that beam is, it's still almost 30 miles across by the time it reaches our satellites! And remember, that's just the half-power beamwidth. The full main lobe is much larger, although much weaker at the same time, and when sidelobes are included it's wider still.

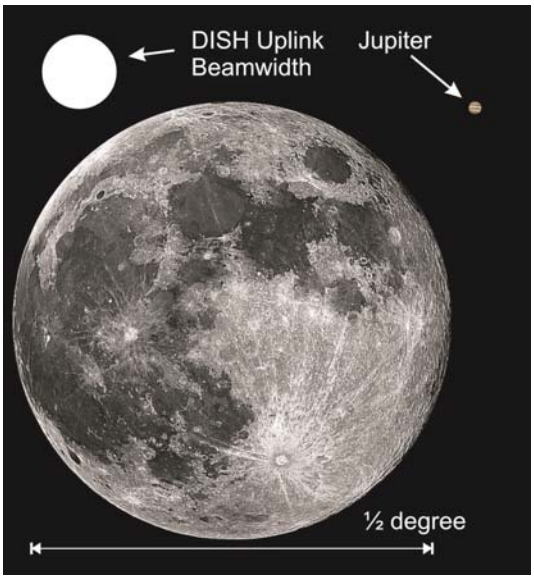

Figure 50: Dish Uplink Beam Width.

As you'd hopefully expect from your newfound understanding of beamwidth and gain, as the beamwidth decreases and the sidelobes get smaller, the gain should increase. Let's look at a few numbers that might help give you an idea of what typical DISH antenna gains are. The largest antenna outside DISH's 90 Inverness service facility has a diameter of about 21 feet and a gain of about 10,000. In contrast, the 43-foot dishes at our Cheyenne and Gilbert uplink centers have gains of about 65,000. The DISH dishes on our rooftops only sport gains of about 250. Bigger antenna, more focusing, larger gain.[33]

Just as an aside, returning to the C band signals received by the large backyard antennas DISH used to sell, if we used C band in our existing uplink antennas the beamwidth would be about 3 times as wide, since the wavelength of the middle of the C band is almost 2 inches, three times as long as the Ku-band wavelength we actually use. What do you think that would do to the gain?

Is the DISH uplink beamwidth of about a tenth of a degree good enough? Well, obviously it is since you're able to receive our DISH signals at your house. But how much of that signal power actually makes it up to the satellite? We'll talk more about that when we get to the *DISH Satellites* chapter.

We've spent quite a bit of time on basic physics so far in this chapter (with a brief excursion to discuss the political realm of FCC frequency allocations) because physics is so important to actually understanding the reason we choose the frequencies, antenna shapes, and antenna sizes we do. However, it's now time to move on to the subject of signal processing. We discussed a lot about signals in the *Prepping the Signal* chapter when we dealt with MPEG and encryption. For most of the rest of this chapter we'll explain the actual coding of the waves we send up into space.

Phase Shift Keying

When last we checked in on our program signal back in the *Prepping the Signal* chapter, it was an MPEG-coded, encrypted series of ones and zeroes—binary code. Let's see how we get this binary code out of our computers and up into space. Remember, we've got a huge amount of data to transmit, so we'll have to figure out how to do this as efficiently as possible. We also need our customers to be happy with the quality of the product DISH delivers, so in addition to efficiency we need to make sure what they receive is as close to what we transmit—as error-free—as possible. As you might have experienced in your lives, efficient, rapid communications don't always mean the most error-free messages. We'll look at how we balance these two competing interests

Pulse Code Modulation. To introduce you to coding, we'll look at a communication code almost everyone has heard of: Morse code. In Morse code, the letters SOS are represented by three dots, three dashes, and three more dots, where a dash is a long signal and a dot is a short one: · · · – – – · · ·. What is commonly overlooked in such a description is the spaces between the dashes and dots, as those spaces are required to keep them from all running together. Figure 51 shows this situation graphically.[34]

Now, while a human ear could take in these dashes and dots in a wide variety of forms (dashes might not be twice as long as dots, spaces might be much shorter than dots, the speed might vary throughout the message, etc.), a computer is much more rigid. Programmers tell them to expect data at a certain rate, and that rate is usually fixed in some manner to the clock speed of the computer. That's why we've labeled the time axis in terms of clock cycles in the figure.

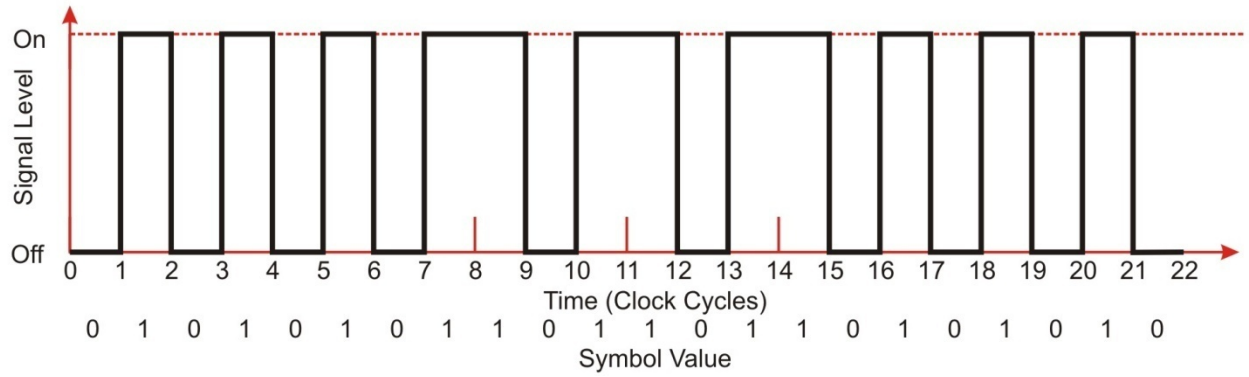

Figure 51: Data Content of a Transmitted Morse Code SOS.

Notice that it takes 22 clock cycles just to convey these three letters S-O-S using a simple binary system like Morse code.

Morse code is a very simple example of a binary system—one in which there are only two states the system can be in. In this case, the two states are signal or no signal. To the telegraph operator, signal meant a sound, no signal was indicated by silence. To the computer, signal means it's getting a certain level of electrical voltage and no signal means no voltage. During each clock cycle, the computer is able to record a single bit of information. (A **bit** is the most basic unit of information—signal or no signal, a one or a zero.) That's what the programmers told it to expect for the code. The bit values below the time axis show what the computer transmitted—a one or a zero representing the on and off states—during each clock cycle.

Of course, there is going to be some noise in any electrical system, with **noise** meaning voltage changes that are not related to the signal. It could be caused by the line picking up electromagnetic waves from a distant lightning strike, by variations in the electrical supply to the computer, or even just because the computer has a non-absolute-zero temperature. To deal with this unavoidable feature, engineers set ranges around the values they tell the computers to expect to represent signal or no signal to reduce the number of erroneous bits. The range is set to deal with the level of noise they expect in the system. If the signal received during a clock cycle is, on average, above the error level, it's counted as a one; below the error level, it's a zero. This situation is shown in Figure 52.

In this case, the engineers have designed the system to correctly deal with the expected noise level by appropriately setting the signal difference between on and off and correctly setting the error level. All the bits, though very noisy, are translated to the correct states of the system—on or off, one or zero.

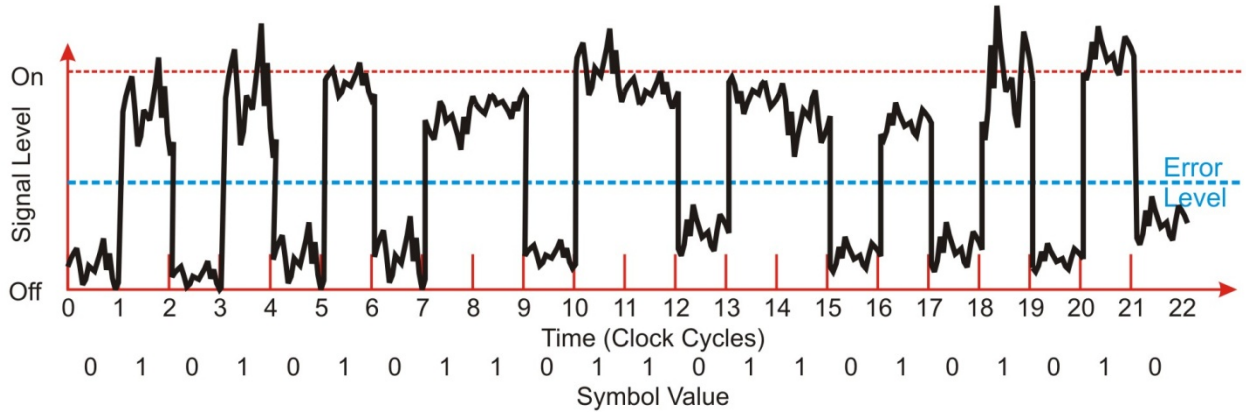

Figure 52: Received Morse Code SOS with Noise.

So we've seen our binary transmission system can transmit one bit at a time. If we were able to adjust our transmitter to send four levels of signal, could we send data more efficiently? The answer is yes, but there are some caveats.

Looking back at Figure 52, we see that by sending the string of seven bits 0101010 (low-high-low-high-low-high-low) we've communicated the letter *S*. With four levels, we can increase the complexity of our code a bit. Let's say that the lowest level of signal represented a zero followed by another zero, the next lowest signal a zero followed by a one, and so on as shown in Figure 53. (These combinations are actually the numbers zero through three in the binary system.)

Using this two-bit system, the letter S could be sent with the signals low-low-low-off or 01 01 01 00 (assuming a second zero for a space following the *S*, since all values used by the code must now represent two binary numbers). We've communicated the same information using four bits instead of seven. This observation brings us to another definition. The **bit rate** is the number of bits (ones and zeros) we can communicate per second. Assume that the clock intervals in Figure 52 are one second each (in most communication systems, they would be far smaller—thousandths or millionths of a second, but for the purposes of this example, let's make the math easy and use one second). For our binary Morse code system, we only transmit one bit during each clock cycle, so the bit rate would be 1 bit per second (1 b/sec). With this new four-level system, we're sending two bits during each clock cycle, so the bit rate is 2 b/sec.

Signal Level	Value Represented
Off	00
Low	01
Med	10
High	11

Figure 53: Four-Level Bits.

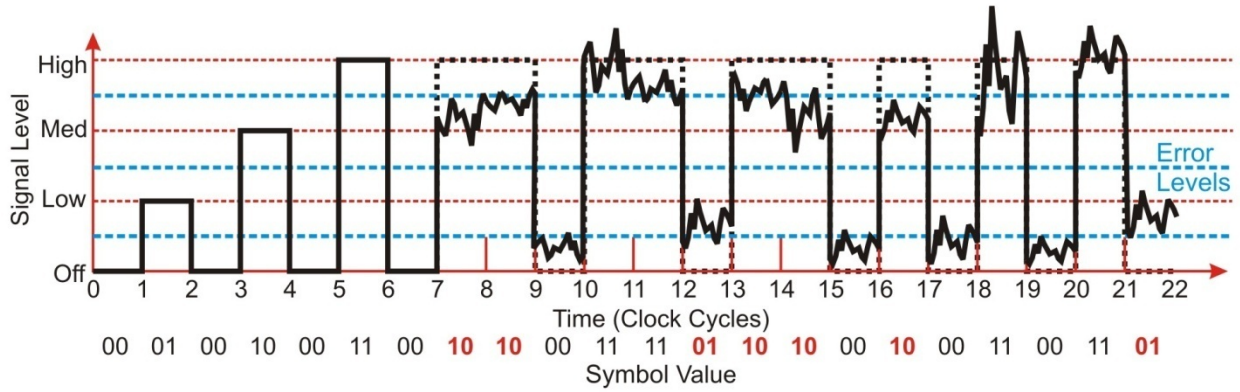

Figure 54: Four-Level Signal, Noiseless and with Noise.

Let's look at a graphical example. Instead of just on and off, let's now assume our transmitter can send off, low, medium, and high levels of signal. That situation is shown in the left three noiseless peaks in Figure 54. We can send twice as much information in the same amount of time with four levels instead of two.

However, this capability comes with a potential drawback. Let's assume we were stuck with the same transmitter capable of delivering the same signal power, and that the noise level was the same as in the Figure 52. We will also set the error levels halfway between each of the desired levels, as before. The noisy peaks on the right side of Figure 54 are exactly the same shapes and signal strengths as the Morse code signal for the letters *O* and *S* we received in Figure 52 (though, in this coding, they don't mean the same thing); the dotted peaks are what we transmitted. Instead of the transmitted and received signals matching up at the same signal levels like they did before, we've got a lot of errors this time. Notice the red numbers in the signal values below the time axis. They show where the noisy solid, received signal differs from the noiseless, dotted transmitted signal. The lesson to be gained here is that while it's possible to add more and more levels to our transmitted signal to increase the rate at which we can send information, it comes at the price of a potentially higher error rate because the difference between the different signal levels are now closer together and the noise level hasn't changed.

Phase Shift Modulation. So that's some basic information about coding, but it's *amplitude* coding, where we vary the value of the signal's strength—its amplitude—to carry the information. That's not what we use in the DISH network. We actually shift the *phase* of the wave to convey the code.[35] What is phase? We're glad you asked!

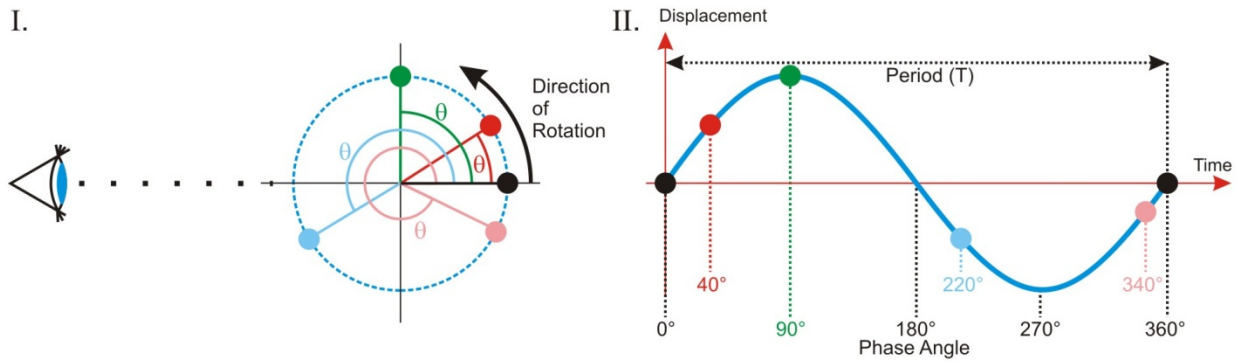

Figure 55: A Wave's Phase.

Figure 31 (p. 49) showed the parts of a wave, but it neglected phase because we didn't need to know about it until now. The concept of phase is described in Figure 55. Imagine you're watching a ball on a string being swung around in a circle as shown in Frame I. Your vantage point is to the side of the circle, not front-on. From that point of view, the ball really just looks like it's moving up and down (although you know it's really moving in a circle).

At time zero, the ball starts at the black position, directly away from you. As it rotates, it passes sequentially through the red, green, blue, and pink positions, returning after a complete cycle to the black position. Frame II shows the position of the ball as a function of time (remember from the discussion related to Figure 31 that having time on the horizontal axis means the distance between any two similar points on the sine curve is the wave's period). Note that the ball's height, traveling in a circle, appears to trace out a sine curve when viewed from the side. Its plotted displacement on the graph in Frame II corresponds exactly to the vertical positions shown on the circle in Frame I.

If we view the circle from the front, it's apparent that we can describe the position of the ball using an angle measured from the ball's starting point. The colored θ symbols (the Greek lower case letter *theta*) correspond to the angle of the similarly colored balls. Notice that when the ball makes a complete cycle, it's gone through an angle of 360°, and that it's made a single, complete period of oscillation on the sine curve. These two actions, the angle related to the circular motion and the oscillation of the sine curve, are so closely related that we're going to define the angles as the **phase** of the oscillation. The phase angles of the colored balls are indicated at the bottom of Frame II.

You can see that the right angles (0°, 90°, 180°, 270°, and 360°) occur at special places on the sine curve—either at the maximum positive or negative displacement, or where the displacement is zero. In fact, the phase of 360° looks

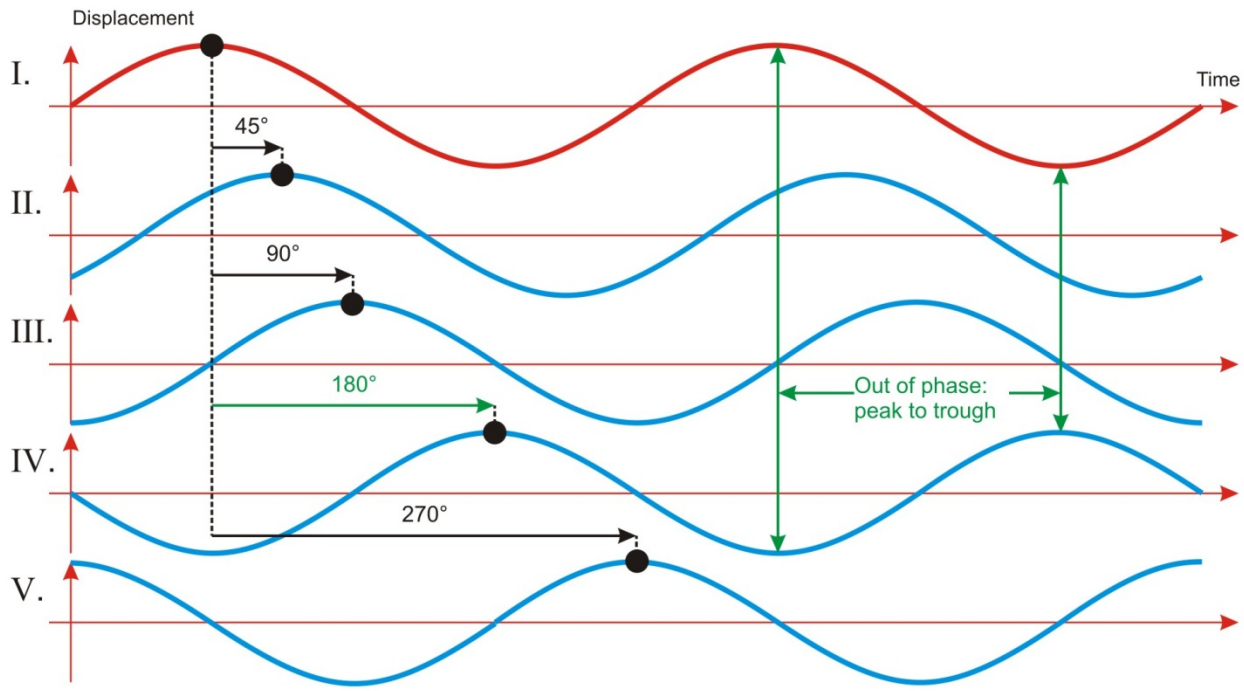

Figure 56: Phase Shifts.

exactly like the phase of 0°, so we really don't need to ever use a phase angle of 360° or greater—we'll just repeat the phase angles again starting at zero.

So that's what the phase angle—or just phase—of a wave means. But we said we used phase *shifts* to code our information. How does that work? Figure 56 shows what we mean by a phase shift. It shows five sinusoidal curves each having the same period (peak to peak distance) and amplitude (maximum displacement value). The only difference between the curves is that they are progressively shifted further to the right.

The top curve, Frame I, is your basic sine wave (sine waves always start their upward trajectory from the origin). The curve in Frame II starts its upward trajectory a short time to the right of the origin, a little later in time. On each of the curves, the position of the first peak is highlighted with a black circle. You can see from the drawing that the peak of the curve in Frame II is shifted by 45°. We say that curve *lags* the original sine wave by 45° because the peak happens at a later time (remember, time increases as you move to the right on this graph). Frames III to V show additional sine curves with phase shifts of 90°, 180°, and 270°, respectively.

The plots in Frames IV and V are somewhat special, but for different reasons. Frame IV's curve is what we call *one-eighty-out* from the sine wave—it's got a

phase shift of 180°. Its peaks occur when the sine curve's valleys happen, and vice versa. Were you to add the amplitudes of these two curves together at all times, they would completely cancel and leave only a straight line at amplitude zero. Frame V's curve is actually a cosine curve because it starts its downward trajectory from its maximum value at time zero. That means that the cosine curve lags the sine curve by 270° degrees (or leads the sine curve by 90°, depending on your point of view).

Now that you understand the concept of the phase shift, let's look at how it can be used to code data. The top part of Figure 57 shows our old friend the Morse code SOS, just like we saw in Figure 51. It's coded using one-bit amplitude modulation, or by just varying the strength of the signal to be either on or off, so the bit can either be a one or a zero. The bottom part shows the same SOS, but this time coded using one-bit **phase shift keying** (PSK). Since there are only two allowable states, this version of PSK is known as **2PSK**. Here the bits that are zeroes are just sine waves (phase = 0). Check that by referencing what a sine wave looked like in Frame I of Figure 56. In Figure 57, the first zero-coded bit is highlighted in red for easy visualization. In contrast, the bits that represent ones are sine waves but shifted by 180°, as you can also check with Figure 57, Frame IV, and visualize from the blue-highlighted segment of the curve in this figure. Since sine curves (regardless of phase) are smooth, you can easily spot where the phase changes in this figure by seeing the sharp spikes. See why we first used

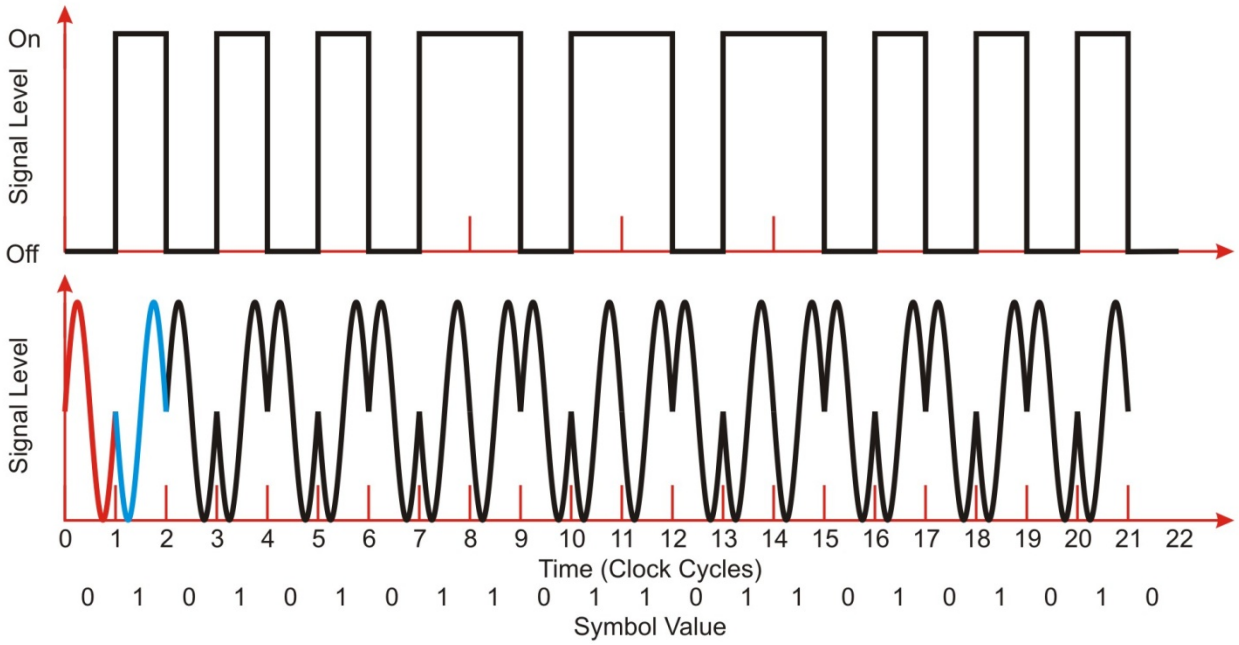

Figure 57: Phase Shift Keying.

2PSK One bit per clock cycle		QPSK Two bits per clock cycle		8PSK Three bits per clock cycle	
Angle	**Symbol**	**Angle**	**Symbol**	**Angle**	**Symbol**
0°	0	45°	01	22.5°	011
180°	1	135°	00	67.5°	010
		225°	10	112.5°	000
		315°	11	157.5°	001
				202.5°	101
				247.5°	100
				292.5°	110
				337.5°	111

Figure 58: Symbols for 2PSK, QPSK, and 8PSK Modulation.

amplitude modulation to do the basic explanation? It's a whole lot easier to see on a graph, isn't it?

Just like the four-level amplitude shift example we saw in Figure 54, we can make a four-level phase shift system as well. Typically, the shifts used are odd multiples of 45° (45°, 135°, 225°, and 315°). This version of PSK is called quadrature PSK (**QPSK**) because it uses four angles to convey information. We can also use eight levels (notice the binary progression of the number of bits we use in these codes, from two to four to eight), and the angles associated with these phase shifts are odd multiples of 22.5° (22.5°, 67.5°, etc.). Since there are eight of these angles in a complete circle, this version of PSK is called **8PSK**. The codes associated with the phase angles in 2PSK, QPSK, and 8PSK are shown in Figure 58. The reason for the seemingly strange order of the coefficients is beyond the scope of this text, but is related to the concept of two's complement coding.[36]

To demonstrate the additional efficiency of transmitting more information per clock cycle enabled by the additional bits of information, look at how the code for SOS is transmitted using PSK in Figure 59. However, just as we saw in our amplitude-modulated system in Figure 54 (p. 72), noise becomes more of a problem as we try to squeeze in extra bits. It's harder to show for the phase shifts, but the concept is the same.

So far, we've talked about phase shifts as an absolute thing. However, you can only tell if the phase has shifted when you *compare* the signal to a reference signal of some sort. When we introduced the concept of phase in Figure 56 (p. 74), notice that the shifted signals were all compared with the original sine wave

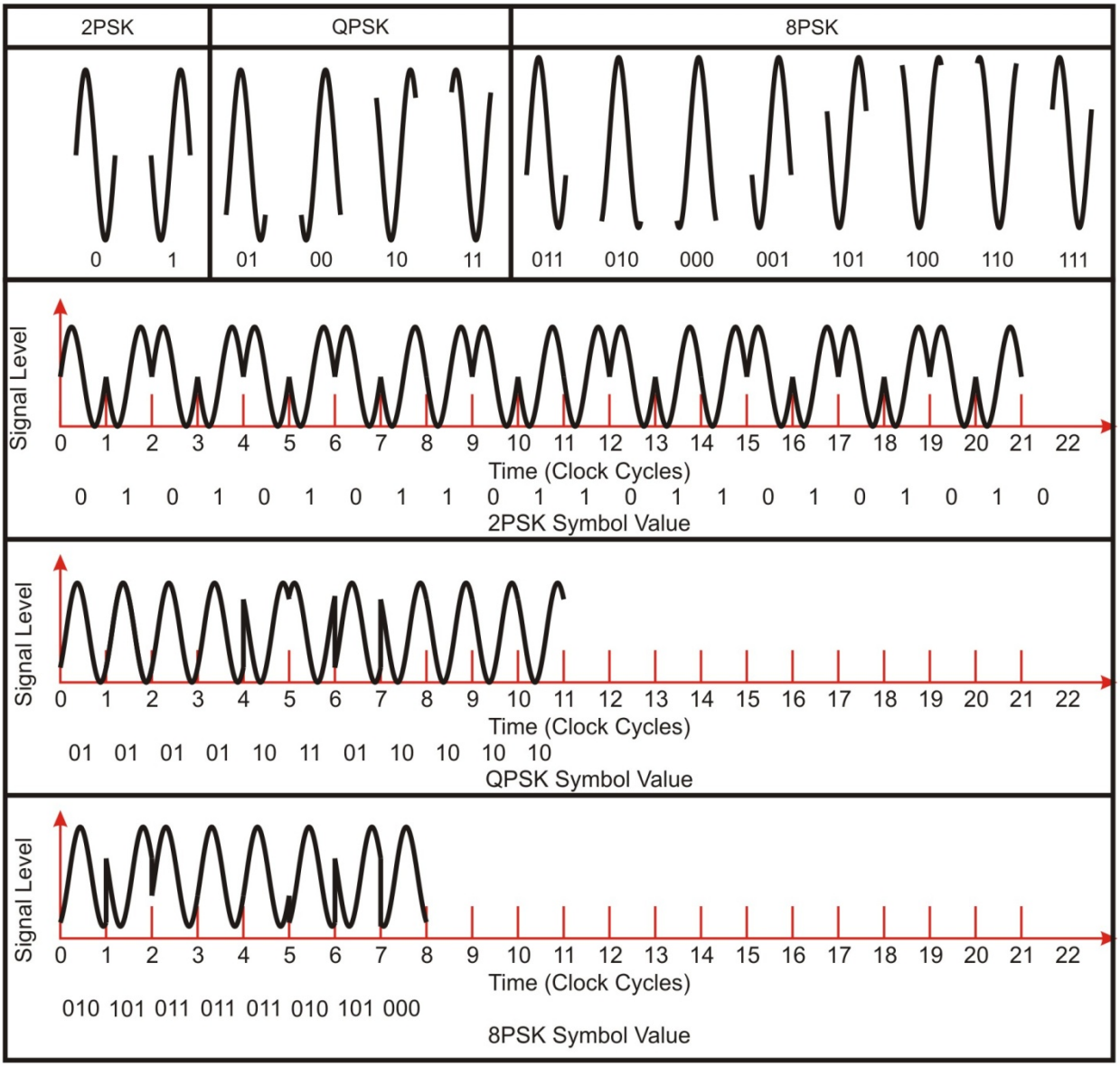

Figure 59: Transmission Time Efficiencies Gained Through Multi-Bit Transmissions.

in Frame I. That sine wave was the **reference signal**. Without it, all the other waves would just have been sine waves or cosine waves depending on where we'd set the zero time axis.

So, for us to detect a phase shift in our signal, the receiver needs to have its own reference signal that's *exactly* in synch with the transmitter's reference signal. How close to *exactly* does it have to be? Well, let's look at a very small number. How about a hundred billionth? That's pretty small. So let's say that our transmitter and our receiver have signal generators that are that close together—

they are *so* good that their signals only drift away from each other at a rate of one part in a hundred billion. Sounds pretty tight, doesn't it?

The only problem with that tiny hundred-billionth tolerance level is that our Ku-band uplink signals oscillate at a rate of about 17 billion times per second (17 GHz), so that means that in a little over two seconds they're now 180° out of phase. Look back at Figure 56 to recall what 180-out means (the peak of one of the signals lines up with the trough of the other), and then look at the zero and the one symbols in 2PSK in Figure 59. Those two symbols are 180-out from each other. If we were using 2PSK, that means every 2-ish seconds, when we send a one our receiver thinks it's receiving a zero! Not a very good result! And it only gets worse as we move to 4PSK and 8PSK, since the phase shift required to get an erroneous symbol gets smaller for those codings.

In order for our scheme of having our transmitter and receiver's reference signal generators synched up to work, the signals have to be virtually *perfectly* aligned.

Perfection is a pretty high bar to meet when you're building a real device, so let's see if there isn't a better way to generate a reference.

Instead of relying on two separate clocks generating two separate (but identical) reference waves, one for the transmitter and one for the receiver, what if we as receivers instead *used the incoming wave as our own reference*? That's exactly what we do when we employ the **differential phase shift keying** (DPSK) scheme.

Take a look at Figure 60. The incoming data stream is a series of one-cycle sets of symbols. In this example we're using the binary 2PSK scheme for simplicity, but it could be any phase shifting coding. When the signal hits the receiver, it is immediately put into the input bin. The way the receiver decides whether the

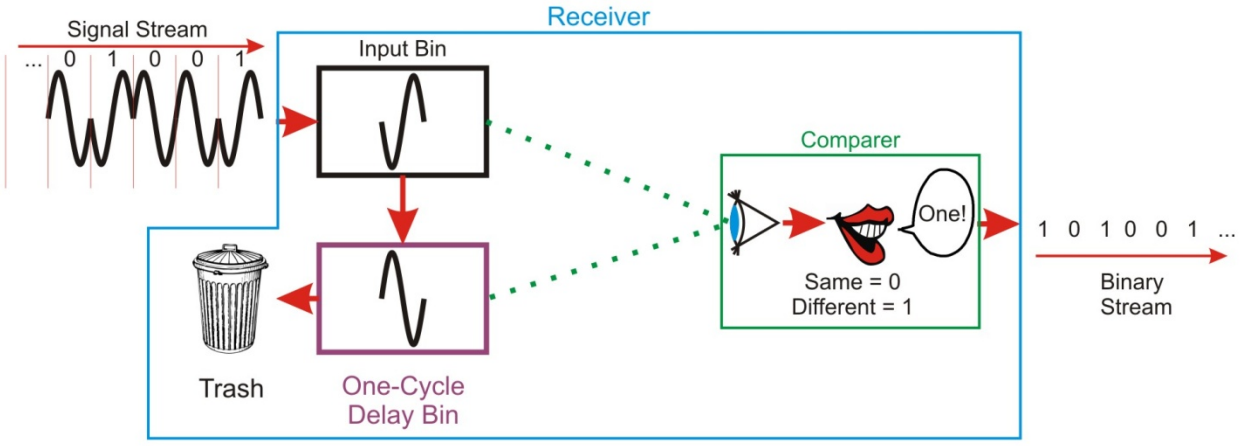

Figure 60: Differential Phase Shift Keying.

incoming symbol is a one or a zero is to compare it with the waveform that was received *one clock cycle earlier*. That waveform was moved from the input bin into the one-cycle delay bin just as the symbol in the input bin arrived. The receiver then looks at the two waveforms to see if they're the same or different, and from that comparison decides whether the symbol should be interpreted as a one or a zero. When the clock moves again, the symbol in the delay bin is thrown away, the symbol in the input bin moves to the delay bin, and the new symbol moves to the input bin. In this way, the signal stream is continually converted into a binary data stream without reference to anything but itself.

Now, there are a lot more details that make DPSK work, but they're beyond the scope of this text.[37] And just for reference, DISH uses DPSK operating on 8PSK and QPSK signals to convey our signals to and from our satellites.

The bottom line of this discussion about different forms of PSK is that we can transmit three times more data in the same time with 8PSK than just a binary modulation like 2PSK. We even get a 50% increase in data rate with 8PSK over QPSK. And data rate is a money maker when we have limited bandwidth, two concepts we'll discuss next.

Data Bandwidth

Did we just say *bandwidth, two concepts*? We did, and we don't have a problem with subject verb agreement. *Bandwidth* is a term bandied about in many publications willy-nilly, but it has two very different meanings in very different contexts. We'll discuss both of them below, as they're equally pertinent to DISH operations. To keep confusion to a minimum, we'll always refer to the two kinds as **data bandwidth** and **spectral bandwidth**. The first refers to the speed at which information can be transferred, and is measured in bits per second (bit/s). The second refers to the band of frequencies required to transmit our signal and is measured in hertz. The two quantities are closely related because the broader the spectral bandwidth of the signal, the more information you can send on that signal.[38] Since our business relies on transferring huge amounts of information quickly, let's look at some of the things we can do to increase our data bandwidth.

First, we've already discussed one way to get faster data transfer: make the bit rate faster. We talked earlier about bit rate and how we could increase it by sending more than one bit per clock cycle by using such advanced coding techniques as QPSK and 8PSK. By adding bits to the transmitted symbols, we directly increase the data bandwidth; twice as many bits per symbol translates to twice as much data bandwidth. That's a pretty good payback, isn't it?

Well, there's a drawback to this method. We also discussed how adding noise to the signal increased the error rate (recall Figure 54, p.72). When we divided our maximum amplitude into smaller and smaller regions, the noise variations became more and more significant and began introducing more and more errors.

Figure 61 shows a series of signals against a constant average background noise level. Notice that the noise is always present, and it just adds its amplitude to our signal values. As the bars representing the signals get taller, the ratio of the signal's height to the average height of the noise level gets larger. This ratio is called the signal-to-noise level, or SNR. For example, the bar on the right has five times as much signal amplitude as the average amplitude of the noise, so the SNR is five. The higher the SNR, the less significant the noise will be on our ability to transmit data.

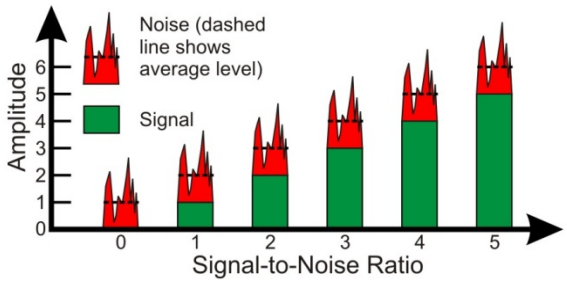

Figure 61: Signals and Noise.

Now, we saw that doubling the number of bits we transmitted per clock cycle doubled our bit rate. Doubling our SNR doesn't do quite as well for us. Look at Figure 62. It shows the factor by which you multiply the data bandwidth as you change the SNR. Since the curve tails off more and more as the SNR gets larger, there are diminishing returns from increasing the DISH uplink signal strength. That's a good thing, because increasing our signal strength means adding more powerful transmitters, and those are pretty expensive.

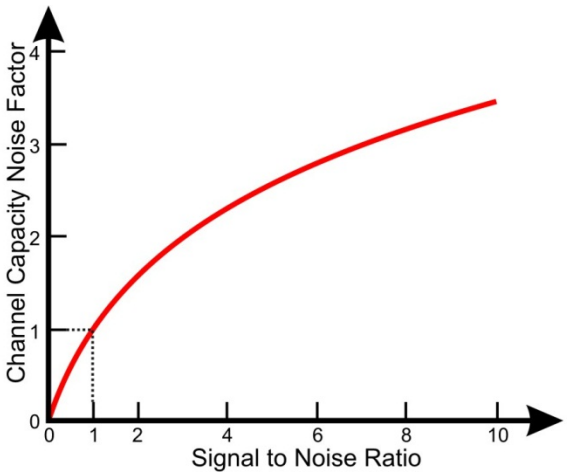

Figure 62: Noise Affects Data Bandwidth.[39]

Noise is always present in one form or another. In Figure 61, the noise is amplitude noise, which as we discussed earlier, can come from lightning, the Sun, other nearby radio spectrum broadcasts, and many, many other sources. While we can't eliminate noise, one way to reduce its impact is to send our signal in a way where the background noise isn't as prevalent.

We actually already take advantage of this tactic by using phase shifts as our

coding method rather than amplitude modulation. We explained SNR in relation to amplitude noise just because it's way easier to draw. But noise also exists that can affect the phase of the received signal. Phase noise can come from a variety of sources, including passing a signal through the ionosphere, and spurious inputs from the Sun. However, probably the biggest input to phase noise is the design of our transmitters. It's virtually impossible to build one that puts out a perfect sine wave all the time. There are always little frequency jitters in the wave, and those jitters show up as phase shifts. The best we can do is to make just as stable an oscillator as we can, and design our signal levels to be very error-resistant based on the known hardware limitations. In fact, the reason we use 8PSK instead of 16 PSK is related to that problem—the phase jitter is enough to generate errors when the phases look as similar to each other as they do with 16 levels. However, the noise induced by these oscillator jitters and other effects is generally well below the level we'd have to deal with were we using an amplitude modulation scheme like the flashlights we discussed in *The Basics*.[40]

So we've discussed two of the three primary ways to increase our data bandwidth: increase the number of bits we send in each symbol (we use 8PSK to send three bits per symbol) and increase the signal to noise ratio (we use relatively high power transmitters and take advantage of the fact that phase noise is lower than many other types of noise). The third way to increase our data rate is to increase the spectral bandwidth of our signal. Like increasing the bit rate, that relationship is pretty straightforward: if you have double the spectral bandwidth you can send twice as much information. Again, a pretty good payback. But just what do we mean by spectral bandwidth?

Spectral Bandwidth

Spectral bandwidth just means how wide the frequency band is that we use for our signal. As you might expect, some signals require more spectral bandwidth than others. The human ear is capable of hearing frequencies between about 20 Hz and 20 kHz, essentially a spectral bandwidth of 20 kHz (20,000 - 20 = 19,980, which is pretty close to just 20,000). However, our brains can decipher the content of speech using only the lower frequencies, so a typical telephone signal only uses about 4 kHz of spectral bandwidth. Digital music is usually double-sampled for fidelity, so it uses about 40 kHz of bandwidth. Full HD television signals like DISH transmits are much, much heavier users of bandwidth. Our satellite TV signals require almost 30 MHz of spectral bandwidth—over a *thousand* times what the human ear can process.

But we've been talking about radio frequencies, with the Ku band being in the range of about 10 GHz, about a thousand times higher than the spectral bandwidths we've discussed here. What's the relationship between these two frequencies?

The frequencies DISH is licensed to use by the FCC are called **carrier frequencies**. They're the ones in the GHz ranges. The frequencies of the information we're trying to transmit are called **modulation frequencies**. For HD television, these are in the MHz ranges. In order to transmit our program information at the carrier frequency, we have to mix them, which just means to use an electronic device called a mixer that multiplies the two signals together.

Before we go into the mixing process in detail, we really need to take a look at two different ways to look at a wave. You've already seen one of the ways, but understanding the second way will make it a *lot* easier to understand the subject of mixers and spectral bandwidth when we get to them.

We've discussed a wave's frequency and amplitude already (see p. 49 for a review), and we've already used the presentation format in the left-hand frames of Figure 63 to illustrate those concepts. Those frames show amplitude on the vertical axis and time on the horizontal axis. In that format, waves look like the wiggles you've come to know and love. In Frame I, for example, we show a blue wave we'll call the carrier signal (for reasons that will become clear in just a little while). This wave oscillates up and down ten-billion times a second, or at a frequency of 10 GHz. You can't tell that directly from the figure, though, since the horizontal axis is labeled with time, not frequency. From your review, you should remember that the relationship between frequency and period, the time between successive wave peaks, is f = 1/T. Counting the number of full waves between the zero and one tic marks, we see there are ten of them. That means the period of

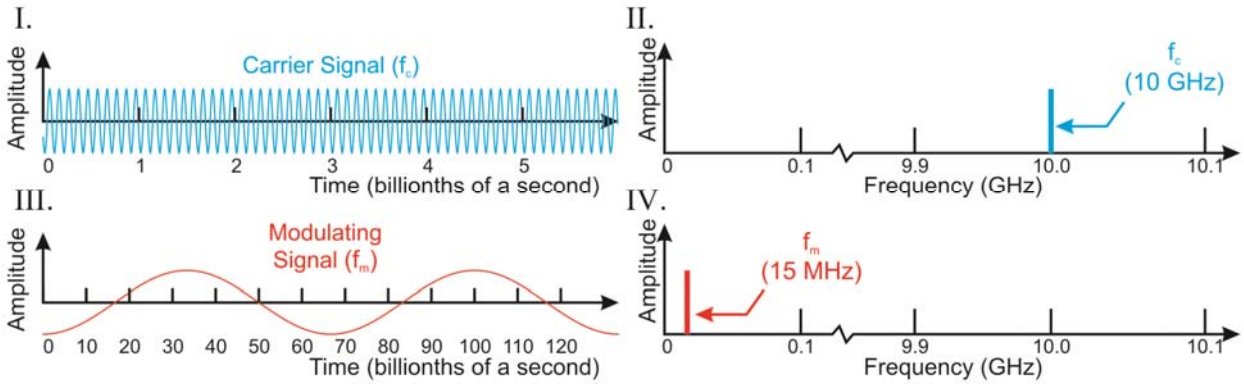

Figure 63: Two Ways to Visualize a Wave.

the wave is ten oscillations every billionth of a second, or one oscillation every ten-billionth of a second. One over one ten billionth of a second is ten billion oscillations per second, or 10 GHz. Anyway, all seems well in the world because you've seen and understood this way of looking at a wave before.

Moving down to Frame III, we see a different, red wave we're calling the modulating signal (again, the reason for that name will become clear in a minute). Figuring out what frequency this wave has is a little trickier, since a full oscillation doesn't seem to match up exactly with an easy-to-identify number on the horizontal axis. We'll help you out a bit and tell you the wave repeats itself every 66.67 billionths of a second. Doing the same one-over-period math as above, that works out to a frequency of 15 MHz. Again, nothing new here you haven't seen before.

Let's rock your world a bit now and change things around a little. Look closely at Frames II and IV. They show *exactly the same waves* as are shown in Frames I and III. Really! It doesn't look like the wiggles you're used to, but trust us, the same information is contained here. Notice the amplitudes are exactly the same, shown by the heights of the blue and red bars in Frames II and IV and the displacement of the wiggles from zero in Frames I and III. However, the big difference is that the horizontal axes are labeled with *frequency* instead of *time*. (The spiky squiggle to the right of 0.1 GHz on the frequency axis means that we've left out some of the values on that axis so we could jump between places that are interesting, in this case, being able to show the 10 GHz and 15 MHz signals on the same format axes.) Now, instead of having to do math and count wiggles, we can directly read off the frequencies of 10 GHz and 15 MHz. That's a *way* easier thing to do if the information we're interested in is frequency, isn't it?

It's very important to realize that both of these presentation methods give you exactly the same information—the amplitudes and frequencies of signals. It's just that sometimes seeing the wiggling wave gives more insight and sometimes seeing the exact relationship of the frequencies is better.

So now let's look at what combining, or *mixing*, these two very different frequencies does for us. Figure 64 show two examples of a modulated carrier wave on both the time and frequency axes. The big difference between this figure and the previous one is that only one signal was shown in each frame in Figure 63 and multiple signals are shown in Figure 64. Again, the blue carrier wave oscillates at 10 GHz and the red modulating signal (representing the highest frequency of the signal we use to transmit our HD program) oscillates at 15 MHz.

(Yes, in the figure that 10 GHz signal should oscillate over a thousand times for every time the 15 MHz signal oscillates once. That would be very hard to draw, so we've drawn the blue wave oscillating *much* less frequently in Frames I and III for clarity.) Note that these frequencies aren't really the ones DISH really uses—they're close, but we chose them because they made the additions easier to do.

Also notice that we've shifted the curves up on the amplitude axis in Frames I and III so the waves are separated for clarity. That means that in these frames it's the relative amplitudes that matter here, not the actual vertical location.

Anyway, when we vary the amplitude of the carrier with the modulat*ing* signal, we get the green curve, which represents the modulate*d* signal. Notice that it continually oscillates at similar rate as the blue carrier curve, but the amplitude of those oscillations has been modified to fit the shape of the red curve. We won't go into the math of why that happens, but it's not too difficult for those who are interested

Frame I shows a situation where the amplitude of the modulating signal is quite a bit less than that of the carrier. Frame II shows the **frequency spectrum** of Frame I, that is, it changes from a horizontal *time* axis to show exactly the same information on a *frequency* axis. Notice how much less cluttered that diagram is,

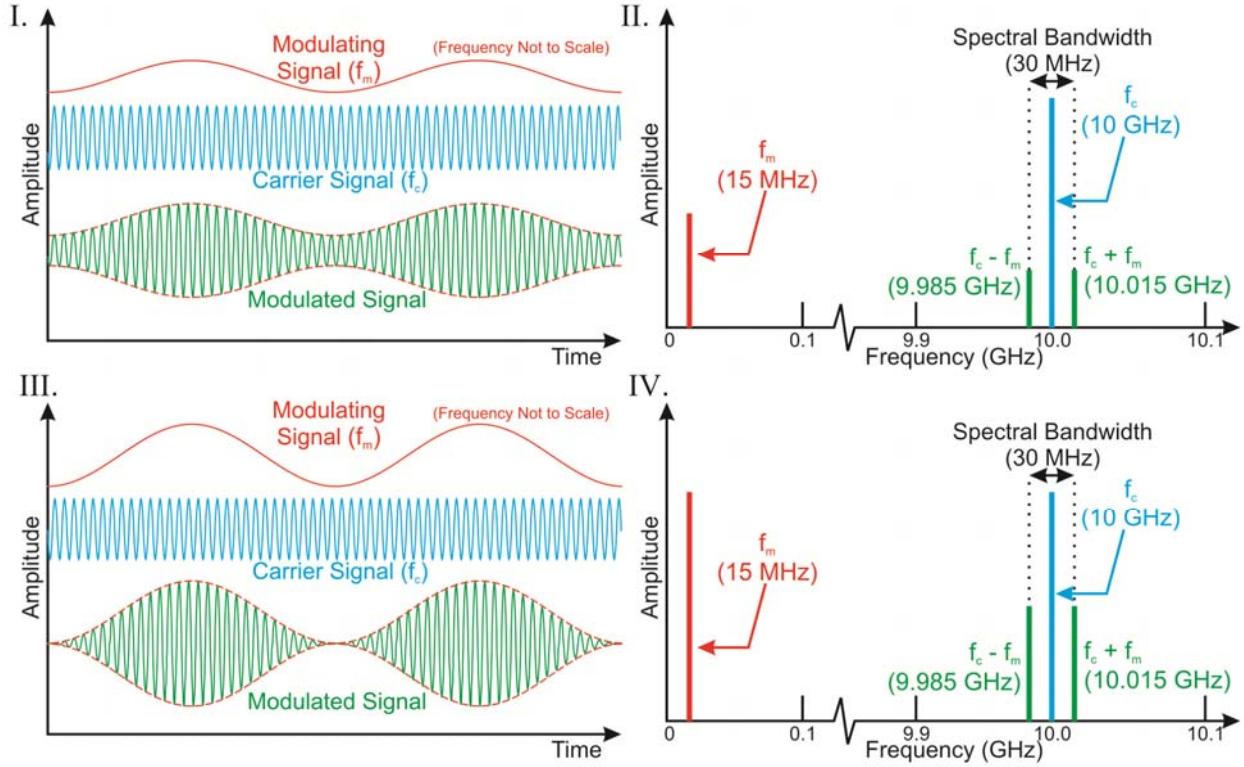

Figure 64: Modulation and Sidebands.

but also notice it doesn't show the wiggles and the envelope that may make more intuitive sense to you. Sometimes one way of looking at things is better than another, depending on the information you're trying to convey. In this case, we trade the ease of mentally understanding the picture of the wave for simplicity of drawing and seeing the spectral content—the frequencies that make up the wave—when we move from time to frequency points of view.

In any case, in Frame II you can see there is a red bar representing the modulating signal at a frequency of 15 MHz, as advertised before. There is also a blue bar representing the carrier signal at 10 GHz. Those are the two input signals. As we saw in Frame I, the amplitude of the blue carrier signal is quite a bit larger than the amplitude of the red modulating signal.

You may be wondering at this point why there are two green bars. Where did they come from? We won't go into the math,[41] but when you mix two signals of different frequencies like we've done,[42] you end up with two additional frequencies that appear at both the *sum* and *difference* of the frequencies we mixed. In this case, you can see the green lines representing the sum and difference frequencies, and by inspection you can see that they're separated from the blue bar by the same amount as the red bar is separated from the vertical axis, that is, they are at 10 GHz +/- 15 MHz, or 9.985 GHz and 10.015 GHz. These smaller bars are called **sidebands**. The difference between these two sideband signals is 30 MHz, and that total spread is called the **spectral bandwidth** of the signal.

The only difference between these frames and Frames III and IV is the level of the modulation. In this case, you can see that the amplitude of the modulating signal is the same as the carrier. This situation is called 100% modulation. The envelope of the amplitude oscillations in the modulated signal *just* goes to zero when carrier and modulating signal amplitudes are equal. With 100% modulation, we get just as much power into the sidebands as possible. Were we to crank up the modulating signal any more, though, we'd start to over-modulate and would see a marked decrease in signal quality on our TVs.

Up until this point in our carrier/sideband discussion we've only discussed modulation by a *single* frequency at the very edge of the spectral bandwidth. We did this for clarity, since it's easier to discuss how one frequency mixed with the carrier ends up forming two sidebands separated from the carrier at the frequency of the modulating signal. Let's now turn to a brief discussion of what real modulation of an HD program might look like, a situation shown in Figure 65.

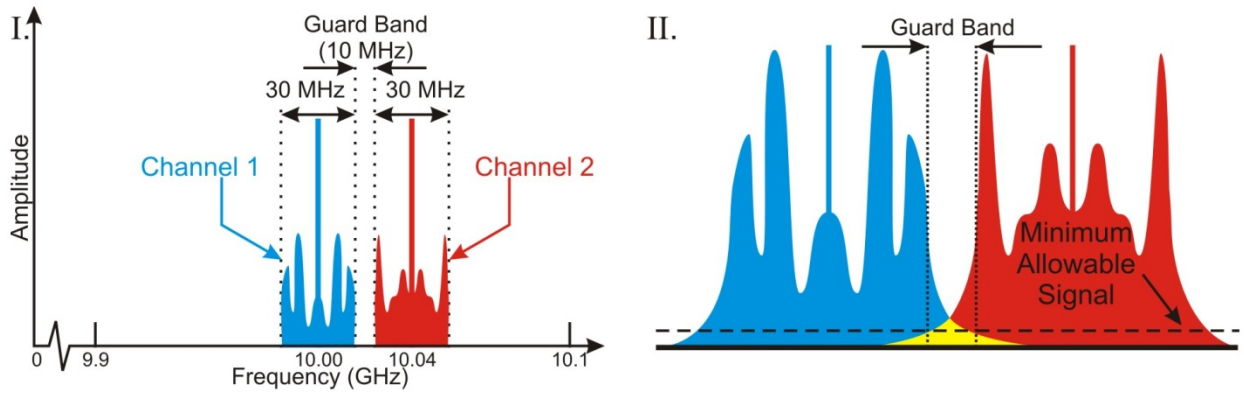

Figure 65: Channel Separation.

In Frame I, instead of a single modulating frequency we have a whole slew of signals between zero and our maximum modulating frequency of 15 MHz. We can tell this, even though the **baseband signal** (the actual modulating signal that would be on the horizontal axis near the origin) isn't shown because we know that the mixed signals end up generating frequencies above and below the carrier—sidebands—that each have the same spectral width and amplitudes as the original modulation. The blue region shows this situation in the figure. These continuous peaks and valleys of different frequencies look much like a real signal might.

But we've added something else here as well—a second channel, denoted by the red region. Notice the red and blue regions don't overlap. We have to keep the channels separated in frequency so they don't interfere with each other. Remember how the FCC kept different users of the radio spectrum from interfering with each other (recall Figure 38, p. 57)? Well, we have to do it within the spectrum we've been allocated as well. Each of the channels has the same spectral bandwidth of 30 MHz, but just to be on the safe side we've decided to add a 10 MHz guard band between the two channels.

So why do we need the guard band? It's because we haven't been exactly accurate in drawing our signals up to this point. We've assumed they were exactly single frequencies and dropped off from their maximum amplitude to zero immediately. Real signals don't act like that. Even if we try to transmit on a single frequency, it has a certain frequency range because it doesn't drop off immediately. Take a look at the region between the red and blue regions in Frame II. You can see that, in contrast to Frame I, the blue and red regions don't go to zero amplitude right at the edge of the guard band. Instead, they drop off gradually.

The yellow-shaded region shows where the red and blue regions overlap—and where they overlap they interfere. We can't allow that while insisting on consistently high-quality picture and audio quality for our customers! That's why we have guard bands between channels—to help prevent interference.

But notice that the yellow overlap region extends past the guard band. That actually happens in real life, and we have to figure out how to work around it. The way we ensure that we don't have any meaningful interference is by setting a minimum allowable signal level throughout allocated spectrum. That means that we ignore any signals lower than that level. By setting guard bands and the signal floor, we don't have to worry about any interference from the yellow overlap region.

To close out this section on carriers, sidebands, and spectral bandwidth, let's take a look at how DISH actually allocates its precious spectrum. Figure 66 shows the frequency band between 17.3 and 17.8 GHz, our FCC-authorized uplink region. We divide that frequency range into 16 segments separated by guard bands. Each of the segments is 26.16 MHz wide. The lower frequency guard band is 10.92 MHz wide and the upper guard band width is 25.52 MHz. The guard bands between the signal segments are 3 MHz each. It's a pretty efficient way to use our spectrum, but not the most efficient way. What if I told you we actually had 32 transponders on our satellites, with each transponder being capable of handing one of our 30 MHz segments? Why would we need twice the number of transponders as frequency segments? Why do all of the transponders we've labeled in the figure have odd numbers? Why is there such a large guard band on the right side of the figure? Well, it's because we've not yet told you the whole story, and the rest of the story will have to wait for just a little bit until we discuss the concept of polarization.

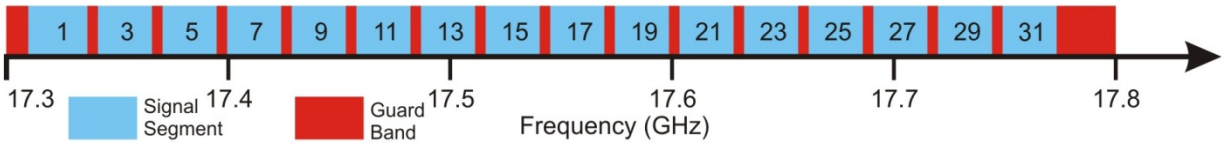

Figure 66: The DISH Uplink Spectrum.

Radio Versus Visible

But before we move to polarization, we've got one final bandwidth consideration to investigate. Since the whole point of discussing data and spectral bandwidths is to show how we can use both concepts to maximize our data rate, we need to emphasize one concept that may seem a little confusing. *It's the spectral*

bandwidth of the content, the modulating signal, that is the limit on the data bandwidth that can be carried by a wave. That bandwidth remains the same regardless of the frequency of the carrier wave. If a telephone call with a certain data bandwidth takes up 4 kHz of spectral bandwidth on a 10 MHz carrier signal, it will still take up 4 kHz of spectral bandwidth on a 10 GHz carrier.

However, an important measure of how efficiently a carrier uses that spectral bandwidth is the ratio of the two numbers. In the first case above, 4 kHz/10 MHz = 0.0004 ($4x10^{-4}$), while for the second case 4 kHz/10 GHz = 0.0000004 ($4x10^{-7}$). The smaller that number, the more efficiently the range of frequencies around the carrier can carry those modulations. By *efficient use*, we mean how many different channels a single transmitter, designed to deliver a certain fixed range of frequencies, can carry.

Let's look at a hypothetical example. In this example, a parabolic radio antenna like we use at DISH uplink centers might be designed to carry signals across the entire Ku band, from 12-18 GHz, a 6 GHz spread. Since a HDTV channel requires about 30 MHz of bandwidth, the maximum number of channels that antenna could possibly transmit side-by-side (were the FCC to decide to allow that to happen) would be 6 GHz/30 MHz = 200. The Ku band takes up about 2% of the radio spectrum (0-300 GHz).

Let's say we had a device that was able to transmit across 2% of the *visible* spectrum, which runs from 400-789 THz, as shown in Figure 67. 2% of this range equates to 7.78 THz. Note that 7.78 THz is only a fraction of any of the spectral widths of the colors shown. Since the information content of our HDTV channel only depends upon the spectral bandwidth, it stays at 30 MHz. However, this optical device could theoretically carry almost 260 *thousand* channels (7.78 THz/30 MHz) side by side! Now, practical considerations

Color	Frequency Range (THz)	Spectral Width (THz)
Red	400-484	84
Orange	484-508	24
Yellow	508-526	18
Green	526-606	80
Blue	606-647	41
Indigo*	647-673	26
Violet	673-789	116
Visible Spectrum	400-789	389

Figure 67: Frequencies Ranges for the Visible Colors.

* Note that indigo is not normally recognized by scientists as a division of the visible spectrum. Instead, the color blue runs from 606-668 THz and violet runs from 668- 789 THz.

don't yet allow us to make an optical device that can transmit at high power with good frequency selectivity across that wide a range of the spectrum, but we're only using the idea to show how much more efficiently we could use the visible spectrum were it possible.

In addition, returning to diffraction limitations, using optical frequencies would allow us to use much smaller antennas to get the same directivity for our beam (remember the equation for beamwidth said to make it smaller we need to make the dish larger or the wavelength smaller, and smaller wavelengths are the same thing as higher frequencies). Conversely, were we to make the same size antenna (or lens/mirror, since we're talking about visible light), our directivity would be phenomenal—less than 3 *millionths* of a degree—small enough that we'd actually have to worry a lot about aiming in order to hit our satellites since our beam would be about 300 feet across trying to hit an antenna that's about 5 feet across at a range of well over 20,000 miles. The upside, were we able to aim that well, is that a much higher fraction of our transmitted power would actually hit the satellite than is the case with our Ku-band operation.

So why, if higher frequency gives us such great advantages in beamwidth and data bandwidth, don't we move DISH operations to the optical window where absorption is just about as low as in the radio window? Well, the answer to that final question comes from Figure 34, reproduced below as Figure 68 for your convenience. In our business, we need to be able to get our product to our customers 24/7/365. If we provided much less availability than that they would quickly turn to a competitor. Look at the yellow region in the figure. It shows the additional atmospheric absorption in the presence of visible moisture—clouds and fog. Notice that the optical window is no longer a window—something you're very aware of when you can't see the stars or even the Sun when it's

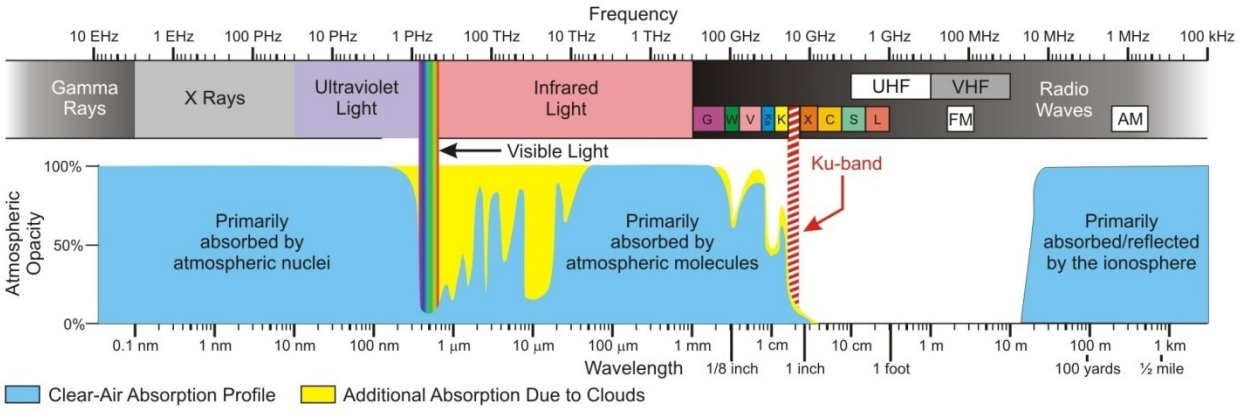

Figure 68: Atmospheric Transparency (revisited).

cloudy outside.

On the other hand, the radio window is barely affected at all by clouds. That's the ultimate reason for the radio versus optical window choice. We just can't get light through clouds and atmospheric turbulence effectively enough at the present time to make optical communication to satellites suitably reliable for a commercial operation. That's not to say that people aren't trying. Recently, there have been a number of reasonably successful experiments using high-power, very short pulse lasers that have been able to penetrate moderate cloud cover and still transmit significantly greater data bandwidth than the best radio frequencies can attain.[43] However, commercial applications reliable enough to use in our line of business still appear to be many years away.

So we've now covered the major reasons why DISH uses the Ku band for our satellite operations. First, it's in one of two regions of the spectrum where the atmosphere is transparent to electromagnetic radiation, and we're in the only window that's not affected by clouds. Second, we're about as high up in the radio window as we can get, meaning our wavelength is as short as possible so our antennas can be as small as possible because of diffraction. And finally, we're using a part of the spectrum that we've leased from the FCC—we can't legally operate anywhere else!

Polarization

Before we end this chapter, we need to return to one last topic on electromagnetic waves. We know we promised earlier that we were done with that subject, but you needed to understand the concept of phase before we discussed polarization. You also needed to see that we actually have twice as many transponders on our satellites as we seemingly can use before you saw the need for this discussion.

DISH uses right- and left-hand circular polarizations (RCP and LCP) to communicate with our satellites, effectively doubling the amount of information we can squeeze into our limited spectral bandwidth. That information can be found in many of our public and internal documents, but what does it really mean?

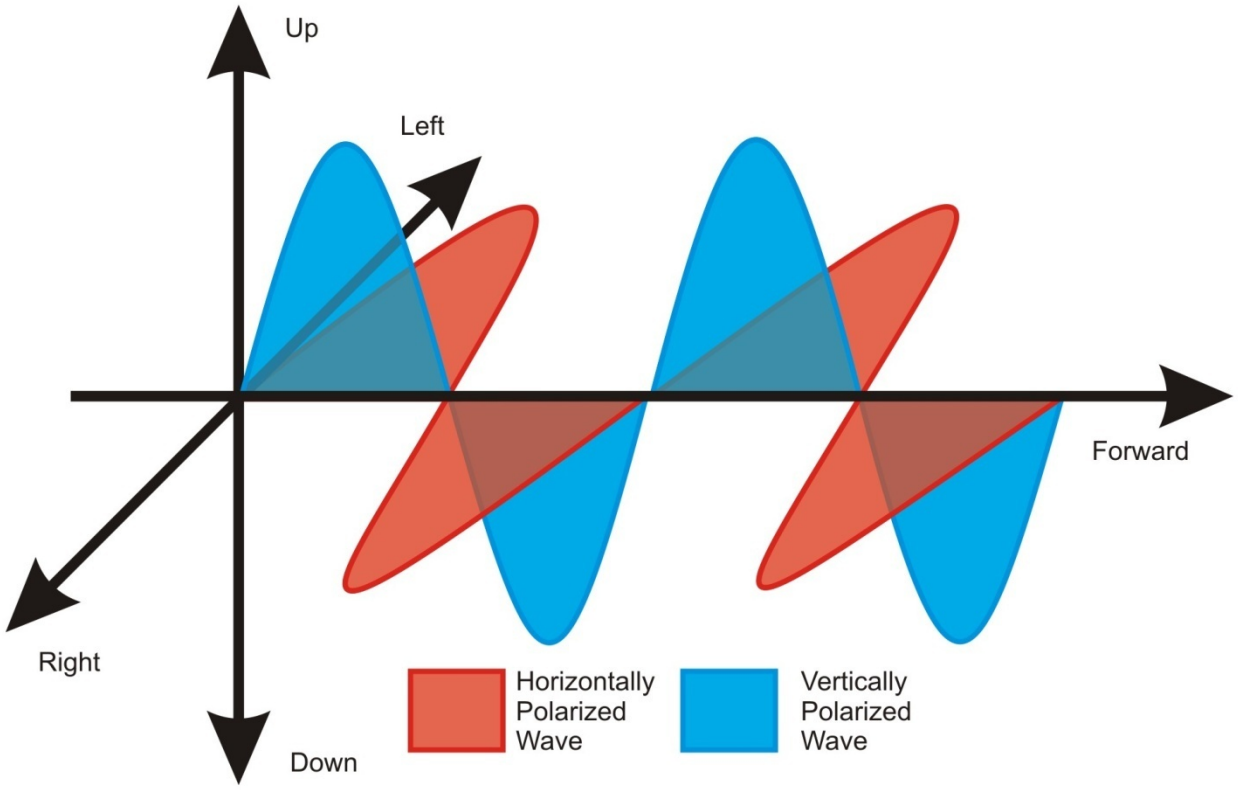

Figure 69: Horizontally and Vertically Polarized Waves.

Before we get to circular polarization, let's look at the simpler but related case of linear polarization. Figure 69 shows two waves, one shaded red and one shaded blue, traveling to the right. Notice that the angled axis is labeled left and right from the point of view of looking at the waves along their direction of travel. The blue wave only oscillates up and down; we call this wave *vertically polarized*. Conversely, the red-shaded wave only oscillates right and left; this one is *horizontally polarized*.

So why do we care about polarization? Because we can use the concept of polarization to double the data bandwidth of our signal. We can send one signal on the horizontally polarized wave and a completely different signal on the vertically polarized wave! But how do we separate them at the receiving end? Simple. You've all probably seen or worn polarized sunglasses at one time or another. We simply use a polarized filter similar to the lenses in those glasses to separate the signals. Figure 70 shows this concept.

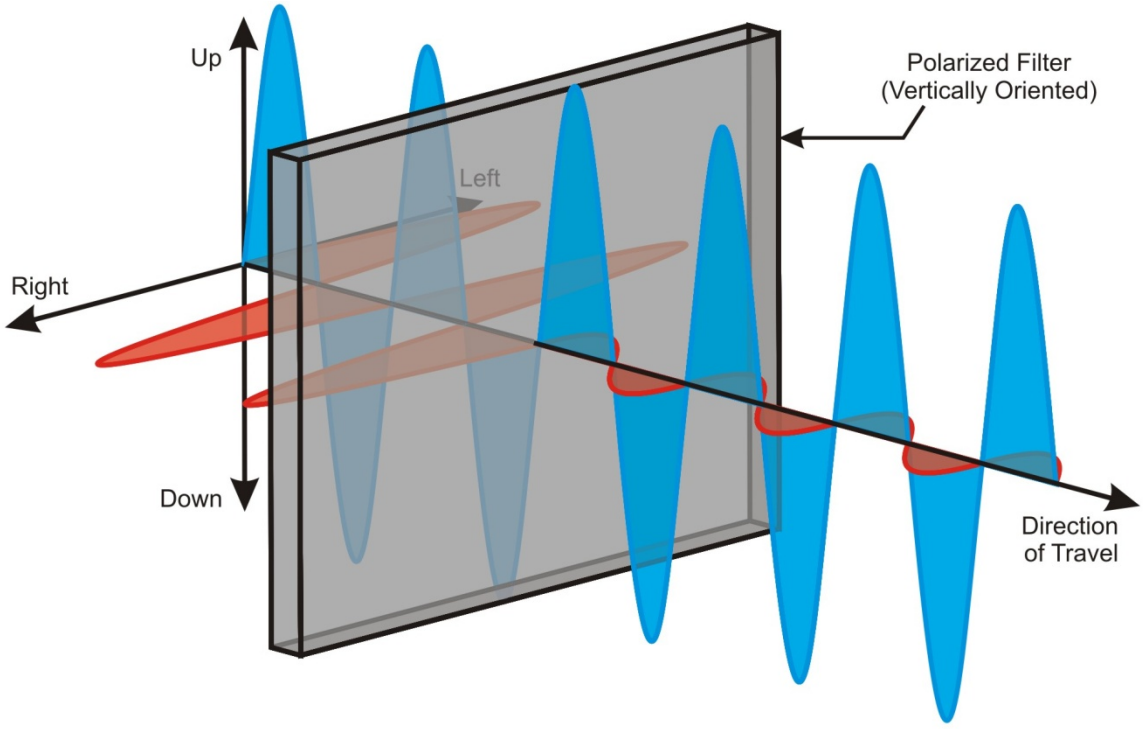

Figure 70: Separating Signals Using a Polarizing Filter.

Two polarized waves, red and blue, move to the right in this figure. When they encounter the polarized filter, oriented to preferentially pass vertically polarized waves, the blue wave passes through almost unscathed while the amplitude of the horizontally polarized red wave is almost extinguished. Were we to rotate the filter 90°, the red wave would pass through and the blue wave would be killed off. With an orientation somewhere in between, both waves would be diminished to varying degrees, depending on whether the filter was closer to vertical or horizontal. The red wave that has passed through the filter in the figure contributes to the noise level in the system, but while we can reduce its strength with better (and more expensive) filters, it's impossible to eliminate it completely.

However, DISH doesn't use *plane polarized waves*—waves that only oscillate in one direction (up-down or right-left). We use *circularly polarized waves*, and for good reason. What do you think would happen if we installed our receiving antenna a little out of alignment? We'd end up reducing the signal that we've paid so much to transmit, and when we're talking about the powers and distances required to get to our satellites, even a small loss can result in poor signal quality and lost customers.

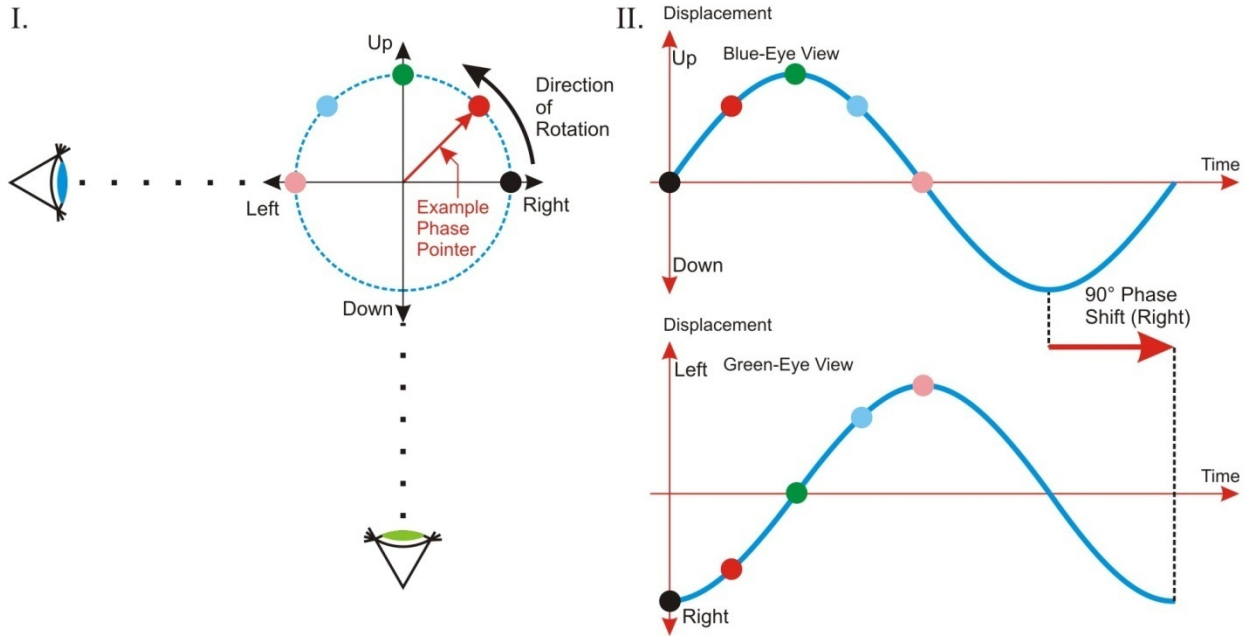

Figure 71: Two Phased Waves for Left-Hand Circular Polarization.

Circularly polarized waves are just a combination of two plane polarized waves with a special phase relationship (now is where the concept of phase comes into play, the one we needed before we could understand circular polarization). Look at Figure 71. Frame I shows a circle with five colored dots on it. It's the same concept we used when first explaining phase way back in Figure 55 (p. 73). Imagine string connecting the ball to the center of the circle now has an arrow pointed on it. We'll call that arrow the *phase pointer*. The phase pointer starts pointed at the black ball, then the red, then green—it's basically a rotating arrow with its base at the center of the circle and the tip moving around the circle in the counterclockwise direction, carrying the ball with it. The ball and arrow change color in this figure just to help us see where they are at different times.

We're first going to imagine that we're looking at this circle from the left, using the blue eye. Notice that as the colored ball moves from its black starting position at the right side of the circle to its pink position on the left and back again, the blue eye perceives the ball as moving up and down in a sinusoidal motion in the vertical plane, as shown in the top part of Frame II.

The green eye, at the bottom of Frame I, sees things a little differently. It sees the ball start at the far right and then describes a sine-wave motion to the left and back again in the horizontal plane, as shown in the bottom part of Frame II. However, this motion is phase-shifted 90° to the right when compared with the blue eyed view. Both eyes see sinusoidal motion, but with a 90° phase shift

between them. When these two motions are combined, taking into account both the horizontal and vertical planes, the two oscillations perfectly describe the circular motion of the ball as it moves in the counterclockwise direction.

Imagine now that we're looking at the vertical and horizontal planes along the direction of motion of the waves from Figure 69 (p. 91), that is, the waves are moving away from us. We're going to define this combination of right-left, up-down oscillations as left-hand circular polarization because if we were to take our left thumb and point it in the direction of travel of the wave, the fingers of our left hand would curl in the direction the ball is rotating. Try it yourself!

The big point to get out of this discussion is that a combination of two *straight-line* sinusoidal oscillations at right angles to each other with a 90° phase shift between them can create *circular* motion. The phase shift between the two curves is a critical component for this result.

What does this circular motion look like in three dimensions? Figure 72 shows it. The red solid line describes a spiral around the direction-of-travel axis while the green arrows show the phase arrow of the wave timed at every 22.5° (1/16 of a circle). Seen from directly behind, it would look just like the rotating circle from Figure 71.

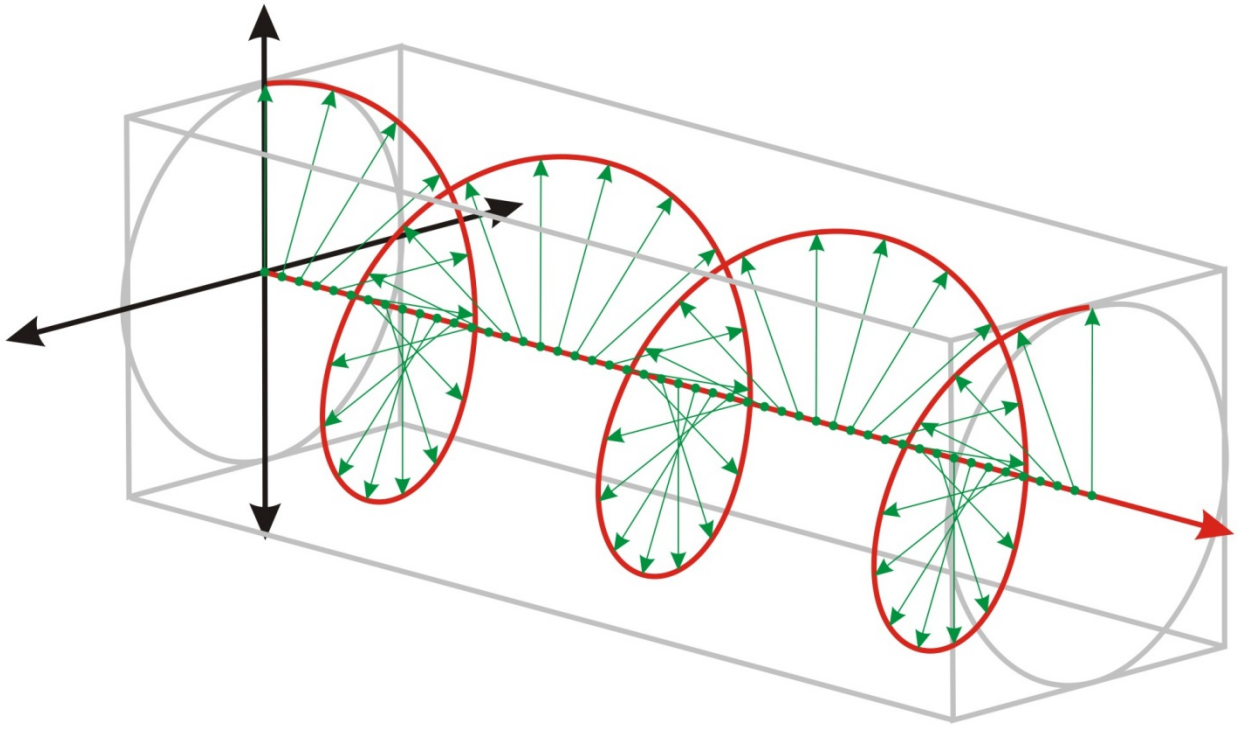

Figure 72: Circularly Polarized Wave in Three Dimensions.

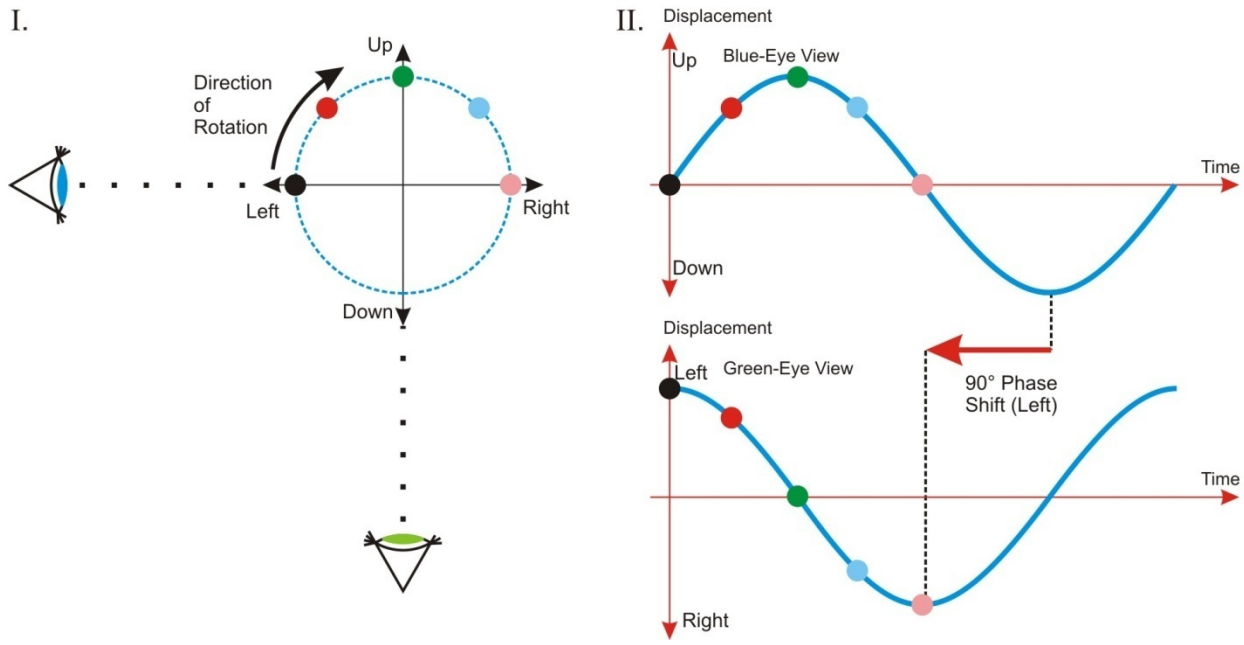

Figure 73: 180° Phase Shift Gives Right-Hand Circular Polarization.

A right-hand circularly polarized wave is exactly the same thing, but instead of a 90° shift to the right, the left-right wave is shifted 90° to the left. Figure 73 shows how this works.

While not as easy to draw as the plane wave polarizing filters we showed in Figure 70, it's possible to make filters that only pass RCP or LCP waves, and that's how we use circular polarizations when transmitting our signals up to the satellites. We'll show you how those filters work later in *The Dish on Your Roof* chapter.

So now that we understand polarization, we can return to the questions we posed earlier about why we only showed odd transponders on the DISH uplink spectrum allocation. The odd transponders receive RCP signals, while the even transponders receive LCP signals, effectively doubling the amount of data we can send up over our precious spectral allocation. This doubling is shown in Figure 74.

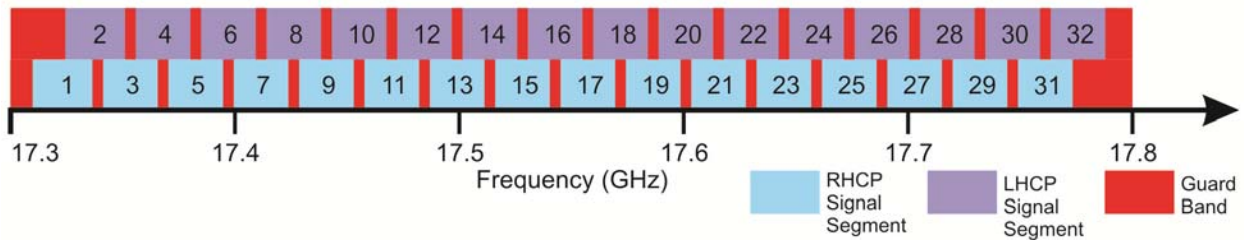

Figure 74: DISH Uplink Transponders.

Chapter Summary

In a nutshell our business is data delivery. We've got a limited amount of spectrum we can use to deliver that data. We're doing all we can to squeeze as much out of that spectrum as we can to get just as high a data bandwidth as possible. We're using high-power transmitters to increase our signal-to-noise ratio; we're using 8PSK to encode our signal, allowing us to send three bits per symbol; we're minimizing the spectral bandwidth of each channel by only modulating our carrier signals by the minimum required for HDTV channels (30 MHz); we're stacking our carrier signals just as close together as we can in our allocated spectrum without allowing them or their sidebands to interfere with each other; and we're using two non-interfering kinds of polarization to double the number of channels we can send within our FCC allocation.

We're also doing this in the Ku-band, which is just about the best place we can get in the electromagnetic spectrum to guarantee us weather-independent access to our satellites through an atmosphere that absorbs most radiation while simultaneously minimizing the size requirements of our transmitting antennas.

A lot goes on at the uplink center, but much of it is designed in from the start because of requirements derived from basic physics and atmospheric science, constraints we've discussed at length in this chapter. We're betting after reading this section you'll never see another satellite uplink dish in the same way!

3. DISH Satellites

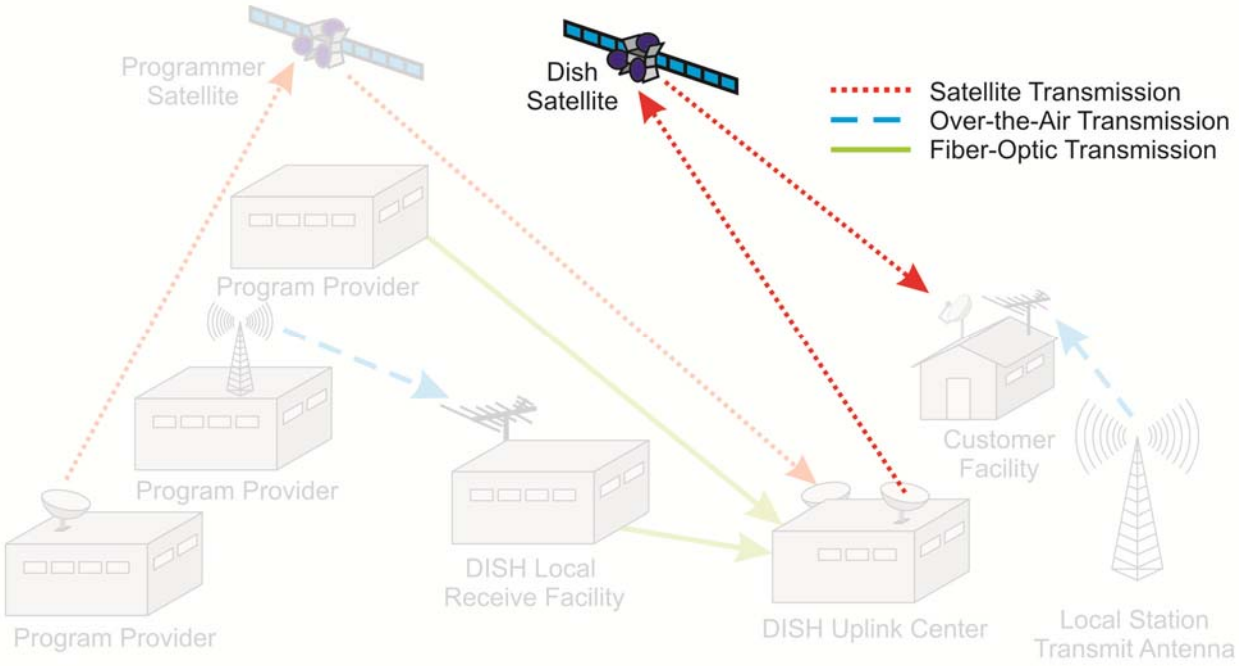

Figure 75: The Signal Flow Roadmap—DISH Satellites.

Our satellites sit
In fixed spots quite far away
Transceiving signals.

The Basics

(Continued from p.39) The whole point of the uplink center is to transmit a radio wave containing DISH entertainment and information to our satellites in space. So let's take a look at what happens when that signal actually gets to our satellites.

Did we say *our* satellites? To be completely candid, DISH does not own all of the satellites we use. We own a few and our sister company, EchoStar, owns and leases the rest of the satellites that carry our signals. The fleet is operated by a small staff of EchoStar employees co-located with the uplink centers in Cheyenne

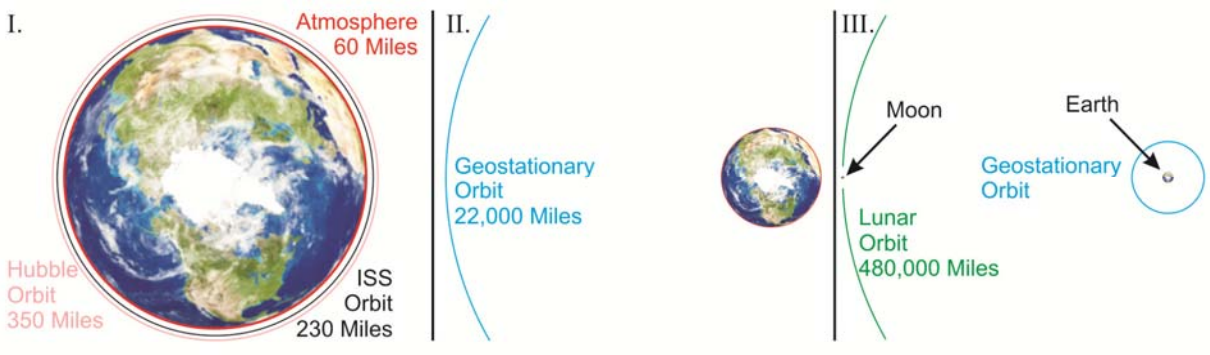

Figure 76: Geostationary Orbital Distances in Perspective.[44]

and Gilbert. For the purposes of this document, though, we'll usually refer to the entire fleet as DISH satellites.

Satellite Orbits. DISH satellites are really far away in the **geostationary** belt.* Figure 76 shows a few orbits to give you an idea of just how far *far* is. Frame I shows the Earth looking down on the North Pole. We need to use a polar view like this since geostationary satellites orbit above the Earth's equator. The red band around the Earth represents the thickness of the atmosphere, about 60 miles deep (and that number is really stretching it—it's a commonly used definition of where space begins, but no aircraft currently fly higher than about 15 miles because the air is too thin even at that relatively low altitude). The black circle shows the orbit of the International Space Station (ISS), about 230 miles up. The pink circle shows the altitude of the Hubble Space Telescope, about 350 miles above the Earth's surface; it's about as far away as the Space Shuttle ever got from Earth.

We change scales in Frame II to show just how large a geostationary orbit, the blue arc, is. At this scale, the atmosphere and the ISS and Hubble orbits are all but invisible. Because of Isaac Newton and physics, geostationary satellites all must orbit 22,236 miles above the Earth's surface, directly above the equator. We'll get into the reason for that in the *Technical Discussion* later, if you're interested.

Finally, we'll expand the scale once again to give you a little more perspective on the size of our planetary neighborhood. Frame III shows just how far away the

* Many people refer to this belt as the *geosynchronous* belt. While all geostationary satellites are also geosynchronous, the reverse is not true. Geosynchronous means a satellite orbits the Earth in 24 hours. Geostationary means a satellite orbits the Earth in 24 hours and does not appear to move when viewed from the Earth's surface. Such orbits can only be circular and must lie in the plane of the Earth's equator. See the *Technical Discussion* for a more complete explanation.

Moon is compared to geostationary distances. The Moon's orbit is about 480,000 miles in diameter. At this scale the Earth, with a diameter of almost 8,000 miles looks pretty small, and the Moon, with a diameter of about 2,200 miles, is just a speck in between the two green lunar orbital arcs.

So, let's get back to the original problem for this chapter, getting our signal to our satellites. The first thing we have to do is find the satellite. That's actually a pretty easy thing to do, even with them being so far away. If you've ever looked at a very large parabolic dish antenna used for satellite communication, there's one thing that becomes very obvious very quickly: they're boring. They don't move. Ever. They aren't like the air traffic control radars that turn round and round every couple of seconds. They just sit there. And that's a good thing. Being stationary makes them much cheaper than they'd be if they actually had to point at a moving object. Since the antennas don't move, it's a pretty good guess that the satellite they're aimed at doesn't move either. And from the point of view of the uplink center's antenna, the satellite really is stationary. But that's not the whole story.

Pick up a rock and drop it. It falls straight down. Pick it up higher and drop it and it still falls straight down. Take it to the edge of a tall cliff and drop it and it still falls straight down. If you were able to take it up over 22,000 miles in altitude and drop it, *it would still fall straight down*. And that fact starts to look like a paradox: how *do* those satellites stay in one place without falling out of the sky?

The answer is somewhat surprising. They're actually moving. Really fast, in fact. They move at almost 7,000 miles an hour in a circle around the Earth, a circle that's almost 52,000 miles across. But at that speed and at that distance from the Earth, the satellites take 24 hours to go around that circle (why that is will be discussed in the Technical Discussion below), exactly the same time that it takes the Earth to turn once on its axis. Since they take 24 hours to go around and the Earth takes 24 hours to go around, they *appear* to be motionless in the sky.

This very special kind of satellite orbit is known as a *geostationary orbit*. As you can imagine, lots of people want to put satellites up at the geostationary altitude since it lets them use cheap, fixed antennas on the ground. Much like the way we saw the FCC licensing the use of the electromagnetic spectrum in the previous chapter, the International Telecommunications Union licenses the use of the geostationary belt. There are hundreds of satellites in the belt, but the licensing process keeps them far enough apart they don't crash into each other and their incoming and outgoing radio waves don't interfere with each other. In fact, the

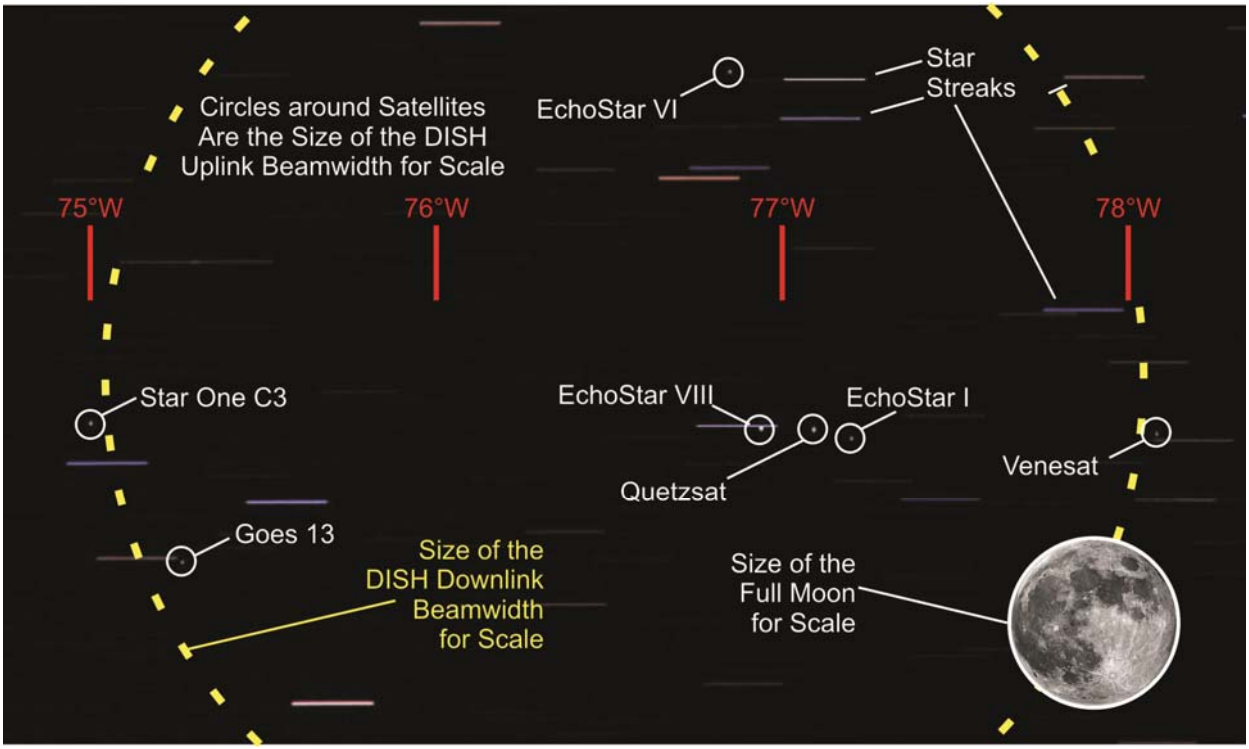

Figure 77: Image of Satellites in Geosynchronous Orbit against a Background of Star Streaks.[45]

numbers you may have heard being related to our satellites—61.5, 72.7, 77, 110, 119, and 129—actually refer to the longitude on the equator the satellites are licensed to park over. Figure 77 shows a long-exposure image taken with a 200 mm telephoto lens of a few satellites in geostationary orbit. The camera was fixed, so during the course of the 1-minute exposure the stars appeared to move about ¼°. When things move during a picture, they blur, so the "moving" stars appear as horizontal streaks in this image. In contrast, the small dots in the image are the satellites. They do not appear to move with respect to the fixed camera—they're moving at the same rate as the Earth's rotation.

Satellite Communications. So we've now figured out where the satellites are and can presumably do a little geometry so we can point our transmitting antennas at them. The signal can now get to the satellite. What happens when it gets there? We need it to bounce back down to the Earth. In fact, the very first communications satellite, Echo 1*, did exactly that: bounce signals back down.[46] Launched in 1960, Echo 1 was essentially an aluminized balloon that acted just like a mirror to reflect radio waves back down from space all the way across the

* Note that EchoStar frequently shortens the names of their satellites in informal writing to "Echo 16," "Echo 14," etc. The Echo 1 we're discussing here is not the same as the satellite EchoStar might refer to as "Echo 1."

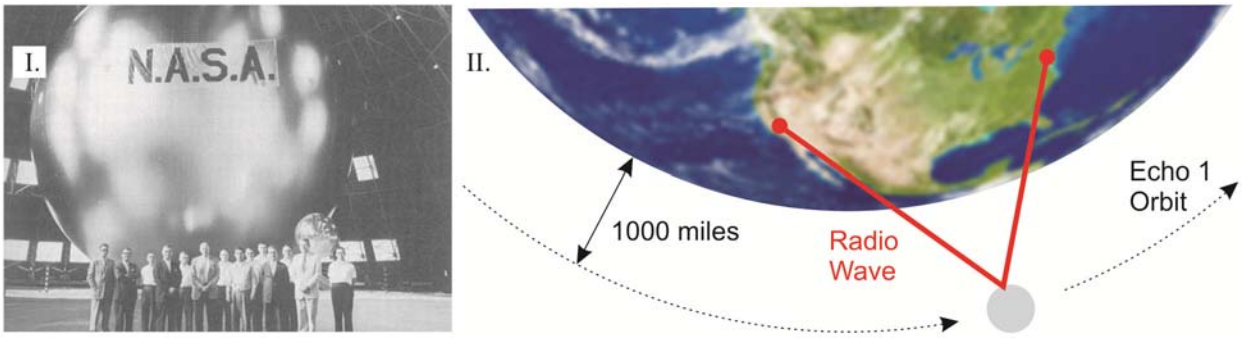

Figure 78: Bouncing a Signal off Echo 1: A Very Basic Bent Pipe.[47]

United States, as shown in Figure 78. Being able to do that in 1960 was a miracle; we take it for granted now.

But bouncing a signal like we did with Echo 1 isn't very effective because only a very, very small fraction of the signal NASA sent up actually made it back down to the place they wanted to receive it. How can we do better? If we could collect more of the signal, amplify it, and retransmit it, we'd probably be a lot better off. And in fact, that's exactly what our satellites do.

Why do we need to amplify the signal? Well, by the time it travels those tens of thousands of miles, it's a lot weaker than it was when it left the uplink antenna. In fact, seeing the uplink center from space would be like looking at about 30 60-watt light bulbs from 25,000 miles away. And don't forget that the signal is divided up into 32 transponders, so really each transponder's "brightness" is about like a single 60-watt light bulb. To say that's not very bright is a huge understatement.

In order to collect enough signal to be useful, we use a large parabolic dish antenna much like the one we used to transmit the signal. It's nowhere near as large, since we have to fit it onto a rocket to blast it into orbit, but the physics behind it is the same. The receive antennas on our satellite are about 12 feet across. Those large antennas collect a lot of the signal we worked so hard to aim at the satellite and concentrate it at the parabola's focus, where we've conveniently placed a detector. However, even with all of the antenna's collecting area, only a few *millionths* of a watt of the 2,000 we transmitted actually gets collected. The detector then directs that tiny signal into a transponder. A transponder is an electronic device that takes the signal, amplifies it, shifts it to a different frequency, amplifies it again, and directs it to one or more of the many transmit antennas on the satellite.

We mentioned shifting the frequency. Why do we have to do that? Remember how weak the uplink signal was when it got to us? Well, our satellites' transmitters have powers of a few hundred watts, millions of times stronger than the received signal. If we used the same frequency to downlink our signal through the transmit antennas, our transmitters would bleed over into the receive antennas and drown out the weak uplink signal. By shifting the frequency, we minimize the interference from our transmissions. We noted in the last chapter that we were licensed by the FCC to uplink signals to our satellites in the 17.3 to 17.8 GHz band. We're similarly licensed to use the 12.2 to 12.7 GHz band to send them back down to Earth.

Once they've gone through the transponder, the signals go to the transmit antennas. Our satellites typically have two basic sizes of transmit antennas: bigger ones that are about 12 feet across, and smaller ones that are about 8 feet across. Recalling what we learned in the last chapter, the smaller antenna produces a wider beam. We use this antenna to send signals across a very wide area—the entire continental United States (CONUS). The larger antennas are used to send signals to smaller regions, something you'd want to do with things like local channels. That concept is shown in Figure 79.

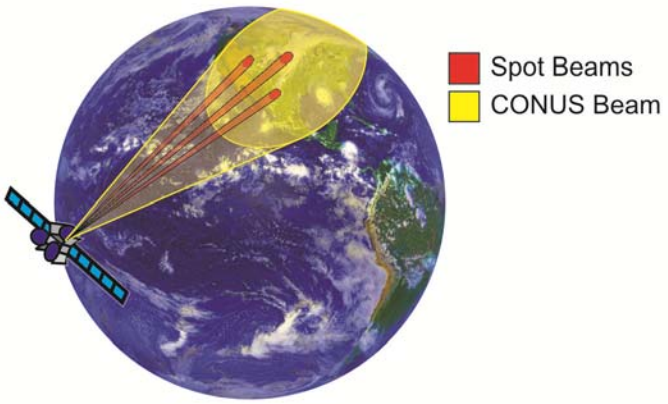

Figure 79: Multiple Transmission Beams from a Single Satellite.

That's really the big picture of what our satellites do. They take a weak incoming signal, amplify and shift its frequency, and send it back down to Earth.

From this basic discussion, you should remember

- We have seven primary satellites for broadcasting all DISH television signals
- The satellites are in geostationary orbit (about 22,000 miles above the Earth)
- Geostationary satellites orbit the earth every 24 hours, so they appear not to move in the sky
- Not moving in the sky means we can use cheap, fixed antennas on our customers' facilities

- The satellites take the incoming signal from the uplink center and shift its frequency so that it doesn't interfere with the uplink frequency before they send it back down to Earth
- We use two different kinds of beams to broadcast our signals: CONUS beams for stations that need to go to all our customers and spot beams for local content

If you're only reading *The Basics*, you can continue your study on page 131. However, we do encourage you to dip your toes into the waters of the technical discussion below. We promise it won't be as dense as the one for the last chapter!

Technical Discussion

The devil is in the details, as they say. Yes, the big picture of what our satellites do for us is really pretty simple. However, there's a lot that goes on behind the curtain to make that "simple" stuff happen. In the remainder of this chapter, we'll look at some of that magic in a little more detail.

A Primer on Satellite Orbits

DISH signals pass through satellites that orbit over 22,000 miles above the equator—almost three times as far away as the Earth is across. Did you ever wonder why they can't be closer? Or why they can't sit over, say, Denver? Being able to have your own satellite nearby and right overhead would solve a whole lot of signal problems, from significantly decreasing the power we'd have to transmit to reducing the delay it takes between transmitting and receiving signals. It would also make it a whole lot cheaper to launch the satellites and to build them, since we wouldn't have to pay for the energy to get them all the way out to geostationary orbit (GEO) and wouldn't have to size the antennas and transmitters to handle those huge distances.

Gravity. Unfortunately, there's a branch of physics that dictates where satellites can go, and where it allows them to be isn't always the most convenient to those of us who want to make money from them. This section will go into some detail on the physics behind satellite orbits, again, without a lot of math. However, by the end of this section we think you'll be very comfortable understanding what can be a relatively confusing subject: orbital mechanics.

Before we go much farther, we need to talk about gravity, since that's the driving force behind everything related to satellites. It's a common misconception that there's no gravity in space. It's easy to see why many of us believe this "fact" to

be true—who hasn't seen pictures of astronauts cavorting in space, throwing fruit and droplets of water to each other while floating all around inside their vehicles? Gravity does have a lot to do with their abilities to do all of those things, but not in the way you might think.

Gravity is everywhere. The Earth's gravity keeps you on its surface. It also keeps the Moon in orbit (aided by the Moon's own gravity). We've already seen that our astronauts on the space station are much, much closer than the Moon. So why, if the Earth's gravity affects the Moon, would it not affect our astronauts?

Let's look at a simplified version of the equation that Sir Isaac Newton came up with to describe gravity. He said $F_g = k/d^2$, where F_g is the force of gravity, k is a constant that depends on, among other things, the two masses in the system that are attracting each other, and d is the distance from the center of mass. Let's look at the case of a system consisting of the Earth and an astronaut. The astronaut's mass is so insignificant compared to the Earth's that the center of mass of the Earth-astronaut system is essentially the center of the Earth. That means that the only variable we have to play with to see what happens to the force of gravity is d, the distance the astronaut is from the center of the Earth.

When the astronaut is standing at sea level, he is one Earth-radius away from the center of the Earth, about 3,963 miles. Since weight is just a measure of the pull of gravity on an object, let's say he weighs 150 pounds there. Should he go to Denver, he's now a mile further from the center of the Earth. Looking back at Newton's equation, since d got bigger, F_g must get smaller. And in fact it does.

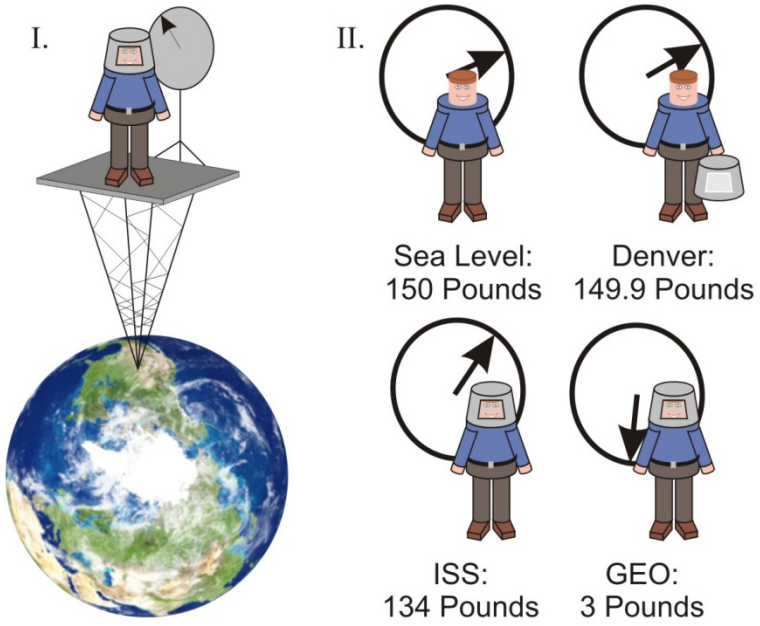

Figure 80: Gravitational Pull at Various Altitudes.

Let's examine the obviously fictional case of an astronaut standing on a *stationary* scale that measures weight. To make such a scale, let's suppose we can build a platform to whatever height we need to support it. That situation is shown in Frame I of Figure 80. In Frame II, you can see the astronaut does in fact

weigh 150 pounds at sea level, but when he moves the scale to Denver, he loses about 1/10th of a pound. Now, let's move him up to the altitude of the ISS, 230 miles above sea level or 4193 miles from the center of the Earth. There, he's lost a bit more weight because gravity isn't pulling on him as much. But notice that he doesn't weigh zero, the answer you'd expect from having seen astronauts floating in space. Were we to move him all the way out to geostationary orbit, over six Earth radii out, he now only weighs less than 1/36th (one over six squared) of his original weight, or about 3 pounds. But still he doesn't weigh zero. In fact, he'd still weigh *something* even if he went an almost infinite distance away.

So who is wrong? "Ah," you say. "Sir Isaac, I've finally got you! Your silly equations don't work!" Unfortunately, they do, but not in the way you'd think. Let's look at another example of weight that you'll be familiar with: riding in an elevator. Figure 81 shows a person in an elevator at various stages of the ride. In Frame I, he's just gotten on and the elevator is standing still. The scale in the background shows that he's at his normal weight of 150 pounds. Once the door closes and the elevator starts down, though, he feels a little light. A glance at the scale in Frame II shows that he's indeed lost a bit of weight. That's because the elevator is being accelerated downward away from his feet, as shown by the red arrow, and he feels less force on them. He hasn't lost any of his "*real*" weight,[48] but it sure feels to him like he has. Things go very badly for him in Frame III. The elevator cable breaks and he begins to free fall down toward the bottom floor, as indicated by the longer red arrow. His last living memory is a look at the scale behind him which now registers zero. Again, his "real" weight didn't change, but the scale and he both thought it did. And therein lies the answer to why our astronauts appear to float. As shown in Frame IV, they're in a perpetual state of free fall!

But if they're falling, why don't they hit the Earth? It's because they've got motion in *two* directions. They're falling *straight down* toward the Earth's

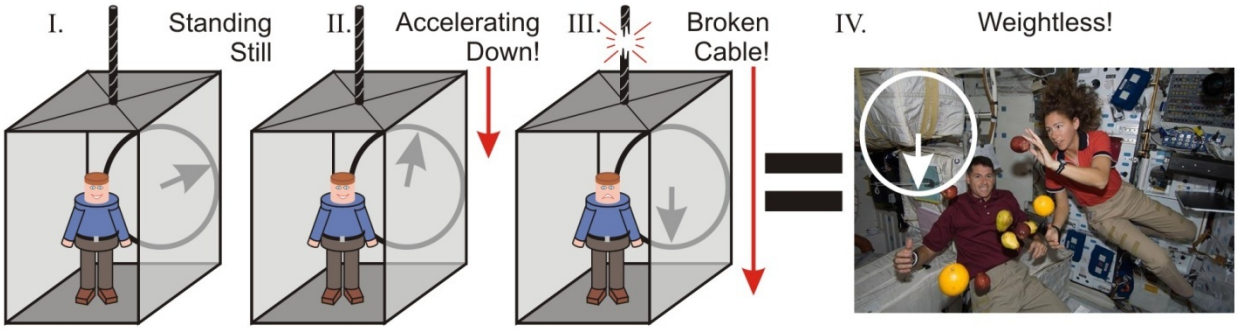

Figure 81: Weightless = Freefall.[49]

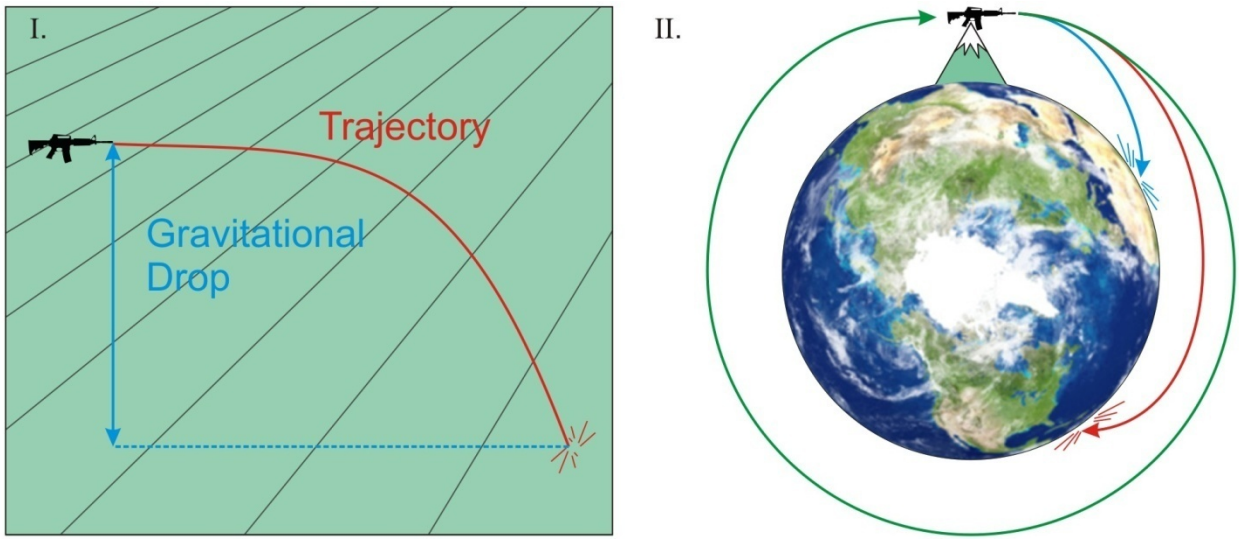

Figure 82: Falling Projectiles.

surface, but they're also moving very quickly *parallel to* the Earth's surface at the same time. Perhaps another illustration might help explain this point.

Frame I of Figure 82 shows a rifle firing a bullet horizontally across a flat plain. Let's assume for a minute that there's no air, so there's no drag on the bullet. Even so, the bullet won't go forever. It is falling out of the sky at the same rate as it would have had it been dropped straight down. Hard to believe, but it's true, as is proven every semester in basic physics classes around the globe. Gravity acts on the bullet whether it's in flight or not, and pulls it down toward the Earth's surface.

Frame II of the figure shows the same rifle mounted on top of a very tall mountain. In this case, the rifle has been modified to shoot its bullets with higher and higher speeds. The first bullet it fires, represented by the blue line, starts off horizontally and falls, just like in Frame I. However, this bullet is going fast enough that we have to start taking the Earth's curvature into account. The Earth's surface curves away from the local horizontal (horizontal from the point of view of the shooter). Eventually, the bullet's rate of fall outpaces the rate at which the curvature is taking the surface away from it, and it impacts the Earth. The red line shows another bullet, also fired horizontally, but this one has an even greater speed. It takes a lot longer for that bullet to impact the Earth, but eventually it does, too. The green line, however, shows a bullet fired with so much speed that even though it's continually falling toward the Earth, the rate at which it loses altitude is exactly countered by the rate at which the Earth's surface curves away from it. At this speed, it falls forever, continually circling the Earth at

the height of the mountaintop from which it was fired. (The shooter had better duck!)

This concept is exactly what happens with an orbiting satellite or our astronauts. They are all falling together but they never hit the Earth because they have just the right speed to match the Earth's curvature. So let's think about what we've learned here: at the altitude of the ISS, our 150 pound astronaut still would weigh about 134 pounds if we could put him on a *stationary* scale way up there. With no horizontal motion, he'd fall like a rock if he stepped off the platform regardless of whether he started at 5 feet, one mile, or 230 miles. No horizontal motion, no orbit, and boom when he hits the ground! If he properly matches his horizontal motion with his altitude, however, he can fall forever. And as we've seen, falling feels a lot like being weightless. The bottom line of this discussion is that *satellites cannot hover above an arbitrary spot on the ground.* They absolutely have to move to stay up there.

But that fact seems to fly in the face of another fact we think we understand: we point the DISH antennas on our roofs at a specific point in space where we know the satellite will be. If satellites can't hover, if they can't stay in one spot, how can we do that?

Orbital Velocity. The point of this digression about gravity is to get you to realize that every orbit has a special velocity that will keep it at the same altitude, preventing it from ever hitting the Earth's surface. Sir Isaac quickly realized that he could calculate these orbital speeds by combining his famous second law and his law of gravity. We won't go into the details here, but the end result is that the speed equation is basically $S = k/d^{1/2}$, where *S* is the required orbital speed, *k* is our old friend the arbitrary constant, and *d* is the distance from the center of the Earth to the orbit. Don't worry about the one-half power (*d to the one-half* is how you'd say it)—it just means that as d gets bigger, $d^{½}$ gets bigger, too, just not quite as quickly. Anyway, what happens to the required orbital speed as you get higher? That means d gets bigger, so $d^{½}$ gets bigger, so the right side of the equation gets smaller, so S gets smaller (whew! That was a lot of <u>so</u>s). The higher you go, the less speed you need to stay in a circular orbit.

What kind of speeds are we talking about? Well, the ISS travels at about 17,500 miles an hour. That's fast enough to go from Denver to Colorado Springs in about 12 *seconds*. At that rate, it takes just about 90 minutes for the space station to go all the way around the Earth. When we move up to geostationary altitudes, the required speed goes way down (bigger d, smaller S). Geostationary satellites

"creep" along at *only* about 6,800 miles an hour. Not only is the speed slower than the ISS', but the circle a GEO satellite has to travel to go around the Earth is a lot bigger. At that speed—the speed *exactly* matched to the 22,000-mile-high orbit—a satellite in GEO takes almost exactly 24 hours to go around the Earth.

Orbits. Now, that seems like a strange coincidence, doesn't it? The satellite takes 24 hours to go around the Earth and the Earth takes 24 hours to spin on its axis, as shown in Figure 83. Hmmmm…maybe that's why the satellites appear to hover, why we can point our dishes to a spot in the sky and know our satellite will be there! The satellite is going around the Earth with exactly the right speed so that it matches the Earth's rotation. It's not really stationary after all, but its speed is matched to the Earth's so it *appears* to be stationary from the ground. In the figure, it will not move from over the grey dot. That's why we call these satellites **geosynchronous**, from *geo*, meaning *Earth*, and *synchronous*, meaning *in time with*. They move in time with the Earth.

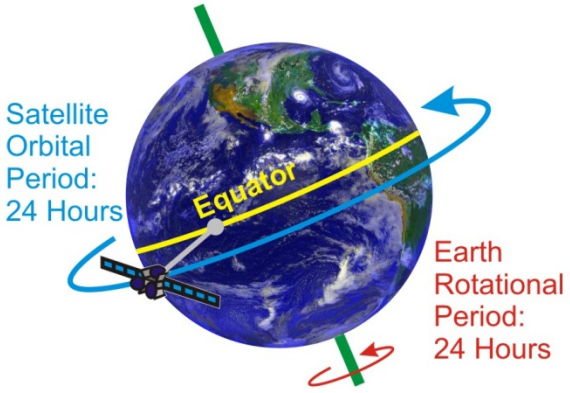

Figure 83: Geostationary Orbit.[50]

Well, that answers our question about why satellites have to be so far away. That's the only place where their orbital speed will match the rotational speed of the Earth. But why can't we park it above Denver? Again, the answer is gravity. Which way does gravity pull? Down, of course. But the direction of *down* depends on where you are. Take a look at Figure 84. Frame I shows that down on one side of the Earth is the exact opposite direction of down on the other side. No matter where you draw this satellite, the direction *down* would eventually pass through the center of the Earth.

Since gravity is making the satellites fall down, and the satellite's speed has to be perpendicular to that to keep it in orbit, the orbit has to be in a plane that contains the direction of both its speed and the direction it wants to fall. Since the direction it wants to fall is down, and down is always toward the center of the Earth, *the plane of <u>any</u> orbit <u>must</u> contain the center of the Earth*. Frame II shows three orbital planes: green and blue allowable ones that contains the center of the Earth and a red, bogus one that doesn't. Allowable orbital planes don't have

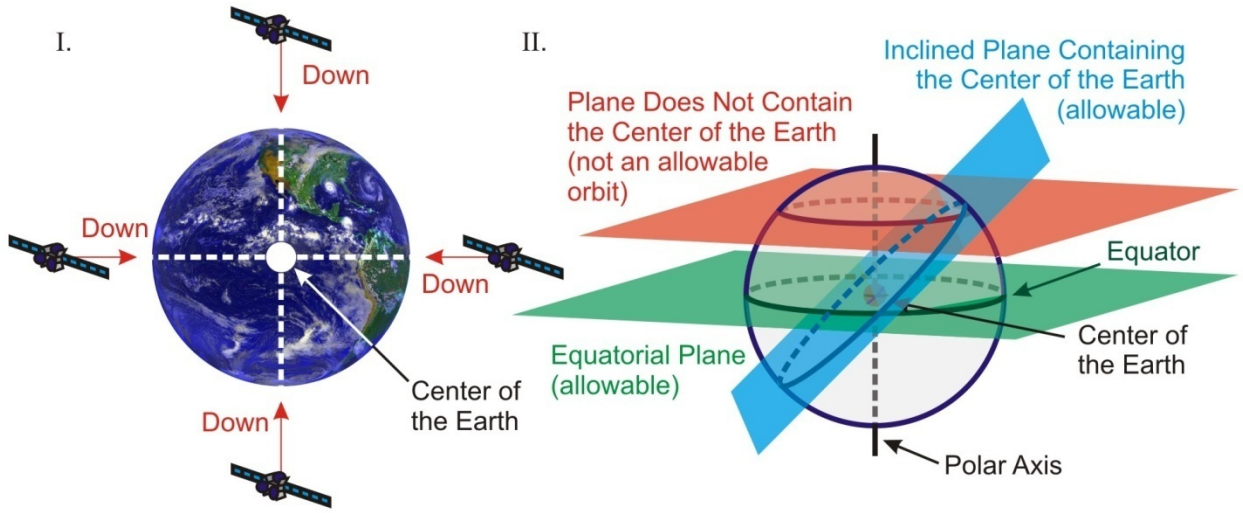

Figure 84: Orbital Planes and the Center of the Earth.

to be horizontal; they can be tilted at any angle just so long as they contain the center of the Earth.

There's one last tidbit we need to know about geosynchronous orbits. There are a lot of ways to make a geosynchronous orbit; all that's required is that the satellite completes a single orbit in exactly the same amount of time it takes for the Earth to turn once on its axis. However, not all geosynchronous orbits remain over one spot on the Earth. Both of the orbits shown in Figure 85 are geosynchronous. The green orbit is the same one we showed before, orbiting directly over the equator. The blue one, however, is in a circular orbit with an orbital plane that's tilted with respect to the equator. In fact, in this example it's tilted by about 45°. While it still takes 24 hours to go around the Earth, as you can see from the series of grey dots under the satellite, it does not stay over one point. It moves up and down between 45° north latitude and 45° south latitude (yes, those limits are always the same as the tilt of the orbit), making one cycle each day. It stays over the same *line* of longitude, but not over the same *spot* on that line.

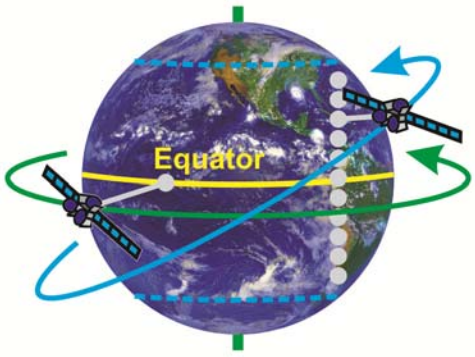

Figure 85: Geostationary vs. Geosynchronous Orbits.

While a tilted orbit like that can still be called geosynchronous, it wouldn't be of much use to us here at DISH because our antennas would have to move, tracking it up and down, north and south, every day to keep it in sight. So what we need

to do so we can use fixed antennas is to only put our satellites in what are called **geostationary orbits**. A geostationary orbit is *a circular orbit directly over the equator that has an orbital period of 24 hours*. However, most people just say geosynchronous and use it to mean geostationary. But, now you know the real difference.

That's all you really need to know about the physics of satellite orbits. And you now understand more about orbital mechanics than 99 percent of humanity. However, just like when we learned about the electromagnetic spectrum, there's more to it than just physics. Politics comes into play here, as well.

Political Considerations. Remember our old friends the International Telecommunications Union (ITU), the international organization that licensed use of the electromagnetic spectrum? They also license use of the geostationary belt. Why do they need to do this? Because, just like the radio spectrum, the geostationary belt is highly valuable and many groups want to put their satellites there for the now-obvious reason that you can use cheap, fixed antennas to point at them. How popular a place is it? Check out Figure 86 and see. The figure shows the orbital locations of the almost 300 commercial communications satellites currently in orbit in the geostationary belt. Notice that's *only* communications satellites (there are other uses for the belt), and *only* commercial users (there are government users as well, using slots for things like weather monitoring, intelligence gathering, and communications).

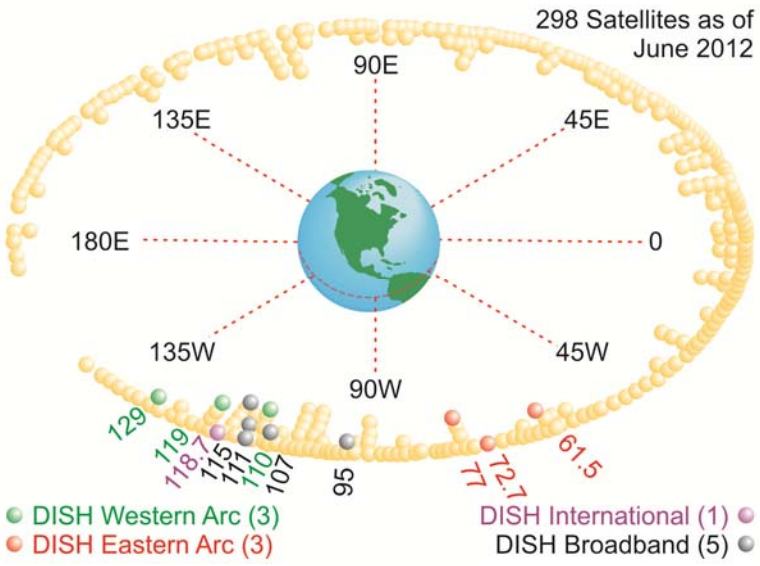

Figure 86: Communication Satellites in Geostationary Orbit.[51]

You'll notice a lot of satellites grouped between about 60° and 120° west longitude; these are highly-coveted orbital slots that can serve the lucrative North American market. There is also similar heavy use of the belt over Europe and East Asia. Can you figure out why there aren't many satellites between 140°W and 180°E?

The television satellites DISH uses are owned or

leased by our sister company, EchoStar. The satellites we use for Internet delivery are either owned by Hughes, a subsidiary of EchoStar, or by the independent company ViaSat. DISH resells bandwidth from these companies under the dishNET name. The locations of all satellites DISH employs are shown on the figure, and color coding indicate whether DISH uses them for television or broadband. The numbers next to those satellites are their assigned orbital slots, indicated by the longitudes where they're allowed to be. They look really closely packed together in this diagram, but in reality they're very far apart. Just doing some quick math, if there are 300 satellites around a 330° arc (excluding the hole over the Pacific), then there must be on average 1.1° per satellite. Since the satellites are all 22,000 miles above the Earth (26,000 miles from the center of the Earth), that translates into…um…carry the one…they're separated by about 500 miles or so (on average) since they do tend to drift a bit due to things like perturbations from the Moon, pressure from the solar wind, and other things. The satellites' owners are responsible for guiding them back to their proper locations if they drift by more than a certain amount. In the case of our DISH satellites, we have to keep them in a box that's about 40 miles on a side.

To give you an idea of what parts of the country DISH satellites are parked over, Figure 87 shows the satellite locations over the equator and the projections of their longitudes up through the United States. The 129 satellite is due south of the easternmost point in Alaska, and, in addition to providing programming across the Lower 48 States, it provides programming to Hawaii and the southern part of Alaska. While the 119 and 118.7 satellites appear to overlap, they're actually over 150 miles apart in their geostationary orbits.

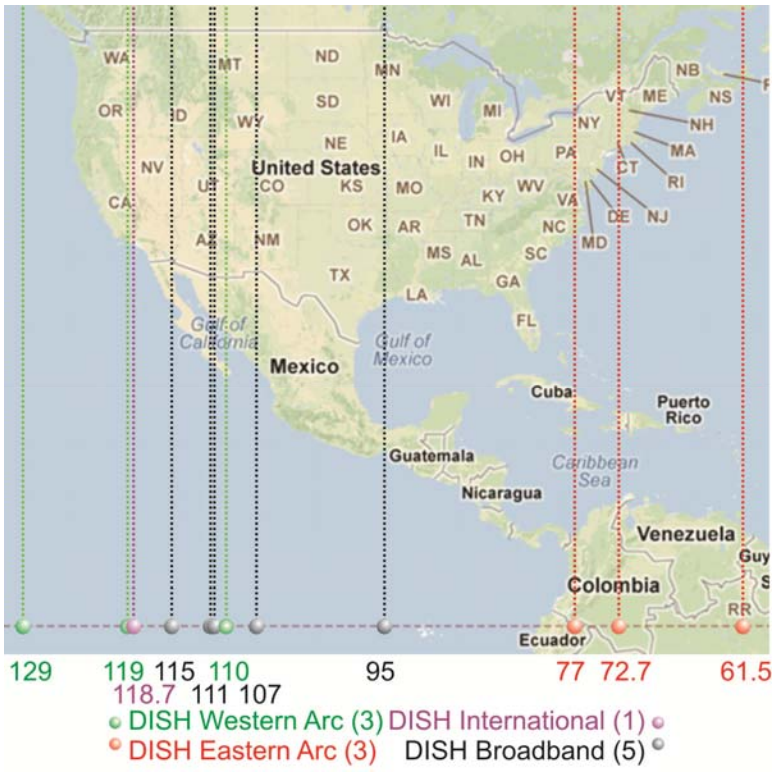

Figure 87: DISH Satellite Longitudes.[52]

Well, that's enough about why our satellites orbit where they do. Now that we know why they're there,

let's see what that means to us as we follow our signal up from the uplink center and back down to the familiar DISH rooftop antennas.

Satellite Design Considerations[53]

As we saw earlier, we sent some thrice-coded electromagnetic waves (MPEG-coded, encryption-coded, and QPSK/8PSK-coded) from some very large parabolic dishes at our uplink centers out into space. The whole point was for those signals to be received by antennas on the satellite and then retransmitted from the satellite back to the ground. This concept, where the satellite only performs very basic functions on the signal between receiving and retransmitting it, is called a **bent-pipe** system. That's really all the satellite needs to do, but the antennas and transponder that perform that function aren't the only things on the satellite. There are a number of other things that satellites do to make the bent pipe possible, but in essence a communications satellite is basically just a life support system for the bent pipe. The bent pipe is the **payload** for our satellites. The payload is the reason the satellite is up there and is the "load that pays us," or thing that makes us money.

Payload Support Systems. Excluding the payload, there are four basic systems on any satellite: attitude and orbital control; communications and data handling; power; and environmental control.

The *attitude and orbital control system* is what keeps the satellite pointed in the right direction and in the right place in space (*attitude* means the direction the satellite is pointed). Attitude is sensed by such things as star sensors, gyroscopes, and magnetometers. When these sensors determine the satellite is not pointed within parameters, they command small thrusters or reaction wheels to act, reestablishing proper pointing. Ground controllers are constantly monitoring satellites to make sure they're in their proper locations. We mentioned before that there are a number of things that are constantly trying to move satellites away from where we want them to be. When the ground controllers determine the satellite has moved too far from its specified location, they will command a small thruster burn to correct the drift. Fuel is required for these orbital maneuvers (and for some attitude control maneuvers), so they must be done judiciously since the fuel supply is limited. In fact, running out of maneuvering fuel is the most likely reason for a satellite to cease functioning. Just before it runs out of fuel, controllers normally command GEO satellites to raise their orbit to get them out of the precious GEO belt, making room for a replacement.

The *communications and data handling system* is what allows ground controllers to command the satellite to make orbital and attitudinal maneuvers, turn on and off specific subsystems, and take other actions. It also allows those systems to communicate their health and status to the ground to give controllers early indications of malfunctions. While communications is the bread and butter of DISH satellites, this particular communication system is independent of the payload communication system, and is only used to communicate with the satellite itself. It has nothing to do with transmission of our signal.

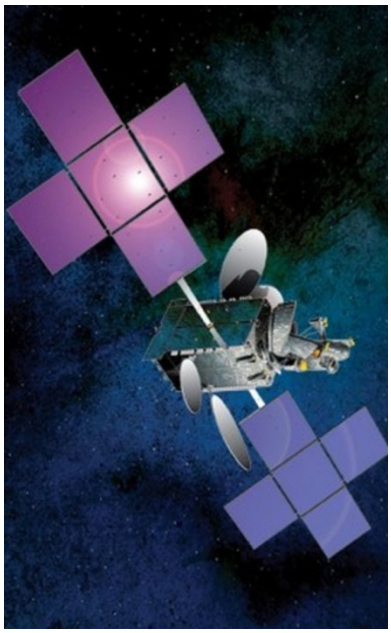

Figure 88: EchoStar XVII Showing Its Prominent Solar Power Arrays.[54]

Electrical power is generated and distributed by the satellite's *power system*. Its two main components are solar panels for electricity generation and batteries for electrical storage. As demonstrated in Figure 88, perhaps the most obvious features of most satellites are their large arrays of solar panels which convert sunlight into electricity the satellite can use—the two sets of five purple and blue squares in the image. Solar panels are made up of thousands of individual solar power cells. They are only able to convert about 20% of the Sun's energy that falls on them into electricity, so it's very important to keep them pointed as flat-on to the Sun as possible. It's no easy feat to keep the solar panels oriented correctly at the same time as keeping the communications and data handling antennas and the bent-pipe antennas pointed in the right direction!

Being so far from the Earth and not orbiting in the same plane as the Earth orbits the Sun, GEO satellites only enter the Earth's shadow for no more than 72 minutes on a few days twice a year (about 12 hours after they encounter the solar interference we'll discuss in detail in *The Dish on Your Roof* chapter). But when they do, they need to have charged batteries capable of keeping the satellite operating while they're not seeing the Sun. This is a significant difference between GEO satellites and satellites like the ISS orbiting in low Earth orbit (LEO), which encounter the Earth's shadow for about 45 minutes out of every 90 minutes. For this reason, GEO satellite batteries don't undergo nearly as many charge-recharge cycles (these cycles are a major constraint on battery life), but they do drain much more deeply because of the longer time away from the Sun.

For this reason they need to be quite a bit larger than those on a LEO satellite with the same power requirements.

The *environmental control system* makes sure the satellite doesn't get too hot or too cold. Space is a strange place. On the side of the satellite facing the Sun, temperatures on the satellite's surface can reach +250° F, while on the side away from the Sun it can be a frigid -250° F.[55] On average, though this means the satellite's interior stays at about 80-120° F. Batteries are relatively sensitive to temperature, working well in a range from about 40° F to about 150° F.[56] Satellites use radiators and other devices to regulate the internal temperature of the satellite to keep it within this range so the satellite will operate properly.

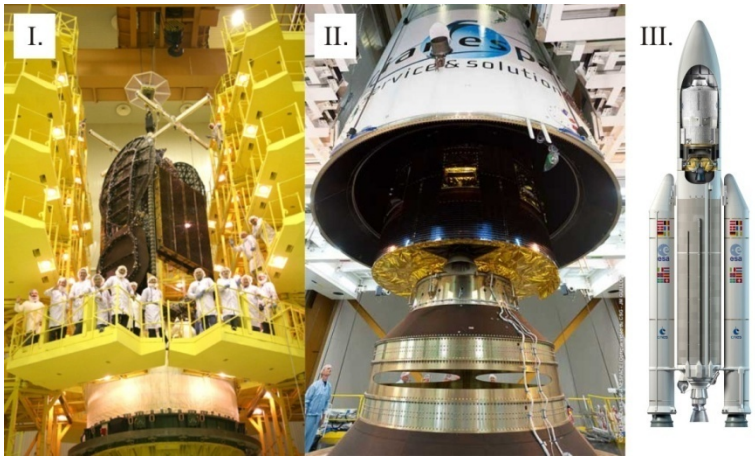

Figure 89: Satellite and Fairing.[57]

That's a very high-level overview of some of the systems common to all satellites. Oh, and by the way, all of these systems and the payload have to fit inside the fairing atop the rocket, and those fairings are generally only designed to hold satellites that are less than about 18 feet wide and 40 feet tall.[58] Frame I of Figure 89 shows EchoStar XIV being prepared for launch. It weighs about 14,000 pounds, which is a moderately heavy payload to get all the way to GEO. You can see the folded antennas and solar cells on the sides of the satellite. In Frame II you can see the fairing being lowered onto EchoStar XVII. Frame III shows Arianespace's Ariane V rocket like the one that launched EchoStar XVII. While the payload shown in the cutaway of that frame is not the EchoStar satellite, ours fits inside the rocket in a very similar manner.

The Bent Pipe—Receive Antennas

Earlier we described a communications satellite as a life support system for the payload. Now that we've looked at the life support system, it's time to look more in depth at the bent pipe itself. The first part of the bent pipe that encounters the signal is the receive antenna. On DISH satellites, these are typically only about 12 feet across. We talked about beamwidth earlier in the Antennas and Diffraction section of *The Uplink Center* chapter on p. 66, but it's not only transmit antennas

that have beamwidths. Receive antennas have them, too. What beamwidth means on the receive end, though, is that if a transmitter is located within the main beam, the antenna will amplify its signals strongly, allowing them to be detected above the noise level in most cases. If the transmitter is not within the main beam, the signals don't make it to the receiver very well at all.

The same considerations for beamwidth as for transmit antennas apply as well: at a fixed wavelength (or frequency) a narrower beam is associated with a larger dish. Since these satellite receive antennas are a lot smaller than our transmit dishes, you'd naturally expect the beamwidth to be larger (and the gain to be less). Plugging in numbers, the beamwidth is actually about 0.6° for an 11-foot DISH satellite receive antenna. That translates to a circle about 300 miles in diameter on the ground directly below the satellite at the equator. In contrast to the 65,000 value for the uplink antenna gain, these antenna can only boast gains of about 4,500 (4,500 times the strength of what the antenna would receive were it not directional at all).

Frame I of Figure 90 shows **footprints**, the shape where the antenna beam intersects with the ground, from a GEO satellite above the equator using a receive antenna about the size of the ones on DISH satellites. The footprint directly below the satellite is circular, but that shape changes as the boresight direction is moved away from straight down. The circle becomes more and more elongated as the antenna is pointed away from the equator and/or the satellite's longitude. Note that by the time it gets as far north as our uplink center in Cheyenne, it covers an area of about 350 miles by 250 miles. Frame II demonstrates that larger footprints (from smaller antennas) show even greater degrees of

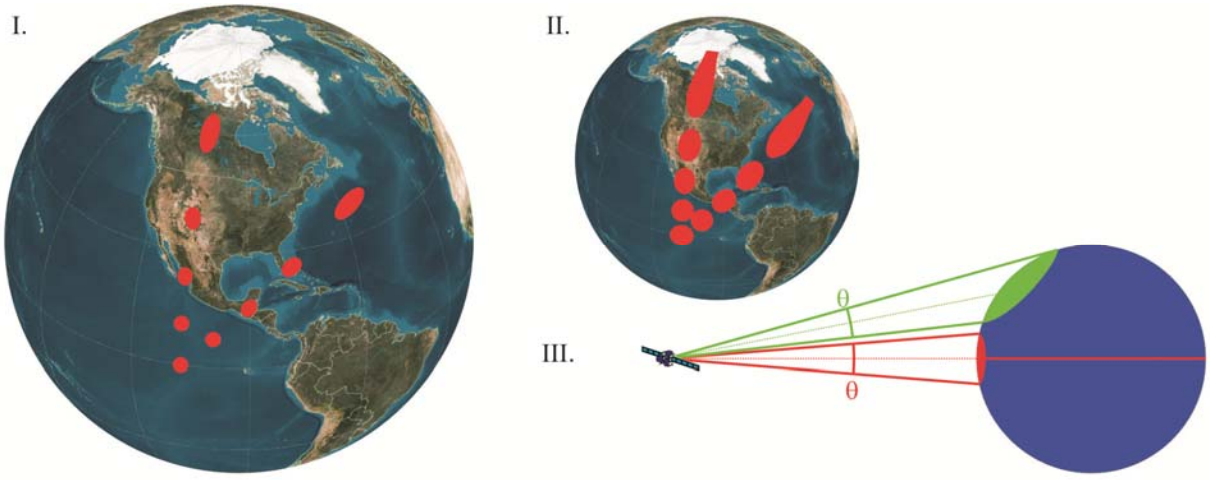

Figure 90: Footprints Become More Elliptical as They Shift from Directly Below a Satellite.[59]

elongation. The largest footprints are squared off at their tops because the satellite drops below the horizon and is not visible for observers at more distant locations. Frame III shows a side view of this situation to clarify why the shape changes and gets bigger. The green and red angles labeled θ are exactly the same, but they intersect the spherical Earth in very different ways.

The point of this discussion is that to prevent interference in the receive antenna, we can't place more than one uplink center in the main beam. That's one of the physics constraints on where we locate our uplink centers. And it's really not all that difficult to interfere with the signals on our satellites. Any similarly-powered and similarly-sized dish aimed at our satellite could do it, as long as it is in the footprint of the receive antenna, as could a smaller antenna with larger transmit power.

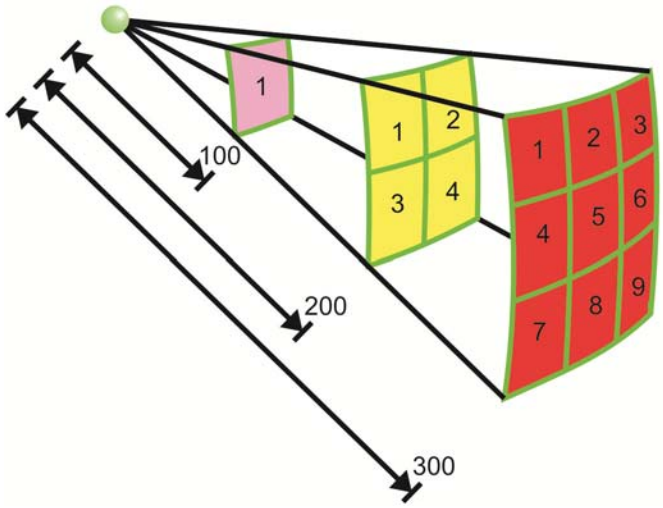

Figure 91: Power Reduces with Distance.

As our signal travels through space, the power hitting a specific area decreases rapidly as the signal gets further from the antenna. Figure 91 shows why that's true. Regardless of how large an object is, when you move far away from it the object eventually just looks like a dot—a small sphere. So, from GEO distances of at least 22,000 miles, our seemingly huge antennas definitely become dots. That dot is shown as the green sphere in the figure. Let's say that as the signal gets 100 miles away from the dot, the area the main beam of the signal passes through is one unit square, as indicated by the pink square with the number one in it (yes, we know the main beam isn't square, but for the purposes of illustration…). By the time the signal is 200 miles away, it now passes through an area of four unit squares (the yellow two-by-two square region in the figure). Twice the distance, four times the area. The red region shows that at three times the distance, the signal fills an area nine times (three times three) as large. Thus, only one-ninth of the power that passed through the pink square passes through any one of the red squares. That's actually a very general physical phenomenon—power falls off very rapidly as you get farther and farther away from the source. (In fact, the law is called the *inverse square law*: one over one squared equal 1; one over two

squared equals ¼; one over three squared equals 1/9; and so on…) and it's related to the surface area of a sphere, but we won't go into that in any more detail here.

The practical meaning of all that for us is that although we're transmitting 2,000 watts of power from our uplink center antenna, by the time the signal makes it all the way out to GEO our receive antennas only capture a signal of about a few millionths of a watt. Can you see why it's in our interest to make our receive antennas just as large as possible? Not only will that increase our gain (by narrowing the beamwidth), but it gives us that many more "squares" from the figure from which we can capture the signal's power.

The Bent Pipe—Transponders

So let's move on from receive antennas and address why and how we change the frequencies of our signals before we send them back down to Earth. We've talked about interference repeatedly, and we've also shown you how broadcasting from an antenna creates sidelobes, with some of the lobes even pointing behind the broadcasting antenna. These two problems conspire in a transponder to create the need for us to use a different frequency for our transmitters and receivers.

The satellite transponders are relatively powerful, emitting a signal of between about 150 and 450 watts (equivalent to 2 ½ to 7 ½ 60-watt light bulbs). Now, that may not sound like a lot of power when compared to the 2,000 watts of the uplink center transmitter, but compared to the few millionths of a watt we're effectively receiving from the distance-reduced incoming signal, a couple of hundred watts is huge! Even the relatively minor sidelobes pointing backward from our satellite transmission antennas could be strong enough to interfere with our received signal…if, of course, they were in the same frequency band as the incoming signal. Fortunately, they're not.

Shifting the frequency on which the satellite transmits away from the frequency on which it receives is the primary function of the transponder. Up until now, we've talked about the frequency band that DISH owns for uplink—17.3 to 17.8 GHz. We also own the band from 12.2 to 12.7 GHz, and it's there that we downlink our signal to our customers.

Fair warning: the next page or so covering mixers and filters may go over your head. This document is trying to hit a balance between being informative for both non-technical and technical readers alike, so we do need to cover how the frequencies really are shifted. We encourage you to read through the description

of the process, but if you're spending more than about five minutes on it or you find your head hurting (and we mean *really* hurting—no pain, no gain, you know), substitute the phrase *and then magic happens* for what you haven't covered and skip to the heading *The Bottom Line on Mixers*. In that paragraph there's some addition and subtraction you *will* need to understand both here and later when we discuss how the signal gets inside houses.

But how does the transponder shift the frequency? Before we answer that question, we first need to understand the concept of a filter. We're sure every one of you has seen a filter of some sort at some time in your lives. Colored plastic wrap, sunglasses, all sorts of common items are filters. If you've ever looked at the world through rose colored glasses, those glasses made everything look rosy because they only let visible light with the color combination we call *rose* through, blocking everything else.

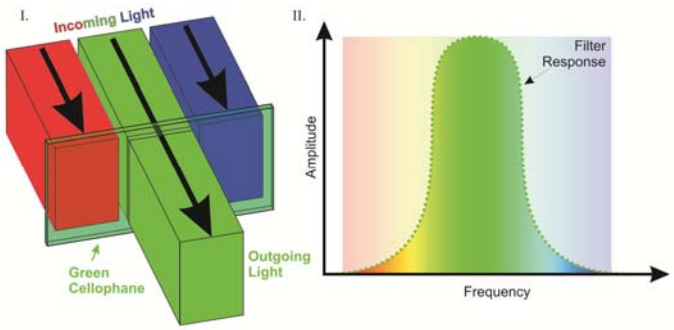

Figure 92: Color Filter.

Frame I of Figure 92 show this concept for incoming light of three different colors. As that light hits the green-colored cellophane, the red and blue light is blocked, but the green light passes straight through unhindered. Looking at Frame II, we see the frequency response of the filter. It's a bit curved, letting through a little red, orange and yellow at the left side, before shooting up rapidly to let lots of green light through. It then falls off rapidly before only letting through very small amounts of blue and violet light.

Figure 92 showed an example of a visible light filter because it was relatively easy to draw, but there are also electronic devices that filter radio signals as well. We'll use one of those radio filters in a moment as the critical final step in shifting the incoming frequency to the outgoing frequency.

To shift the frequency, we use an electronic device called a *mixer*. We'll use Figure 93 to illustrate how the mixer and the filter work together to get us to the desired outgoing frequency. (If these wave and frequency spectrum diagrams look foreign to you, you probably skipped the section in the *Spectral Bandwidth* part of this document where we introduced them. You should review that discussion starting on p. 81.)

The only real difference between the mixer we're discussing here and the ones we discussed earlier is that the frequencies we're mixing are much more on the same scale. Previously, we mixed a relatively low frequency modulating signal containing the information content of our programs (15 MHz) with a much higher carrier frequency (10 GHz) that the information essentially rode on. The mixing process we discussed there kept the carrier frequency and the two sidebands created at the sum and difference frequencies. (If all this sounds foreign to you, we *really* suggest you read the section on *Spectral Bandwidth* starting on p. 81.) The only thing we didn't mention before was that we filtered out the low-frequency modulating signal before uplinking the carrier and sidebands.

So, let's look at how this bent-pipe mixing differs from the carrier/modulator case we've already discussed. Frame I shows the incoming signal at 17 GHz (we'll use whole numbers in the figure and this discussion to simplify the math a bit instead of using the exact frequency ranges). Remember, this 17 GHz signal is the *carrier* wave that we mixed with the modulating signal to get our information uplinked. For the purpose of clarity, we'll ignore drawing the sidebands on this carrier, but keep in mind they're there and they'll undergo mixing in this process as well.

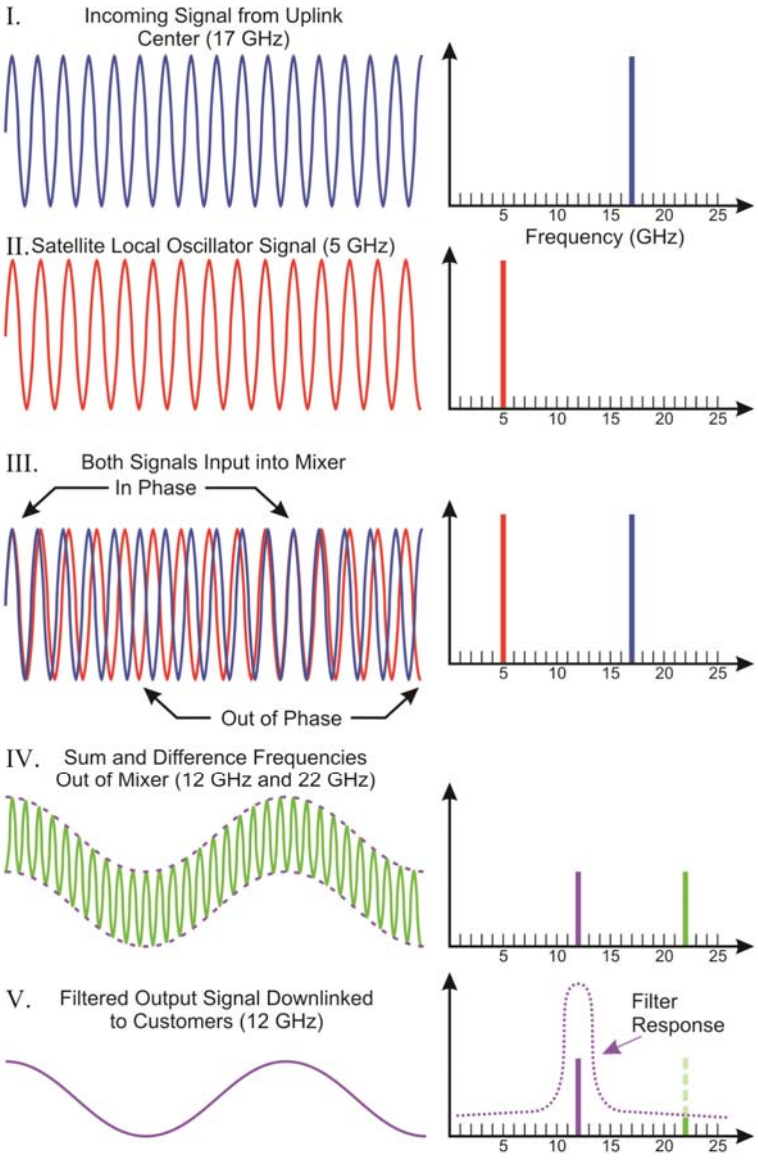

Figure 93: Shifting Frequencies with a Mixer and a Filter.[60]

The satellite has a very precise radio wave

generator called a *local oscillator* that outputs a wave with a frequency of 5.1 GHz at relatively low power, low when compared with what we'll eventually send back down to Earth. Frame II shows this signal, but uses a frequency of 5 GHz to make the math easier for this discussion. Those two signals, our incoming signal and the local oscillator signal, are then fed into the mixer. Notice in Frame III how those signals sometimes are in phase and sometimes out of phase.

What the mixer does is to electronically multiply the amplitudes of both signals together. There's a little trigonometry involved when you multiply two sine waves together,[61] but the rather amazing end result is that the output has two *different* frequency components, one at the sum of the original frequencies and one at the difference. Frame IV shows this result. For our 17 and 5 GHz signals, the sum is 22 GHz and the difference is 12 GHz. In the left of this frame, you can see how the green wave wiggles up and down much more rapidly than the incoming red and blue waves, but the green wiggles are modulated by the slow wiggle of the dashed purple wave. The two new frequency components are shown in the right side of that frame.

Now is where we apply our newfound understanding of filters. We're only authorized to transmit from space to the Earth in the 12 GHz region of the spectrum, not the 22 GHz region. To get rid of the unwanted frequency, we send the mixer output through a filter that allows 12 GHz-ish signals to pass but blocks others. That result is shown in Frame V. The 17 GHz signal we received from the uplink center has now been transformed by the transponder in our bent pipe into a 12 GHz signal.

Now, we've been using whole numbers to illustrate this process, but in reality, we're transforming an entire block of signals, 17.3 to 17.8 GHz, into an entire new block of signals ranging from 12.2 to 12.7 GHz. Notice that our 5.1 GHz local oscillator will exactly accomplish that transformation using the difference frequency out of the mixer (17.8 – 5.1 = 12.7 and 17.3 – 5.1 = 12.2).

Returning to the carrier/modulator mixing we looked at earlier, in that case we kept three of the four signals involved in the mixing: the carrier and both the sum and difference frequencies generated by the mixing. We filtered out the modulating frequency. In this bent-pipe case, we only keep the difference frequency, discarding both the LO and the original carrier as well as the sum frequency. Although we haven't mentioned it explicitly, the frequency we keep (the *downlink carrier*) also has the modulating frequency sidebands attached to it.

The Bottom Line on Mixers. OK, here's what you really need to get out of the above discussion on mixers. When we mix two signals together, the result is that we get back two different signals oscillating with frequencies at the sum and difference of the two we mixed together. We only want to use one of them, so we have to filter out the other. That's all there is to it! In our uplink carrier to downlink carrier example, the two signals we mixed were the uplink band (17.3 – 17.8 GHz) and the signal from an oscillator onboard the satellite, the local oscillator (LO) at 5.1 GHz. Add those together and we get two more bands: 22.4 – 22.9 GHz and 12.2-12.7 GHz. We only want the lower band, so we filter out the upper one and, *voila*, we're left with the downlink band we need.

Figure 94 shows the satellite transponders and the actual downlink frequencies DISH uses to send our signals down to Earth. The transponder frequency bands are 29.16 MHz wide (corresponding to the modulating sidebands), including the 3-MHz-wide guard bands that separate the signals. The center frequencies are the downlink carriers. Compare this chart with Figure 74 (p. 95), and you'll see that it's essentially the same layout, just shifted down in frequency by 5.1 GHz.

Transponders are the workhorse of the bent pipe, but they're really not all that complicated. They take the incoming signal and rebroadcast it at a different frequency to avoid interfering with the input. Now that we've got that clear, let's move on to the last part of the bent pipe, the transmit antenna.

RHCP		LHCP	
Transponder	Center Frequency (GHz)	Transponder	Center Frequency (GHz)
1	12.22400	2	12.23858
3	12.25316	4	12.26774
5	12.28232	6	12.29690
7	12.31148	8	12.32606
9	12.34064	10	12.35522
11	12.36980	12	12.38438
13	12.39896	14	12.41354
15	12.42812	16	12.44270
17	12.45728	18	12.47186
19	12.48644	20	12.50102
21	12.51560	22	12.53018
23	12.54476	24	12.55934
25	12.57392	26	12.58850
27	12.60308	28	12.61766
29	12.63224	30	12.64682
31	12.66140	32	12.67598

Figure 94: DISH Downlink Frequencies.

The Bent Pipe—Transmit Antennas and Satellite Footprints

We've discussed antennas in detail already during this text, first in *The Uplink Center* chapter and then again in this chapter when we talked about receive antennas. We won't beat that dead horse again except to add a little extra information specific to getting the signal down to the places on Earth where we can use it to make money.

CONUS Beams. Earlier in this chapter we discussed how satellite footprints appeared to be circles or ellipses on the ground, and that's true if we're using perfectly parabolic antennas. However, it's pretty difficult to design an ellipse that only hits the parts of the ground we'd like to target with our signal. Wouldn't it be more efficient if we could tailor the footprint so that it more accurately matched the region on the ground we're trying to target?

Frame I of Figure 95 shows an approximation of an elliptical footprint that would cover the United States from our DISH satellite over the equator at 129° west. Notice how a very large portion of the signal we pay so much money to get to space and back ends up in Canada and Mexico or over the ocean, places where we don't make any money from it.

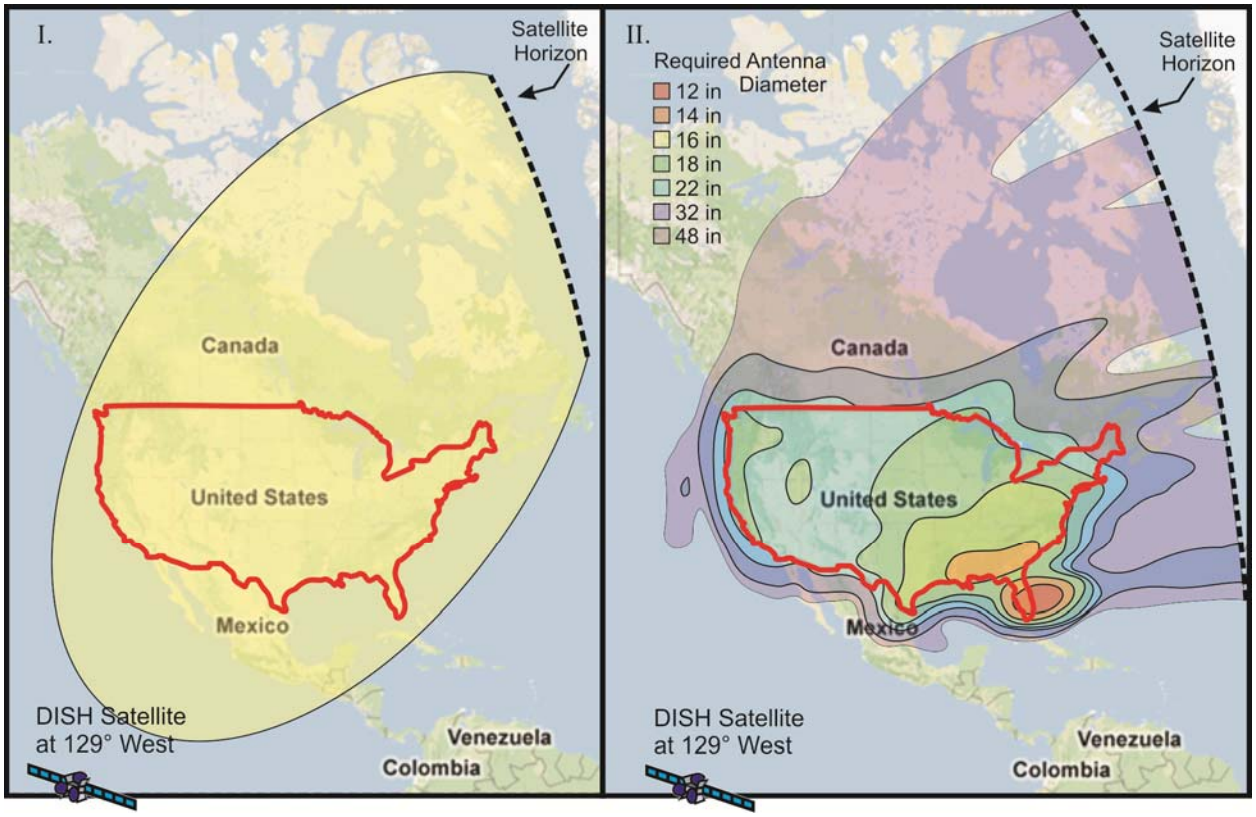

Figure 95: Tailored Satellite Footprints.

Figure 96: Antenna Shaping for Tailored Footprints.[64]

In contrast, Frame II shows the actual CONUS beam from that satellite.[62] The strength of the signal at different places on the ground is indicated by the shaded regions, where in this case the relative signal strength is measured by the size of the antenna required to effectively capture the beam. Notice how the higher-power (smaller rooftop antenna size) regions mostly fall within the Continental United States (CONUS), our market area.

Why are the two beam shapes so different? It's because the actual beam shape has been *tailored* to better fit our requirements to provide satellite coverage to our market by subtly changing the shape of the transmission antenna away from that of a pure parabola.[63] Figure 96 shows an example of a shaped antenna used to generate tailored footprints. While it looks substantially misshapen, especially since we're used to looking at very smooth, perfectly manufactured surfaces, a great deal of math, time, and money went into determining that shape since it will put most of our signal in places where we can make money from it. It's still kind of parabolic, though, isn't it?

Spot Beams. So far, we've discussed footprints and beams that cover the entire CONUS region. However, it's pretty typical for DISH satellites to only have one or two antennas devoted to CONUS coverage. The rest of the transmit antennas on the satellite are devoted to spot beams that cover fairly small areas that are usually about 300 miles across. We use spot beams to deliver local channels to specific market areas. For example, the DISH satellite at 129° W, the one we used as our example for the tailored CONUS beam above, has 53 spot beams that individually cover much of the CONUS as well as Alaska and Hawaii. Figure 97 shows the 20-inch rooftop antenna contours of the beams aimed within the CONUS (these contours show the outer edges of the locations where a 20" antenna would receive an acceptable signal).

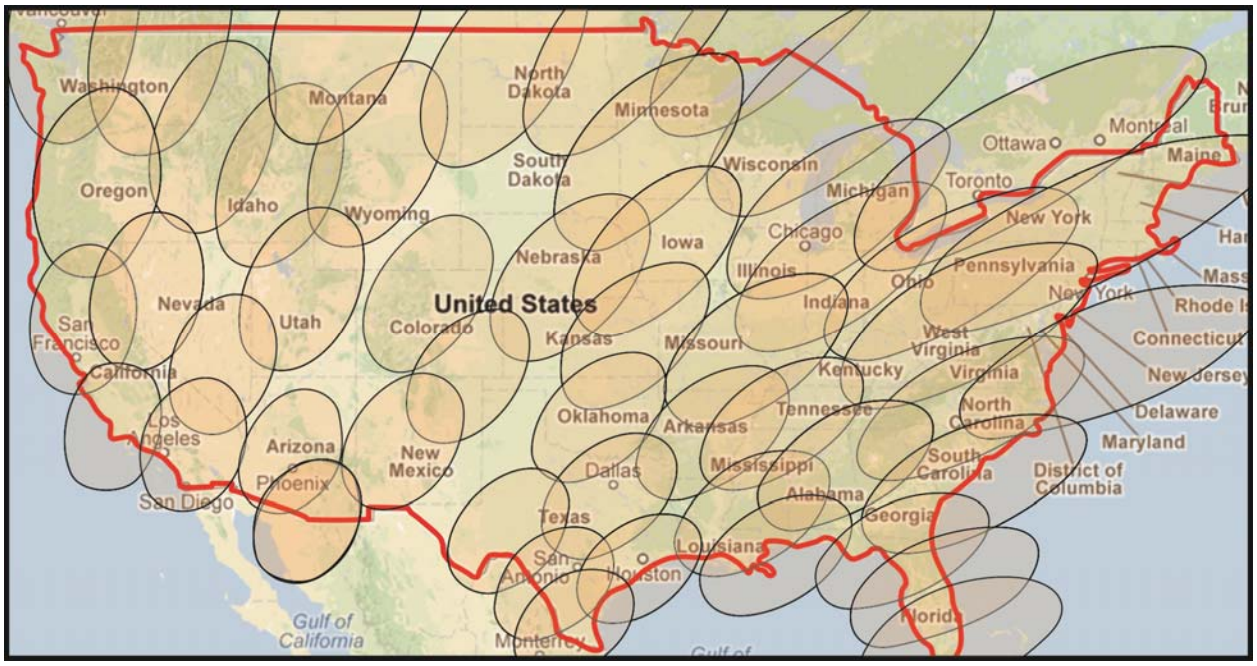

Figure 97: Spot Beams.[65]

Figure 98 shows the antenna configuration on EchoStar XIV, a satellite similar to the one that delivered the spot beams in the previous figure. Notice that it only has four large transmit antennas, and two of them, labeled "National," are used for shaped CONUS beams. How can we get all of those spot beams out of the two remaining antenna?

When we originally talked about antennas, we drew a diagram that showed how electromagnetic rays beginning as circular waves coming from the focus of the parabola would emerge as a beam of linear wavefronts. If you need to review it, that discussion is on p. 62. The original figure is reproduced in simplified form in

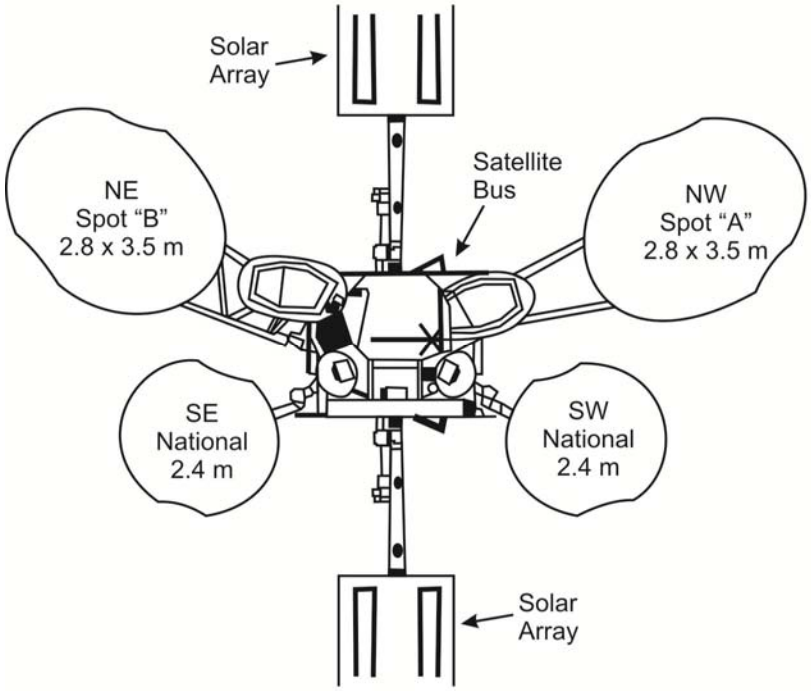

Figure 98: EchoStar XIV Antenna Configuration.[66]

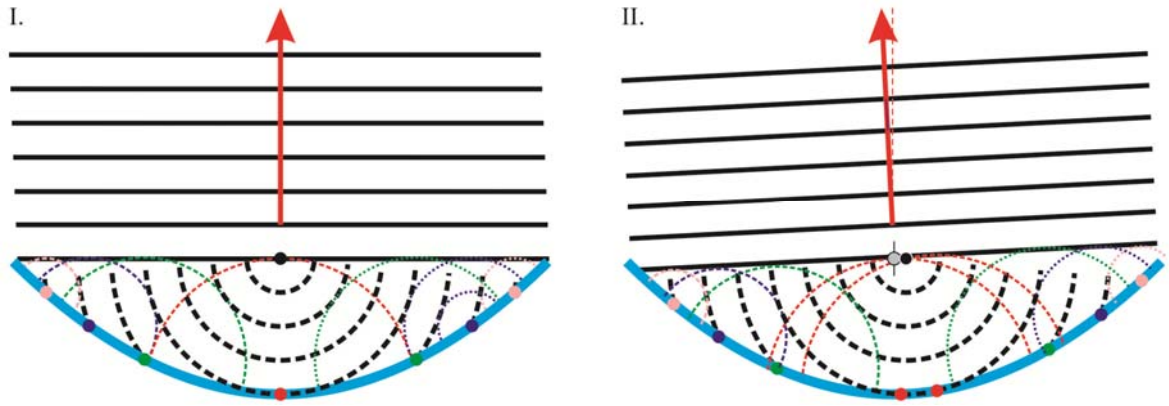

Figure 99: Off-Focus Feeds Shift the Beam Direction.[67]

Frame I of Figure 99. In this frame, the circular waves originate at a black feed-point located at the focus of the parabola. The black dashed lines show the wave crests between the feed and the parabola, with one wavelength between each circle. Dots on the blue parabolic dish show the points where those black wavefronts touch the dish. Circles of corresponding color indicate the reflected circular waves from those colored points. Notice that there are four black wavelengths between the feed and the red dot, five to the green dot, and so on. To reduce the clutter in the picture, we've only indicated a circular wave at a distance of one wavelength *from* the pink dot, two from the blue dot, and so forth. That means that the black line joining all of the colored circles right at the top of the parabola are all eight wavelengths from the focus. They all line up to form the heavy black wavefronts moving straight up, away from the parabola.

Frame II shows a slightly different situation. The black feed point has been moved slightly to the right, away from the gray focus. Notice that this shift moves the points where the black dashed circular waves touch the parabola. It's hard to see at this scale, but the wavefront passing through the red dots really does intersect the parabola in two places, with the portion of the wavefront between those two dots actually being past the parabola. When a similar construction of wavefronts is used, you can see that the black linear wavefronts don't move straight up the page anymore. By shifting the location of the feed, we can shift the direction the beam goes.

That's exactly how we get different spot locations using one antenna: we have multiple feeds that are precisely shifted away from the focus. Figure 100 shows an example of just such a feed horn array with 18 feeds that would be used to deliver our signal to a single antenna to produce 18 spot beams.

With just that technical discussion, spot beams would be little more than a nice bit of trivia. However, there's a business reason we use them: frequency reuse. We've talked at length about the available frequency band and the amount of spectral bandwidth it takes to broadcast all of our channels. We just don't have the spectrum available to transmit *all* of our national channels and *all* of the many, many local channels across the entire United States as we would have to do if we only had the CONUS beam shown in Figure 95 available to us. By using spot beams, we can reuse frequencies in any of the footprints that don't overlap. For example, let's say that the local NBC channel from Denver is broadcasting on transponder 17 from our satellite at 129° W. That same transponder frequency, 12.45728 GHz, could be used to broadcast the local Fox station in Salt Lake City, the local CBS affiliate in Los Angeles, and a multitude of other stations in other footprints. Frequency reuse means we don't have to launch many more very expensive satellites.

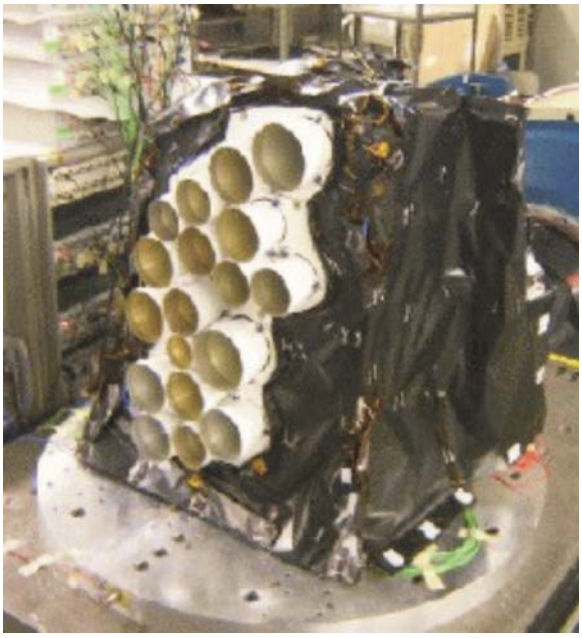

Figure 100: Feed Horn Array for Spot Beams.

Frequencies Revisited. Speaking of frequencies, we've previously noted that DISH uses two Ku frequency bands: 12.2 to 12.7 GHz for downlink and 17.3 to 17.8 GHz for uplink. Notice that the downlink frequency band is lower than the uplink frequency band. That arrangement is not by chance, and if you look at just about any satellite communications system you'll see they follow the same pattern: downlink lower than uplink.[68]

There are two technologies that drive the downlink frequency band to be the lower one: antennas and transmitters. First, think about what we're trying to do with our uplink and downlink antennas. For the uplink, we're trying to hit a very small satellite very far away, so we want our beam to be just as tight as possible. As we saw earlier in *The Uplink Center* chapter, that's the reason we make our uplink dishes so very large. But remember the equation that determined the beamwidth: $\theta = k * \lambda/D$, where θ is the beamwidth; k is just a constant that depends upon the shape of the antenna and the definition of beamwidth we're

using (there are several); λ is the wavelength of the electromagnetic radiation being focused; and *D* is the diameter of the parabolic dish. So, to make the beamwidth smaller we have two things we can adjust—wavelength and dish diameter. We've made the dish large already, so the only other way is to make the wavelength as small as we can. Since wavelength goes up as frequency goes down (see the top and bottom scales on Figure 34, p. 52 if you need to re-convince yourself of this relationship), that means we need to use a smaller wavelength or a larger frequency. Voila! That's what we do! The 17-ish GHz range we use for uplink is larger than the 12-ish GHz range we use for downlink.

For the downlink problem, antenna size isn't nearly as critical. Instead of trying to hit a tiny satellite, we're targeting the entire United States with CONUS antennas and 300-mile-wide regions with our spot beams. We can afford to have smaller antennas and lower frequencies for downlink since our beamwidth doesn't need to be nearly as tight.

As for the other technology—transmitters—the design constraint comes from a different direction. We need to make our satellites weigh as little as possible since we've got to pay to get any weight into GEO. In general, transmitters of the same power are more complicated and therefore weigh more as the frequency goes up. By using a lower frequency, we can use a lower-weight transmitter on our satellite. It's really nice when design constraints like these—antenna size and transmitter weight—work together to come up with the same solution for determining which frequency band to use going up and coming down.

We've got one last thing to clear up before we finish our discussion of the bent pipe. In this chapter, we discussed receive antenna and transmit antenna separately, since their functions in the bent pipe were quite a bit different. However, as you may have noticed in our drawing of the Echo XIV antenna configuration, there weren't antenna labeled "receive" or "transmit." That's because we actually use the same antennas to do both jobs. If you compare Figure 3 (p. 12) and Figure 97 (p. 124), you'll see that all the uplink centers lie in completely different spot beam footprints. Since those spot beams are related to separate feed horns, we can use those different feed horns as receivers for the different uplink centers at the same time as we're using them to downlink metropolitan area-specific content to those footprints. It's a really efficient way to do business, considering what the cost of launching the weight of separate receive and transmit antennas all the way up to GEO would be. That fact should

also make it even more apparent why we have to use different transmit and receive frequencies.

A Brief Return to Content Acquisition

Now that you've got a good handle on antennas and footprints, we need to revisit a topic we skipped way back in the *Prepping the Signal* chapter: frequency reuse for uplink stations. Remember how we discussed the fact that we use two primary and four secondary uplink centers (Cheyenne/Gilbert and Monee/Mt. Jackson/New Braunfels/Spokane)? Well, the reason for that will be quite apparent now that we've talked about spot beams for downlink.

Most of our local channels are sent to the secondary uplink centers nearest them. Those uplink locations are spaced more than a receive-antenna footprint apart, so more than one uplink center can send signals to the same satellite. The satellite can then route those signals to the appropriate transmitter so they end up in spot beam for the right local market. There's obviously a lot of routing going on in the satellite that we won't cover here, except to expose you to the fact that it exists and why it needs to happen. The bottom line is that we've got a very limited bandwidth to use. Spot beams allow us to reuse frequencies for both uplink and downlink, making our operations much more efficient.

Chapter Summary

And with that, we come to the end of this chapter on DISH Satellites. We've covered quite a lot of ground, starting with some basic thoughts about gravity to figure out why our satellites had to be so darned far away and over the equator. We saw that in order for us to be able to have very cheap, fixed antennas that we could put on our customers' roofs, we had to make our satellites appear to be stationary. We now know they're not really stationary, but are at just the right altitude so that they go all the way around the Earth in 24 hours, the same time it takes the Earth to turn on its axis. They're not really stationary at all—they just look that way from the ground.

We then looked at some of the major subsystems on most satellites such as the solar cells and the thrusters that keep the satellite pointed in the right direction, but we soon realized that they were just there to support the money-making bit: the bent pipe. We saw that the bent pipe really was made up of three things: receive and transmit antennas and a transponder. The receive antenna's job is to take the weak signal from the ground, concentrate it, and send it on to the transponder. It needed to be large to collect as much signal as possible and so

that its beamwidth was as narrow as possible to prevent interference from transmitters on the ground other than our uplink center.

The transponder's job was just to shift the frequency of the transmitted signal away from that of the received signal so it didn't interfere with and overwhelm the incoming signal. It did that through the use of an electronic mixer which uses a precise local oscillator and a filter to output the frequency-shifted signal.

We then saw how the transmit antennas are designed to send the signals where we could make money from them. For the large CONUS beams, we saw how the surfaces of those antennas are deformed from pure parabolas so they can concentrate their beams in non-elliptical footprints that don't send signal to the oceans or other countries where we don't do business. We also saw that larger antennas could be used to form spot beams to deliver specific programming to local regions.

That was a lot to cover, but it's definitely not the end of our signal's journey. In the following chapter we'll continue following our signal to the next place it comes to land: at the small dishes on top of our customers' roofs.

4. The Dish on Your Roof

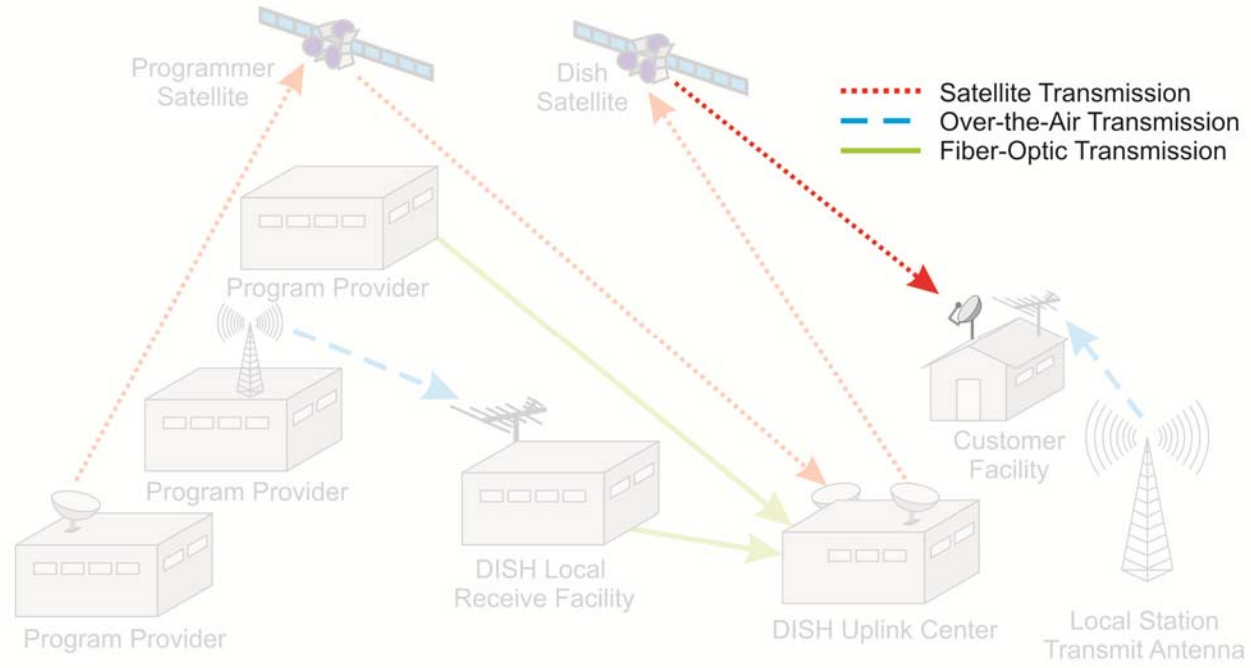

Figure 101: The Signal Flow Roadmap—Customer Reception.

Whispers from the sky
Grow strong and are directed
Deep inside a house.

The Basics

(Continued from p. 97) They are perhaps the most ubiquitous and visible feature of our company: the millions and millions of small dishes attached to rooftops, walls, balconies, fences, and hundreds of other objects around the country. Each of them bears the DISH logo, silently advertising our wares. We tend to take these icons for granted. After all, what could be simpler than a curved piece of metal pointed toward the sky? Even the overview diagram for this chapter, Figure 101, is as simple as can be.

But that simplicity is deceiving. There's an entire world of technology hidden inside the dish, technology without which our business couldn't exist. In this

Figure 102: Rooftop Dishes Patiently Receiving Programming.[69]

chapter, we'll look at some of that technology and see how it really is a critical link in connecting our customers to their programming.

We'll pick up where we last saw our signal as it left the transmit antennas of our satellites, following it through space then air then clouds and rain down to where it finally touches the dish antenna. That trip contains many perils that threaten to prevent signal reception. We've designed our system to overcome most of these perils, but there are some we actually surrender to and accept temporary signal loss. However, for the vast majority of the time for the vast majority of our customers, our services are delivered seamlessly, with all of this technology being, as it should be, transparent to their experience.

We'll see the signal transformed by components of the dish from the waves that traveled so far through space into a form that can pass through the small black or white cables connecting the dish to our receivers inside homes and businesses across the country. We'll also look at the components that help direct that signal to the places it needs to go, all in preparation for our final chapter where our signal finally reaches our customers' screens. Ready to get started? Here we go!

A Link Budget Overview. Let's start our discussion in this installment of *The Basics* with an examination of the statement we made about not being able to deliver signal to all customers at all times. We just can't be 100% effective in delivering programming because that would be prohibitively expensive. The bottom line is that in order for our signal to be useable to our customers, we've got to make sure the signal has a minimum strength when it reaches the antennas at the focal point of the dishes on our customers' rooftops. If the signal is stronger than that amount, they get their programming. If not, they call and complain. The technical term summarizing all the math we have to do to figure out whether we'll have that minimum signal strength is a *link budget*. Link budgeting means we have to add up all the positive things we do to increase our signal strength and subtract off all the things that happen to the signal to make it weaker to find out what strength it has when it reaches our antennas. If it's not

strong enough, we have to redesign some of the hardware we control until the signal is sufficient to meet our business goals.

There are a lot of things we can do to increase signal strength, but all of them cost money, with more strength invariably costing more money. Many times, spending increasing amounts of money only results in marginally increasing amounts of signal strength. The law of diminishing returns kicks in and business sense takes over to prevent directing lots of cash to places it won't do that much good. For example, let's say we wanted our main beamwidth to be sized perfectly so that it was the same size as the receive antennas on our satellites. Sounds reasonable, doesn't it? We wouldn't be wasting nearly as much of the signal we do now, when our beam is about 30 miles wide by the time it gets to the geostationary belt. Using a frequency near the middle of the Ku band, as we must because of FCC licensing issues, and the 12-foot receive antenna on the satellite, we'd need to build a transmit antenna *over 100 miles across* to meet our beamwidth goals. If *that's* not prohibitively expensive, we don't know what is! We wouldn't want to be the engineers who took that proposal to senior management, as they'd likely be looking for jobs shortly after the briefing.

In *The Uplink Center* chapter we've already discussed most of the things we can do to make the effects of the signal-weakening factors as small as possible. We've selected a wavelength/frequency band that's in the radio window, where the atmosphere is relatively transparent. (Remember, frequency and wavelength are just two sides of the same coin.) We've chosen a band that has just as short a wavelength as possible to minimize spreading of the beam. Within those constraints we've done pretty well, so we'll just have to live with what nature still takes away from us. With those limitations in mind, let's look at the chain we have to make unbreakable in order for us to deliver signal to our customers 24/7/365. In this way, we'll see the factors—good and bad—that have to go into our link budget.

The "good" comes in four parts over which we exercise control: transmitter power, transmit antenna size, receive antenna size, and receive antenna sensitivity. To make these things "good-er," we can increase all of them. But as we've already noted, increasing them costs money. If our link budget tells us we don't have enough signal to provide a satisfying customer experience, we'll have to do a trade study to find out the most economical and effective way to increase one or more of these factors until our link is sufficiently strong.

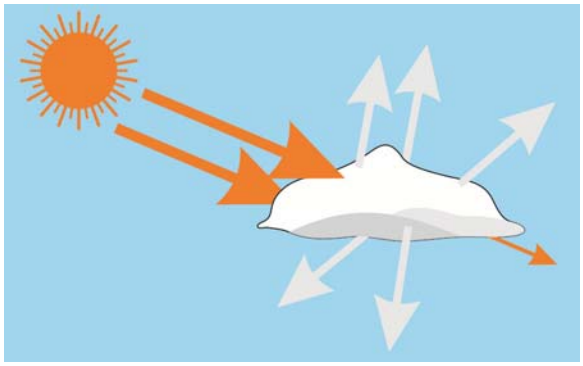

Figure 103: Clouds Scatter Our Signal.

Now let's look at the "bad." We really don't have much control over this part, once the wavelength has been chosen. But we've got to design the "good" to overcome the "bad."

Rain and Snow Fade. Our signal has to pass through our atmosphere, both on the way up to and on the way down from our satellites. As we saw in *The Uplink Center* chapter, even clear air in the atmosphere isn't completely transparent to the wavelengths of electromagnetic radiation, the Ku band, we have been licensed to use. Portions of the air, primarily water vapor and oxygen molecules, absorb some of our signal so there are losses there. And the discussion we've already had didn't even deal with clouds or rain and snow!

In addition to the losses due to clear air, *visible* moisture blocks visible light from space, as anyone who's tried to see the Sun on a cloudy or foggy day can attest. It also blocks Ku-band signals, although not to as great an extent. We call the losses due to visible moisture *rain and snow fade*, and that fade can be quite significant depending on how much moisture is suspended along the signal's path. Our signal weakens as it passes through moisture because rain, snow, and clouds all tend to scatter our signal as it passes through them, as shown in Figure 103. The heavier the rain and the longer our signal's path is through that rain, the more the signal disappears. We do our best to have strong enough transmitters on our satellites and sensitive enough antennas on the ground to overcome rain fade, but sometimes the rain is just too heavy and we lose signal. Again, we could build better, stronger, more sensitive equipment but eventually the cost becomes prohibitive. Very heavy rain is one of the factors that means we can't guarantee programming to our customers 100% of the time.

Solar Interference. Another factor that we just can't overcome is the Sun. Now, most of the time the Sun doesn't play in our satellite communications game at all. However, the fact that our satellites are in the geostationary belt above the equator means that twice a year, in March/April and September/October depending on the latitude of the receiving dish, our satellites appear to pass in front of the Sun. When they do, the Sun's radiation goes right down into our dish and drowns out the signals from our satellites. While these alignments only last

for a maximum of 14 minutes per day over the course of eight or nine days, there's nothing we can do to prevent these outages. The Sun is just too bright, as shown in Figure 104. We pretty much surrender to nature on these twice-yearly occasions and don't even try to build in countermeasures for them in our link budgets. Notice these solar outages only occur on the downlink, since the uplink receive antennas on the satellite are always looking at the ground and will never see the Sun.

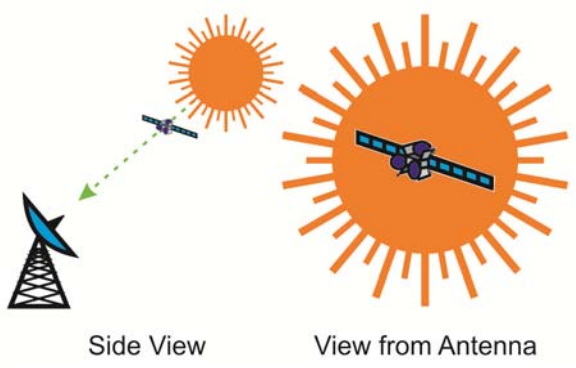

Figure 104: Sun-Satellite Alignment: Which One is Brighter?

Distance Losses. Interference from the Sun and atmospheric losses aren't the only perils we've got to overcome, either. As we learned earlier, as much as we try to focus our transmissions into a very, very narrow beam, it still spreads out. We try to limit this spreading by making our antennas larger, but as hard as we try we never can really make a beam that doesn't spread. The hundred-mile-across antenna example we used a little while ago should prove that.

Anyway, as the beam spreads out, the fraction of it that hits the satellite becomes less and less the further away the satellite is from the transmitting antenna, as shown in Figure 105. This loss is called the free-space loss, or in non-tech-speak, just a loss due to distance. In the figure, the antenna, beamwidths, and satellites are all the same for both the left and right systems. The only difference is the distance between the satellites and the antennas. The antennas are both sending out the same amount of energy into the triangle that represents the beam here.

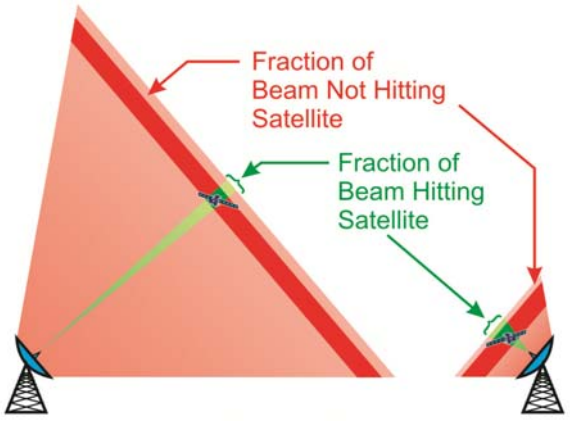

Figure 105: Signal Losses Due to Beam Spreading.

As you can see, the fraction of the beam hitting the satellite from the left antenna, the size of the green portion of the beam at the distance of the satellite compared to the size of the red portion of the beam, is pretty small compared to the green fraction from the right antenna. Since that fraction of the beam area is smaller, the amount of energy hitting the satellite is smaller. Larger distances mean greater losses, and the fact that we have to put our satellites

over 22,000 miles away in the geostationary belt means we always have large distance losses to deal with. As we've said, making larger transmit and receive antennas are a way to counter this kind of loss, but those counters are never even close to 100% effective, as our 30-mile-wide-at-GEO beam will demonstrate.

Link Budget Summary. Let's summarize the things that go into the link budget, first looking at things we can do something about and then listing those detracting factors that nature throws in our way, based on the frequency bands we have been assigned by the FCC. The good (things we can control and spend money to improve):

- Increasing transmitter power
- Making larger transmitting and receiving antennas
- Making more sensitive receiving electronics.

The bad (natural givens):

- Atmospheric scattering and absorption
- Precipitation scattering and absorption
- Large distances between transmitter and receiver
- Solar Interference.

While calculating a real link budget involves a great deal of math, conceptually these factors are really about all that's involved. Now that we've looked at the link, let's talk about some hardware.

The Thing We Call the Satellite Antenna. The most noticeable part of the thing we call the satellite antenna is the large, curved dish. In *The Uplink Center* and *DISH Satellites* chapters, we discussed how these dishes are shaped like parabolas to focus the transmitted beam and to collect the received beam. We also learned that bigger was better when it came to dishes. The dishes on our roofs are much smaller than the dishes at the uplink centers, but they're big enough for most of our needs. They are the part of the antenna that collects much of the very weak signal that finally makes it down to Earth from the transmitters on our satellites.

Did we say very weak signal? You may be wondering just how weak it is. Our satellite transmitters are rated at between 150 and 450 watts, but remember that power is shared equally by 32 transponders. If we're trying to see an individual transponder it's like trying to see a 5 to 15 watt night light from over 20,000 miles away—and that's not counting any losses due to the atmosphere or precipitation! When you think about it, it's amazing that we get any television reception at all!

That's where the dish comes in. Without a dish, you'd only be looking for that light bulb with a very tiny "antenna"—the size of the pupil in your eye. Our satellite dishes have an area about 40 thousand times as large as your pupil, so if you had that kind of light-gathering ability, you'd effectively be looking for a half-million-watt light bulb instead. Since that light bulb is way out in GEO, it's still a difficult task but not nearly as bad as looking for a night light. That's what the dish does—it concentrates the signal from the satellite to make it easier to "see" by focusing the entire signal hitting its parabolic shape onto the parabola's focus.

However, the dish itself is only a very minor (but essential) part of the antenna. The brains of the operation are actually out on the end of the pole attached to the dish, situated at the focal point of the parabola, as shown in Figure 106. Out there, we've located what's technically known as a low-noise block down-converter. However, most people just call it an *LNB*.

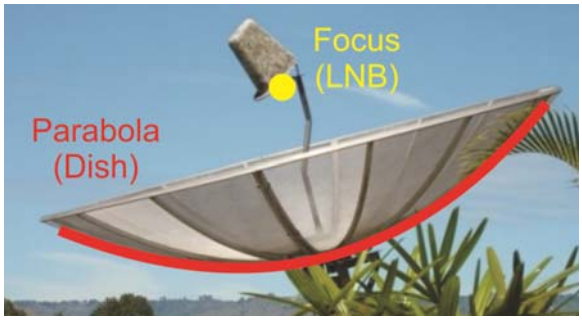

Figure 106: The LNB is at the Focus of the Dish.

The LNB is a very sophisticated bit of hardware that does several very important things. First, it's the place where the actual antenna is located. The antenna takes the electromagnetic waves that travel through space and turns them into electrical signals that can be processed by electronic devices. After receiving the signal with the antenna, the LNB amplifies them and then shifts their frequencies into a range that can travel through the coaxial cables that connect many of the devices we have in our homes. This frequency-shifting is a critical function, as our engineers have come up with some ingenious ways to have a simple cable carry two different signal bands at once without them interfering with each other—and that's no easy task.

With the advent of more modern receivers and the desire of our customers to have fewer holes poked in their walls during installation, we've developed a number of additional devices that pack even more signals on a single cable. These devices, known as *nodes*, shift wavelengths yet again, enabling the cable to carry a third set of signals inside the house. There are other *signal flow devices* than just nodes, but nodes are the most sophisticated. If you're interested in the others, they're discussed in the *Technical Discussion* below.

As we'll see in the next chapter, some receivers have more than one tuner and can supply programming to more than one television set through a system called home distribution. With the appropriate signal flow devices in place, those receivers can move that signal back down the same cable that's bringing in the signal from the LNB but on another frequency band. Our Hopper and Joey receivers also use a fourth section of that cable to send yet another set of frequencies between them through a system called MoCA (*MOE-cah*), but since they don't use the home distribution system those frequency bands don't actually ever overlap.

Figure 107 shows the bands we use at DISH and what they're used for. Since a coaxial cable isn't very good at moving signals that oscillate up and down much more rapidly than 3 billion times a second (technical people would say 3 gigahertz, 3 gig, or 3 GHz), and since we can't send two signals in the same frequency band at the same time, you can see from the figure we're rapidly approaching the limit of what can be sent simultaneously on a single cable with our existing technology.

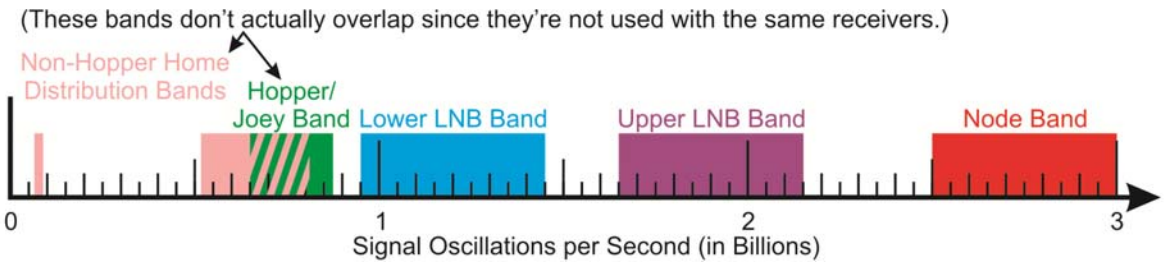

Figure 107: How DISH gets Maximum Use from a Single Coaxial Cable.

Summary. Fast and furious: that's a good summation of this overview of a quite intricate process. As usual, we recommend a deeper dive into this subject by reading the following *Technical Discussion*, but if you're pressed for time here's what we think you should have gotten out of this installment of *The Basics*:

- The dish is only part of our antenna system, the part that concentrates the signal so the actual antenna has a chance of detecting our very faint signals from space
- The LNB contains the actual antennas, and after receiving the electromagnetic waves from our satellites, it shifts their wavelengths into forms that can be sent over a coaxial cable
- We use signal flow devices to help put our signals where they need to go inside the house

- Some signal flow devices shift wavelengths of our signals again so we can send more of them over the same cable without the bands overlapping
- The path down from space is full of perils that threaten to degrade our signal to the point where our antennas can't find it
- Some of these perils include rain, clouds, the Sun (only at very specific times of the year), and the large distances between our satellites and the dishes on our roofs.

If you're only reading The Basics, you can continue your study on page 189.

Technical Discussion

While this chapter's title may lead you to believe we'll be talking about dishes, there's more here than initially meets the eye. We'll discuss the dish for just a bit, and then will show why it needs to be as large as it is. That will lead us to a discussion of what engineers call a link budget, a calculation that helps them figure out just how large, powerful, and/or sensitive our hardware needs to be to ensure a good customer experience. We'll then move past the dish to discuss how the signal gets moved inside customers' homes, a surprisingly complicated process.

The Satellite Dish

We've probably beaten this dead horse enough in previous chapters, but your basic rooftop dish is just a smaller version of the parabolic dishes we use at DISH uplink centers and on our satellites. The major difference is that they're much smaller, typically two to three feet across. Hopefully you'll all remember that smaller antennas have much wider beamwidths (and smaller gains), so they can take in signal from a much broader set of angles than larger dishes. However, don't be fooled into thinking that's a really broad look at the sky. A three-foot dish has about a 3° beamwidth (and a gain of about 250). To give you some scale, the full Moon's diameter as seen from Earth is about half a degree. Frame I of Figure 108 shows this relationship.

What does this beamwidth mean for us operationally? It means that we can't put our satellites closer together than 3° as viewed from the surface of the Earth (equivalent to about 2.5° of longitude, since longitude is measured from the center of the Earth and beamwidth from the Earth's surface). If we did, the downlinked signals both satellites use would end up at the same focal point for the parabola—and since we reuse the same frequencies on all of our satellites, that means a significant interference problem. If they're that close together, both

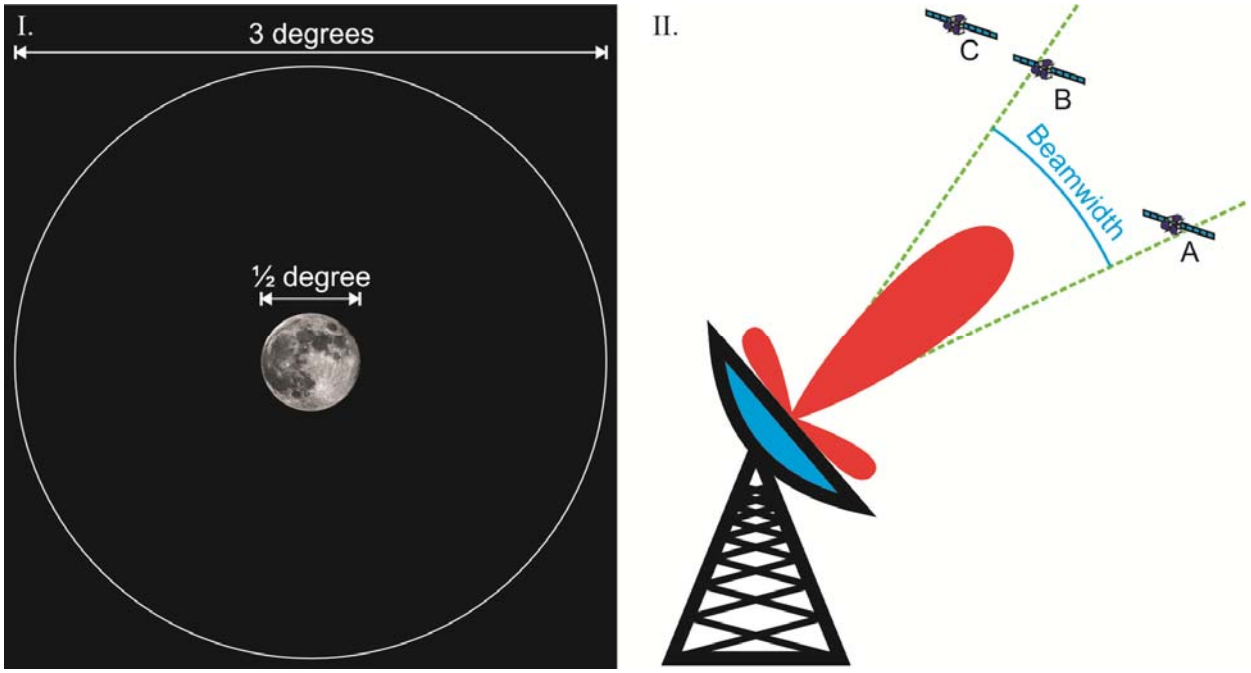

Figure 108: Beamwidth of DISH Rooftop Antennas.

of them could possibly be in the beamwidth of the antenna at the same time, as shown by the satellites in positions A and B in Frame II. Satellite C is more than a beamwidth away from satellite A, so they can't interfere with each other.* Of course, if satellites A and B used different downlink frequencies, the problem is moot.

Figure 109: A DISH Rooftop Antenna.

Just as we discussed when we talked about spot beams and multiple transmit feed horns on our satellites, we can use multiple feed horns on our receive antennas. For reasons we'll explain later in this chapter, we call these specific feed horns LNBs, or low-noise block down-converters. Figure 109 shows an example of one of our antennas with three LNBs designed to see the three satellites in the

* The observant reader may remember that we have a Western Arc satellite at 119° W and our international programming satellite at 118.7° W. OK, even we're not that observant, so we'll just remind you there are satellites in those positions. The international satellite actually uses a different frequency band that we didn't discuss—14-14.5 GHz uplink and 11.7 to 12.2 GHz downlink. Those different frequencies are required to prevent interference between satellites, but the Ku-band physics we studied still applies.

Figure 110: A Mesh-Style C-Band Antenna.[70]

DISH Western Arc. The LNBs are located away from the parabolic dish's focus, so their receive beam boresights are pointed in different directions. Notice that the longitudes of the satellites are all separated by much greater than the 2.5° required for effective frequency reuse. Also notice that the dish is solid.

As a brief digression, you may have seen some large C-band satellite dishes that were constructed of wire mesh, such as the one shown in Figure 110. This technique is used to reduce wind loading, weight, and snow accumulation. The lighter load from mesh construction also reduced the required size of the actuators these older systems needed to point the dish at different satellites, since they only had single LNBs. In order for a mesh antenna to work, the spacing of the mesh must be small with respect to the signal's wavelength so the dish can reflect most of the incoming signal.[71] Waves that are long *with respect to the mesh spacing* essentially "see" a solid surface and reflect from it as if it were solid. The frequencies our eyes see are very small compared to the mesh spacing, so we see right through it.

Surface smoothness for the dish must be better than about a sixteenth of a wavelength.[72] For C-band wavelengths of 1.5" to 3", this constraint translates to a mesh spacing of no greater than about 1/16" to 3/16". Typical perforations in mesh-type C-band antennas are less than ½" across, which is in the 1/3 to 1/6 wavelength range, so design inefficiencies have obviously been allowed. One other drawback of mesh antennas with larger-than-optimum mesh spacing is that they increase the background temperature the feed horn sees because heat from the Earth leaks through the mesh. As we'll see shortly in the *Uplink and Downlink Differences* section, higher temperatures mean reduced antenna performance.

Construction of mesh type Ku-band dishes is less practical because the mesh spacing would have to be less than 1/32" and the wire used to make them would need to be very thin and fragile. Ku dishes designed for the improved, higher-

power satellites in current use also don't have to be nearly as large as the older C-band dishes, so the need for weight reduction and wind loading aren't as much of a problem. The additional cost of building a 1/32" precision mesh dish compared to the cost of a small solid dish therefore just isn't worth it.

Before we end this section on rooftop dishes, we've almost forgotten one of the most basic rules of satellite TV, one that may seem quite obvious: we need a clear path between our dishes and our satellites. While it would be nice to be able to install our receive dishes without regard to buildings, trees, and bushes, that isn't possible. Electromagnetic radiation will not penetrate foliage reliably at frequencies above 400 MHz (tropical forests) to 1 GHz (temperate forests without liquid water on the leaves),[73] and they won't penetrate buildings or glass at all. Thus, we have to have an unobstructed view of all Eastern or Western Arc satellites at our installation point.

That's really about all we need to say about the satellite dish itself. But before we go on to discuss more of the hardware DISH uses, we need to back up a bit to look at some things that can affect our signal between the satellite and the rooftop antenna.

Rain/Snow Fade

We spent a lot of time in *The Uplink Center* chapter talking about electromagnetic radiation, concentrating on absorption to help us understand why we chose the Ku band for DISH operations. However there are two more ways that the atmosphere influences our goal of delivering quality DISH programming to our customers. Those phenomena are atmospheric scattering and absorption. Absorption happens regardless of whether there are actual raindrops. At radio frequencies scattering is of little concern except when there *are* rain drops or snowflakes in the air.

Scattering. We discussed absorption at some length in *The Uplink Center* chapter (see p. 52 for a review, if necessary). *Scattering* differs from absorption in that with absorption, the radio wave energy goes *into* the atoms in the atmosphere where it is eventually lost as heat energy. With scattering, some of the signal energy is *redirected* to different paths, meaning it doesn't travel in the straight line to our antenna that we'd planned. Figure 111 shows this concept, where millions and billions of raindrops in the storm combine to weaken the satellite's signal.

The heavier the rain, the more pronounced the scattering effect. Relative reductions in signal strength due to rainfall are shown in Figure 112 (hint—look at the scale on the left side of the figure, not the right), and as in previous figures the red Ku band is the one we're most interested in. Notice that for the tropical downpour (curve 7) we lose about 90 percent of our signal strength for every kilometer (about 3000 feet) of the storm the signal has to pass through. That's a really tough challenge to meet—OK, it's all but impossible to deliver signal in those conditions.

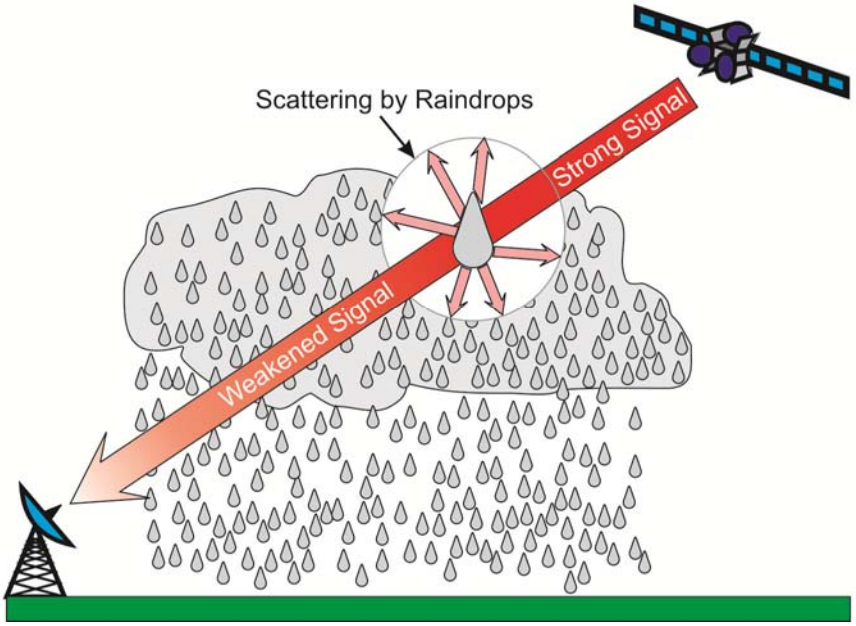

Figure 111: Rain Scattering of Satellite Signals.

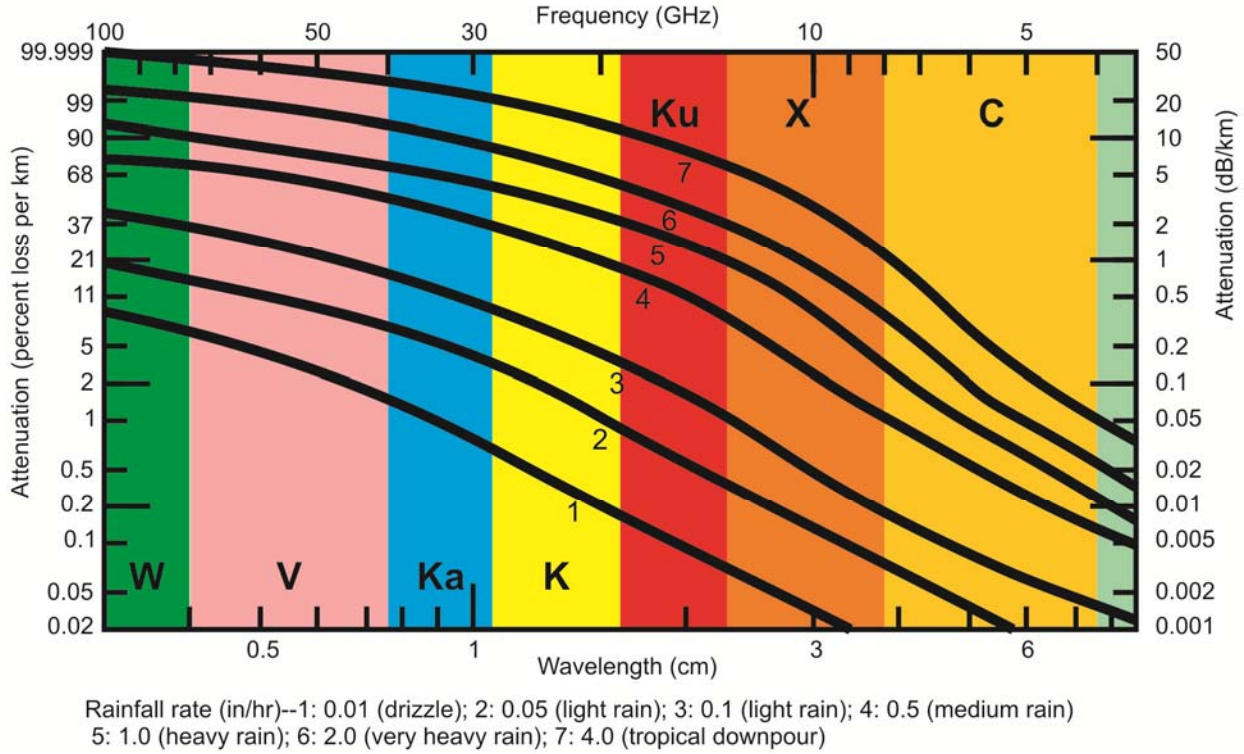

Figure 112: Atmospheric Attenuation Due to Rainfall.[74]

However, by the time we get down to rain falling at a still-respectable rate of half an inch per hour (curve 4), we only lose about 10 percent of our signal per kilometer, and that seems much more doable.

As the realtors say, it's all about location, location, location. What do we mean by that? Not all rainfall is the same because of the geometry of our downlink beams. Let's use our go-to satellite at 129° W again—the one that's due south of the easternmost part of Alaska. The closest major city we serve to that satellite is San Diego. From there, the satellite is almost 23,300 miles away (note that it's further than the 22,000 miles we cited as geostationary orbit altitude because it's further to the Earth's surface at San Diego's location than it is straight down to the equator). Since San Diego is so close, you have to point your dish pretty high up to hit the satellite—50° above horizontal to be exact.

On the other end of the scale, Portland, Maine is about as far as you can go in our 129 satellite's service area to find a major city. Portland's straight-line distance to the satellite is almost 25,000 miles—about 1,650 miles or 7% further than San Diego. It's so far away that the satellite is barely above the horizon, with an elevation angle of only 14°.[75] Another look at Figure 97 (p. 124), the spot beams figure, and the near-circular vs. highly-elliptical footprints in California and Maine might help you visualize these distance and elevation angle differences. Let's see what the differences in those two elevation angles mean to us when we're talking about rainfall.

Figure 113 shows the geometry of downlink beams from these two locations. We've set up a fairly challenging precipitation situation: a solid layer of rain blanketing the entire region from the surface all the way up to 20,000 feet. Notice how much further the signal headed to Portland, with its lower elevation angle, has to travel through this layer. That extra distance is significant because, as we saw in Figure 112, the attenuation due to rain (and snow) was measured in percent loss *per distance*. The longer the distance, the greater the loss. Figure

Figure 113: Location Influences Rain Effects.

114 compares the cumulative effects of rain loss across these two paths. While both systems deal well with drizzle, Portland is effectively blocked for anything much heavier than a tenth of an inch per hour. The numbers show that San Diego is much better off. However, notice just how sensitive our signals really are to even relatively common amounts of rain. Precipitation is definitely not our friend!

Rainfall Type	*Rainfall Rate (in/hr)*	*Percent loss per kilometer*	*Total Percent Loss at San Diego*	*Total Percent Loss at Portland*
Drizzle	0.01	0.1%	0.7%	2.3%
Light Rain	0.1	1.6%	12.0%	33.4%
Moderate Rain	0.5	10.9%	60.0%	94.5%
Heavy Rain	1.0	22.4%	86.7%	99.8%
Tropical Downpour	4.0	80.0%	100.0%	100.0%

Figure 114: Total Rainfall Losses at Two Locations.

Uplink and Downlink Differences. Now, you may be wondering why we didn't discuss the effects of rain in *The Uplink Center* chapter, since the signal has to go through rain on the way up just like it does on the way down. Well, there are a couple of reasons we saved it for this chapter that deals with downlink. First, we've smartly located our primary uplink centers in locations where it doesn't rain or snow all that much: Wyoming and Arizona. There really is method to that madness. Second, scattering and absorption due to rain affect the downlink much more than the uplink. The reasons for this difference are rather complicated, but basically it depends upon the background temperature where the *receiving* antenna is pointing.

Why do we care about background temperature? It's because it's buried in the rather involved equation for the signal-to-noise ratio, SNR.[76] We want the SNR to be as high as possible (high SNR means signal is big and noise is small—see the discussion related to Figure 61, p. 80, if you need to refresh your memory about SNR). If we only look at temperature as a variable in the equation for SNR and treat all of the other factors as constants, the equation becomes SNR = k/T, where *k* is the ever-present arbitrary constant and *T* is the background temperature. Remembering our easy-math rules, if we want SNR to be big, *T* needs to be small. The lower the temperature, the lower the background noise level.

When rain enters the picture for the uplink, the receiving antenna is on the satellite looking down. The temperature of the rain isn't all that different from the temperature of the hot background of Earth, so *T* doesn't change all that much between the clear-air and rain situations. However, there's a huge increase

in *T* on the downlink (with the upward-looking receive antenna). Although clouds and rain are cold, they're nowhere near as cold as the background of space. The huge increase in *T* seen by our rooftop dishes on rainy days means that SNR goes down markedly for the downlinked signal when there's rain present. When SNR is low, it's more difficult to receive the information contained in the signal.

In addition to absorbing and scattering our signal, rain also affects the signal's phase a bit,[77] and since we use phase-shift keying, anything that affects phase is of concern to us. Remember that with 8PSK, we're squeezing eight different levels of phase into each symbol we send (if you need to, see Figure 52, p. 71, to review how increasing the number of levels makes our error margin decrease). That means we've got very little margin for phase error, and the error induced by rain can mean the difference between understandable signal and gibberish. Phase shifting of our signal by rain is another way the level of noise in the system Is increased.

In addition to the extra noise we get from the temperature difference between looking up and looking down through rain, there's another reason why rain affects the downlink signal more than the uplink. That reason is an operational and economic decision. We don't operate our uplink transmitters at maximum power all the time. They're built with some margin in their designs to allow us to increase power for short amounts of time for the very purpose of overcoming the reduction in signal we've now seen is caused by rain and snow. With that additional power available to them, uplinks can almost always overcome problems caused by rain and snow.

In contrast, the transmitters on our satellites are on very tight power budgets. We can't afford to send all the extra weight and complexity up into geostationary orbit it would take to be able to change the power levels in response to atmospheric conditions. Satellite transmitter power levels are thus relatively fixed, and since we can't increase their power as needed, the downlink is what it is, regardless of whether it's clear or raining.

So, precipitation can have a huge effect on whether we deliver our product to our customer. However, it's not the only thing that can degrade our signal during downlink before it gets to the rooftop antennas.

Solar Interference

One of few the drawbacks about having satellites in fixed locations over the equator is that twice a year the satellites pass between the Earth and the Sun. As

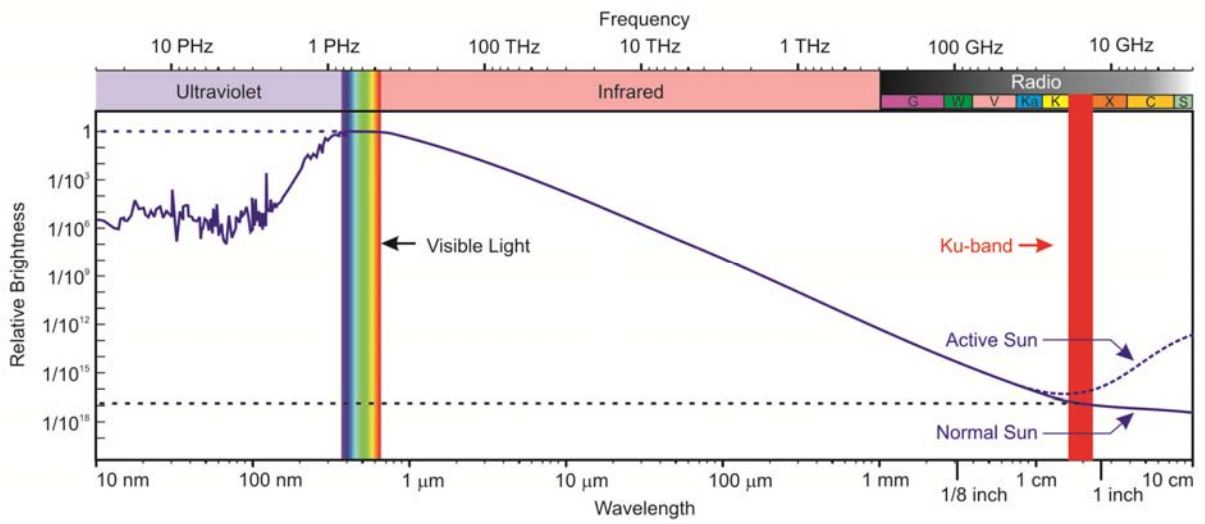

Figure 115: Frequencies of Electromagnetic Radiation Emitted by the Sun.[78]

shown in Figure 115, the Sun doesn't just emit visible light but rather it emits electromagnetic radiation across a broad spectrum of frequencies. In contrast to Figure 34 (p. 52) which showed the amount of radiation that made it through the atmosphere to the Earth's surface at different frequencies, this figure shows the spectrum of radiation the Sun actually emits. The vertical scale is calibrated to show the "brightness" of the Sun at different frequencies relative to its peak brightness, which occurs in the visible range. This vertical scale is also calibrated such that each division is ten times larger than a similarly-sized one below it, so the apparently small vertical change in the curve between the visible band and the Ku radio band is actually huge.

Reading across on the lower dotted line, you can see the Sun is only a *billionth of a billionth* (that's a one divided by a one with eighteen zeros after it, or $1/10^{18}$) as bright in our satellite band as it is in the visible. However, remember our transponders only emit the equivalent of a ten-ish watt night light, and they're at least 22,000 miles away. That's a pretty weak signal, weak enough that even the Sun's extremely weak radio emissions would still overwhelm it if our rooftop antennas were pointed directly at the Sun.

When the Sun's signal in our antenna is stronger than our satellite signal, it's commonly called **solar interference**. Frame I

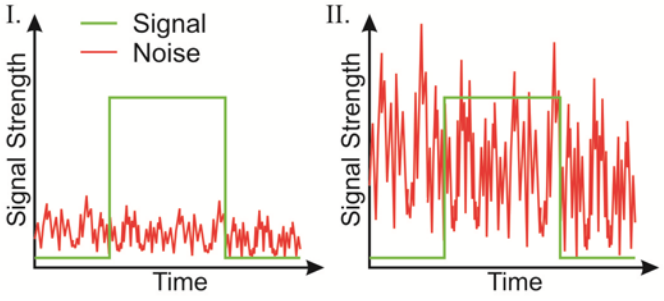

Figure 116: Solar Noise Overwhelms Our Satellite Signal.

of Figure 116 shows a representative signal level when compared with normal background noise in the Ku band. Here the SNR is pretty high. However, when the Sun is directly behind the satellite in the rooftop antenna's beamwidth, the SNR drops markedly and its noise overwhelms the satellite's signal, as shown in Frame II.

The reason for the twice-yearly solar interference phenomenon is demonstrated graphically in Figure 117. Note the distance to the Sun and the relative sizes of the Sun and the Earth are not to scale in the figure. Frame I shows how the Earth revolves around the Sun on its annual trip. The position of the Earth at its equinoxes (the days when the day and night are of equal length) and its solstices (the days when the day or night is the longest) are shown in this frame. While we're looking down on the Earth/Sun system in this view, we'll look at it from the side in the remaining frames, and the observer's position for Frames II and III is

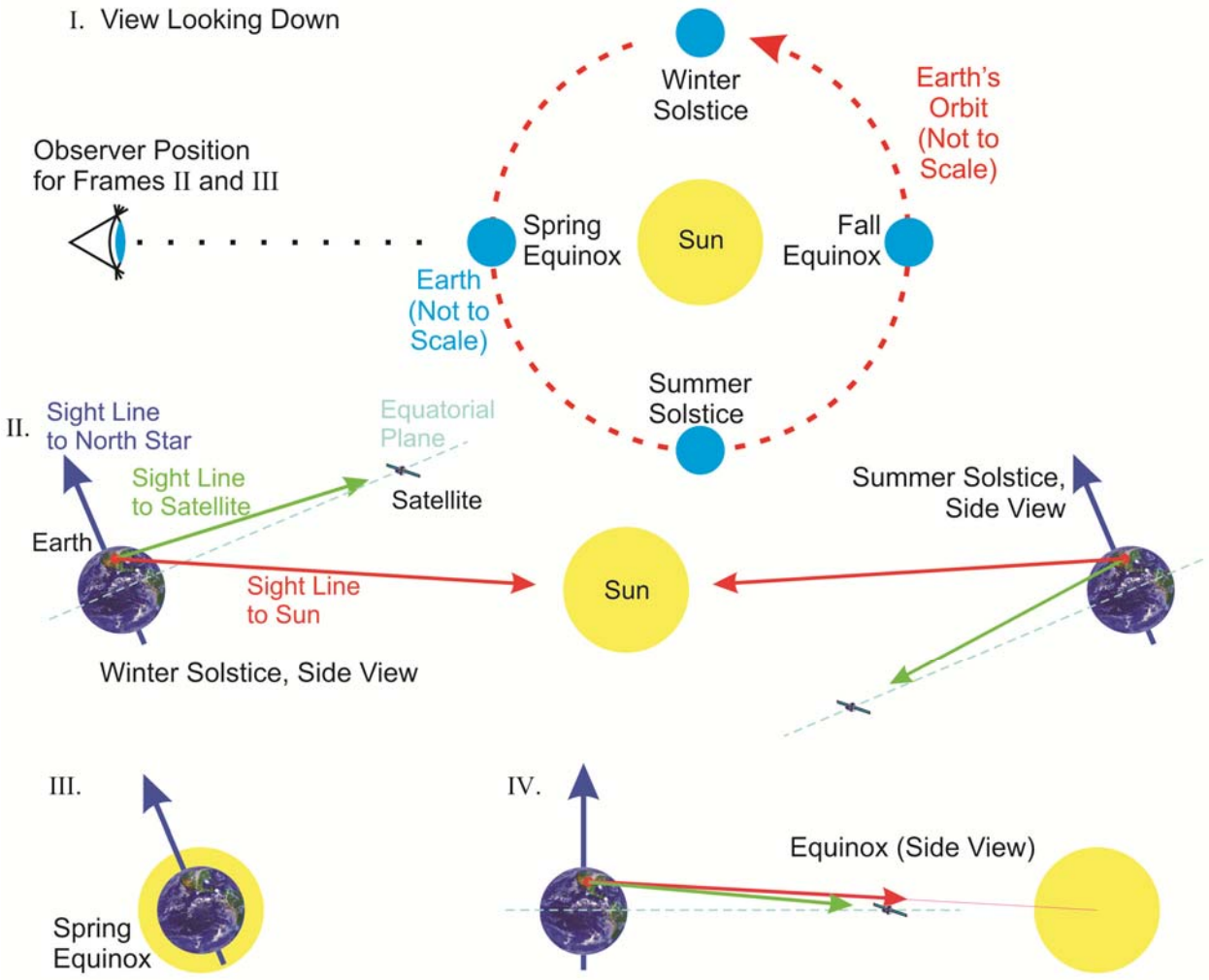

Figure 117: Geometric Explanation of Solar Interference.

shown in Frame I.

In Frame II we've moved to the observer's position and can see the Earth at both the winter and summer solstice positions. Notice that the Earth spins on an axis that is tilted 23° from the plane it orbits around the Sun. Also note that the direction of the north pole points in the direction of the North Star in both the left and right views of the Earth; the direction of that axis doesn't change as the Earth moves around the Sun. The left side of this frame shows the situation at the (northern hemisphere) winter solstice, where the noon Sun is at its lowest point in the sky. At this time for the observer in Denver, located at the red dot, a satellite in geostationary orbit directly toward the Sun would be seen significantly *above* the Sun's location, as shown by the difference between the sight lines to the satellite and the Sun.

Six months later at the summer solstice, the situation has changed markedly. The noon Sun is now just as far north in the sky as it gets. A GEO satellite directly toward the Sun would be seen significantly *below* the Sun. The sight line to the satellite didn't change (else we'd have had to move our fixed antenna to track it, which runs counter to the whole point of putting our satellites in GEO in the first place). The sight line to the Sun, however did change.

Since in one case the satellite appeared above the Sun and in the next case it was below the Sun, at some time in between these two solstices the sight lines must have been along the same direction! It turns out that time is very near the equinoxes, the times in the spring and fall when the day length is the same and the Sun appears to be directly over the equator. We've drawn this situation in Frame III from the same vantage point as Frame II, but from that point of view all of the interesting bits of the figure are still between the Sun and the Earth, so they're not visible from here.

Frame IV shows the equinox situation but with our vantage point moved 90° so we can see the interesting bits (either near the bottom or the top of Frame I, depending on whether you want to look at the spring or fall equinox—both cases would look like this frame). Don't worry, the sight line to the North Star didn't change; it's just tilted into or out of the page a bit but it appears to be straight up and down from here. Notice that now the sight lines from the observer to the Sun and the satellite are much closer together. It turns out that only when the observer is actually on the equator is the time when the sight lines coincide exactly on the day of the solstice. Imagine a second observer on the blue dashed line looking at the satellite and you'll be able to see this. The further an observer

is north or south of the equator, the further away from the equinox is the time of coinciding sight lines, which is why in this frame, right at the equinox, the two sight lines for our observer in Denver don't quite match up.

Figure 118 shows a plot of the date of maximum solar interference for different latitudes. For example, Denver is at a latitude of about 40° North, so a receive antenna there would experience maximum solar interference on about the 5th of March in the spring and the 8th of October in the fall. To see this, find Denver's latitude on the vertical axis and follow the green arrow across to the green curve. Then, move vertically up or down to read the fall and spring dates of maximum solar interference as indicated by the two remaining green arrows.

That covers how to determining the *date* of maximum solar interference, but it doesn't tell us at what time or for how long. The time it will happen is a complicated function of the relationship between the observer's and satellite's location, and there are several calculators on the web[80] that can help you figure this out. However, a good rule of thumb is that if your satellite dish is pointed east (the satellite's longitude is less than your longitude), the interference will occur in the morning and if it's pointed west it will occur in the afternoon.

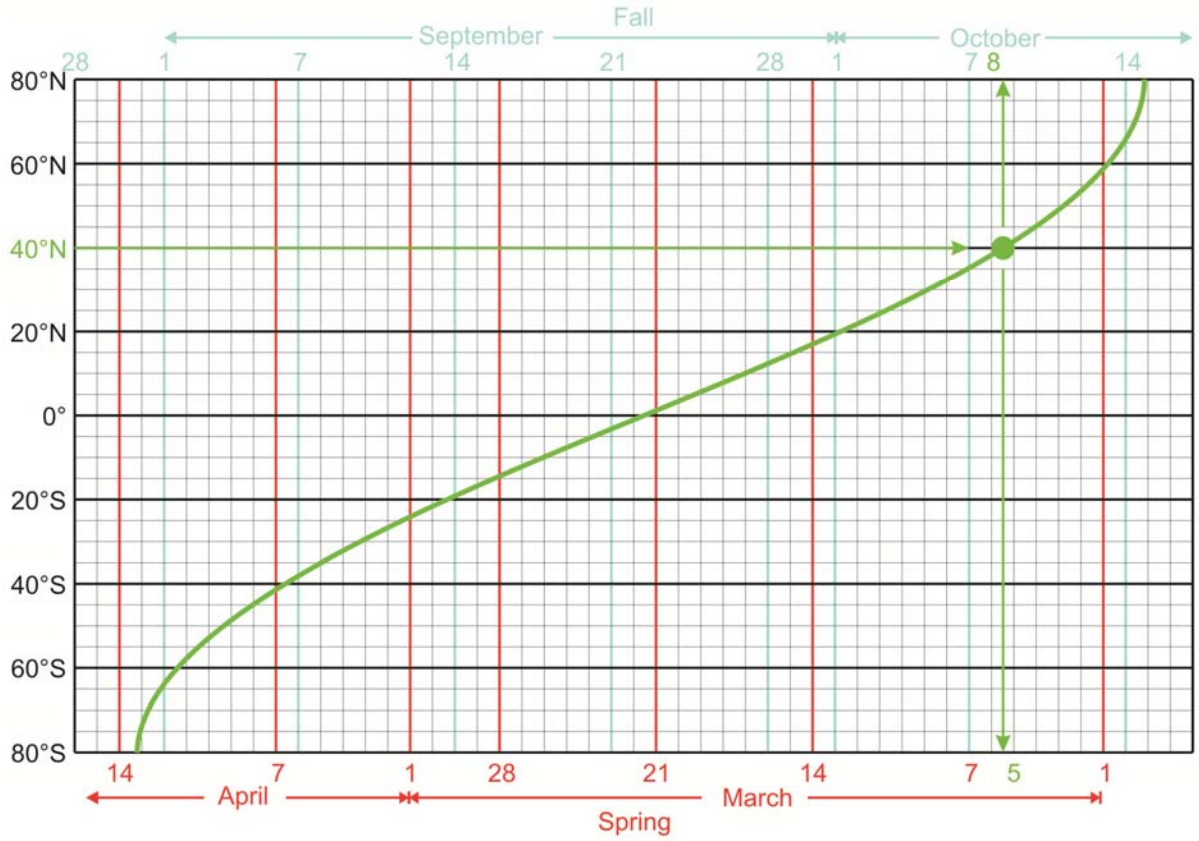

Figure 118: Annual Dates of Maximum Solar Interference.[79]

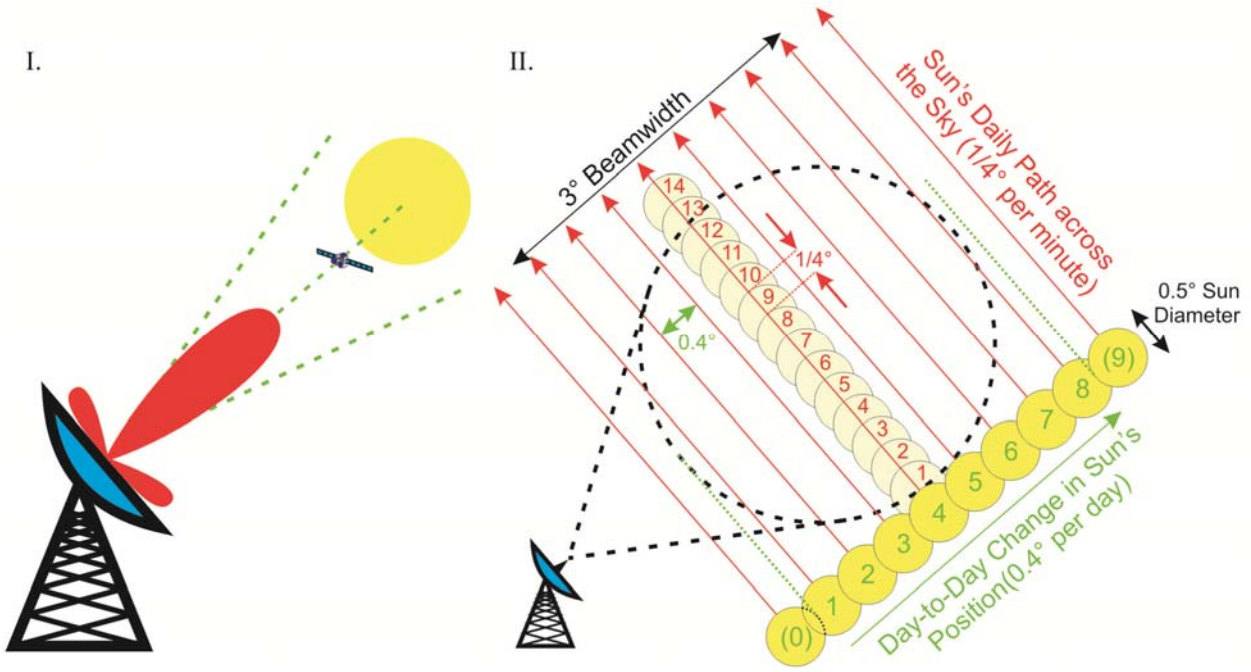

Figure 119: Solar Interference and Antenna Beamwidth.

Denver's longitude is about 105° West, so interference from all Western Arc satellites occurs after noon and all Eastern Arc interference would be morning events.

As for how long the interference will last, that depends upon the beamwidth of the receiving antenna (we discussed beamwidth in the Antennas and Diffraction section of *The Uplink Center* chapter on p. 66). In this chapter we've already talked about why beamwidth limits how closely together we can place our satellites because their transmitters, if they're on the same frequencies, would interfere with each other. Since we've already seen that the Sun emits in our frequency band, similar considerations apply: if it's closer than 3° to our satellite, our antenna will see it.

Frame I of Figure 119 shows an antenna with its main lobe pointed at a GEO satellite; the Sun is directly behind the satellite. The dotted green lines indicate the width of the beam. This situation, where both the satellite and the Sun lie along the antenna's boresight line, is when maximum solar interference occurs.

However, the Sun can interfere with our downlinked signal for longer than just the instant this perfect alignment occurs. Frame II shows how long the interference can last. The Sun appears to move across the sky during the day at a rate of about ¼° a minute, as shown by the red color coding in the figure. Counting the paler yellow suns running from lower right to upper left in the frame

shows that at that rate, the maximum time the Sun could be within our 3° antenna beam is less than 14 minutes. As also shown in that frame, the outages will occur on days before and after the date of the maximum outage. For our 3° beamwidth, since the Sun moves up and down in the sky near equinox at about 0.4° a day, as shown with the green color-coding and darker-yellow suns. Counting these suns gives a maximum of 8 or 9 days in the spring and fall where the Sun could be in the antenna's field of view. Note that the duration of each day's outage will be shorter than the one shown with the lighter-yellow suns since that's the only one that runs all the way across the full diameter of the beamwidth.

Start Interference	End Interference	Peak Interference	Duration	Increase in Noise Level
3/1/2013 8:54	3/1/2013 9:02	3/1/2013 8:58	07m39s	1.7x
3/2/2013 8:52	3/2/2013 9:03	3/2/2013 8:57	10m50s	2.1x
3/3/2013 8:51	3/3/2013 9:04	3/3/2013 8:57	12m34s	2.7x
3/4/2013 8:50	3/4/2013 9:04	3/4/2013 8:57	13m24s	3.0x
3/5/2013 8:50	3/5/2013 9:04	3/5/2013 8:57	13m30s	3.1x
3/6/2013 8:50	3/6/2013 9:03	3/6/2013 8:57	12m52s	2.8x
3/7/2013 8:51	3/7/2013 9:02	3/7/2013 8:56	11m23s	2.3x
3/8/2013 8:52	3/8/2013 9:00	3/8/2013 8:56	08m36s	1.8x

Figure 120: Example Solar Interference Timing, Duration, and Intensity.[81]

Figure 120 shows a typical set of real-world calculations for solar interference. This specific example shows interference between the DISH satellite at 61.5 W and a receiver located in Meridian, Colorado using a 3-foot diameter rooftop dish with a 3° beamwidth. Notice that the increase in noise levels over the normal condition (normal = 1) can be over a factor of three at the peak of interference on 5 March, and significant interference lasts for between about eight and 14 minutes over the course of eight days. Also notice this calculation predicts the outage occurring at about 9 o'clock in the morning, as should be expected since the satellite's longitude is well to the east of Meridian. The peak of the outage also occurs on 5 March, in agreement with the information shown in Figure 118.

We talked earlier about sidelobes, but never really explained why they were important. We've included the first sidelobe for our hypothetical antenna in the antenna figures, but only with Figure 121 will we discuss why. Here, the antenna is pointed directly at the satellite but the Sun lines up directly in the

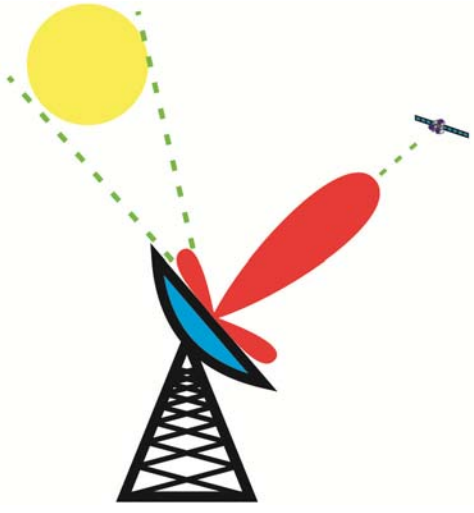

Figure 121: Sidelobes Can Be a Source of Interference.

boresight of one of the sidelobes. We just talked about the fact that if a transmitter is in the main beam of the receive antenna it would be strongly amplified by the dish and if it wasn't, it would be suppressed. However, if it's in line with one of the sidelobes, it will also be amplified. Recall Figure 48, p. 66, showing that the length of the lobe, whether main or side, displays the relative amount of gain or amplification in that lobe. The fact that our first side lobe is much shorter than the main lobe graphically illustrates that the amplification the antenna provides to that lobe isn't nearly as strongly as the main beam. That means in this situation, the satellite's signal is being strongly amplified and the Sun's emissions only weakly amplified, but received and amplified nonetheless. That's why it's so important to design antennas and feed horns well so that the sidelobes are suppressed as much as possible.[82] Don't fret, though. DISH rooftop antennas have small enough sidelobes that solar interference through them is negligible.

It may be obvious to you, but we need to go ahead and say it anyway: *solar interference only affects the downlinked signal.* It goes into the *receive* antenna on the downlink. The receive antenna on our uplink is on the satellite, and it's always pointed at the Earth. It will never look directly at the Sun.

The Sun can affect our signals in other ways, too. The Sun goes through an eleven-year cycle during which its surface goes from almost blemish-free to sunspot-filled. Sunspots are large—many times the size of the Earth--intense magnetic storms on the Sun's surface, and they frequently lead to phenomenon called coronal mass ejections, also known as solar flares, that can significantly influence the Earth's protective magnetic field and its ionosphere, layers of charged particles between about 50 and 500 miles above the Earth's surface. Figure 122 shows the Sun near the time of maximum sunspots in 2001 (the red spots on the yellow Sun in the left image are all sunspots), the sunspot cycle from

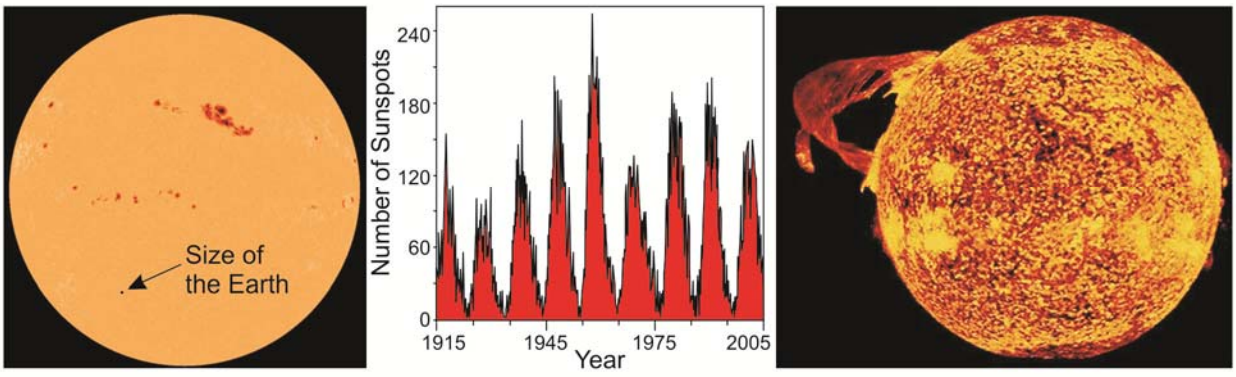

Figure 122: Sunspots and Flares.[83]

1915 to 2005, and an image of a solar prominence linking two active sunspots.

Solar flares[84] are associated with large bursts of x-ray and visible electromagnetic radiation as well as a large amount of ejected electrons, protons, and helium nuclei (two protons and two neutrons). Traveling at the speed of light, the *electromagnetic radiation* reaches Earth about eight minutes after the flare. If the flare is pointed just right, the ejected *particles* will reach the Earth a few days later since they're "only" traveling about a million miles an hour. Both the electromagnetic radiation and the ejected particles interact strongly with the Earth's ionosphere, increasing a phenomenon known as ionospheric scintillation. **Scintillation** is another word for *twinkle*. Stars appear to twinkle because shifting high altitude winds slightly alter the trajectory of the starlight as it passes through the atmosphere. Ionospheric scintillation causes radio signals' amplitude and phase to shift randomly, and since our coding is based on phase shifts, this effect can cause an increase in the number of errors our equipment will receive. It takes a significant solar flare aimed directly at the Earth to dramatically affect DISH signals, so you won't notice this effect very often.

Link Budgets

We've been talking about how our signal travels from our uplink antennas to our satellites and back down to our rooftop dishes for the better part of three chapters now, and we finally have all the information we need to see what it takes to develop what communications engineers call a link budget. The **link budget** is a long, hairy equation that calculates just how much power will be received at the far end of a communications link based on the equipment and the physical limitations along the entire link path. If the power is too small, the link won't be made and our customers won't get their TV. That means they'll be more likely to call and complain and might cancel their subscriptions to our product, all of which is bad for business. If the power is way too high, it means we're spending money we don't have to and the company doesn't do as well financially as it should. Again, something that's bad for business.

We've looked, from a high level of course, at everything that goes into that equation. We know we promised in the *easy math* introduction that we wouldn't be using any complicated equations, but we really think you'll be able to see how this one works now that we've talked about all of the inputs. We'll work up to the equation bit by bit. Just remember the goal is to find out what our received power will be. We'll call the received power P_r (pronounced *PEE-sub-ARE*) as we build the equation. However, remember that there's always going to be some

noise, some unwanted power, in the system. That's why we've talked the signal-to-noise ratio (SNR) several times already. What really matters to our customers is that their receivers can dig the signal we're sending out of the background noise that's always there. We'll call the power due to noise P_N, (*PEE-sub-ENN*—you get the picture) and the signal-to-noise equation we'll be examining will thus be

$$\frac{P_r}{P_N} = Big\ enough\ for\ our\ receivers\ to\ dig\ out\ the\ signal.$$

Since it may be confusing to some to have a fraction on the left side, we'll just call that ratio of powers the SNR from now on, like

$$SNR = Big\ enough\ for\ our\ receivers\ to\ dig\ out\ the\ signal.$$

How will we know if that's true? Well, we can rephrase the equation to show a little more math, which will look like

$$SNR = \frac{Bunch\ of\ helpful\ stuff}{Bunch\ of\ unhelpful\ stuff}$$

The reason we write it like this is that when the helpful stuff gets bigger, the SNR gets bigger; when the unhelpful stuff gets bigger, the SNR gets smaller (easy math, remember?). [85]

Speaking of inputs, let's discuss what goes into the link budget. We'll start with things we learned in *The Uplink Center* chapter, move on to the *DISH Satellites* chapter, and finally return to the things we've examined in this chapter. Figure 123 shows all of these inputs graphically *for the uplink only*, with the helpful stuff shaded green and the unhelpful stuff shaded red.

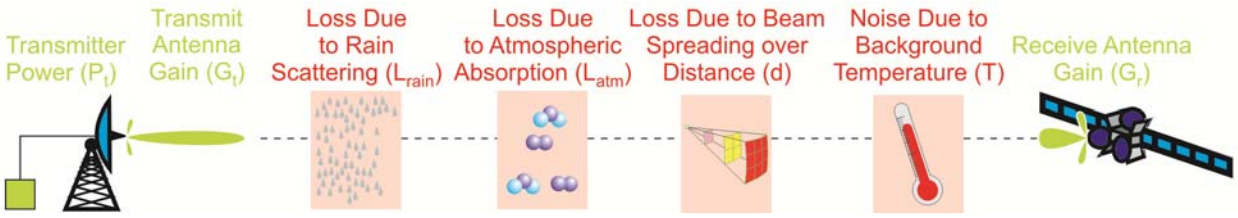

Figure 123: Factors Affecting Uplink Signal Strength.

The first factor that mattered was the output power of the uplink transmission amplifier, which we'll call P_t, for power of the transmitter. The higher that power, the better the chance we've got of completing the link all the way to our customers' facilities. P_t is helpful, so it goes into the equation like

$$SNR = \frac{P_t \times other\ helpful\ stuff}{Bunch\ of\ unhelpful\ stuff}$$

Following along in the figure, we then feed that amplified signal into our huge transmit antenna. We made it so large because that meant its beamwidth would be small and we'd get very high gain out of it (see the discussion on diffraction following p. 58 for a review of beamwidth and gain). We'll call the gain due to the transmit antenna G_t. High antenna gain is helpful, so it goes on the top of the equation like

$$SNR = \frac{P_t \times G_t \times other\ helpful\ stuff}{Bunch\ of\ unhelpful\ stuff}$$

As you can see from the figure, the only other helpful thing we've got going for us is the gain of the receive antenna, which we'll call G_r. Notice, since our receive antennas on the satellite are much smaller than the transmit antennas at the uplink center, the gain is much smaller in the figure. The top of our equation is now complete

$$SNR = \frac{P_t \times G_t \times G_r}{Bunch\ of\ unhelpful\ stuff}$$

Other than those three things, everything else works against us. What are all of those bad things? We've got a lot of losses. The first loss is due to precipitation, which we'll denote L_{rain}. It represents the scattering effect we discussed in this chapter with the San Diego and Portland example starting on p. 142. The higher the amount of rain, the larger the losses. Can you see that when it's in the bottom of the equation, a larger number would make the signal-to-noise ratio get smaller?

$$SNR = \frac{P_t \times G_t \times G_r}{L_{rain} \times other\ unhelpful\ stuff}$$

The next unhelpful thing we come across is the atmospheric absorption we discussed in *The Uplink Center* chapter (p. 54). Remember at the Ku-band frequencies we're dealing with, that absorption is primarily due to water vapor and oxygen molecules. We'll call these atmospheric losses L_{atm}. We also saw in the *DISH Satellites* chapter (p. 116) and in *The Basics* section of this chapter that the fact that the beam spreads out as it travels through space causes it to get weaker. Weaker is unhelpful, so we'll put the distance the signal travels, *d*, on the bottom of the equation as well. Finally, when we were discussing rain effects in this chapter we saw that the background temperature as seen by the receive

antenna affected the noise level in the antenna. Higher temperatures were bad, so we'll put *T* in the bottom. Solar noise, N_s, which only occurs on specific days and only for the downlink, is also bad and on the bottom. Finally, no equation would be complete without an arbitrary constant, *k*. Since we can't do anything about that constant, it doesn't matter whether it's on the top or bottom. With that final insertion, we finally can show a complete link equation:[86]

$$\boldsymbol{SNR = \frac{P_t \times G_t \times G_r}{L_{rain} \times L_{atm} \times d \times T \times N_s} \times k}$$

With this (hopefully by now) relatively straightforward equation, you should be able to tell anyone what design features you'd want to see in a very effective communications link. You want the stuff that makes the things on the top big, and you want the stuff that makes the things on the bottom small! We want high transmitter power (big P_t), and receive and transmit antennas that are just as large as we can make them (makes the G_t and G_r big). We also want clear skies (makes L_{rain} small), a transmit frequency that isn't absorbed very much (makes L_{atm} small), a satellite that's as close as possible (small *d*), and as cold a background (small *T*) as you can get. How hard can that be? (Note that in the uplink case, N_s = 1, because multiplying or dividing by 1 neither helps nor hurts us.)

But that's only half the problem! To this point in the link budget we've only delivered our signal to the satellite! With one sporadic exception, the downlink equation is just the same, but the factors have different values. The exception is solar interference, which as we've seen only affects the downlink. For downlink, N_s will always be 1 or larger. Figure 124 shows how the downlink factors work.

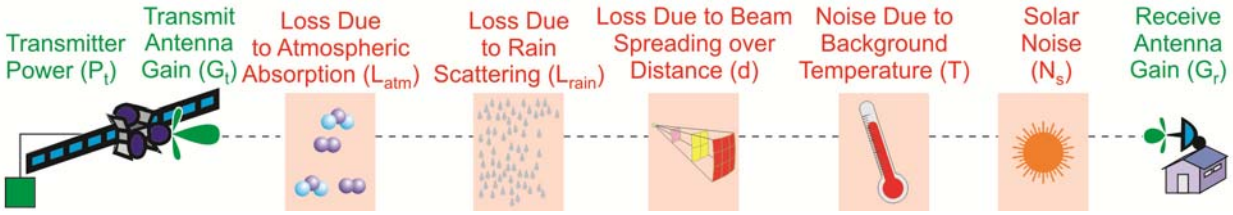

Figure 124: Factors Affecting Downlink Signal Strength.

From our equation and the figure, can you see why the downlink is so much harder than the uplink? Rain scattering, background temperature, and solar noise factors are all larger on the downlink than the uplink, and our antenna gains and transmitter powers are smaller. None of those things work in our favor.

Antenna Type	Use	Antenna Diameter (ft)	Frequency (GHz)	Beamwidth (deg)	Gain	Transmit Power (W)
Rooftop	Downlink Receive	3	12.2-12.7	3.0	250	N/A
Satellite CONUS	Downlink Transmit and Uplink Receive	8	17.3-17.8 (up) 12.2-12.7 (down)	1.1	1500	150-450
Satellite Spot	Downlink Transmit and Uplink Receive	11	17.3-17.8 (up) 12.2-12.7 (down)	0.6	4500	150-450
Cheyenne Uplink	Uplink Transmit	43	12.2-12.7	0.15	65000	2000

Figure 125: Summary of Approximate Antenna Characteristics.

Figure 125 summarizes many of our hardware capabilities and limitations, the things we have the most control over, and is a good place to compare uplink and downlinks. As you can see, our downlink transmit power and antenna are vastly smaller than the uplink case, the noise due to the change in background temperature when it's raining is much larger, our receive antenna is much smaller, and in many cases the distance to our downlink receive antennas is a lot longer. (Poor Maine—it comes out on the short end of the SNR stick for so many reasons when looking at the Western Arc!)[*]

So, how do we use these numbers? We've already looked at our wish-list (big antennas, high-power transmitters, etc.). This is the time the bean counters arrive to tell us how much things cost. We can't have as big a dish as we want. We might not be able to afford the highest power transmitter. So what do we do? We plan a system that's *good enough*. But what does *good enough* mean? Well, as a company we've determined that our customers will be satisfied if they get service 99.85% of the time. What does that mean in real terms? It means on average, they'll only lose service for about 13 hours a year. Why 13 hours? It was a tradeoff between what we want to do (100% up) and what we can deliver with affordable equipment.

Also, remember that while all of the other factors are well-known, the loss due to rain scattering is based on statistical models. We have very good numbers on how much it normally rains and how hard for our entire coverage area, but from year to year there will be differences in what really happens and what the models say. In fact, some of the footprint tailoring we discussed in the *DISH Satellites* chapter (p. 122) is designed to counter precipitation loss problems. Figure 126

[*] Don't feel too sorry for Portland. We've used the example of our Western Arc satellite at 129° W, which is the worst-case satellite for Maine. Don't forget we have the Eastern Arc as well, and most New England locations use those satellites instead.

reviews the tailoring from the CONUS antenna on DISH's satellite at 129° W. Notice that, even though they are in general farther from the satellite's location, the highest-power regions from that beam tend to occur over areas where rainfall is more prevalent: Florida, the East Coast, and the Northwest. That tailoring is no accident. It gives us a signal boost in places that are more likely to need it.

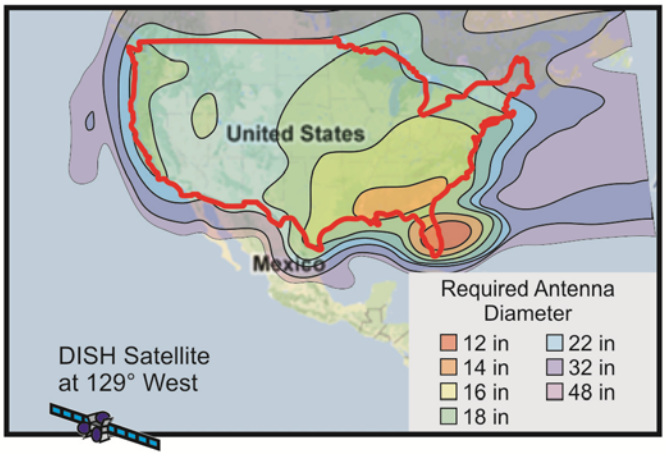

Figure 126: Tailored Footprints Revisited: Precipitation Mitigation.

Now, the 99.85% availability number we quoted is actually for the downlink side of the house, which we've already seen is the hard part. For the uplink, the DISH goal is 99.99% up, or only down for about 50 minutes a year. Think about it, though: the uplink is way more critical than the downlink. If our uplink fails, we lose channels all over the country. If the downlink is down, it's usually because of weather and is pretty localized to a spot beam or two or a small fraction of the CONUS beam.

And with that discussion of link budgets, we've now completed the space-based portion of the path our signal travels between content acquisition and content delivery to our customers. The remainder of this chapter will deal with actual hardware we use to take that faint (but more than 99% of the time useable) signal and get it inside our customers' houses and businesses.

Low-Noise Block Down-Converters (LNBs)

We've already discussed the rooftop dishes, but remember the whole purpose of that parabolic surface is to direct the incoming waves onto the parabola's focus. We saw a picture of the feed horns in their places at the focus of an antenna back in Figure 109 (p. 140), but Figure 127 now shows one of them close-up. Up to this point we've been calling them feed horns, but the name we really use for the entire unit is a **low-noise block down-converter**, or **LNB**.[*] Why the long name? Because it's pretty descriptive of what the entire unit does. It takes the incoming signal from the satellite and, using the same mixer process we've already seen in

[*] You may sometimes see a device like this called an *LNBF*, where the *F* refers to a feed horn built into the unit. For our purposes they are essentially the same thing and we will only use the *LNB* designation.

the *DISH Satellites* chapter, (p. 119), down-converts or shifts that block of frequencies to a form that will pass through the coaxial cables we use to pipe the signal into our customers' houses.* That explains the last part of the name, block down-converter. The first part, low-noise, just says that the LNB has to be really quiet electrically (it doesn't produce a whole lot of internal, interfering signals), since the signal we're receiving from space is so very faint. If it had a lot of noise, that noise would get mixed with the incoming signals and amplified, making the signal going to customers screens a lot less pristine.

Figure 127: A DISH LNB.

LNB Mixers. We talked a lot in the *DISH Satellites* chapter about mixers (see p. 119 for a review) and how our 17.3 – 17.8 GHz uplink frequency band was shifted to the 12.2 – 12.7 GHz band for downlink. We won't go through the whole explanation of mixers again, but we will do a little addition and subtraction to show how the LNB makes it possible for us to get signals inside of our customers' homes. We'll discuss the reasons in the section of the chapter on *Coaxial Cables*, but the little black cables you've seen connecting satellite dishes to receivers can't handle the relatively high frequencies coming down from space. They work very well for very low frequencies, but when you start getting much past 3 GHz, they're very ineffective. So, our 12 GHz downlink frequencies are *way* out of there; they just won't work inside a cable.

It's also much more difficult to design electronic components that work at higher radio frequencies (RF). Even other circuit designers jokingly consider much of what goes on in the RF world to be black magic. RF signals seem to want to go everywhere except where you really want them. It takes a lot of design and hardware investment to make them work well, so there's another reason to shift the frequency down a little.

To get around this hardware limitation, we use *two* more mixers. In the LNB, those mixers are called down-converters. Hopefully you'll remember that in the satellite, the mixer took an incoming signal of about 17 GHz and combined it with a signal from a very accurate local oscillator that produced a signal of about 5

* With an eye toward brevity and with apologies to our business customers, we'll just say *houses* or *homes* from now on instead of *houses and businesses* or *facilities*.

GHz. That combination produced two signals, one at the *sum* of those two frequencies (22 GHz) and one at their *difference* (12 GHz). We used a filter to screen out the sum frequency, and only transmitted the 12 GHz difference frequency down to Earth. Instead of 5 GHz, the LNB's local oscillators work at 11.25 GHz and 14.35 GHz.

So why do we need two local oscillators in the LNB while one worked just fine in the satellite? Hardware limitations again. Remember that we used two polarizations, right-hand-circular and left-hand-circular, so that we doubled the amount of information we could send within our FCC-limited bandwidth (see p. 90 for a review of polarization, if necessary). Our odd transponders use signals sent with RHCP and our even ones with LHCP. (Figure 128 reviews how the different transponders are stacked within our downlink spectral bandwidth—the rainbow at the top has been added to make the figure conform with Figure 129, which we'll discuss shortly.) Well, polarization works very well for electromagnetic waves traveling through space or the atmosphere. It's not effective at all inside of electronics or coax cables. That means we're going to have to finally separate the even and odd transponders so they use different frequency bands that don't interfere with each other.

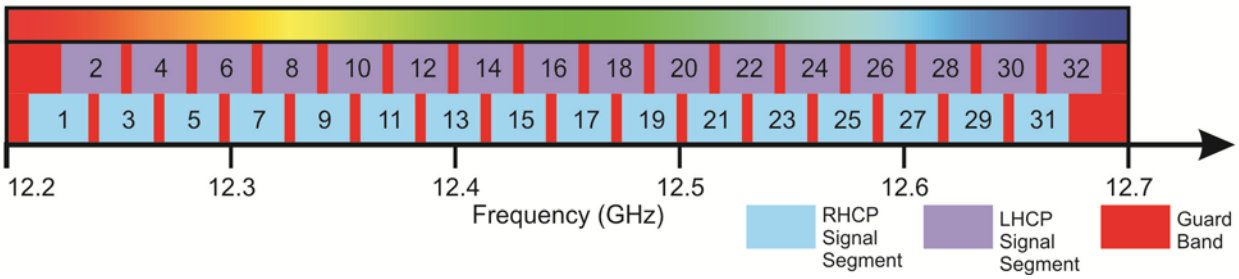

Figure 128: DISH Downlink Transponders.

But what about the FCC's spectral bandwidth limit, you may be asking yourselves at this point, patting yourself on the back for your technical insight. We've only got a half-GHz-wide allocation (12.2 – 12.7 GHz for the downlink case), and as you can plainly see in this figure, each of the odd and even transponder bands take up half a GHz by themselves. While that's great insight, the FCC only controls signals traveling through air and space, not inside our electronics. That means we can use as much bandwidth as we want in our cables (or, actually, as much bandwidth as our cables can handle—stay tuned for more on that limitation in a few paragraphs).

So, let's see what our LNB mixers do to the signal. The first mixer, the one with the 11.25 GHz local oscillator (LO), works almost exactly like the one we looked at

inside the satellite. Under most conditions, it acts on the RHCP odd transponder signals (we'll discuss how the polarizations are separated in the next section). Those signals are at 12.2 – 12.7 GHz. Let's just look at the lower boundary of the downlink band, 12.2 GHz, for now. When we mix it with the 11.25 LO frequency, we get a sum and a difference frequency, 23.45 GHz and 0.95 GHz.* We then filter out the sum frequency and are left with the 0.95 GHz signal. Since the upper bound of the downlink band is just 0.5 GHz higher, the upper bound of the difference frequency band is also 0.5 GHz higher, or 1.45 GHz. Thus, the RHCP downlink signals from our odd transponders end up passing through our cables as unpolarized signals in the 0.95 to 1.45 GHz band. Simple, huh?

Well, the way we treat the LHCP signals is almost the same, but not quite. The mixer for the even transponders has an LO that works at 14.35 GHz. It also acts after the LHCP signal has been separated from the RHCP ones. This LO signal is then mixed with the downlink band to get sum and difference frequencies, which in this case are 26.55 GHz and -2.15 GHz for the lower downlink band limit (12.2 GHz). We then filter out the sum frequency and…hmmm…that difference frequency, the one we're planning on using, is a *negative* number. Let's see what mixing the upper downlink frequency band limit (12.7 GHz) gives us to see if this problem can be resolved. 12.7 minus 14.35, carry the one, um, that's a negative number as well, -1.65 GHz. So the down-converted band for our even transponders runs from -2.15 to -1.65 GHz.

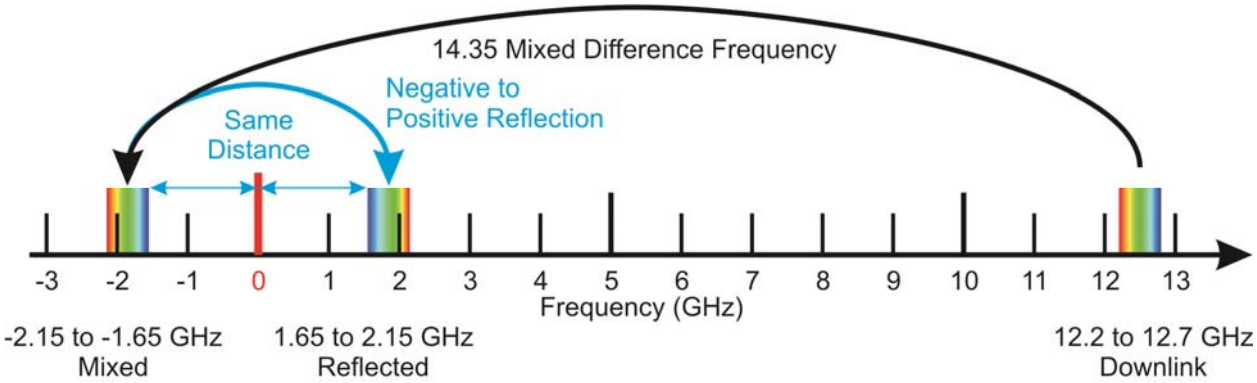

Figure 129: Negative Frequency Reflection.

What the heck does a negative frequency mean and how does it apply to getting our signal into a cable? A picture is worth a thousand words, so behold the words conveyed by Figure 129! It shows the satellite downlink band of 12.2 – 12.7 GHz as a rainbow near the right side of the figure. The rainbow is actually an

* 12.2 + 11.25 = 23.45; 12.2 – 11.25 = 0.95

important visualization tool in this figure because it shows the order of the transponders the same way it did in the expanded version in Figure 128. In this example, red would be near the location of transponder 2's frequency while blue is near transponder 32's. The long, curved black arrow pointing right to left shows the result of subtracting 14.35 GHz from the downlink frequency. As expected, it gives the resulting negative frequencies in the range we calculated above, -2.15 to -1.65 GHz. Note that the rainbow is still shown as red on the left and blue on the right—subtracting didn't do anything to change that result.

Now, here's where the "what does a negative frequency mean" question gets answered. There's no such thing as a negative frequency. A wave oscillating at negative ten cycles per second goes up and down exactly the same number of times as a wave oscillating at ten cycles per second. The only difference is that when the positive frequency wave goes up, the negative frequency wave goes down, as shown in Figure 130.* The fact that negative and positive frequencies are essentially the same means that we can just ignore the negative signs in the mixer difference frequencies we calculated, so the band really becomes 1.65 – 2.15 GHz.

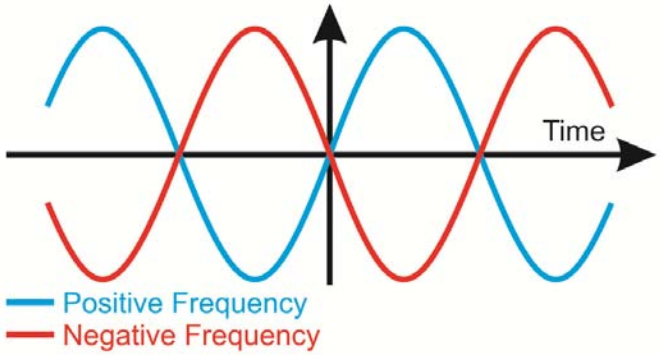

Figure 130: Negative and Positive Frequencies.

However, there's an unexpected consequence to this "reflection" of the frequencies about the zero-frequency line: it's called *spectral inversion*. The blue curved arrow in Figure 129 shows the reflection, and the blue straight arrows show that the distances from zero to the blue band in the rainbow are the same for both the positive and negative frequencies. Not explicitly shown but resulting from the same process, the distances from zero to the red band in the rainbow are also the same. Remember that the blue portion of the rainbow was associated with transponder 32's frequency and the red with transponder 2's frequency. As a result of this reflection, we have *inverted* the order of the transponders' frequency bands! The order of the rainbow's colors in the figure are reversed, blue is now on the left and red on the right. Thus transponder 32's frequency is now lower than transponder 2's.

* The technical term for this is that the wave undergoes a 180° phase change, but that's unimportant for our purposes. See p. 71 for a review of phase.

So we now have moved the signals from both sets of transponders into *non-overlapping* frequency bands that can flow simultaneously inside the same coaxial cable. The only thing we have to clarify is that for the sake of clear explanation above, we fudged the truth just a bit. We didn't want the terminology to become too tedious, so to this point we talked about even transponders and odd transponders being moved to specific bands within the cable, evens to the upper band and odds to the lower. That's not really the case. The LNB is actually smarter than that. It can move *either* set of transponders to *either* one of the cable bands, depending on what's needed by the receiver based on the channels selected by the customer—more about how the receiver does this in the next chapter. We'll also discuss why we need this flexibility for either even or odd bands when we get to the section on nodes later in this chapter. From now on, we'll refer to the LNB-down-converted bands as the upper and lower LNB bands. The locations of these bands are shown in Figure 131. Notice that regardless of which satellite transponder band goes into the upper LNB band, the order of the transponders will be reversed.

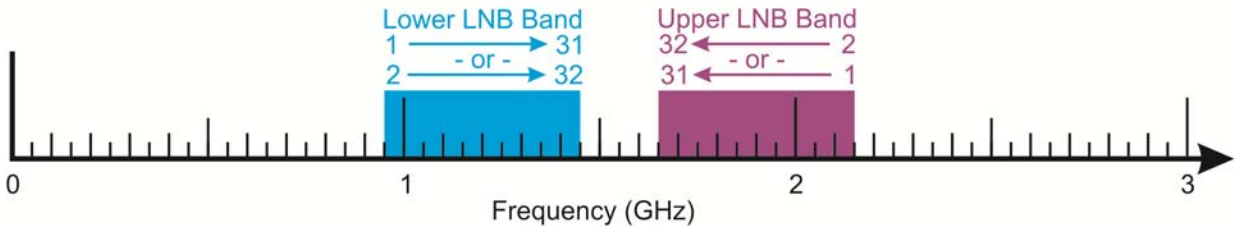

Figure 131: Upper and Lower LNB Frequency Bands.

One last tidbit about the mixers in LNBs: in addition to sending electrical signals down the cables, LNBs receive electricity back up through those same coax cables. They are kept powered up at all times—even when no one's been watching TV for days—to keep their temperatures stable. That temperature stability helps the local oscillators used in their mixers to stay very close to the specified frequencies needed to down-convert into what goes into the cables.

LNB Polarization. We now return to the subject of polarization (if you need a review, it starts on p. 90). We've stated without much justification that DISH satellites use circularly polarized waves to transmit their signals. Have you wondered why we do this, instead of going with what would seem like a much simpler system that uses linearly-polarized waves? Well, if you have, your appetite for knowledge is about to be whetted. If not, you'll be fed anyway, and you'll like it!

To this point we've talked about the parabolic dish as if it were the real antenna in the system. It's very common to talk about a "dish antenna," but in reality all antennas boil down to being just a straight piece of wire inside which we are able to generate an oscillating electrical signal in response to an oscillating electrical wave traveling through space or the atmosphere. The antenna is the way we take waves and convert them to signals inside electronic devices. The dish is not a straight piece of wire, and therefore it's not the antenna. All the dish does is to concentrate as much of that wave as we can in one place so the signal gets boosted to levels where we can receive it. The three important bits of this entire antenna system are the Ku-band radio waves, the dish that concentrates those waves, and the tiny line of wire—the antenna itself.

A good (but somewhat gruesome) analogy to what the dish does is the childhood game of frying ants with a magnifying glass, as shown in Figure 132. In this analogy, the Sun's rays are the waves, the magnifying glass is the concentrator, and the ant is the antenna that receives the concentrated waves. Under normal conditions, Frame I in the figure, the Sun's rays aren't strong enough to elicit a response from the ant. However, when they're concentrated as they are in Frame II, they become a signal that's hard for the ant to ignore.

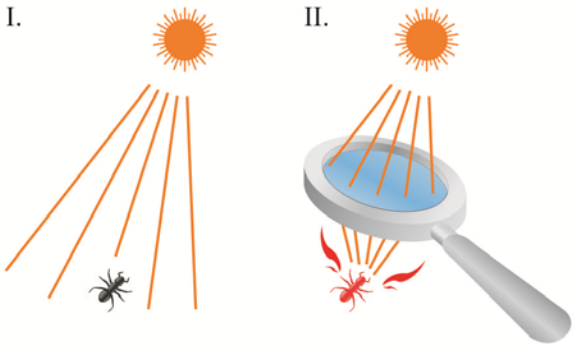

Figure 132: A Magnifying Glass Concentrates Light Waves.

But unlike the ant, the antenna is only sensitive to one polarization direction of incoming electromagnetic waves. If waves having their polarization oriented in a

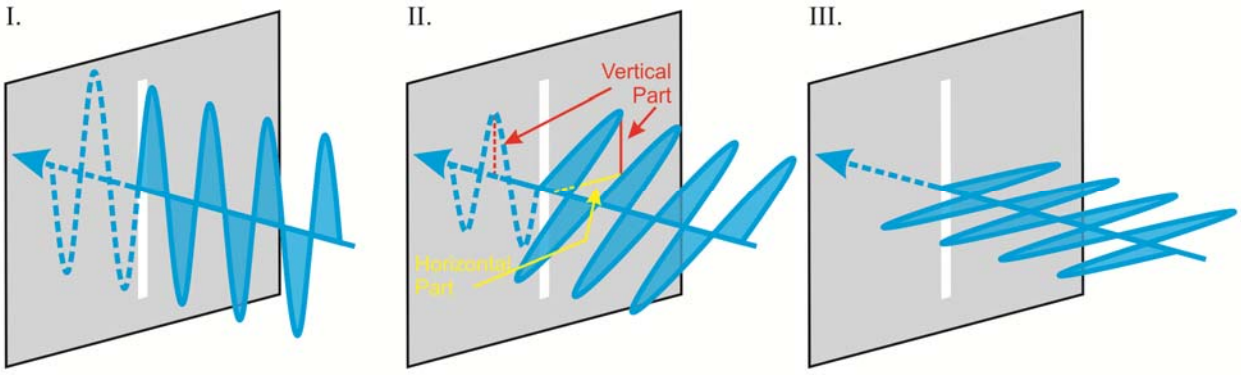

Figure 133: Linearly Polarized Waves Striking a Linear Antenna.

different direction hit the antenna, they aren't nearly as effective at exciting a response inside the electronics. Figure 133 shows this concept, illustrating how a wave might look passing through a narrow slit. While an antenna isn't really a slit, how a wave interacts with an antenna can be illustrated using a slit.[87] In Frame I, the vertically-polarized blue wave matches up perfectly with the vertically-polarized slit. In Frame II, the polarization of the wave has been rotated clockwise a bit. Only the vertical part of the wave makes it through the slit. The horizontal part is blocked. In Frame III, the wave is oscillating horizontally and *none* of it makes it through the slit. Imagine how much harder it would be for our installers if they had to not only point the satellite dish in the right direction but

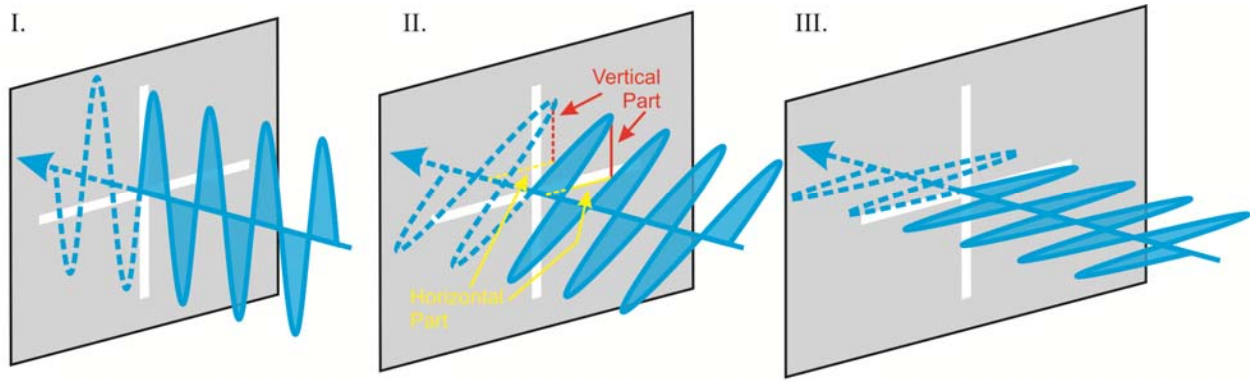

Figure 134: Linearly Polarized Waves Striking Two Perpendicular Linear Antennas.

simultaneously had to align the antenna so it matched the polarization direction of the incoming signal from the satellite. Since our link margins are so thin even a slight misalignment might mean the difference between getting good TV service and none at all.

So how does circular polarization solve this problem for us? We're glad you asked! Look at Figure 134, which shows linearly-polarized waves interacting with two perpendicular slits. In Frame I the wave passes through just as it did with only the vertical slit present. In Frame III, the wave completely passes through the horizontal slit. But in Frame II, the horizontal part of the wave passes through the horizontal part of the slit and the vertical part of the wave passes through the vertical part of the slit. These two parts are recombined on the far side to reconstruct the complete, original wave. With this antenna set-up, it doesn't matter what twist the satellite dish has when the installer puts it in because *any* orientation will work.

OK, so that shows how a linear wave with any polarization direction could pass through the slit, but what's that got to do with circular polarization? Let's revisit

some figures we first saw in *The Uplink Center* chapter. First, Figure 135 should remind you that a circularly-polarized wave is one where the direction of polarization continually changes direction, spiraling around the wave's direction of travel. (If this doesn't ring a bell, we recommend you review the *Polarization* section starting on p. 90.) Next, Figure 136 should remind you that a circularly-polarized wave was made up of two linearly-polarized waves—one wiggling up-down and one right-left—but with their oscillations out of phase by 90°. So, since circularly polarized waves are the same thing as two linearly-polarized waves wiggling perpendicular to each other, having these two perpendicular waves being received by two perpendicular antennas means we can receive circular polarization regardless of the twist of the antennas' directions relative to the satellite beam's polarization. That makes our installers happy!

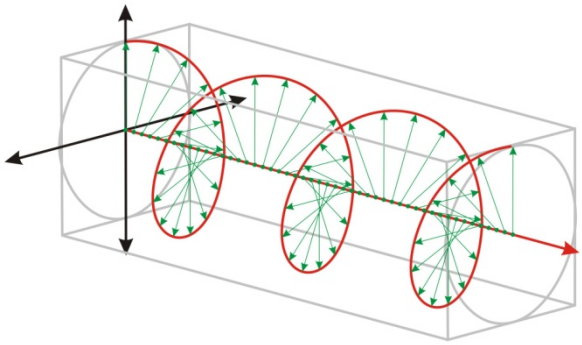

Figure 135: A Circularly-Polarized Wave.

We're about to put in the last piece of the circular polarization puzzle here. We use both left- and right-handed circular polarizations (LHCP and RHCP) to be able to squeeze twice as much information through our bent pipe, but how do we tell

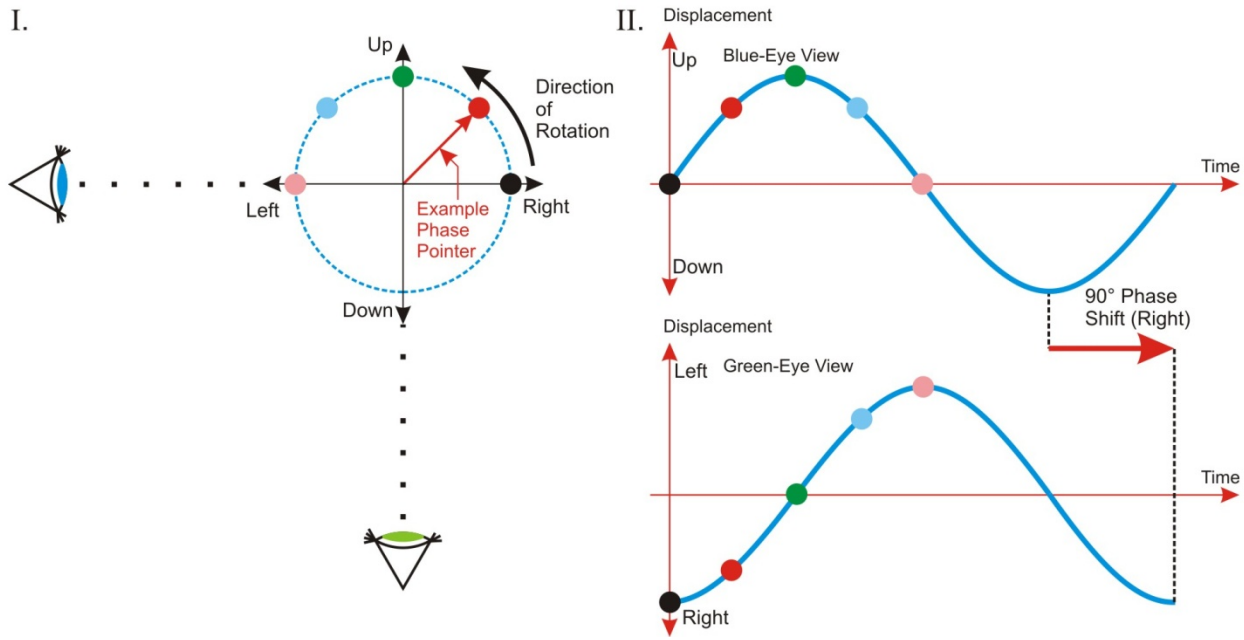

Figure 136: Two Perpendicular Linearly-Polarized Waves Can Make Circular Polarization.

them apart? It looks like both will pass through the perpendicular slits equally well. If we spent the time to properly align the antennas, it seems like it would be easier to distinguish one from the other if we used vertical and horizontal linear polarization and independent vertical and horizontal antennas. This circular business sounds a lot harder, except for ease of installation.

It's not, really. Take a look at Figure 137, showing RHCP waves and compare it with the previous figure that showed LHCP. What's different between the two figures? (*The astute reader might start to answer this question by reading the caption for the second figure…*) That's right! The only difference between RHCP and LHCP is the direction of the phase shift! In the right frames of both figures, the upper up-down wiggle is the same, but the bottom left-right wiggle is a total of 180° out of phase, one shifted 90° to the left, the other 90° to the right. That phase difference is the key to telling the two kinds of circular polarization apart.

In order to get the strongest signal into our LNB, we'd like the highest amplitude portion of the up-down wave to match up with the highest portion of the right-left wave. The way we do this is to use an electronic device to put a time delay on the signal received by *one* of the antennas equal to ¼ of a period, or 90° of phase. Take a look at Figure 138, which shows the up-down and left-right waves for the LHCP (Frame I) and RHCP (Frame II) situations we saw in the previous two figures. (The way we've drawn it here, a delay means to shift the curve to the *right* so it arrives later in time, with time getting bigger as we move to the right.) The

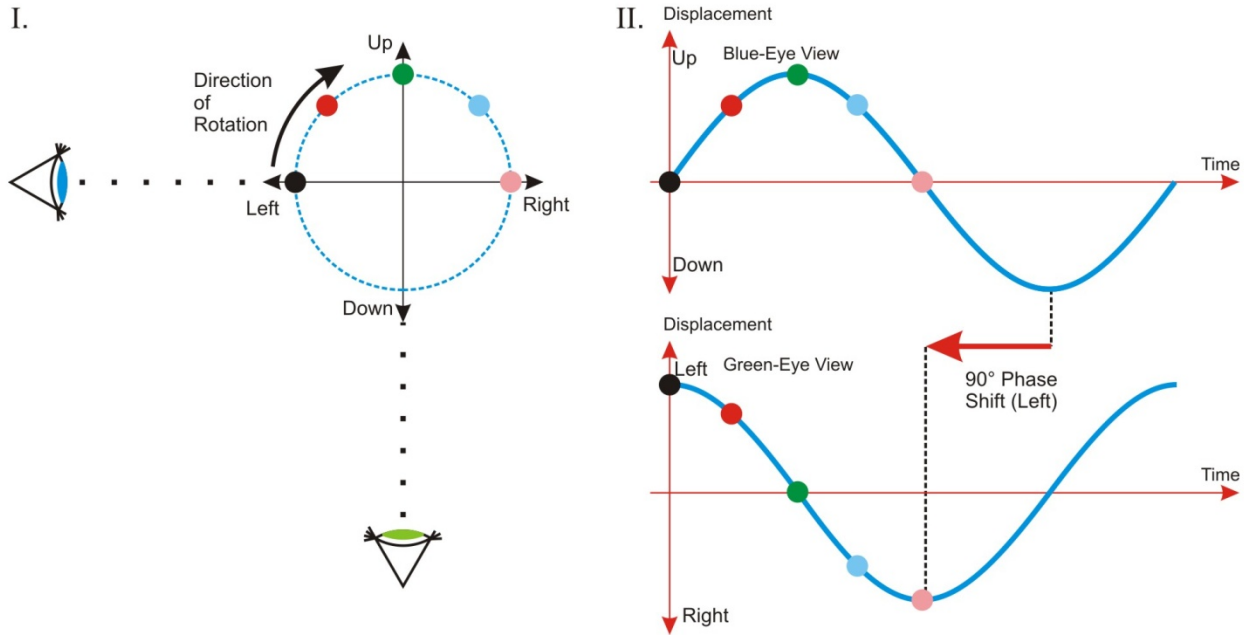

Figure 137: RHCP Differs from LHCP only by the Direction of the Phase Shift.

dotted blue curves represent the original, un-delayed waves, with the delay and the delayed waves being indicated by the black arrows and solid blue curves, respectively. In this figure, we've delayed the *up-down* wave by 90°.

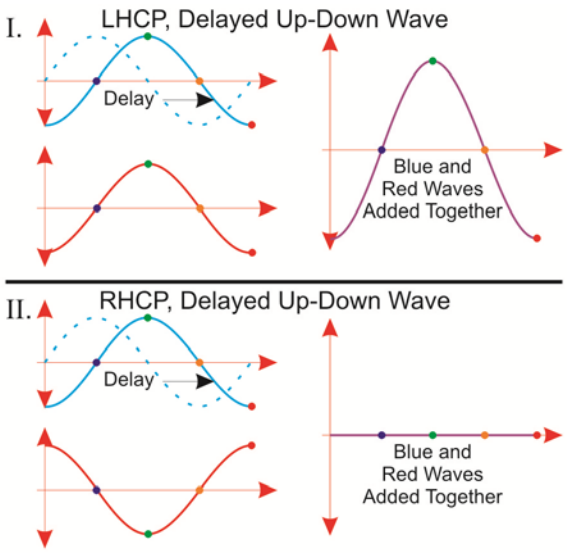

Figure 138: Effect of Delaying the Up-Down Wave.

You can see for the LHCP situation in Frame I, the delayed wave's peak now exactly coincides with the peak for the right-left wave. The purple curve shows the effect of adding the amplitudes of the signals from the red and blue curves giving us a signal that's twice as strong. Delaying that wave amplifies the LHCP signal.

Now look at Frame II. If we do the same thing—shift the up-down signal to the right by 90°—the peak of the up-down wave coincides with the valley of the right-left wave, exactly cancelling it out! Again, the purple curve shows this amplitude addition and the resulting flat-line. Thus, by inserting an electronic component that results in a 90° phase delay on the up-down signal we amplify LHCP signals and destroy RHCP ones.

We won't describe it in similarly glorious detail, but Figure 139 shows what happens if we put in a 90° phase delay on just the *left-right* signal. The end result would be that we amplify and destroy the *opposite* polarizations.

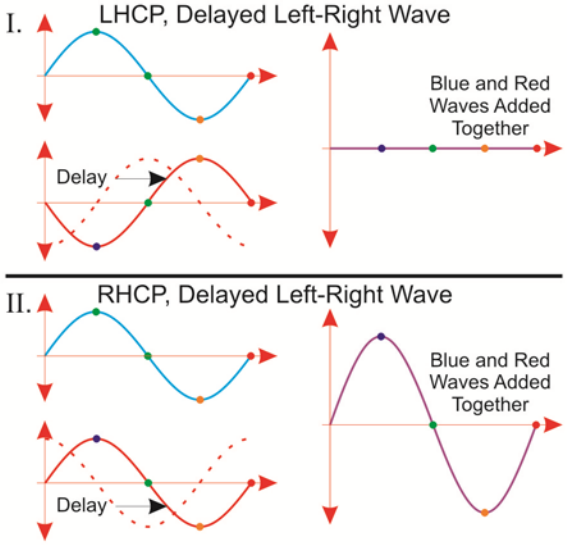

Figure 139: Effect of Delaying the Right-Left Wave.

Thus, the way we separate circularly polarized signals is to capture them with the *same* set of two perpendicular antennas then send them down two *different* paths, one with a delay on one antenna and one with a delay on the other. In the first path, we'll

only get the even transponders and only the odd ones will pass through the second path. Game, set, match. Together we've now conquered the subject of how the LNB takes two independent polarized signals that travel well together in space and converts them to two unpolarized signals that can travel together well in a coaxial cable.

Switches, Nodes, and Other Signal Flow Devices [88]

We're almost ready to move inside the customer's house to talk about receivers, but we still need to describe a few outside devices you may hear about before we do. (In fact to be completely accurate, many times these devices will actually be installed inside the house, but we really want to reserve the next chapter for receivers since they're a nice, self-contained unit of study.)

We've seen how the LNB takes the signal and changes it so it can pass through relatively cheap coax cable. However, there are many times we need to route these LNB signals to different places, sometimes separately and sometimes combined with other signals. The pieces of hardware we use to do this routing are collectively called **signal flow devices**.

Schematics of representative signal flow devices DISH uses are shown in Figure 140. There's a lot of information in this figure, but no worries—we'll step through

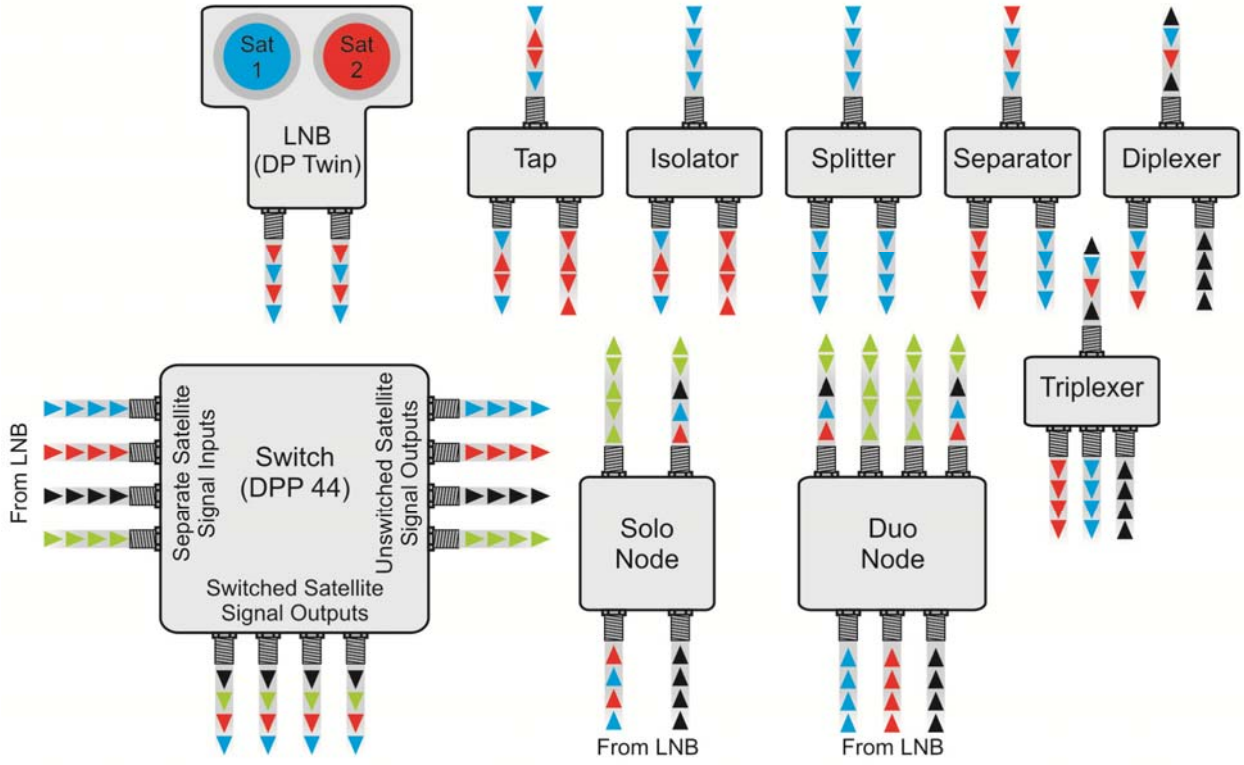

Figure 140: DISH Signal Flow Devices.

each of the devices one by one. We need to present them together, though, so it's easier to see the similarities and differences. Also, you don't need to memorize what each device does unless you're doing installations for a living. We're just discussing them here so you can get a big picture on how signals are moved from the satellite dish throughout the customer's home.

LNBs. The first device in the figure is actually not a signal flow device, at least that's not its primary purpose. It's our old friend the LNB. We're showing it here because there's actually a switch built into its design. We'll look at examples of purpose-built switches in just a second, but what the switch in the LNB does is to allow us to choose which signal flows through a cable. As you can see, the LNB we've chosen as our example is a twin, that is, it is able to look at two satellites simultaneously. We've chosen to color-code the individual feed horns red and blue, and we show the signals produced by, for example, the red LNB with little red arrowheads in the cable indicating the direction the signal flows. In this case, you can see that there are two coax cables attached to the LNB, and signal from both satellites 1 and 2 *can* flow through either cable

One important note here: just because the diagrams show that the signal *can* flow in any of these devices doesn't mean it *is* flowing. We'll discuss how signals are chosen to flow through the cables in detail in the next chapter.

***Switches*.** Moving down in Figure 140 below the LNB we come to an example of a **switch**. What a switch does is to allow a signal from any of the inputs to be routed to any of the switched outputs. This particular switch is designed to take in signals from four LNBs, as indicated by the inward-pointing arrowheads from the left side of the diagram. The incoming signal from each specific LNB feed horn in this figure is color-coded. This switch has two sets of outputs. The first set, at the right side of the switch, is really just a pass-through. The outputs carry exactly the same signals in the same order as the inputs, as shown by the color-coded arrowheads. The purpose of these outputs is to allow us to attach a second switch downstream from the first, taking the output of the first switch and putting it into the input of the second. Since the switch doesn't really perform any operations on those outputs, they're called *unswitched*. That's pretty basic.

The outputs on the bottom side of the switch, however, show that any of the inputs can be routed to any of these outputs, depending on what's selected by the tuners in the house. That result is indicated by having an arrowhead of each color coming out of each of the outputs.

To get an idea of why we need switches, you might imagine a home with four TVs (and four satellite receivers, not shown), as illustrated in Figure 141. Each of the inputs is connected to a different LNB and each of the bottom outputs is connected to a different TV inside the house. The unswitched pass-through outputs are unused and capped off for this example. Each LNB feed horn is pointing at a different satellite, and for the most part we don't duplicate TV channels on different satellites—i.e., the HD version of HBO is on the satellite at 110° W and it's not on other satellites or transponders in the Western Arc. That means if a customer chooses to watch HBO, the signal must come from the feed horn pointed at that satellite. In this house, the first TV is tuned to ESPN, the second and third are watching the same program on HBO, and the fourth is watching the local NBC affiliate. In this example, ESPN is on the satellite being monitored by feed horn 1, HBO is on the satellite being received by feed horn 3, and NBC is on feed horn 2's satellite. As you can see in the figure, the switch allows the four channels to be routed from the proper feed horns to the proper TVs. The signal from feed horn 4 isn't needed now so the switch doesn't send it anywhere.

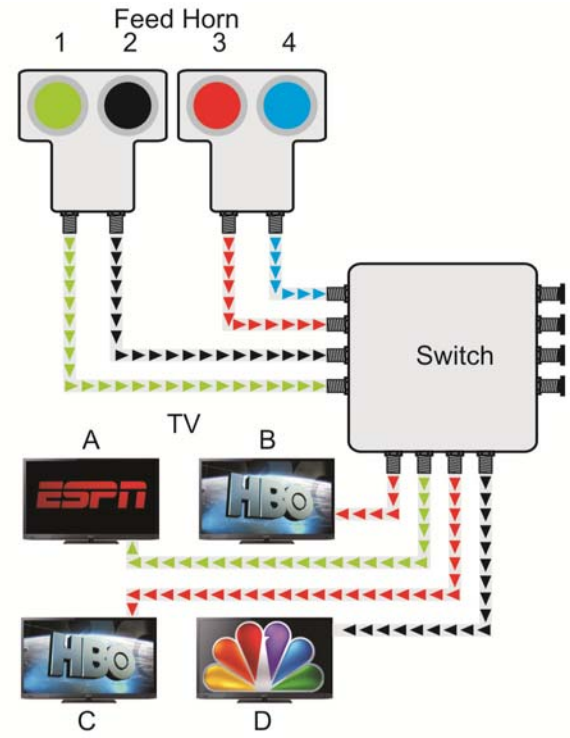

Figure 141: Signals Passing Through a Switch.

Note that in this example, we would have to punch four holes in our customer's wall to get all the cables inside. There are better, cheaper ways (for both DISH and our customers) to accomplish this task of getting signals to four TVs, but this simplistic set-up is an easy way to understand what a switch does. Again, switches just take a set of separate incoming signals and route them to the output where they're needed. Figure 142 shows the other three switches DISH currently uses. The only difference between their operation and the one we've already shown is the number of inputs and outputs.

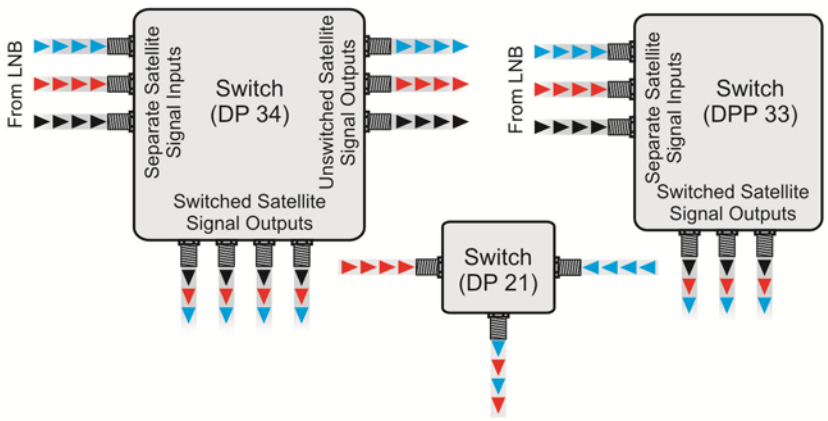

Figure 142: Additional DISH Switches.

Nodes. Switches are just one way we get signals routed inside the house to the proper receiver. For our Hopper and Joey systems, instead of switches we use a **node**. The node operates very differently from the switch. As shown in Figure 143, instead of deciding which cable needs which signal, the node combines *three* LNB signals on a single cable. Now, we've hammered and hammered the fact that we absolutely have to avoid interference of our signals at all costs, because interference means we can't tell which information we need to pay attention to. It was the reason we geographically space out our uplink centers, it was the reason we need small spot beams, it was the reason we needed different uplink and downlink frequencies in the satellite, and it was the reason we had two local oscillators (LOs) in our LNBs to produce the separated bands for the upper and lower LNB bands in the coaxial cables. So, after all that trouble we went to separating frequencies, how are we going to put another signal in the cable, given we only have two bands coming from any of our LNBs?

The answer to that intriguing question is that the nodes *also* have LOs inside them! Our Solo Node has one, operating at 4.65 GHz, and our Duo Node has an additional LO at 3.95 GHz. Being astute readers, we're sure you all immediately recognized that both of those LO frequencies are higher than either of the transponder bands we use (0.95 – 1.45 GHz for the lower LNB band and 1.65 – 2.15 GHz for the upper), which means if we use the difference frequency between the two, we'll end up with a negative

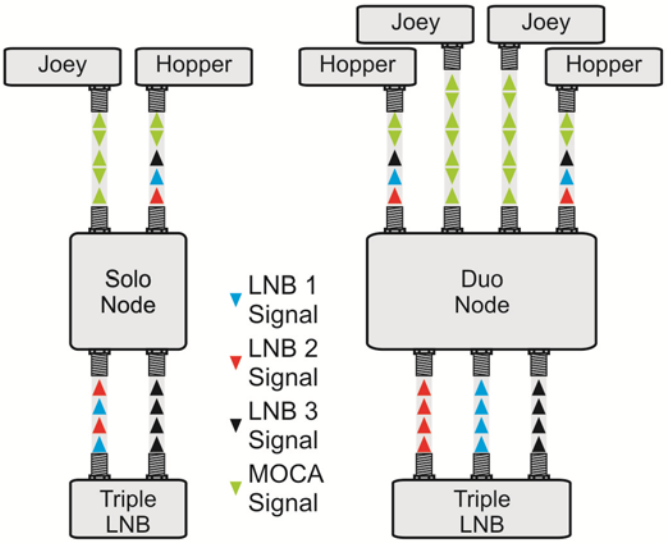

Figure 143: DISH Nodes.

frequency again like we did with the LNBs (see Figure 129, p. 162, and its accompanying text for a review, if necessary). Let's see how that works.

The Solo Node's LO up-converts only the *upper* LNB band, so to get the difference frequencies, we subtract the LO from the upper band limits (1.65-2.15 GHz) like we've done before. 1.65 – 4.65 = -3, but we drop the negative sign; 2.15 – 4.65 = -2.5, which also becomes positive 2.5. Thus, the Solo Node up-converts the upper LNB band into the 2.5 to 3.0 GHz range. What that means is we now can send *three* overlapping, ½ GHz-wide bands in the same cable, as shown in Figure 144.

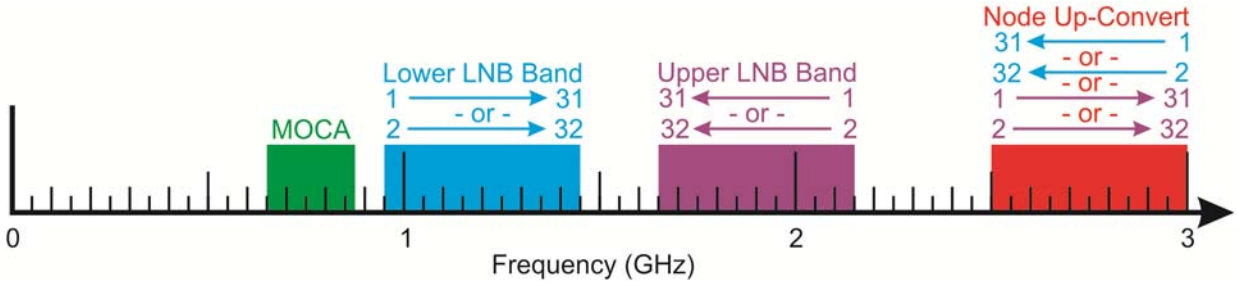

Figure 144: Frequency Bands Coming Out of a Node.

Earlier we discussed how the LNB needed to be able to put either transponder band (even or odd) on either one of the LNB bands (upper or lower), depending on what our customer had chosen to view. Using a node and a Hopper/two-Joey system is a good example of why this additional three-band flexibility is necessary. We'll use Figure 144 and Figure 143 to illustrate the point.

The Hopper has three independent tuners (more on that in the next chapter). Imagine that those three tuners are all tuned to different channels that all come from odd transponders on different satellites. We have a triple LNB pointed at three satellites but it only has two cables coming out of it to a Solo Node. The LNB pointed at the first satellite puts the odd transponders' signals on the lower LNB band of the first cable. The LNB pointed at the second satellite puts its odd transponders' signals on the lower LNB band of the second cable. The third LNB would like to use the lower band as well, but it sees they're both occupied. It instead, knowing there's a Solo Node downstream that can up-convert the upper LNB band, puts its odd transponders' signals on the upper LNB band of the first cable. To this point, there is one signal on one of the cables and two signals, the maximum possible from one of our LNBs, on the other.

When all of these signals get to the Solo Node, the node sees the first cable has both upper and lower bands on it and passes those directly through the node to

the cable connected to the Hopper. It then takes the signal from the third LNB and up-converts it to the Node Up-Convert band and puts it on the same cable. In that way, all three odd-transponder signals make it to the Hopper, where the tuning decisions are made. Two of those signals are then converted to MoCA standard and are sent back out to the Joeys, which display the signal the Hopper has chosen for them. (Yes, we know we haven't explained MoCA yet. It's coming soon. Have patience.) That way, all three odd-transponder signals can be watched simultaneously. Brilliant, huh?

The Duo Node's second LO does the same up-conversion, but it works on the *lower* LNB band. It uses a 3.95 GHz signal working on the 0.95-1.45 GHz band to get exactly the same difference frequency band of 2.5 – 3.0 GHz.[*] Note that this negative-frequency up-conversion inverts the signal, as indicated by the color-coded transponder orders in the figure. With those three bands coming out of two cables going into the house, the LNB combined with the switch can now deliver up to six independent TV channels to two Hoppers and six Joeys (note that two of those eight devices will have to watch a channel determined by one of the others since there are only three tuners in each Hopper).

MoCA. We asked you to be patient, and look what happened: we're about to discuss that pesky MoCA concept from a few paragraphs ago. If you look again at the diagram of the nodes in Figure 143, you'll see we haven't yet discussed some of the connections: the ones carrying the green signals labeled "MoCA." In addition to the three LNB signals, a node also enables the distribution of information between the Hopper and all of its associated Joeys. That information also flows through the cable at the same time as the LNB signals, but it flows both directions simultaneously instead of just inbound to the receiver. A glance back at Figure 144 will show how that's done, by sending signals in yet another, fourth, band within the same cable using the frequency range of 0.65 - 0.875 GHz.

The MoCA band is very different from the other bands in the cable. We don't use an LO to just shift frequencies for these signals. The Hopper actually takes in the signal from the LNB, digests it, and rebroadcasts it in a different format. The way MoCA information is coded is very different from the 8PSK, encrypted, MPEG-4 signals that are in the LNB bands. The coding method is called **MoCA**, for *Multimedia Over Coax Alliance*, the group that developed that particular standard. Our sister company, EchoStar, is represented on the MoCA Board of Directors.

[*] 0.95 – 3.95 = -3 which becomes 3; 1.45 – 3.95 = -2.5 which becomes 2.5.

MoCA uses the same sort of information coding that the Internet does to deliver streaming video to your computer. It sends IP-based packets that are routed to the appropriate Joey where the packets are reassembled and converted back to television programs. The only real difference between this set-up and the Internet you're probably used to is that the information packets flow over coax (hence the *OC* in MoCA) instead of through an Ethernet cable. In essence, MoCA is just the Internet streaming through a cable type it normally doesn't use.

Notice that Joeys *never* receive signals directly from the LNB. They wouldn't know what to do with them even if they got them. They're only designed to understand Internet-speak as delivered over the MoCA cables. So nodes, used with Hopper/Joey systems, allow us to have three tuners and four TVs connected to our broadcasts with only two wall penetrations (six tuners and eight TVs with a Duo node).

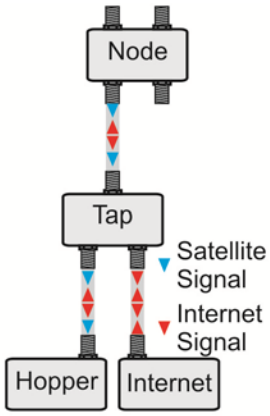

Figure 145: Example Tap Use.

LNBs, switches, and nodes are by far the most important signal flow devices. If you're pressed for time, feel free to skip ahead to p. 178. You'll miss out on six less important signal flow devices, but they're not critical to understanding the big-picture of how we move signal around in our customers' homes.

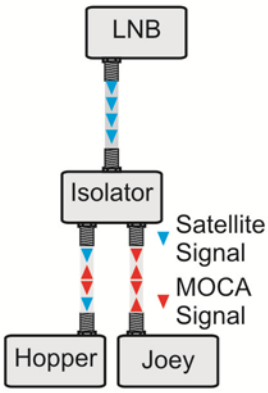

Figure 146: Example Isolator Use.

Taps. The next six signal flow devices all have similar physical layouts, but do very different things. They each have two connections on one side (except the triplexer, which has three) and one on the other. The **tap** is used to insert a two-way signal into a one-way flow. A typical use for this device is to insert Internet information onto the same cable that is carrying the satellite signal, as shown in Figure 145.

Isolators. An **isolator** is very similar to a tap, but it doesn't let the two-way signal out the one-connector side. A possible use, shown in Figure 146, is to isolate the LNB from the two-way MoCA signals passing between a Hopper and a Joey.

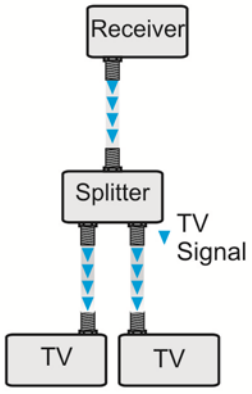

Figure 147: Example Splitter Use.

Splitters. A **splitter** does exactly what its name says: it

takes a single signal and splits it into two identical signals. In this way, it can be used to take the output from one receiver and allow exactly the same program to be viewed simultaneously on two televisions like the example shown in Figure 147. Splitters cut the signal power in half, so if many of them are used sequentially the signal may have to be amplified before it gets to a receiver to be useable.

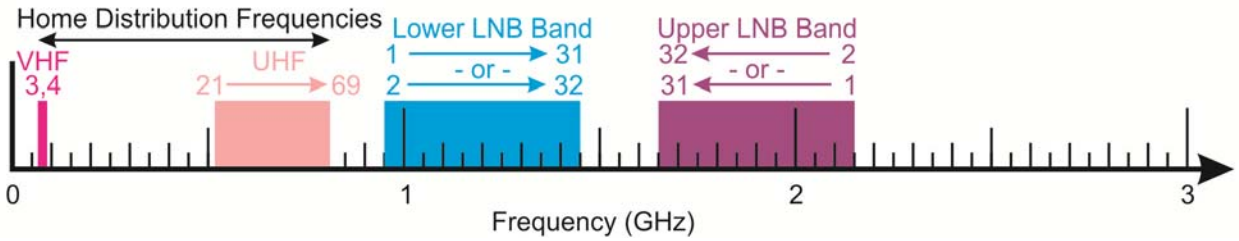

Figure 148: TV 1 and TV 2 Signals Coexist with Transponder Bands.

The subject of splitters brings up a point we've not yet discussed: distribution of TV signals in non-MoCA systems. We've already talked about how the Hopper and Joey use a non-overlapping MoCA frequency band to communicate between themselves. Some of our other receivers also are able to send the TV 2 signals across coax to television sets not located in the same room as the receiver. They use channels 3 and 4 for communicating with the local TV 1 and can use over-the-air channels 21-69 to send TV 1 and TV 2 signals through home distribution.* Figure 148 shows the locations of these frequency bands and graphically illustrates how they don't overlap with the satellite signals that can also be present in the same cable. Note that the UHF band would overlap with the MoCA band, but our receivers either use one or the other, so they'll never both be present to interfere with each other.

Separators. The **separator's** name is also very descriptive. It takes two signals flowing in the same cable and separates them into single signals on a cable. A typical use is to take the signals from twin LNBs that are both on the same incoming cable and separate them to go into the SAT 1 and SAT 2 inputs on a dual-tuner receiver, as shown in Figure 149.

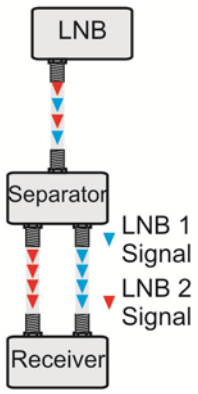

Figure 149: Example Separator Use.

Diplexers. The function of a **diplexer** is to allow both LNB and

* They can also use cable channels 72-125, but those frequencies are roughly the same as the UHF frequencies shown in the figure.

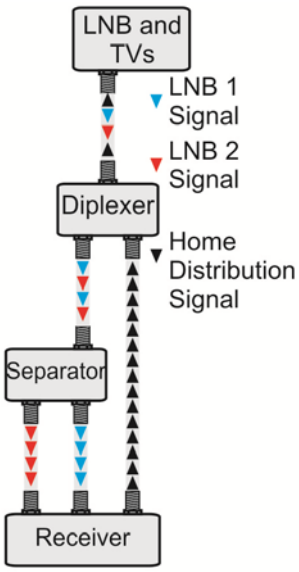

Figure 150: Example Diplexer Use.

home distribution signals to exist on the same cable. However,

it doesn't allow the home distribution signal to go back to the receiver that generated it. Figure 150 shows this function, but also includes a separator in the mix since we'll need that device to explain the last of our signal flow devices.

Triplexers. As shown in Figure 151, a **triplexer** can take the place of a separator-diplexer combination. It also allows inbound LNB and outbound home distribution signals to exist on the same cable, but is able to handle two LNB signals with a single device. A glance at the top cable and the three cables running into the receiver in the previous figure and comparing them with the cables hitting the triplexer will confirm its function.

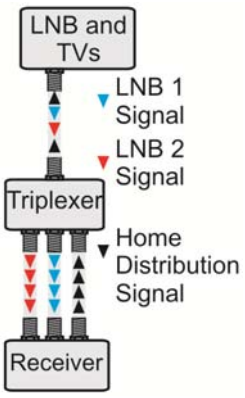

Figure 151: Example Triplexer Use.

That wraps up our discussion of signal flow devices. Now we'll move on to the most basic bit of hardware that allows us to move our signal around between our customer's dish and their TVs: coaxial cable.

Coaxial Cables

During our discussion of LNBs, we said one of their primary functions was to separate the two polarizations and then shift the satellite frequencies they received down to two different blocks of frequencies that could pass through the coaxial cables, or **coax**, we use throughout our customers' houses. Coax is how our signal gets from place to place once it's been processed by the LNB. There are many, many kinds of coax, each with different strengths and weaknesses. Some are suitable for lower frequencies, some for higher. However, almost none of them are designed to handle much above 5-10 GHz. In fact, we've already discussed nodes, devices which up-convert our signal frequencies while staying below the highest frequency DISH uses once we're past the LNB, 3 GHz. Let's see why it's in our best interest to keep those frequencies below the 3 GHz coax "limit."

Resistance and Heat. In order to understand why cables don't perform as well at higher frequencies, we'll have to understand a little bit about electricity and a little bit about heat. Let's tackle heat first. What we all sense as *temperature* is just being able to tell how fast the atoms in a material we're touching are moving. Pretty cool, huh, that you have atomic speed sensors built into your body? Heat is just a measure of how much microscopic energy of motion objects have. For the air, *hot* means the air molecules are zipping around from place to place very quickly compared to how fast they move when the air is cold. Figure 152 shows a schematic of cold (Frame I) and hot (Frame II) gas molecules—the blue dots—with the hot molecules shown moving much faster.

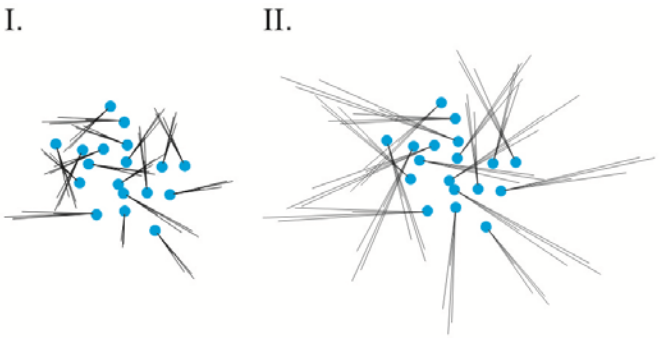

Figure 152: Cold and Hot Gases.

For a solid material like the metal in a wire, the atoms can't move from place to place like they can in a gas. They're pretty much fixed in one location, bound to their surrounding atoms by spring-like forces. Figure 153 shows this situation. Here, the atoms are represented by vibrating blue dots, the springs by jagged red lines, and the normal, non-vibrated positions of the atoms by a regular pattern of light purple dots. When solids get hot, their molecules just vibrate about their undisturbed positions, back and forth, right and left, and up and down. The molecules in the cold solid, in Frame I, are all pretty close to their undisturbed positions and their springs aren't compressed or stretched very much. Contrast that with the much more chaotic scenario of the hot solid in Frame II, where there's much more motion.

What happens when you add lots and lots of heat? More heat than is shown in this figure? The vibrations become strong enough to overcome and break the bonds that normally don't let the atoms in the solid move. When the atoms start to move around more freely, we say the metal has melted. Hopefully this is ringing a bell from what you got in middle

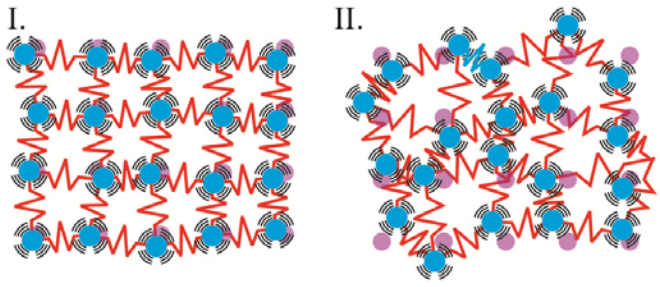

Figure 153: Cold and Hot Solids.

school and high school science, if not in college.

Enough about heat. Let's talk about electricity. More high school science for you: atoms are made up of protons and neutrons in the central part of the atom called the nucleus. They also have much smaller and faster electrons surrounding them in a cloud. In most materials, electrons are pretty tightly bound to their atoms. However, in some materials we call *conductors*, the outermost electrons can flow *relatively* freely from atom to atom. Most metals are good conductors, and the conductors we use in our cables are called *wires*, which are nothing more than long, thin objects made out of metal.

When electrons flow from place to place, we call that an *electrical current*. All the signals inside the cable we've been talking about in this chapter are made of electrical currents. Since our signals oscillate billions of times a second, the currents in the cable that transmit the signals also oscillate back and forth billions of times a second. Earlier, we said that electrons flow *relatively* freely. What we mean by that is from time to time an electron will run into an atom in the wire, restricting its free movement.

You can see what happens during such a collision by imagining dropping a small marble (representing the moving electron) into a collection of springs and balls (representing the atoms in the wire), similar to what's drawn in Frame I above. More often than not, the marble would pass right through the objects. However, if that marble hit one of the balls, the collision would slow the marble down and start the ball moving. That ball's motion would then be transferred to the other balls by way of the springs, and the whole ball/spring collection would eventually vibrate more than it was before you dropped the marble. Energy would be taken away from the marble and would be transferred to the vibrations of the balls and springs. And what did more motion in the balls mean? The balls and springs heated up!

This analogy shows how collisions take the energy of motion of the electrons and turn it into heat within the wire. Collisions mean our electricity doesn't flow as easily within the wire. When talking about electricity, "not flowing as easily" is described by the word *resistance*. More resistance, more heat, more energy loss from the electrons, electricity doesn't flow as easily. Stay with us! We're now well on our way to understanding the coax upper frequency limit.

In general, we can say the resistance of a wire is related to how much wire there is. Thus, a thicker wire has less resistance than a thinner one. That's because for a given number of electrons moving through the wire, there is more open space

available for them to move through in the bigger wire than the smaller one. More open space means less chance for collision and less resistance. To see why this is true, look at Figure 154. It shows a cross-section of two wires, with the atoms in the wire being represented by darker orange circles within the lighter orange cross-section. The density of the atoms, represented by the number of atoms per area, doesn't change between the two wires. On average, there's the same amount of space between atoms in both wires.

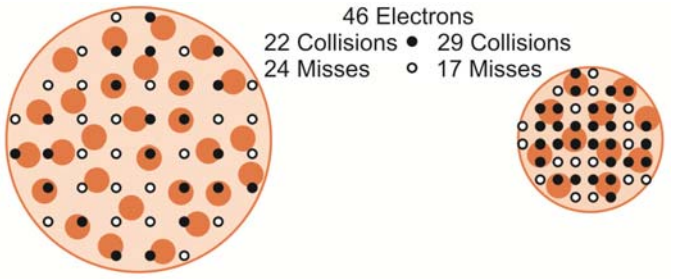

Figure 154: Smaller Wires Have More Resistance.

We've also drawn electrons on these wires, represented by the grid of white and black dots. We need the same amount of electric current—and current is related to the number of electrons in motion—so we've drawn the same number of electrons on each wire. Since the right wire is smaller, in order to fit the same number of electrons we had to put them closer together. We've indicated electrons that do not touch any atoms with white circles and ones that do touch—ones that collide—as black dots. As you can see, there are more collisions in the smaller wire than the larger one because the electrons are closer together and thus have a better chance of hitting atoms. While electrons flow relatively freely in a wire, forcing more and more of them into a smaller and smaller area by making the wire thinner means they're more likely to run into the atoms. When they run into the atoms, they make them vibrate and generate heat. Thus, a smaller area for the electrons to flow through means more resistance and it becomes more difficult for electricity to flow.

Additionally, the further the electrons travel down the wire, the more opportunity they have to run into atoms and give up their energy of motion to heat energy. The end result is that electrons won't travel forever in a wire because eventually they hit enough atoms that they just can't go any further. So two things we've discovered which will cause higher resistance are:

1) Cramming electrons closer together in a smaller area and
2) Having them go longer distances in a wire, where their opportunity to run into atoms increases.

Frequency Dependent Resistance. At this point, you may be wondering what all this stuff about resistance has to do with why we can't use our coax cables at high frequencies. Well, it's now time to examine that question. When we run a *steady* current though a wire, the electrons space themselves out pretty evenly throughout the wire, as we drew in Figure 154 and redraw without showing the atoms in Frame I of Figure 155. However, when we put an *oscillating* current through the same wire, the moving electrons tend to gravitate toward the outside of the wire, as shown in Frame II.[89] The oscillating current thus forces more of the electrons into a smaller area, which from rule 1 above, means we'll get higher resistance. The higher the frequency, the more pronounced this effect, so resistance goes up as we increase frequency.

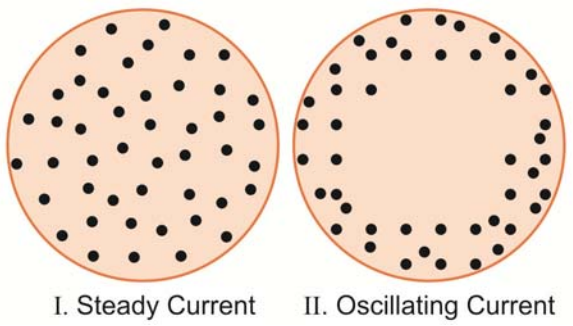

Figure 155: Electrons Concentrate Near the Edge of Wires in Oscillating Currents.

So, getting back to what happens to oscillating signals in a wire, the resistance gets greater as frequency increases because more and more of the electrons are concentrated very, very near the wire's surface. With more electrons in the same place, it gets really crowded and the odds of a collision go way up. That means the distance along the wire it takes before they run out of steam decreases. Eventually, as the frequency gets higher and higher, the signal being carried on the wire drops to near zero after a very short distance along the wire.

Note that for signals that oscillate (i.e., everything we use to get our product to our customers), *all* wires can become long enough that they will eventually have enough resistance to cause our signals to fade below the noise level. So, our problem is to find a cable that can handle all the frequencies DISH uses without having them fade before we can get them to our receivers.

Characteristics of Good Wires. Because we're going to be transmitting high frequencies in our cables, one of the things we want to look for in a wire that will reduce resistance is a large cross-sectional area. The bigger the area, the more the electrons can be pushed toward the edges before they really start getting packed in. We also want the very best conductor we can get, one that allows electrons to flow just as freely as possible. A measure of how well a material allows electrons to flow is called its *conductivity*.

Figure 156 shows the relative conductivity of various metals, where we've set the value of silver to be the maximum and show the other materials relative to how good silver is. The figure clearly shows that silver and copper are examples of very good conductors. Silver is very expensive, so copper is the most commonly used material for wires. However, aluminum, with a conductivity only 60% as good as copper, and copper-coated steel are sometimes substituted as a cheaper substitute when exceptional conductivity is not as critical.

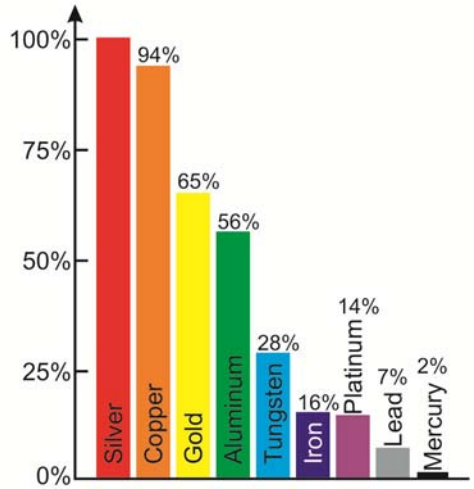

Figure 156: Relative Conductivity of Selected Metals.[90]

Anatomy of a Coaxial Cable. So how do we go about finding the proper cable? Let's look at what they're made of and then see what effect how they're made has on their ability to move our signal around. The differences in the varieties of coax are primarily due to their construction. Figure 157 shows a cut-away drawing of your basic coaxial cable. You can see that it's made up of four parts: two conductors and two insulators. The differences between types of cable are in what these four parts are made of and what their dimensions are.

The inner conductor (also called the *center conductor*) is typically a solid piece of copper wire, but as we've discussed, some manufacturers use cheaper alternatives which reduce the ability of the cable to carry higher frequencies because of increased resistance. This conductor is the wire to which we apply our oscillating signal.

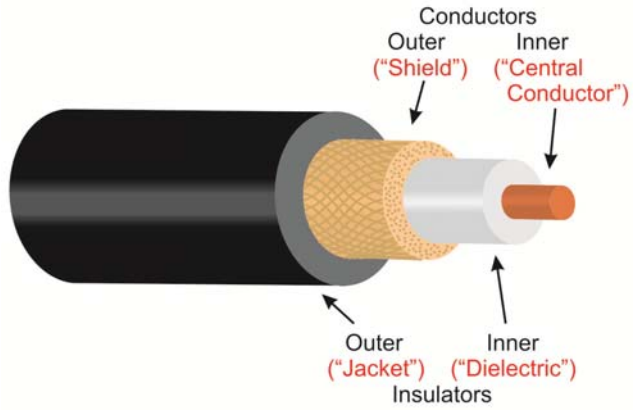

Figure 157: Coaxial Cable Components.

The inner insulator (also called the *dielectric*, which is a general name for an insulator with some specific electrical properties) serves two purposes: it keeps the inner and outer conductors from touching, which is good since the signal would short out were they to touch. It also serves to keep the cable in its round shape and separates the inner and outer

conductors by a very precise distance and shape. These form-factors become more critical at higher frequencies, since our signals actually travel within the space between the conductors—moving as a wave *inside of* the inner insulator. Having a precisely-defined, constant shape allows the signal to travel more freely in that space.

The outer conductor (called the *shield*) is typically braided copper or aluminum to allow it to flex somewhat freely. It is typically grounded, while the oscillating signal is applied only to the inner conductor. Hooking up the cable in this way serves two purposes. First, it helps to ensure our signal is confined to within the inner insulator. Secondly, it also keeps out any external signals that could interfere with the one we've worked so hard to put on the cable. Since the woven outer conductor acts as a shield, keeping the good stuff in and the bad stuff out, coaxial cable is sometimes called *shielded cable*.

While we earlier stated that the FCC didn't regulate what goes on inside our cables, the shielding in our cable is important because the frequencies we use, were they to escape into the wild, are the same ones used by aircraft to navigate, radio astronomers to explore the universe, and many other groups who hold the licenses in that part of the spectrum.[91] If our shielding fails, we'd potentially interfere with those authorized users. Shielding is important enough that some manufacturers put two or even four layers of it in their cables.

The last component of coax is the outer insulator, called the *jacket*. Its purpose is to protect the outer conductor from the elements and to keep it from touching any other conductors. It's a pretty mundane part, but very necessary.

Selecting the Right Cable. The radii of the various parts contribute to the cable's ability to carry higher frequency signals. As we've seen, bigger is better when it comes to conductors. The inner conductor only has a single radius associated with it, but the other three components have both inner and outer radii, both of which are relevant design criteria. When selecting a cable, having a thicker inner conductor and a thicker outer conductor would generally be better. The thickness of the inner insulator is determined by the range of frequencies we'd like to transmit, while the outer insulator only needs to be thick enough to be durable and to prevent physical contact between the outer conductor and external conductors.

While there are many, many types of cable, only two are commonly used to wire homes and businesses today. Let's take a look at them in detail. These cables are called RG-6 and RG-59. The numbering comes from an archaic system developed

during World War II where RG means *radio guide*. The numbers appear to be arbitrarily assigned, as they don't follow an order that would be dictated by any size or electrical qualities. The dimensions of the two cables we'll study are shown in Figure 158.

Cable Type	Inner Conductor Outer Diameter	Inner Insulator Outer Diameter	Outer Conductor Outer Diameter	Outer Insulator Outer Diameter
RG-6	0.04"	0.19"	0.22"	0.27"
RG-59	0.025"	0.15"	0.17"	0.24

Figure 158: Typical Dimensions of Two Common Coax Cables.[92]

We'll start out by saying up front that DISH only uses RG-6 cable, and in fact the RG-6 we use is also certified as being "swept to 3GHz," as you'll easily be able to see since it's printed every few feet on the cable. While the electrical characteristics of non-swept RG-6 *should* be good enough for our use all the way up to 3 GHz, swept cable has been tested to ensure it really does meet specifications. With those caveats, why do we include a discussion of RG-59 at all? Well, for some applications RG-59 is actually fine, and many of our customers' homes are already wired with RG-59. We'll look at the differences so we can understand when the less expensive (and less capable) cable might be OK to use.

As can be seen in the table, RG-6 is larger and has a larger conducting core. This larger core has a larger surface area, so the high-frequency "friction" force we just looked at isn't as concentrated as it would be in a smaller cable. The net result of spreading out those "frictional" losses is that the cable performs better than the smaller RG-59. A common way to measure how well a cable performs is to quote the amount of signal that is lost in a certain amount of distance. With the above

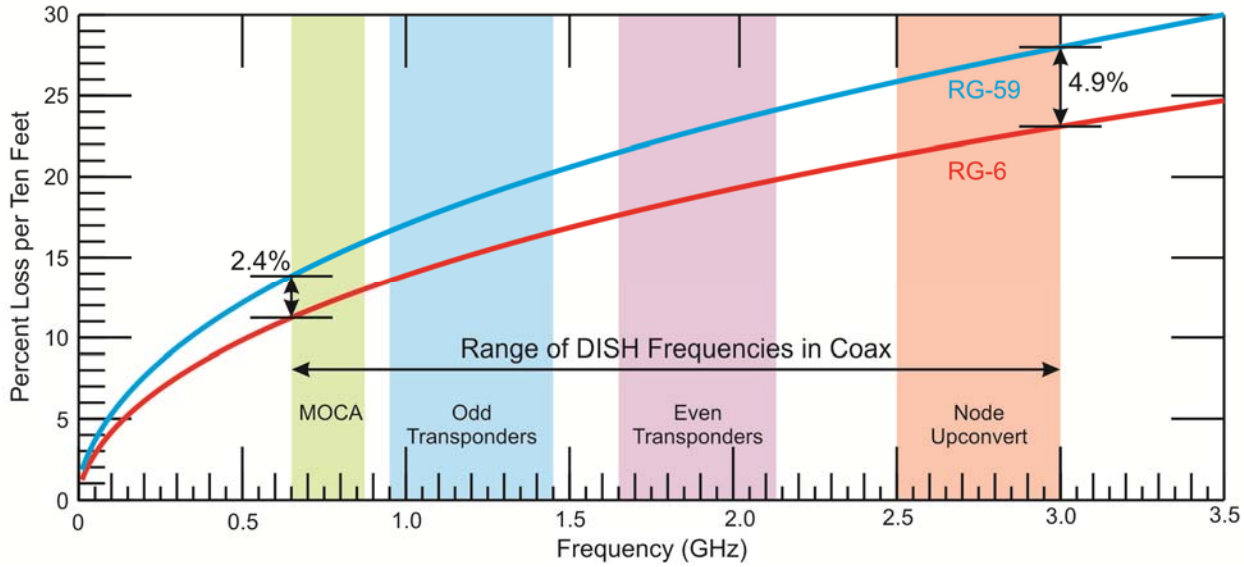

Figure 159: Comparison of Signal Loss in Common Coax Cables.[93]

discussion on resistance we've stated that higher frequencies aren't transmitted in cables as well as lower ones. Figure 159 confirms that cable performance, as measured by the loss of signal in the cable, really does depend upon frequency. We've chosen to measure signal loss as the percent it degrades after passing through a cable ten feet long, and plotted this degradation as a function of frequency. For this factor, values near the top of the figure are worse than those near the bottom. For ease in interpretation of which frequencies are relevant, we've also color-coded the frequency bands DISH uses to move our signals around our customers' homes.

Notice that at higher frequencies (toward the right of the figure), the curves move upwards showing that an ever-increasing percentage of signal is lost over the same distance. Also notice that the red curve, denoting our preferred RG-6 cable, is always below the blue RG-59 curve, showing that it always loses less signal at a given distance and frequency.

DISH uses the frequency band between 0.65 and 3.0 GHz. At the lower end of the band, our cable loses about 11% of its signal every ten feet. That doesn't sound like much, but it quickly adds up. Figure 160 shows how adding additional distance quickly brings the signal down toward zero. The solid red curve in that figure demonstrates the cumulative losses in the cable, and shows that after 100 feet only about 30% of the signal remains.

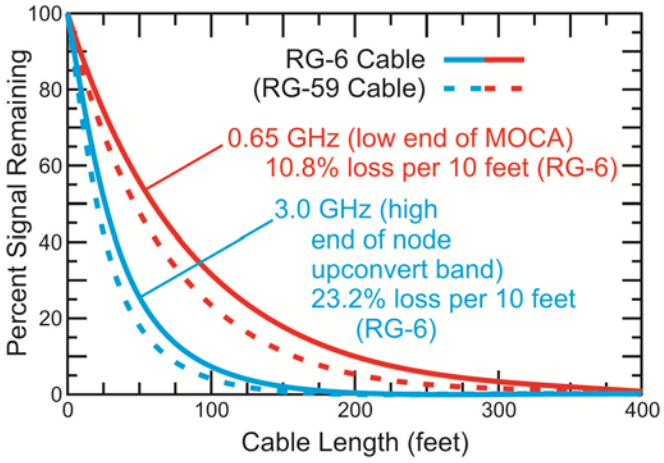

Figure 160: Cumulative Effects of Small Signal Losses Add Up.

But lower frequencies are where cable is supposed to do better. Let's look at the high end of the band DISH uses and see what happens there. As seen in these two figures, RG-6 loses about 23% of its signal every ten feet, meaning that after 100 feet only 8% is remaining. That's a huge drop, isn't it? Well, not when judged against RG-59.

Comparing the loss-per-ten-feet of both cables, RG-59 is almost 25% worse than RG-6 at the low end of the frequency band (losing 14% versus 11 percent), and is 20% worse (a 28% loss versus 23%) at the high end of our band. That translates

to only 4% at the high end and 23% at the low end of signal remaining after 100 feet of cable. The bottom line of this discussion is that while both kinds of cable have a lot of losses RG-6 is definitely a better way to go, especially at the 3 GHz end of our frequency band. Based on this discussion, why would anyone want to use RG-59? Well, it's cheaper. Other than that, we can't think of many reasons to recommend it if RG-6 is available.

However, there is one special application where RG-59 might work. Take another look at Figure 159, paying special attention to where the MoCA band is. Now, in that figure, being near the bottom of the plot is better because there's less loss down there. Compare the worst-case RG-59 loss in the MoCA band (about 16%) with the worst-case RG-6 loss in the other three bands (16.5%, 20%, and 23%, respectively). Now, if 23% is an acceptable loss for RG-6, then the much MoCA smaller loss of 16% in RG-59 should be acceptable as well. Why do we bring this up? It means that in a situation where *only* MoCA information is being transmitted through the cable, RG-59 is perfectly acceptable! If your house is already wired with RG-59 and you get a Hopper, for example, you only really need the installer to run RG-6 from the LNB to the Hopper. Any Joeys in the house can then use your existing wiring with no ill effects.

Chapter Summary

It's been quite the ride so far: four chapters following the signal and we've finally gotten inside our customers' homes. In this chapter, we caught up with the signal as it hit the satellite dish. We saw the dish focus it on the antennas in the LNB. We found out that there were a lot of obstacles to overcome on the way down from space, though, that threatened to prevent the signal from being strong enough to give our customers viewable programming. Some of these obstacles were precipitation, which scattered and absorbed our electromagnetic waves and even raised the background temperature seen by our rooftop dishes. Who would have even thought the temperature of rain might be a relevant problem for satellite communications!

We then spent some time looking at what happens to every dish pointed at a geostationary satellite twice a year—solar interference. We saw that without putting up prohibitively expensive satellites, we just weren't going to be able to outshine the Sun when our satellites lined up with it, so we'd have to deal with solar outages.

We then put together everything we'd learned about conditions from the uplink center to the satellite and back down to our rooftop dishes to find out what was

important to construct a link budget. The link budget discussion showed us that we just can't guarantee a signal to any of our customers 100% of the time, but we do pretty good, getting programming to them for all but, on average, 13 hours a year.

We then moved on from looking at electromagnetic waves in space and air to seeing how we moved the signal around inside the house. We saw how LNBs took the two polarizations of the wave, separated them, and down-converted their frequencies into blocks that can be carried over coaxial cable. We looked at a number of devices used to direct signals from the LNBs and receivers to the right places inside the house. We finished the chapter with an in-depth look at the most basic of signal flow components, the lowly coaxial cable, and saw that we really did need to use quality cable because of the four frequency bands the cable has to carry simultaneously, especially the highest-frequency band we use, the LNB node up-convert band.

We're betting that at the start of this chapter you figured it would be pretty humdrum. We mean, how complicated can a chapter called *The Dish on Your Roof* really be? Well, after almost 60 pages we think you've got a lot better grasp of all the many things that have to go right and the many unsung bits of hardware that are required to get the DISH signal inside a house. We've covered a great deal of information in this chapter, and finally after four chapters and over 45,000 miles of travel, our signal is ready to be transformed by our receivers into the programming that makes our customers happy and our company profitable.

5. Inside Our Customer's Home: Receivers

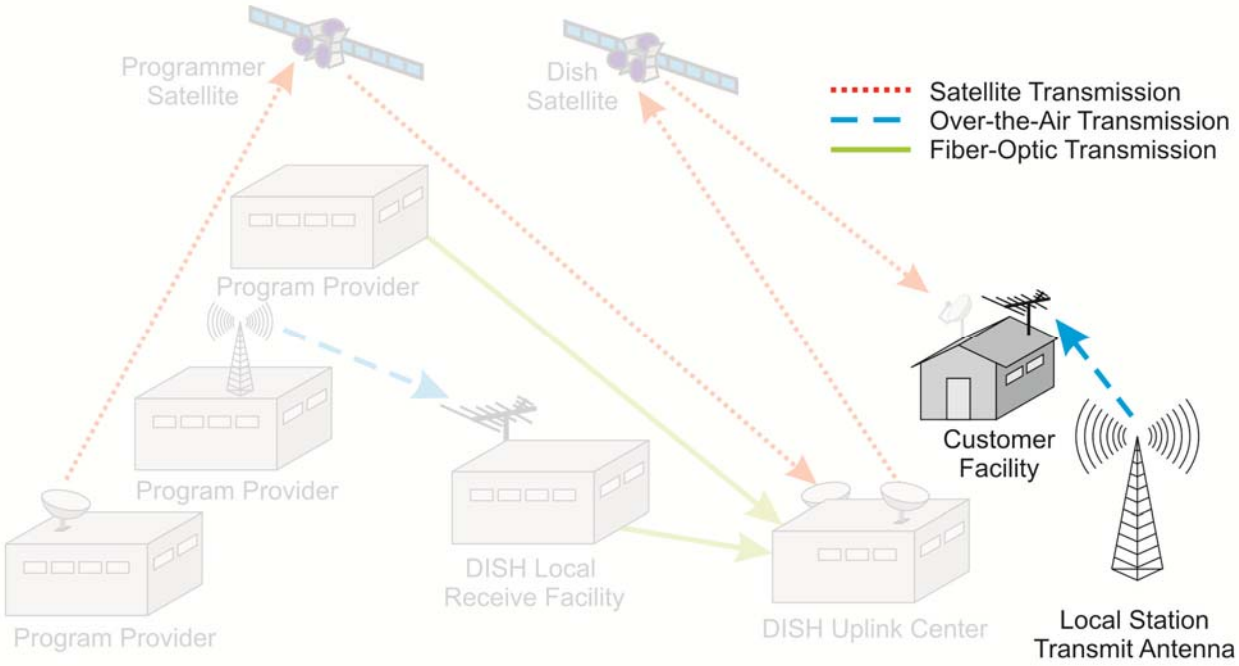

Figure 161: The Signal Flow Roadmap—In the Customer Facility.

Complex yet friendly Boxes find, unlock, and send Channels to the screen.

The Basics

(Continued from p. 131) So, we've finally made it to the last stage of our signal's 45,000-plus mile path from our uplink centers to our customers' homes. How much left can there be for us to learn? Well, lots, to be succinct. By the way, in this chapter you'll see a lot of cross references to information we've covered previously. That's because much of what the receiver is supposed to do is to *undo* the things we did to the signal at the uplink center. If you've forgotten some of these concepts, we strongly recommend you review them when you come across the cross reference.

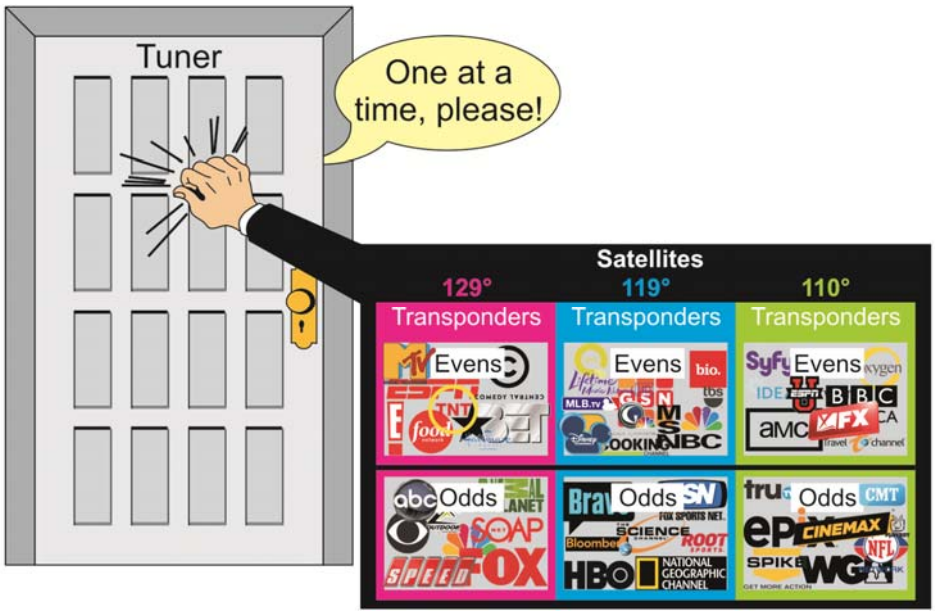

Figure 162: Receivers Select a Single Channel from the DISH Cornucopia of Possibilities.[94]

As shown in Figure 162, what we're starting with in most homes is six signals from three satellites carrying several thousand channels. The signal in the form it has when it gets to the tuner doesn't mean squat to a television set, so the receiver has to convert it into a form that does. Also, the customer typically wants to watch one show at a time (two, with picture in picture), so the receiver needs to do some culling. Taking that huge amount of information and deciding what to do when our customers push a button on their remotes takes quite a bit of prior planning to make both them and us here at DISH happy.

To make the customers happy, they've got to have seamless, continuous access to the programming they've paid for. While that makes us happy, too, we've also got to ensure that they're not getting *more* than they've paid for at the same time. In order to continue to provide them with great entertainment options we've got to stay profitable, and staying profitable doesn't normally involve giving away our product for free.

So let's take a brief tour of what actually happens when our customer chooses to watch, say, a baseball game. The customer looks in the electronic program guide (EPG) and selects the current program, *Colorado Rockies vs. Texas Rangers*. As soon as she presses *select*, a near instantaneous but highly complex process is set into motion.

Before we go into that process, however, we need to understand a bit about what the receiver already knows. There's a surprising amount of information stored in it, and that information is crucial for it to be able to deliver channels quickly and

accurately. We talked about encryption a little bit in the *Prepping the Signal* chapter (see p. 31 for a review), but at *The Basics* level, we said that the smart card in the receiver was continually updated with codes that let the receiver know which channels it was authorized to display and codes that told it how to decrypt those authorized channels. The receiver continually refreshes this information because on *every single transponder* we use there is at least one data stream that the receiver can access containing *all* of the encryption information.

In addition to this encryption information, on every transponder there are other data streams that contain information about how the receiver can find every one of the over 7,000 channels DISH broadcasts—and that's a lot of information! Also, remember we transmit different data streams for audio and video so customers can, for example, choose to listen to a different language for the show. The receiver needs to know where to go to find audio as well as video. The data stream that carries this information thus needs to match up every channel—both audio and video components—with a specific satellite and a specific transponder number where it can be found.

Finally, there are other data streams that contain program information, both for the programs on at the current time and for programs scheduled in the future. That data is used to construct the EPG as well as to provide the text customers see when they press the *Info* button. These data streams containing information about the service DISH is providing are treated a little differently by the receiver than normal channels, which are only let through the door one at a time. As illustrated in Figure 163, those special streams have special access and, like celebrities at an exclusive big city club, are let behind the velvet rope whenever they arrive while the rabble—the audio and video content for all our channels—

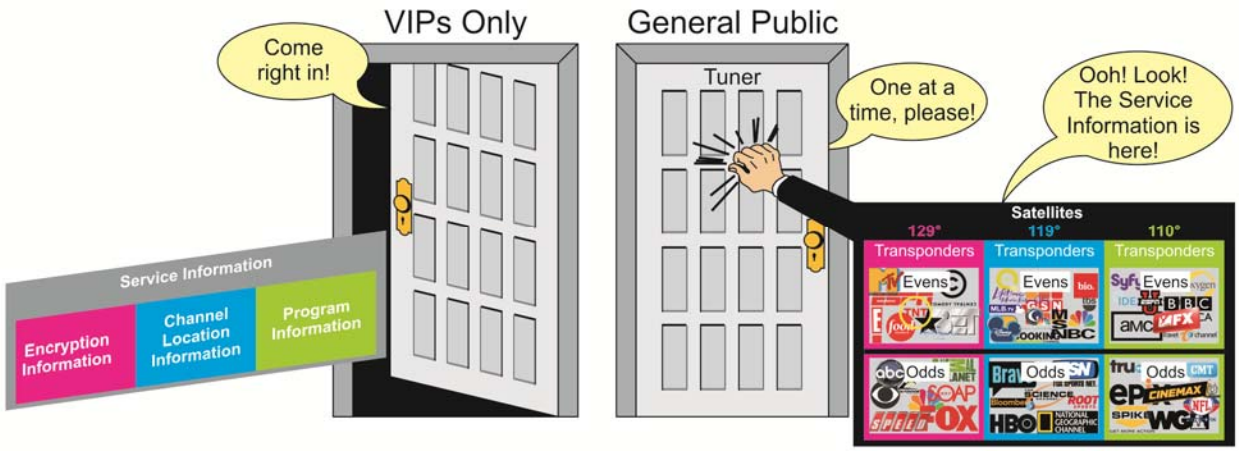

Figure 163: Receivers Need Special Information.

have to wait in line.

Those encryption, transponder, and program content streams are called service information, and the service information arrives quite often. The passwords we use to decrypt channels change every twenty seconds or so (and there's a different password for *every* channel). We're also required to transmit the information about data stream locations and program information many times a second in order to assist with rapid channel acquisition. Bottom line: there's a lot of information coming into your satellite dish and being stored by your receiver that you never see, information that's absolutely necessary for the receiver to operate correctly.

So with all of this service information now safely tucked away inside the receiver, you should be able to see there's enough there to construct the EPG our customer used to see that the game was on Root Sports right now. Let's get back to what happens when she decides to watch the game.

As soon as she presses the remote button selecting the show, the receiver translates that location on the EPG to channel 414, the HD version of Root Sports. It knows channel 414 is Root Sports because of data it got from the service information. It then looks to find where channel 414 is located and discovers that there is a single video stream, an audio stream, and two data streams for closed captioning and scores associated with channel 414. All of these streams are located on transponder 7 of the DISH satellite at 119° W (we'll call this 119/7).* The receiver then sends a signal to the LNB via the coax cable asking it to send down the frequency block carrying the odd transponders from 119.

Simultaneously, the receiver checks to see if it is actually allowed to display Root Sports. Through some very complicated and very closely-held procedures, the smart card knows which channels the customer has paid for. If Root Sports isn't one of them, the process ends here and the receiver will display a polite message something like, "We're sorry, but this channel is not available in your current package." If the receiver is connected to the Internet via broadband or phone line, in some cases there will be an option displayed to immediately add the channel to the customer's package.

However, for the sake of argument let's assume that our customer does have Root Sports in her authorized list. (If she didn't, this section would be really short

* The stream types and numbers, transponders, and satellites quoted throughout this document are not necessarily the actual values. Those numbers are EchoStar confidential information.

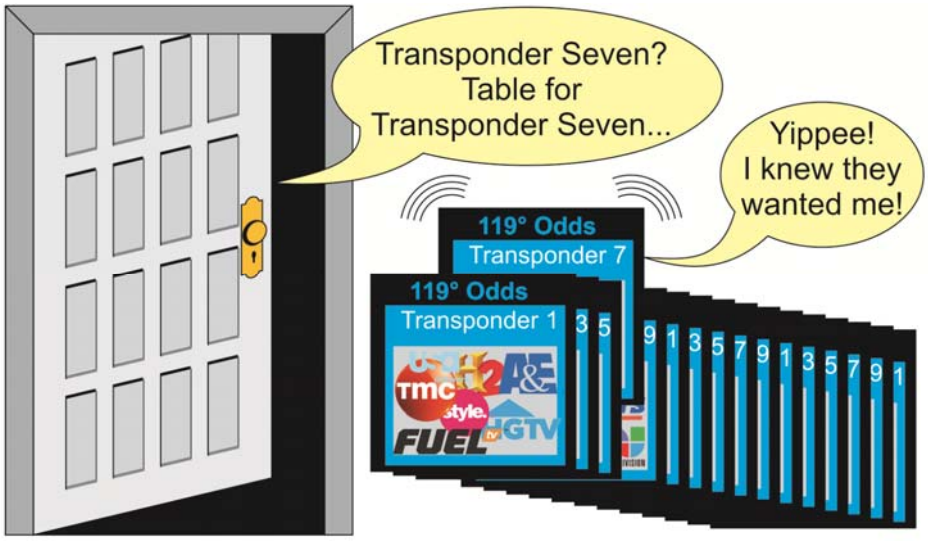

Figure 164: Before Selecting a Channel, the Tuner Allows One Transponder's Frequency Band Inside.

and not as informative as it needs to be.) The smart card will then retrieve the appropriate control word and will have it ready and waiting to decrypt the data stream once the time comes. At this point the Root Sports signal is still multiplexed into all of the other channels that have been transmitted on transponder 119/7, and it's still part of the 16 transponders' worth of information that are knocking at the tuner's door.

After a little incidental signal processing we won't go into here, the tuner does what it's supposed to do and only lets in the signal from 119/7, discarding fifteen other transponders—119/1-5 and 119/9-31 (remember, it's only the odd transponders that the LNB sent to the tuner). This concept is shown in Figure 164. Recall from the *Prepping the Signal* chapter (p. 35), that we *multiplex* several channels' audio and video streams together on a single transponder to more efficiently use our limited spectral bandwidth. That means that 119/7 contains not only the audio, video, and data streams for Root Sports, but for a dozen or two other channels as well. However, those channels are all mixed together and aren't yet ready for separation.

So now we've got a single transponder's information, but it's still coded like it needed to be when it was sent up into space using phase shift keying, either QPSK or 8PSK (p. 69). The receiver takes the phase-coded message, reads the code, and converts the stream back to the digital series of ones and zeroes, as shown in Figure 165. It then checks the stream

Figure 165: Demodulation Converts the Wave to a Digital Stream.

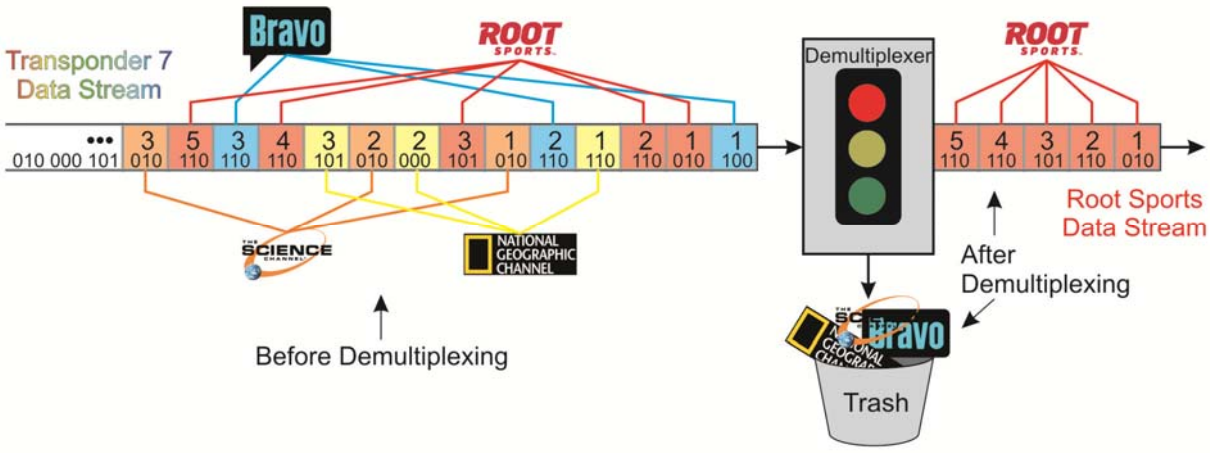

Figure 166: Demultiplexing Separates out the Specific Channel.

and ensures that no errors have made it through the system, correcting any that it finds.

However, the digital stream it creates contains multiplexed data from up to ten or twenty channels and it only needs the one the customer wants displayed. It separates out the digital packets that belong to Root Sports and throws the rest away in a process called demultiplexing. That process is illustrated in Figure 166, where sequentially-numbered packets of digital data from four representative channels on 119/7 are shown on the left. They hit the demultiplexer, which only allows through the packets from the channel the customer has requested, throwing away all the others.

We're finally down to just the data the customer wants, but there's a big problem: that data is encrypted so it couldn't be stolen. The receiver dips back into the service information and, using that information in conjunction to keys stored on its smart card, decrypts the digital stream so it is converted back into MPEG format as shown in Figure 167.

Figure 167: Decryption Makes the Scrambled Data Readable.

The next stop for the signal is the MPEG decoder, which looks at the signal frame-by-frame, figures out which frames should be displayed in which order, and synchronizes the audio and video streams so lips and sound match up, demonstrated in Figure 168. The signal is finally ready to go to the TV. Our customer finally gets to watch her baseball game!

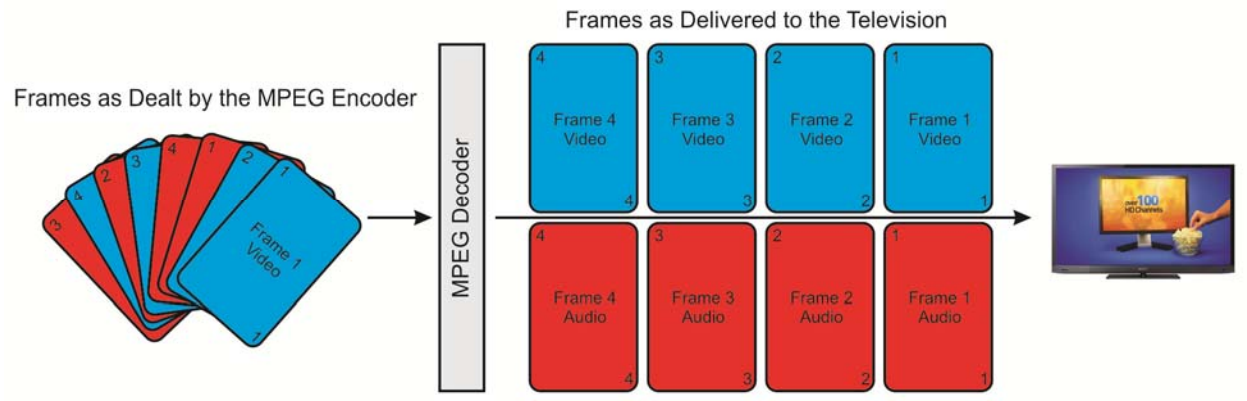

Figure 168: The MPEG Decoder Reconstructs Frames and Puts Them in the Right Order.

Summary. Some of the major points you should understand are:

- In a typical home set-up, we install a dish capable of receiving all three satellites in either the Eastern or Western Arc, meaning there are six possible bands of signals (even and odd transponders) hitting our LNBs
- The receiver is constantly getting and storing for later use updated sets of service information that tell it where on the different satellites' transponders channels are located, what information is included for each channel, information related to the electronic program guide, and passwords for each of the channels
- Selecting a channel number on your remote sets off a complex chain of events inside the receiver that results in only one of thousands of possible combinations of audio, video (and potentially data) information being displayed on a television screen
- The receiver tells the LNB which band of 32 transponders it needs, takes that band and filters out all but the transponder containing the desired channel, and then converts the 8PSK (or QPSK) signal to a digital stream of ones and zeroes
- It demultiplexes this signal and discards all streams not related to the channel of interest
- If the customer is authorized to see the channel it next decrypts the signal into an MPEG-standard digital stream
- The MPEG decoder then converts the digital stream into a form that is TV-ready and sends it to the customer's television.

So this brings us to the end of *The Basics*, which means you're done if that's all you've signed up for. Thanks for staying with us this far! If you've signed up for the full course, there are a few more pages for you to read.

Technical Discussion

We really hope you're a fan of baseball, because we're going to go into much more detail on how to watch the Rockies play that game using your DISH subscription. Watching a game seems like a simple thing to do. You find it in the guide, select it, and, voilà, it shows up on your screen. However, there's a whole lot that goes on behind the scenes inside the receiver to make that seemingly simple thing happen.

The purpose of a receiver is to display the one program requested by the customer on their television set (two programs, if you're watching picture-in-picture). Considering that all the receiver has to work with as raw material is a coded electromagnetic wave wiggling up and down billions of times a second, this feat is actually pretty miraculous.

In *The Basics*, we've just broken down the fundamental steps required to get from wave to entertainment, but as you've come to expect in these *Technical Discussions*, we're going to look at those steps in quite a bit more detail. However, before we get into the guts of the receiver, we really need to revisit the topic of multiplexed data streams.

A Second Look at Data Streams [95]

Way back in our first technical chapter, *Prepping the Signal*, we saw how multiplexing was a way to use our bandwidth efficiently. We have a limited number of transponders available to us, far less than we would need for each of the thousands of channels we broadcast to be assigned its own. Each transponder can also handle over 40 Mbps, but a single channel's programming only takes between about 500 kbps and 12 Mbps. Thus, by sending more than one channel at a time to a single transponder via a transport stream we minimize the number of transponders needed and maximize bandwidth use. If these statements are sounding a bit like Greek to you, look back at Figure 22 (p. 36) for a description of basic multiplexing and transport streams.

However, in that earlier chapter we showed you only a very basic look at how the channels are digested by the multiplexer before they're sent into space. In order to appreciate the complexity of the programming inside our receivers, we really need to delve a little deeper here. Figure 169 will be our starting point for this deeper look. At first glance it looks pretty busy—much busier than Figure 22—but we'll get you sorted out on what it all means pretty quickly.

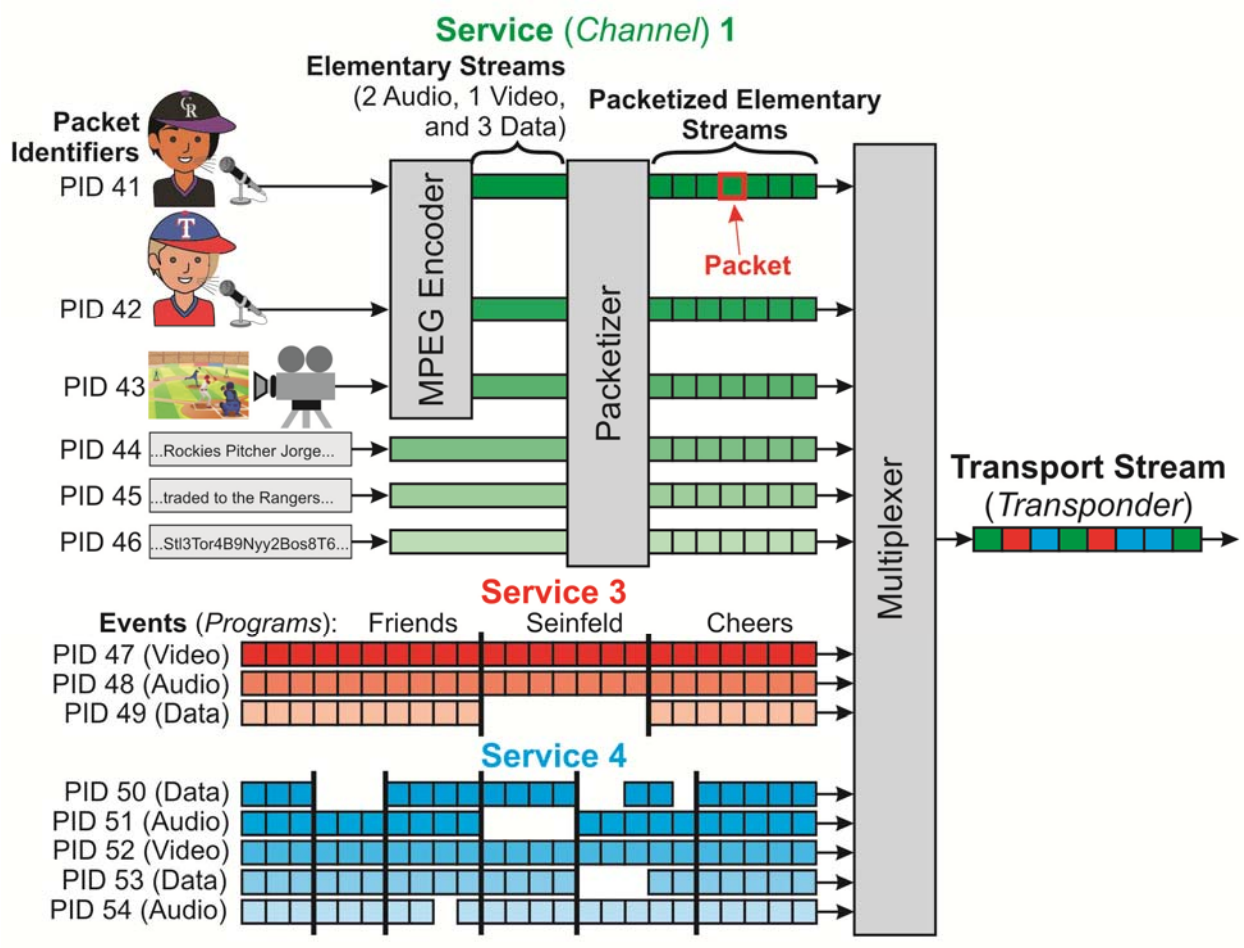

Figure 169: Elementary Data Streams Become Transport Streams.

The figure shows three channels being multiplexed into the transport stream, color-coded green, red, and blue. In receiver-speak, channels are referred to as **services**. Service 1, the green one at the top of the figure, is currently broadcasting an interleague baseball game between the Rockies and the Rangers. In this case, the channel consists of six **elementary streams**. These streams consist of information of only one of three very specific types: video, audio, and data. Most services will have at least one audio and one video elementary stream. Most services will also have several associated data streams.

In this case, the service has two audio streams, one from the announcers who call all the Rockies games and another for the folks who want to hear about how Texas is doing. Both of these audio streams are tied to a single video stream, which is yet another way we sometimes use to employ our bandwidth efficiently. There's not much reason to send two bandwidth-hungry video streams of exactly

the same game, is there?* While our sports broadcast needs separate home and visitor audio streams, there are many other uses for multiple audio streams. The second audio channel for a different language is another example.

This green-colored service also has three data channels, in this case two carrying closed-captioning for the two announcing teams and an additional one that has scores and innings for the day's other Major League Baseball games. Just like we didn't send two video streams for the game, we can also reuse data channels across different services. For example, would it make sense to send the baseball scores as part of every service carrying a baseball game? The trick is to make sure the stream is associated with the correct channels—we'll see how that's done shortly.

You'll notice that flowing in time from left to right, the streams start, for example, with words coming out of an announcer's mouth and then is converted to an electrical signal by the microphone. That signal is represented by the green rectangle to the left of the MPEG encoder. The encoder then takes the signal and converts it to MPEG-standard packets of information, converting the elementary stream into a *packetized elementary stream*.

In our figure, each elementary stream packet is represented by a colored square, as indicated by the red-outlined example packet. As we saw in Figure 22, the packets contain not only audio/video information but a header identifying which stream they belong to. The identifiers used to tie packets to a specific stream in digital video broadcasts are imaginatively called **packet identifiers**, or PIDs. We've arbitrarily numbered the six PIDs for the green service PID 41 to PID 46, but there's no reason for them to have to be sequential. These streams are defined by MPEG standard, and packets can be as long as 64,000 bytes in length, although lengths of 2,000 bytes are more the norm.[96]

Back in Figure 22 we showed the headers of each packet saying, "I'm ESPN packet number 10 million!" Now we know we have to be more specific, saying, "I'm packet number 10 million of PID 43, the primary video stream for Root Sports!" To be even *more* specific, since all of these streams have to be synchronized, it would have to say, "I'm packet number 10 million of PID 43 and this packet starts at 3 minutes and 32.375 seconds after the beginning of this event!" We won't go into much more detail than that on synchronization, but since everyone gets antsy when the lips don't match the sound, our process really does pay a lot of

* We'll ignore the need to broadcast different commercials for the different local markets in this discussion.

computational attention to making sure the streams match up at the right time. Bottom line, elementary stream packets are primarily concerned with telling the decoder what type of information they contain and when it occurs in the stream.

So that's a look at just one service in excruciating detail. Let's look at an overview of another service in the figure. Service 3 contains three red-colored packetized elementary streams numbered PID 47-49. (We'll explain why we skipped Service 2 in a few pages.) The conversion to MPEG isn't shown but you can assume it has already occurred. We've drawn this service to show that, in this example, each stream is divided in time into three **events**, things our customers would call programs. (Although we've drawn only a few packets within an entire program like Seinfeld in our figure, realize millions of them are sent every few seconds for *every* show.) In this case, it appears that while *Friends* and *Cheers* may have closed-captioning available, *Seinfeld* does not. You can see that from the disappearance of the data stream during *Seinfeld*. That was a lot more concise than the description of Service 1, wasn't it? We covered the details there. These other two services are just included to put the frosting on the data stream cake.

Blue-coded Service 4 consists of five packetized elementary streams (PIDs 50-54) divided into six events. Notice that only the video stream and one of the audio streams are continuous across all events. Every service will have at least one video and one audio stream at all times. Also notice that while the second audio channel is either present or not for an entire event, the data streams can start and stop *within* an event.

So we're now back to where we were when we first illustrated multiplexing in Figure 22. The only difference is that we now know that we have to multiplex more than one stream to get an entire channel's information to our customers. The multiplexer doesn't much care how many streams it takes in; it just figures out how fast the streams are coming and puts them out onto the transport stream packet-by-packet as fast as is required to keep any of its inboxes from filling up. Since a video stream takes up roughly ten to one hundred times as much data bandwidth as an audio stream, that means you'd expect to see the multiplexer pull a video packet out of that inbox and dump it onto the transport stream ten to one hundred times more often than an audio packet.

However, just when you thought you'd got this packet thing down, we're going to throw in another kind of packet. The multiplexer chops up these MPEG packets into much smaller bits called **transport packets** (bits—get it? ones and zeroes?). Transport packets are exactly 188 bytes long with only 4 bytes reserved for the

header, also as defined by MPEG, so each elementary stream packet will be split across many transport packets—a typical 2,000-byte elementary stream packet would take up 11 of them. The reason for this extra splitting is because instead of being concerned with content and timing, transport packets are designed to enable robust error correction, a topic we won't go into here.[97] Suffice it to say we'd like to make sure the information our customers receive is the same as what we're sending to space and back down, and error correction is the way we significantly increase the odds of that actually happening.

At this point, you may be wondering why we're revisiting the creation of a transport stream here in a chapter on receivers instead of putting it all together in the *Prepping the Signal* chapter where it might have logically gone. Well, we're counting on two things. First, we're about to discuss how the receiver constructs the program guide and how the receiver goes about figuring out which streams to put together when customers select a specific channel on their remote controls. That decision process is based on packetized elementary streams and their PIDs. And second, we think your memory isn't what it used to be; had we discussed these things way back on p. 35, we're pretty sure you would have forgotten most of it by now! How's that for confidence in our readers? So, we've described the different kinds of elementary data streams in this section with the goal of using this (probably volatile) knowledge very shortly.

Even after understanding data streams, we're still not ready to delve into what happens when our customer selects a channel, though. We've still got to look into how the receiver knows that channel 414 is Root Sports, and how it knows where to go to find that channel. That means we're ready for a discussion of system information tables.

Finding the Correct Channel: Tables

In this document we've strived to use "friendly" words and phrases, staying away from technical terms as much as possible. However, in the previous section we explained that *services* were just *TV channels*, *events* were just *TV programs*, and so on. The reason we've done that is that those terms are part of another standard that works with the MPEG standard we examined at length way back in the *Prepping the Signal* chapter. That standard is the **Digital Video Broadcasting** standard, or DVB. These two standards work together to get our information to our customers. The reason we bring this standard up here is that it has a lot to say about how our receivers find the correct transport streams.

When we first introduced the concept of elementary streams, we stated the three types were audio, video, and data. Our examples of data streams were closed captioning and baseball scores. That's not all that data streams can carry. In fact, there are some mandatory data streams that get pushed out on every single transport stream. Those data streams contain all the information the receiver needs to decrypt and tune programs, data called **Service Information**.

But before we go any further, let's clarify one point: for the purposes of this document, *a transport stream is the same thing as a signal carried by a single satellite transponder*. We studied transponders back in the *DISH Satellites* chapter, and learned that each satellite has 32 transponders, with each transponder capable of carrying up to about 10 HD channels. Now that we've studied data streams, you should realize the multiplexed transport streams that carry the audio, video, and data for each of the channels are exactly the same signal we described as being carried by a single transponder. For clarity, we'll try to only use the term *transponder* from now on.

At the time of this writing, DISH uses nearly 700 transponders to carry over 7,600 channels (over 3,500 of them local channels), with each channel having at least one associated audio and one video stream. The total number of channels changes from week to week, based on requests from programming, legal, marketing, etc.[98] If we just multiplexed them all together, even splitting them across the 700 transponders, we'd have a mishmash of information that would make it virtually impossible for a receiver to figure out which audio, video, and data needed to go together. That's where the service information comes into play. Now, if you go into the DVB and MPEG standards, you'll find both the terms *service information* and *program-specific information* used to describe different aspect of what we're about to study. However, in documents written by the DVB standards group, they're also collectively referred to as program-specific information/service information.[99] Since that's a mouthful, and since we don't really need to understand the nuanced difference at this level, we'll just refer to them collectively as *service information*.

Service information is encoded into special elementary data streams and is multiplexed into each and every transponder's signal. Those streams are also assigned special, unchanging packet identifier numbers (PIDs) so all receivers know how to identify them. It goes without saying that these streams also need to be unencrypted so a receiver can use the information at any time. The service information consists of data that fill up to 12 different kinds of tables. Some of

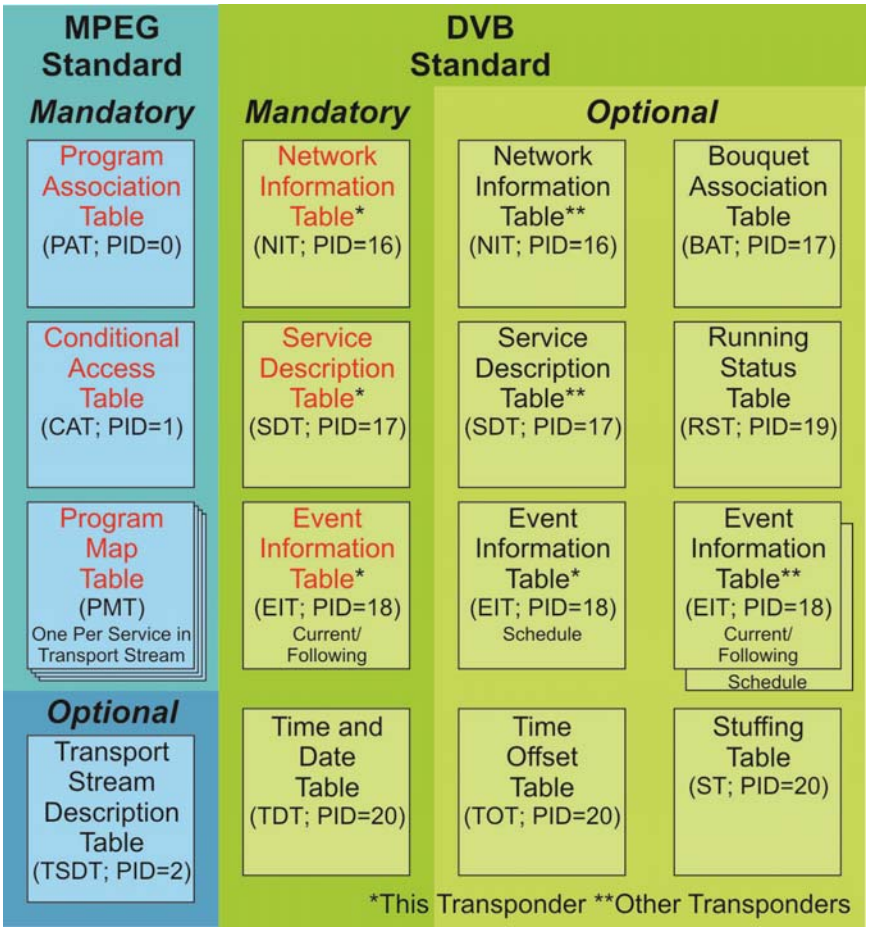

Figure 170: Service Information Tables.[100]

these table types are used more than once to hold information about the transponder the receiver is currently tuned to, other transponders that could be tuned, current information, and future information. Figure 170 shows these different tables, indicates which organization sets their standards, and highlights whether they're mandatory or optional to broadcast. In this document we'll only go into the tables highlighted in red in this figure. Notice that the DVB table names use the terms *service* and *event* instead of the friendlier terms *channel* and *program*.

The Program Association and Program Map Tables. For each transponder, the most important table is the **Program Association Table**, or PAT. The PAT is the place that tells the receiver which elementary streams on the transponder belong together. The PID of the PAT is always set to be zero. (This is going to be fun, isn't it? Keep up with your acronyms!) That way the receiver always knows where to go to figure out what information is on the currently-tuned transponder.

In Figure 171 we've created an example PAT (the top-center table) that would help the receiver sort out the multiplexed transponder information we used as an example back in Figure 169 (p. 197). In that figure we had the baseball game between the Rangers and the Rockies, a channel with several 1990s sit-coms, and a third channel with five audio, video, and data streams.

The PAT shown here sorts all of those things out. It shows that this transponder has eight services on it (remember, a service is just the way DVB refers to a channel—we discussed three of these services in our description of Figure 169, but transponders normally carry more than just three channels so we've added a few more here, the grey ones). Remember how we skipped discussing Service 2 earlier? That's because the green-coded Service 1 in Figure 169 really contained the data necessary to feed *two* channels, as indicated by the two green-colored rows in the PAT—Fox Sports Southwest with a PID of 27 and Root Sports with a PID of 73. Remember, they shared a common video feed.

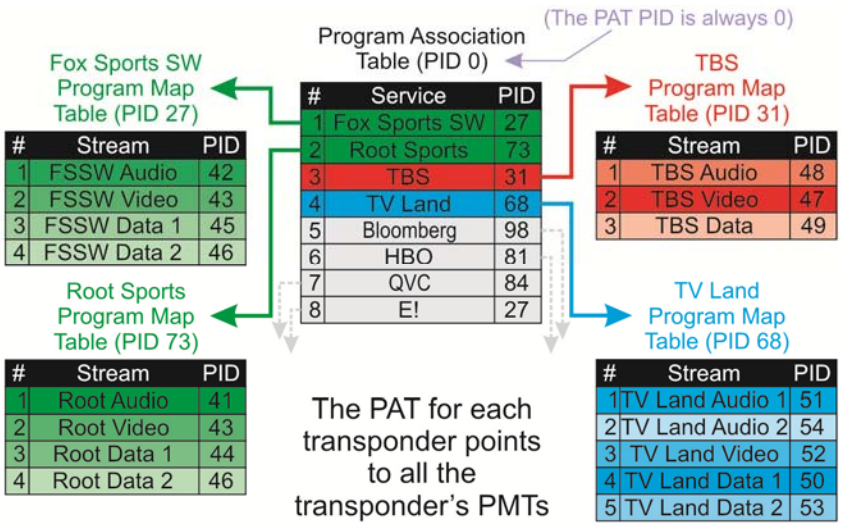

Figure 171: Example Program Association Table and Program Map Tables.

Now follow the arrow in Figure 171 to the left from Service 1. You'll come to the associated **Program Map Table** (PMT) for Fox Sports Southwest, home of the Rangers. It shows that the service consists of an audio stream, a video stream, and two data streams. Again, the streams are named in the figure to help us humans sort them out; the receiver only cares about the numbers and the PIDs. The PIDs are really important here. They tell the receiver which elementary streams to combine to put together the complete program on this channel. If you look back at Figure 169, you'll see every one of the streams associated with the Texas broadcast are referred to in this table. That makes sure we'll get the right video, audio, and closed-captioning for this broadcast as well as the scores for all the other baseball games. If the customer had wanted to watch the Rangers version of the game, the PAT and PMT would have told the receiver it needed to find the elementary streams for PIDs 42, 43, 45, and 46.

Now, look at Service 2 in the PAT. It's labeled Root Sports, which is the home of the Rockies. Following the arrow down and to the left, its PMT shows that it also has four streams, but these streams are Rockies-specific. Notice how we've saved bandwidth by using the same video stream (PID 43) and same stream for the

0	27	73	31	68	98	81	84	27
27	42	41	48	51	22	91	69	53
73	43	43	47	54	34	97	51	75
31	45	44	49	52	88	72		77
68	46	46		50		55		
98				53		40		
81						32		
84								
27								

Figure 172: The Program Association and Program Map Tables without Labels.

league-wide scores (PID 46) as we did for the Rangers broadcast. Looking back at the PAT, you can see that the red- and blue-coded services 3 and 4, TBS and TV Land, are similarly broken out.

Again, the actual tables used by the receiver don't have friendly labeling like "Fox Sports Southwest." The receiver would know from other tables that Fox was the first service in this PAT. We only put names in these tables to make them easier for human readers to understand. In fact, the actual tables could look more like the bland ones shown in Figure 172, where only the PIDs show up.[*] As far as the receiver is concerned, that table contains the same information as is in the ones in Figure 171. The sequence number is understood from the order of transmission. The PAT is blue, the PMTs we discussed at length are rose, and the PMTs we didn't discuss are grey. Notice that the top row of each table contains the PID of the table itself, with the top left number *zero* indicating that table is the PAT. Since the PAT contains the PIDs of all the PMTs (wow again—lots-o-acronyms), the left column contains the same numbers as the top row. Since humans really do need the text names for the electronic program guide, we'll show you where they actually come from in a little bit.

The Network Information Table. So that's how the receiver knows what's on the transponder it's currently tuned to. But what about what's on the other 95 transponders it can see?[†] That's where the next table

DISH Channel Number	Channel Name	Satellite	Transponder (Transport Stream)	Market	Displayed Channel
105	USA	129	9	US	105
105	USA	72.7	30	US	105
106	TVLND	119	13	US	106
106	TVLND	61.5	4	US	106
107	CMDY	110	14	US	107
107	CMDY	61.5	17	US	107
108	LIFE	110	23	US	108
108	LIFE	72.7	30	US	108
109	LMN	129	18	US	109
109	LMN	61.5	14	US	109
110	FOOD	129	9	US	110
110	FOOD	72.7	25	US	110

Figure 173: Example Network Information Table Data for National Channels.

[*] In reality, they'd be even less intuitive, since the numbers are stored in hexadecimal form.
[†] Three satellites with thirty-two transponders each.

comes into play: the **Network Information Table** or NIT. It's basically a place the receiver can go to find out which channel is on which satellite and transponder. It also contains other information such as which market areas are allowed to see which channels, and the mapping of displayed channel numbers to DISH channel numbers (we'll explain that one in a minute). Like the PAT, the NIT has a reserved PID, 16, so the receiver always knows where to find it. The *same* NIT is broadcast on *every* transponder, and is the way the receiver knows how to go from transponder to transponder.

Figure 173 shows an example of some of the information contained in the NIT. The channels in this table are all national channels, that is, they're normally uplinked from either Gilbert or Cheyenne and retransmitted through the satellite's CONUS antenna. As with the PAT and PMTs we showed above, there's really no textual information in these tables, just reference numbers. For example, the satellite at 119 might be reference number 17 and the US market might be reference number 43. The channel names don't appear at all and are only here for your ease of viewing.

A few things are worth noting in this table. There are two channel numbers listed for each channel: one for the SD version and another for the HD. In this example, the HD versions of the USA Network and the Food Network are on the same transponder (129/9), while the SD versions of USA and the Lifetime Network are on 72.7/30. That example shows how more than one channel can be on the same transponder. Also note that the displayed channel number and the DISH channel numbers are the same for each channel name, which is true for many national channels.

In contrast, Figure 174 shows a different portion of the NIT much further down the DISH channel listing. Here we see local channels, as indicated by the different markets. Notice that in every one of these examples, the displayed channel number is 9, even though the DISH channel numbers are different. That's the way we're able to have every market's TV channels show up where our customers (and the

DISH Channel Number	Channel Name	Satellite	Transponder (Transport Stream)	Market	Displayed Channel
6311	KCAL	129	29	Los Angeles	9
6316	WGN9	129	29	Chicago	9
6332	9News	129	30	Denver	9
6333	KNMD	129	5	Albuquerque	9
6350	KMSP	129	3	Minneapolis	9
6377	KMBC	129	3	Kansas City	9
6448	KQED	129	19	San Francisco	9

Figure 174: Example Network Information Table Data for Local Channels.

local broadcasters) expect them to be.

Also notice that in this example two pairs of the stations are on the same satellites and transponders (129/29 and 129/3). But way back in Figure 97 (p. 124) we showed the LA, Chicago, Minneapolis, and Kansas City markets to be served by different spot beams. How can they be on the same transponders? Well, we actually didn't tell the whole story back in the *DISH Satellites* chapter. We said that each satellite carried 32 transponders. What we really should have said was that *any spot on the ground can only see 32 transponders from any one satellite*. Some of our satellites carry many more transponders than 32, especially our spot-beam birds that carry local channels. That's what we meant when we discussed frequency re-use. We have several sets of "32 transponders" on those satellites, with each set feeding the output of only one of the spot beams. The same 32 transponder *frequencies* get reused over and over again, but don't interfere with each other since they're directed to different spots on the ground. Did the fact that we told you earlier that DISH broadcasts over 7,600 channels tickle a part of your brain that knew we only used 7 satellites to do that?[*] There's the answer for you.

Let's compare and contrast a little (our high school English teachers would be so proud!). The NIT doesn't need to link to information about elementary streams because the receiver only needs that knowledge about the transponder it's currently tuned to. The same NIT is broadcast on every transponder. The PAT *does* link to elementary stream information (through the PMTs) because a *different*, transponder-specific version is broadcast on each different transponder. Anyway, with the NIT, PAT, and PMTs stored away in memory, the receiver is ready to find channels for our customers.

The Service Description and Event Information Tables. One last thing about tables before we move on to helping our customer watch the Rockies. We told you the channel navigation tables—NIT, PAT, and PMT—only contain reference numbers, not textual names or information. How does the EPG get built with just numbers? That's the job of the Service Description and Event Information Tables.

The **Service Description Table** (SDT) links the different service PIDs (e.g., PIDs 27, 73, 31, 68 in Figure 171) with text information that would be used in the program guide, things like the name of the channel in English (Fox Sports Southwest, Root Sports, USA, TV Land). Like the PAT, the *mandatory* SDT is transponder-specific; a

[*] 7,600 divided by 7 is almost 1,100 channels per satellite. With 32 transponders, that means each transponder would have to carry almost 34 channels, way more than the 10-20 we've discussed were possible.

different version is broadcast on every transponder. However, it's optional to broadcast the information for other transponders as part of the SDT as well. DISH broadcasts this optional information so, regardless of what transponder the receiver is tuned to, we can build or update the EPG.

The **Event Information Table** (EIT) is where the EPG gets information about individual programs on the different channels. The *mandatory* part of the table links the service PIDs with information on the name of the show currently being broadcast, what the show is about, and what's on next. It's optional to also include information about the currently-tuned transponder's schedule for further in the future, as well as about current and future programs on other transponders. Again, DISH broadcasts this optional information in various locations and at various times known by our receivers so they can build the EPG from the current time out to about nine days in the future.

With that discussion of data streams and look-up tables now lodged firmly in our noggins, it's finally time we looked at the chain of events our customer started into motion with the seemingly simple act of selecting the Rockies game with her DISH remote.

Tuning and Decoding the Channel

What happens next is essentially the un-doing of everything we did at the uplink center when we prepared our signal for transport to the satellite. There, we took a television signal and compressed it using an MPEG encoder into a stream of ones and zeroes. We then added some encryption to keep thieves from helping themselves to our wares. Next we put several channels of information together using our multiplexer, adding some error correction to the packets. Finally, we coded the ones and zeroes using phase-shift keying into a form that would still be understandable by the satellite 22,000 miles above the Earth.

Figure 175 shows that inside the receiver, we'll be undoing those steps in the reverse order. Don't be worried: this figure is really busy; there's a whole lot of information here that we'll repeat in smaller chunks later. (By the way, we're assuming an HD, 8PSK, MPEG 4-coded signal in this figure; remember it could also be SD, QPSK and/or MPEG 2.) What you need to see here is that there are seven steps—the vertical black boxes—between satellite signal and stuff showing up on your TV screen. Those steps reduce the number of signals from a big amount—represented by the rainbow at the bottom left—to a single channel—represented by the thin red line at the bottom right. You also need to notice that during four of these steps we'll require input from the service information tables as shown by

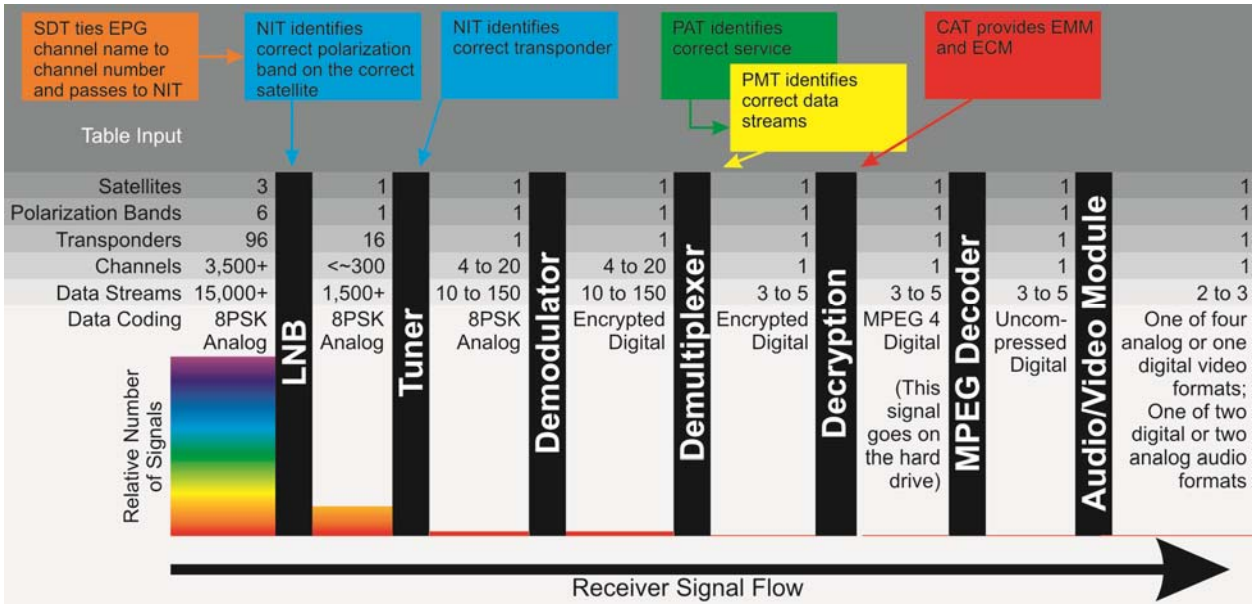

Figure 175: Overview of Signal Flow through a Receiver.

the colored boxes at the top of the figure. This figure is the overall big picture. Let's see what's involved in each of these steps.

Selecting the Right Transponder from the LNB. After what seems like an eternity of table- and data stream-related lead-up to this rather unremarkable event, the customer finally pushes the remote control button. The finely-tuned machine that is the DISH receiver springs into action. First, the receiver notes that she's selected Root Sports. It goes into the SDT and discovers that Root Sports is the same thing as Channel 414. It then goes into the NIT and finds out that Channel 414 is on transponder 7 of our satellite at 119° W (again, fictitious values we'll use for this example). From information it got during its initial installation, the receiver knows the 119 bird's signal can be found on the middle LNB (see Figure 109 on p. 140). Completely capable of dividing by two and sensing remainders, the receiver also knows transponder 7 is an odd transponder (that by definition uses right-hand circular polarization). As Sherlock Holmes would say, "Ah, Watson, I've now deduced which transponder and polarization we need to use!" Your excitement is palpable!

The receiver then sends a coded signal up the coaxial cable connecting it to the LNB. The very simple code used to communicate with LNBs is known as DiSEqC, which stands for Digital Satellite Equipment Control. The acronym is pronounced *die-seck* or *DISS-ick*, depending upon whom you ask. You don't need to know more than that, and we only bring it up because you might hear the word used

over in engineering. Anyway, the DiSEqC signal tells the LNB to send the RHCP signal from the middle LNB down the cable.

So, that sounded pretty easy. What did it do for us? Nothing, except filter out 5/6 of the signals we send into space![*] That's the single largest filtering event the receiver does, and it seemed really simple, didn't it. Take a look at Figure 176. It was shamelessly cut from the previous figure to highlight what the LNB has just accomplished. At the left, before the LNB did its job, we were looking at signals from 3 satellites potentially giving us access to upwards of 3,500 channels. Once the receiver selected an LNB and a polarization, we're now down to a maximum of 16 transponders and perhaps 300 channels. Notice that the LNB did nothing to the signal's form, though. It came in as an 8PSK wave in space and leaves as an 8PSK oscillating signal in a coaxial cable. Also, notice that the job couldn't' have been done without access to two of the tables we so painstakingly described earlier. Step one of the receiver's job is now done.

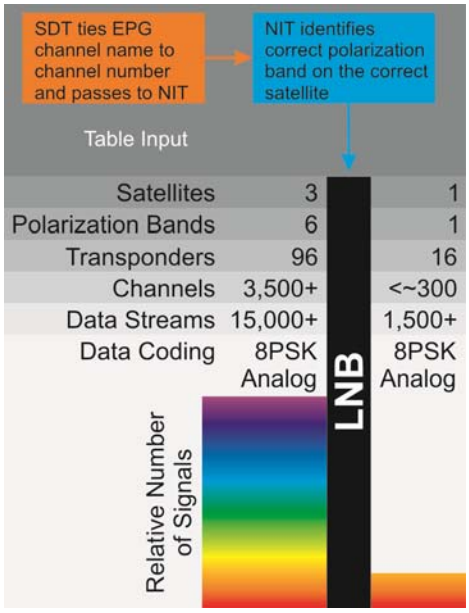

Figure 176: Signal Flow through the LNB.

Selecting the Right Channel from the Transponder: The Tuner. So we now have 16 transponders to choose from. But, thanks to the NIT information we already know we're looking for transponder 7. That makes things pretty simple for us. All the tuner has to do now is to filter out all the signals from the other 15 transponders. However, there's a little more to it than that—you've read our stuff for long enough now to know that's almost always the case when we preface something with the qualifier *pretty simple*.

The signal that hits the receiver from the LNB is in one of the three LNB bands we discussed in the last chapter (see Figure 144 on p. 174). Our tuner can only handle inputs in the 0.95 to 2.15 GHz bands, which works for all of our receivers except the Hopper. Remember, the Hopper can use the extra "node up-convert"

[*] DISH uses seven satellites and they're all visible from just about everywhere in the CONUS, meaning the entire panoply of 7,000+ channels we send really *is* arriving at the customer's home—even the spot beams directed to other parts of the country are there, but they're very, very weak. However, to be technically correct, the directivity of the dish filters out four of the seven satellites. That means only about half of the 7,000 channels and 192 transponders were actually available to the LNB.

band at 2.5 to 3 GHz for its third tuner. If the lower two bands are already being used and the LNB sends the signal on the node up-convert band, we'll have to shift it back down to one of the other bands in order for the tuner to be able to use it. You've read about frequency shifting enough by now to know that to do that we'll just have to mix it with a local oscillator of the appropriate frequency to get it back into the proper range. We've gone over mixing enough that we'll actually spare you the details this time.

Now that we're guaranteed to have a signal that's within the 0.95 to 2.15 GHz specifications for the tuner, the receiver electronically selects the proper filter for the transponder. We've discussed filters before, using the visual examples of rose-colored glasses and green cellophane back in the *DISH Satellites* chapter (see Figure 92, p. 118 for a review). Recall that electronic filters do the same thing as the cellophane, but in this case only allow a certain subset of our LNB band to pass through. Figure 177 illustrates this concept. The figure should look somewhat familiar, as it shows the frequency locations of each of our 32 transponders within the 0.5 GHz-wide spectral band we use. The difference between this presentation and previous ones is that, as you can see from the dual frequency scales along the horizontal axis, we're showing the frequency range after it's been shifted into the bands the LNB uses to send the signal down the coaxial cable, 0.95 to 1.45 and 1.65 to 2.15 GHz. Since any transponder can be sent on either the upper or lower LNB band, both frequency ranges are shown. Also new in this presentation, we've shown the transponders arranged as they're sent to the satellite (normal) and as they'd show up if the band had been spectrally inverted by the LNB (see p. 162 for a review, if necessary). Note that only *one* of these four rows would ever be presented to the tuner at any one time.

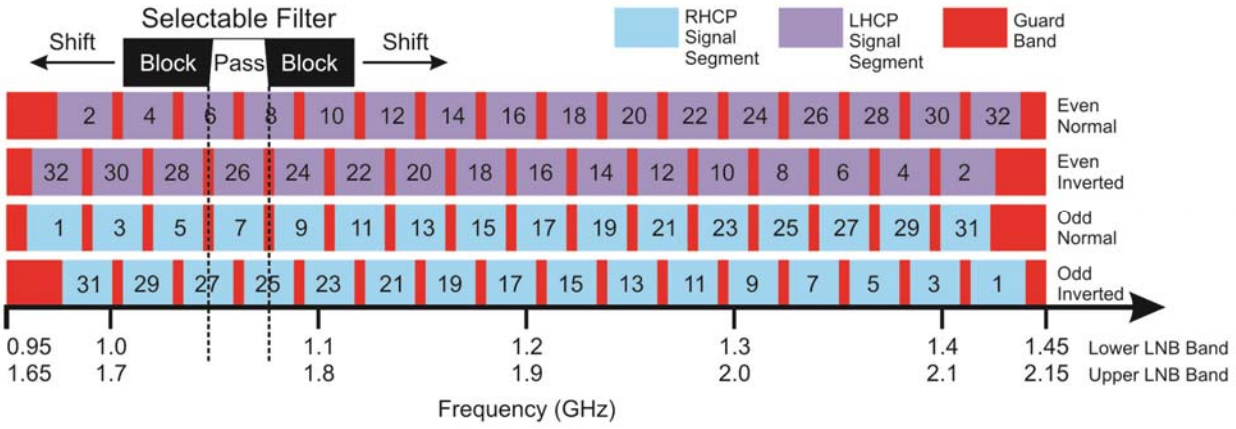

Figure 177: Filters Select the Appropriate Transponder.

At the top left, we've drawn a representative filter in black and white. And, no, it's not describing a football practice drill (shift-block-pass-block-shift). What it's trying to show is that the tuner knows where each of the transponder bands is located and can shift (or *tune*) the *pass* region of the filter to let just one band through, blocking all other frequencies. In this example, the filter is centered near 1.06 GHz (lower LNB band) or 1.76 GHz (upper LNB band)—as indicated by the vertical dotted lines—where it would let either odd transponder 7 or even, inverted transponder 26 through, depending on which band the LNB sent down. This shifting of the filter to the right and left in frequency through the allowed frequency band is the essence of what the tuner does.

From the NIT, the receiver knows that Root Sports is on channel 414, which is on 119/7. It also knows that (as we will stipulate for this example) the LNB sent the odd normal transponders directly to the tuner on the lower LNB band. That means the tuner needs to set the filter to allow frequencies between 1.0484 and 1.07456 GHz to pass, blocking all others (Figure 94, p. 121, showed the center frequencies of the downlinked bands; the 1.0484 and 1.07456 values are the same ones from that figure, but mixed and down-converted a few times). Bottom line: only transponder 7 makes it past the tuner's gate.

What does that mean for us in terms of the signal flow through the receiver? From Figure 178 you can see that we've gone from 16 transponders to one. Instead of around 300, we're now only dealing with between about 4 and 20 channels, depending upon how they're coded (HD vs. SD, 8PSK vs. QPSK, MPEG 4 vs. MPEG 2, etc.), with each of those channels having varying numbers of elementary streams associated with them. However, the tuner doesn't give a hoot about channels or streams. It only cares about tuning transponders, which it's now sorted out for us.

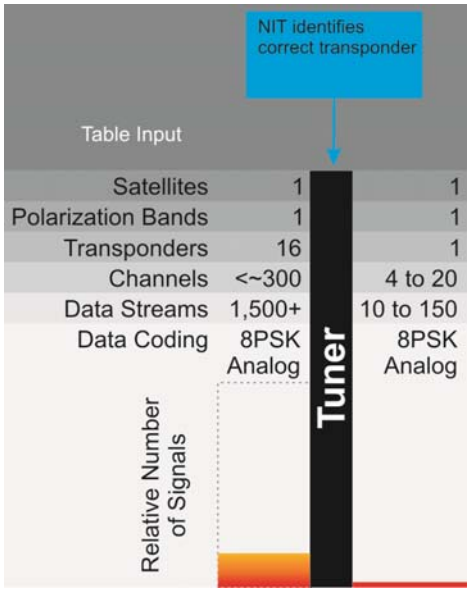

Figure 178: Signal Flow through the Tuner.

Converting Wiggles to Bits: The Demodulator. So now we've got the signal for the single transponder carrying the Rockies game inside our receiver. Remember that we said we'd use the word *transponder* but that it meant the same thing as a *transport stream*? It's now time to start talking about transport

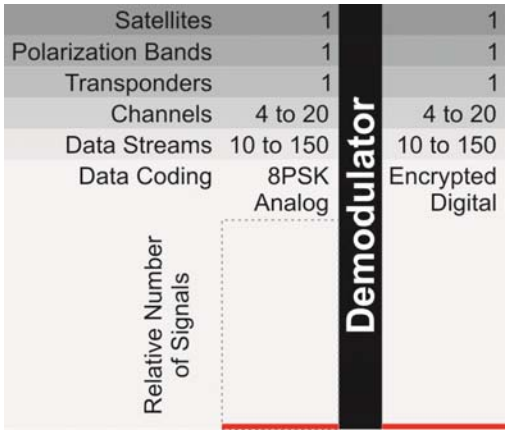

Figure 179: Signal Flow through the Demodulator.

in a defined way to encode a bunch of ones and zeroes onto a wiggling wave. When we demodulate the wave, the point is to recover those ones and zeroes. The concept is shown in Figure 180. But there's more to it than that.

streams again, since that makes more sense in the context of the demodulator. As shown in Figure 179, the demodulator is going to input the multiplexed, 8PSK transport stream for all channels on the stream and output the digital packetized elementary streams for those channels.

To *modulate* means to vary one of the attributes of a wave. We modulated our wave at the uplink center when we transformed it into the 8PSK coding (see p. 69 for a review). There, we varied the phase

Figure 180: The Demodulator First Takes the Input Wave and Converts it to a Digital Transport Stream.

Remember, the transport stream signal that comes in to the demodulator consists of a carrier wave at the center frequency of the transponder band and the actual information being carried, located in sidebands that take up the rest of the transponder frequency band. Frame I of Figure 181 is a review of that carrier/sideband concept, presented earlier in the *Spectral Bandwidth* section of *The Uplink Center* chapter (p. 81). It shows the first five odd transponders with

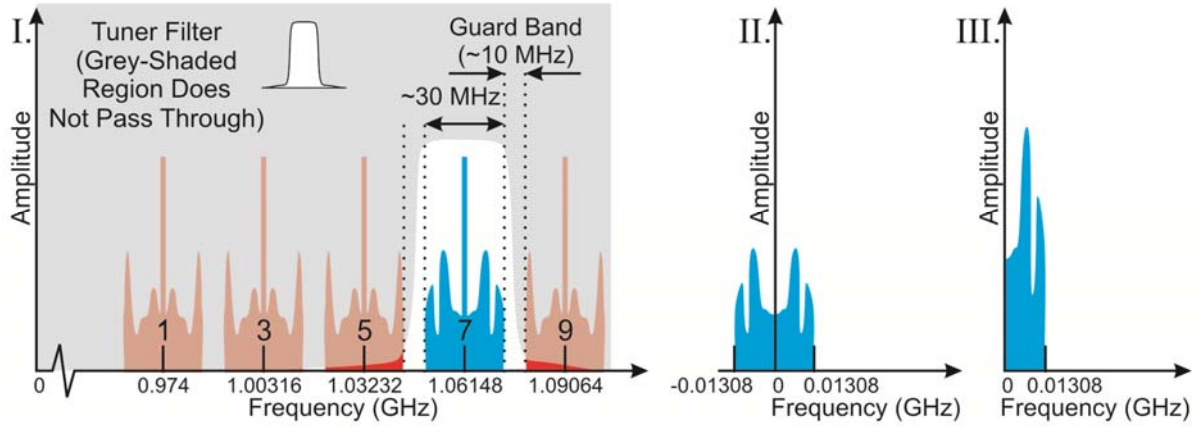

Figure 181: The Demodulator Moves the Sidebands to the Baseband Range.

their coax-shifted center frequencies. It also shows how the tuner's selectable filter has screened out all but transponder 7, the transport stream we're interested in, as indicated by the non-grey-shaded region. Notice each of the transponders has a strong central spike which is the carrier wave. The carriers are surrounded by symmetrical sidebands that contain our program content.

What the demodulator does is to first take transponder 7's band and mix it with a local oscillator working at the transponder's center frequency. It then discards the sum frequencies and only keeps the difference frequencies. We've discussed mixers several times (for a review, see the *Spectral Bandwidth* section starting on p. 81), so you should be able to see that if you subtract the center frequency from the center frequency, you get a frequency of zero. That means that the sideband that is to the right of the center frequency will be mixed down to positive frequencies, while the sideband to the left will be mixed down to negative frequencies. Frame II of the figure shows this situation. Also recall (see Figure 129, p. 162, if you don't recall) that negative frequencies just get folded back over into the positive section of the frequency spectrum, so they end up adding their amplitudes to the signal,[101] as shown in Frame III, where the signal has the same shape but is twice as high as the right sideband in Frame II.

Our signal is now only 13 MHz (0.013 GHz) wide. If you do a little thinking you should be able to see that as long as the local oscillator in the demodulator is set to work at the center frequency of the transponder we're interested in, whether that transponder is 1, 32, or anywhere in between, this mixing will always result in the signal looking like it does in Frame III, with a frequency range between 0 and 13 MHz. This location, where the signal runs from a frequency of zero to its maximum modulation frequency, is called the *baseband.* The demodulator then takes this carrier-less baseband analog signal and looks at the phase shifts of the modulation to reconstruct the digital transport stream. We show Figure 182, originally presented as Figure 60 (p. 78), to remind you of how that's done for differential phase shift keying.

Separating Wheat from Chaff: The Demultiplexer. Wow. After 44,000 miles and many, many pages of text, our signal is finally back to being a digital stream! However, it's still a mish-mash of several channels of information. Our customer only wants the, ahem, bits that relate to the Rockies game. Enter the demultiplexer. We first discussed demuliplexers on p. 35 in the *Prepping the Signal* chapter. There, we saw that demuliplexers basically undid everything the multiplexer worked so hard to do.

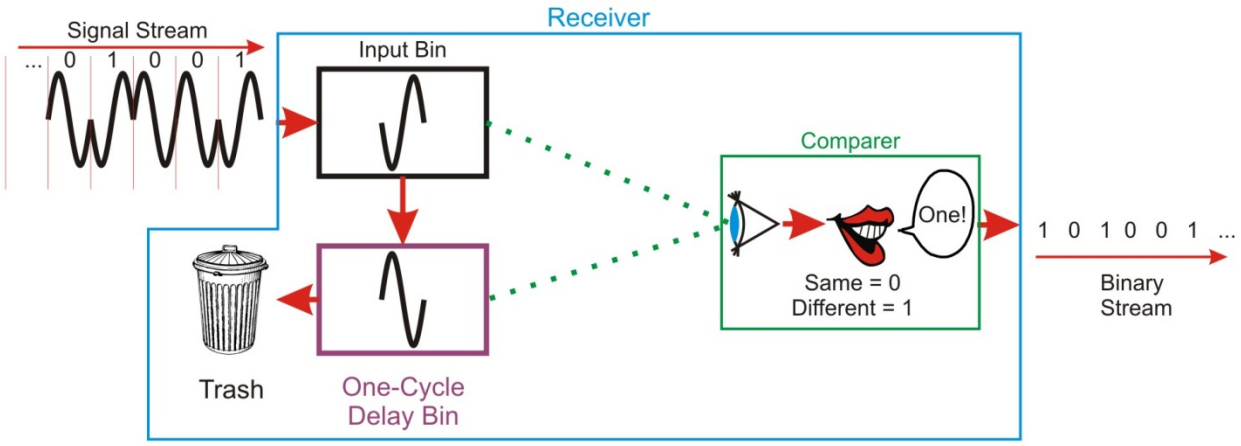

Figure 182: Differential Phase Shift Keying Revisited.

We revisited multiplexers in the first section of this chapter. As you recall from that discussion, the transport packets all had headers that identified which stream each packet belonged to. The packets also had sections devoted to error correction, which we didn't describe in that section. We won't describe them here, either, except to say one of the most important things done to the stream at this point is to make sure the information we've recovered is the same as the information we sent. Error correction enables this critical step.

Anyway, after the errors have been corrected the receiver digs into its memory and figures out what elementary streams belong to Root Sports. Up until now, it only needed the NIT to tell it which transponder to care about. Now the receiver turns to the PAT and the PMT to figure out, as shown in Figure 171 (p. 203) it only needs to keep packets from PIDs 41, 43, 44, and 46 (we told you to brush up on your acronyms, didn't we?). Figure 183 is a sort of mirror image of Figure 169 (p. 197), where we described how the channels were multiplexed. The transport stream we multiplexed there gets demultiplexed in this mirror image. However, based on the PMT input, the figure shows that most of the streams end up on the trash heap. We're now only left with the four streams that matter to watching the Rockies game on channel 414, as shown in Figure 184 where only one channel and a few data streams are left. By the way, we show what the streams eventually become in the blue oval at the right of Figure 183 just so it's more clear what the streams were, but realize that process doesn't really happen just yet.

We're almost done with the demultiplexing step. Remember that there was a difference between transport packets and elementary stream packets? The transport packets were smaller and had a lot of error correction information in them. They were made up of bits and pieces of broken up elementary stream

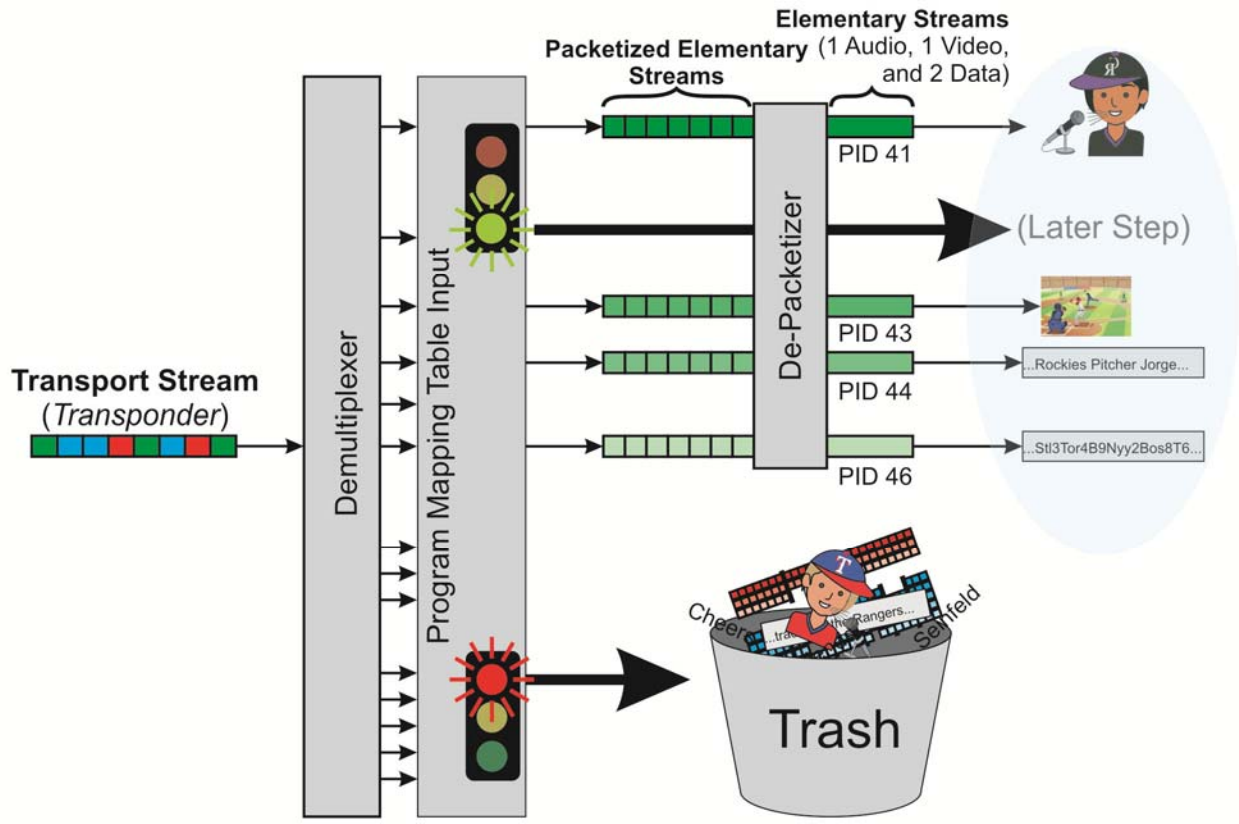

Figure 183: An Informed Demultiplexing Process Discards Unnecessary Data Streams.

packets. What the demultiplexer pushes out are these elementary stream packets, reassembled and put into the correct order. Elementary stream packets, however, still have a lot of extraneous header information that makes them unusable by an MPEG decoder. The last thing our demultiplexing process does is to strip off these headers, combining the packets to leave only elementary streams of encrypted MPEG code. And with that, we're ready to move on to our next step within the receiver.

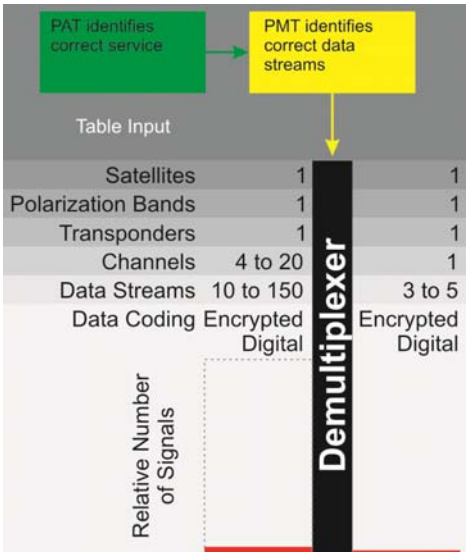

Figure 184: Signal Flow through the Demultiplexer Process.

Smart Cards and Decryption [102]

We talked a little bit about encryption back in the *Prepping the Signal* chapter. Now it's time to see the other side of that coin, decryption, however we'll also revisit the encryption process in much more detail here as well. Earlier, we talked about the two messages that made sure our customers got

to see the programming they bought: the Entitlement Control Message (ECM) and the Entitlement Management Message (EMM). We didn't say it then, but these messages are a part of the DVB standard, and are contained in the last mandatory table we need to discuss (Figure 170, p. 202), the **Conditional Access Table** (CAT). The PID of the CAT is always one, and, like the PAT and NIT, most of it is always transmitted unencrypted so it can be used by the receiver. The CAT contains the PIDs of all streams associated with Entitlement Management Messages (EMM) and Entitlement Control Messages (ECM).

As we mentioned before, we won't go into a lot of detail about the specific encryption/decryption processes we use at DISH because they're closely held. Were too many details divulged, unscrupulous people could use them to crack our security system to get free content from us. However, we can go into more detail here than we did in the *Prepping the Signal* chapter because you've now learned a lot more about packets and streams. The information presented below comes from open sources available on the Internet, so we're not disclosing anything here that isn't already widely known.

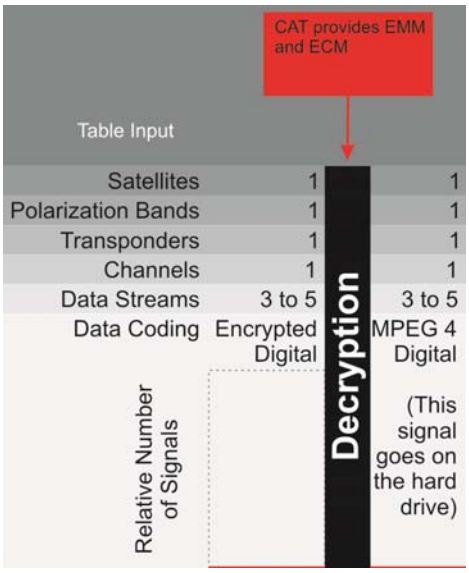

Figure 185: Signal Flow through the Decryption Process.

You can see in Figure 185 that the only thing the decryption step will do is to remove the encryption from our signal. It doesn't change the number of streams or types of data at all, other than to make them readable by the MPEG decoder. Not everything that we transmit is encrypted. We've already mentioned that the DVB tables are sent in the clear so the receivers can find and use them without having access to the current CAT information. We also don't encrypt the headers for the transport stream or for the packetized elementary streams. Doing so would make it very difficult to reassemble the packets in a timely manner. However, DISH does encrypt the payloads of every single content stream.

Now, as we were researching encryption for this document we discovered that it's a rather Byzantine process; it took us a more than a few times through our sources to get our minds around it. Much of this confusion was because no single source really did a good job of showing us all the pieces. We believe we've put it

all together, though, and that the two figures and three key concepts you'll soon see will be very helpful for you to visualize the process. We encourage you to read all the way through the next few pages to get a general view of what information is here, and then return to this point to have another go. That second time through, after you've gotten the big picture, should help you figure out what the details really mean. The first thing we'll do is go over some definitions and relationships between different parts of the encryption system.

Let's review the EMM and ECM briefly so we can see how they're used for decryption. The ECM contains an encrypted list of **control words**, a different one for every single elementary stream. Control words are essentially the same thing as the password that lets you log on to your computer; they let the receiver "log on" to a specific elementary stream. Each elementary stream is encoded using a control words, and the same control word is required to decode its paired stream. The ECM is transmitted about once every two seconds on every transponder. The control words are created by a pseudo random number generator and change fairly frequently, about every 20 seconds or so—and you thought changing your computer password every few months was onerous! We don't send the ECM in the clear, though. It's encrypted with a cipher code—another password—called the **service key**.

The EMM contains an encrypted string of bits that hold information about what each and every receiver should be authorized to view, essentially information on which channels every single customer has paid for. We have millions of customers, and the complete EMM set contains information about every single one of them. Also, since at any one time our customer service agents are changing service levels for hundreds of customers based on non-payment, customers calling to add/drop premium channels, or system-wide changes to package content, you should be able to understand that the EMM needs to be updated quite frequently. In addition to customer information, the EMM also contains an encrypted version of the network-wide *service key* discussed above. The EMM is encoded with another password called the **user key**.

The user key is stored permanently on the smart card in each individual receiver. Since every user key is different, there's a different section of the EMM for every one of the receivers in our millions of customers' homes, and that EMM section unlocks one and only one receiver. We broadcast every section of the EMM a few times each minute in most cases, more often when a service change has been made so that the receiver is more likely to receive and act upon it. Every receiver

sees the complete EMM, but only pays attention to the section that's encoded with its unique user key. It's important to note that even though the EMM is encrypted *using* the user key, the user key is never *transmitted* to the receiver but instead is resident on the smart card. Information about every one of the user keys is held in the DISH subscriber authorization system so we can properly encrypt the EMM.

Thus, to decrypt any specific channel, the receiver needs to have 1) the user key (which it will have if its smart card is inserted in the receiver); 2) the service key (which it gets via the EMM and decrypts with the user key), and 3) the control words for every stream in the service (which it gets via the ECM and decrypts with the service key).

Up to now we've talked mostly about definitions and pieces of the encryption system. Let's now take a look at how the encryption *process* works. We'll move on to the decryption process after we're done with this part. Figure 186 shows the encryption process in block-diagram form. It looks pretty busy, but we'll go over it piece by piece.

The legend is at the top of the figure, and it shows that in this figure red indicates portions dealing with the user key, blue the service key, and green the control words. Encrypted streams are shown as dotted lines while unencrypted streams are solid lines. The black-filled boxes are the starting and ending points of the flow, pieces not themselves involved with conditional access. In this case, the flow starts with an unencrypted MPEG stream coming from the encoder and ends when the stream leaves the

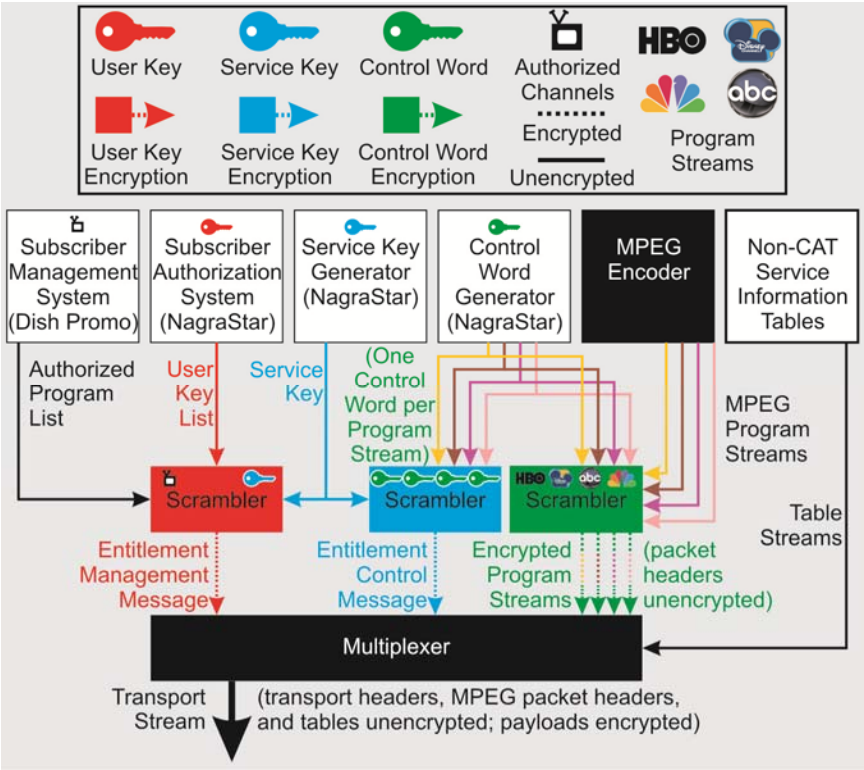

Figure 186: The Conditional Access Encryption Process.

multiplexer on its way to the satellite. But what happens in between those two steps?

Let's start at the top left white box. It represents the DISH Promo, where information about the programming our customers have paid for resides. DISH promo regularly sends updates about each customer's authorized programming to DISH encryption central at NagraStar in Denver. Included in that information are the serial numbers of every receiver the customer has and the serial numbers of the smart cards paired with those receivers. Smart cards will only work in their paired receivers; that's just one additional way we try to prevent unauthorized viewing of DISH programming.

As part of the subscriber authorization system, the second white box in the figure, NagraStar has a database of the user keys that should be on each smart card. For security, they never transmit these numbers but instead use them to generate the EMM. Also at NagraStar is a service key generator, the third white box. The service key is a password that changes every few months. A scrambling circuit takes the *authorized programming list* for a specific receiver, combines it with the system-wide *service key*, and encrypts them using the smart-card-specific *user key* it knows a customer's receiver has. That process is shown by the arrows pointing into the red scrambler box. Coming out of that box as a result of those inputs is the EMM section specific to that individual receiver. The contents of the EMM are shown by the black TV and blue key symbols inside the red box. A different EMM section is then generated for every other receiver in the DISH system, and the entire EMM is broadcast as part of the conditional access table (CAT) available on every single DISH transponder. Key Concept 1: *the EMM holds the service key but is encrypted using the user key*. Told you it was a *key* concept!

The next white box represents the control word generator. Again, the control words are the passwords guarding each individual elementary stream that goes out over a DISH satellite. NagraStar generates a different control word for every data stream about once every 20 seconds. The first place these *control words* go is to another scrambling circuit where they're encrypted using the *service key* as a password. This encrypted set of control words is transmitted to every receiver as another part of the CAT, the ECM. In our figure, we've indicated four control words by showing four differently-colored lines going into the blue scrambler box. The contents of the ECM, the control words, are indicated by the four green keys in the blue box. Key Concept 2: *the ECM holds the control words but is encrypted using the service key*.

As we mentioned above, the control words are used a second time. They're mixed with their paired MPEG stream in yet another scrambler, where the program information is encrypted using the appropriate control word. We've shown four network logos to represent the MPEG program streams that are encrypted by the four different control words in the green scrambler box. Every data stream is thus encrypted differently, as indicated by the same four colors being used for both control words and MPEG streams flowing into the green scrambler box. That means that to watch a program, you'll have to have access to not only the encrypted stream but also to the appropriate control word so you can unlock that stream. We'll see how that happens in just a few paragraphs. Notice in the figure that it explicitly mentions that only the program content is encrypted. The headers for each of the packets are not encrypted to assist in rapid reassembly of the stream once it's decrypted. Key Concept 3: *each elementary stream is encrypted using a separate control word*.

Finally, as we've discussed earlier, some data is transmitted unencrypted so our receivers can find and use the information to construct the EPG and so they can find channels and their related streams on the various transponders and satellites DISH uses. This unencrypted data is then multiplexed with the encrypted program streams, the ECM, and the EMM and sent out as a single transport stream from the uplink center to DISH satellites and back down to our receivers. That part of the process is represented by the last white box on the right and the arrows going into and out of the black multiplexer box.

Figure 187 shows what happens once the stream gets past the demultiplexer step in the receiver and before it gets to the MPEG decoder, the two black boxes in the figure that indicate the beginning and end of the decryption flow. We've finally gotten around to the decryption step in the receiver! As indicated by the arrows coming out of its box, the demultiplexer separates out not only all of the encrypted MPEG streams but the unencrypted service information table streams, the encrypted ECM, and the encrypted EMM. The table information is stored in the receiver's memory for later use, as shown at the right of the figure.

Now let's walk through the decryption process. The still-encrypted EMM is passed to the smart card, as shown at the left of the figure. The user key, already hard-coded on the smart card, is used to decrypt the service key and knowledge of which programs the receiver is authorized to decrypt (remember Key Concept 1). That information is shown with the red lines coming out of the EMM descrambler box.

Still on the smart card, the control word descrambler takes in the ECM and decodes it using the service key it gets from the EMM (Key Concept 2 returns). All of the control words are known at this point, but we don't want to let them off the smart card just yet. The smart card now uses its knowledge of which streams are authorized for the receiver—again, information it got from the EMM—and *only* releases the control words associated with those streams. In our example, this customer has only paid for two of the four channels we're showing in the figure. That means that only two of the four colored lines representing control words that we generated in the previous figure will exit the control word descrambler and leave the smart card.

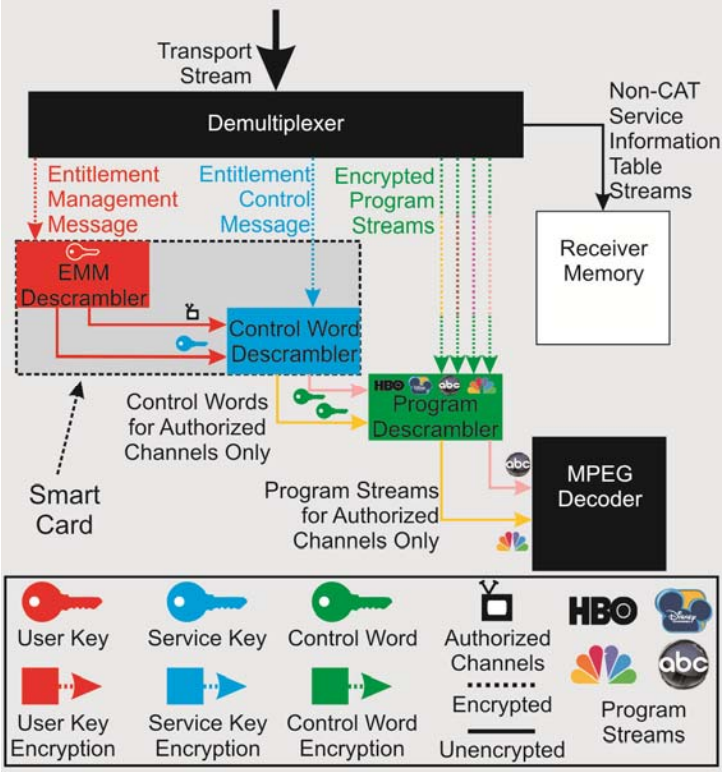

Figure 187: The Conditional Access Decryption Process.

The appropriate, authorized control words are then sent to the decryption chip on the receiver where they're used individually to decode their paired MPEG streams (Key Concept 3). This process shows that of the four inbound MPEG streams, only the two with colors matching the released control words exit the program descrambler where they're directed to the MPEG decoder.

We've mentioned the smart card above, but really didn't talk much about what it was and how it worked. Figure 188 shows a DISH smart card. It looks remarkably like the credit cards most of us carry in our wallets. However, notice the gold region in the inset. That's where the magic of the card starts.

The metallic region is actually a very thin layer of gold used as an electrical contact between the smart card reader in the receiver and an integrated circuit embedded in the card. The areas separated by the black lines on the gold region are individual contacts, and are sized so that there can be some slop in how the

Figure 188: A DISH Smart Card.

card is inserted while still allowing it to work. Each individual contact is connected to a separate input on the integrated circuit by a small wire. The chip on the card is where the user key is stored, and where the processing for EMM and ECM decryption occurs. Obviously, how the circuitry on this chip works is a closely-guarded secret.

At this point, we believe your understanding of what is admittedly a complex encryption/decryption process would be best served were you to go back to the start of the encryption section and reread it, using your newfound big-picture perspective to put the fine details into place. However, keep in mind that what we've described above is a just a generic view of how the DVB encryption/decryption process works. It is close to what we use at DISH, but, as we've explained before, the exact workings are a secret. However, with this description we think you'll be very capable of understanding the importance of the EMM, ECM, control words, keys, and smart cards in helping to keep our signal secure, which contributes significantly to our company's profitability.

From Code to Frames: The MPEG Decoder. MPEG is another horse we've already beaten to death. If you don't believe us, take a penalty lap through pp. 13 to 31. There, we thoroughly discussed how the different frames are encoded, a few using intra-frame techniques (I-frames), and many more using forward prediction (P-frames) and bi-directional prediction (B-frames.) Sound familiar? However, as we also discussed, those frames are almost never sent in the right order. So the job of the MPEG decoder is to take the frames, put them in the right order, and convert the mathematical shortcuts to pixel coloration and brightness values that the TV will understand. The decoder also decompresses and synchronizes the audio with the appropriate video frames. It really doesn't

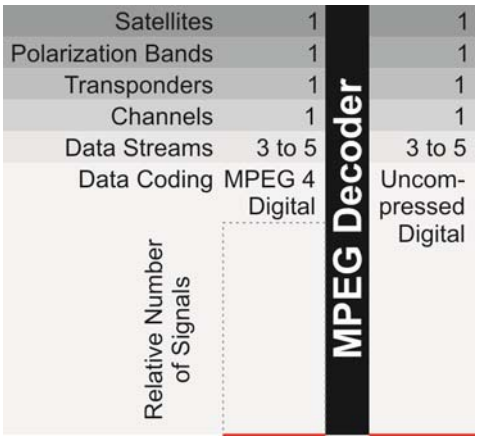

Figure 189: Signal Flow through the MPEG Decoder.

do much else to the number of streams, as indicated in Figure 189.

Finally--To the Television Set! Can you believe we're finally here? Over 200 pages of newfound knowledge and we're at last to the point where that signal we first met as it came in over satellite, Internet, or airwaves is ready to delight our customers. It's now a single audio and a single video channel, perhaps with some accompanying data, with the video consisting of an uncompressed digital stream in YCbCr format just like we discussed back in the *Prepping the Signal* chapter (p. 16). But as you can see in Figure 190, there are a number of ways we can now get those signals to the television. Take a look at Figure 191, an image of the rear panel of the DISH 622 receiver. There are no less than 20 connectors on the back of this receiver, and the majority of them are there to get signal to televisions. Why so many? Because different TVs have different ways that they can accept a signal, and we have to be able to output to every one of them in common use. These different connectors deliver varying degrees of sound and picture quality.

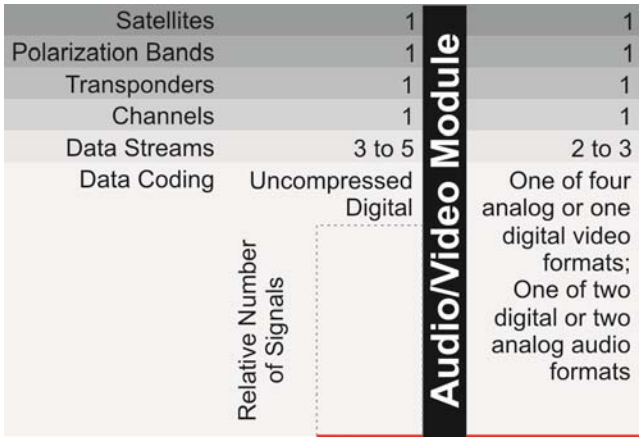

Figure 190: Signal Flow through the AV Module.

We've already looked at coax cable in detail (starting on p. 178) and hopefully you understand its strengths and weaknesses pretty well. The four left coax inputs on this box are to get signal *in*—one from an over-the-air antenna, one from the antenna for the remote control, and two from the LNB. The one on the right sends audio, video, and data signals *out* via the home distribution band on coax cable (p. 177).

The connectors on the right labeled *other* don't deal with TV signal at all, and we

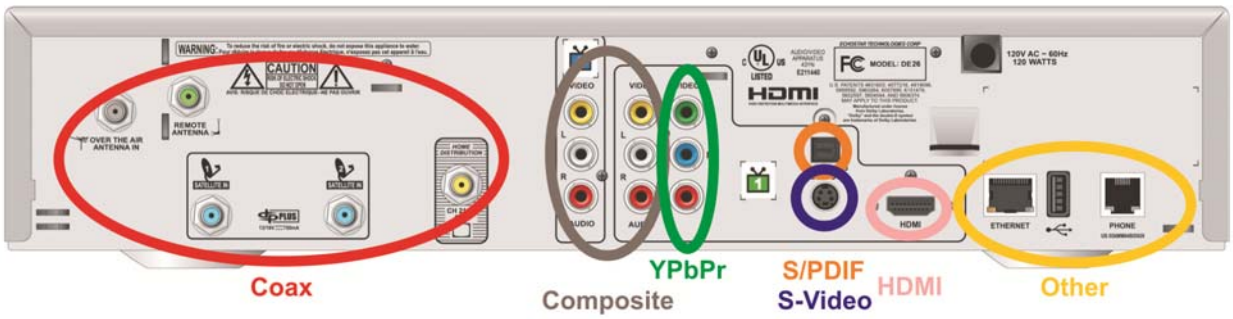

Figure 191: A Receiver Back Panel.

won't discuss them here. The ones in the middle—labeled *composite*, *YPbPr*,* *S/PDIF*, *S-Video*, and *HDMI*—all do get pictures and sounds to televisions, but with varying levels of quality. We'll go over them in order of best picture/sound quality to worst.[103]

Digital vs. Analog Signals. But before we do that, we've neglected a very important part of your education on signals and we intend to correct that omission right now. We've talked a lot in this document about digital signals, but we haven't really defined what we mean by digital and why they're better for information transmission than analog signals. Let's take a look at an analog signal—the kind that nature produces—and see how we turn it into a digital series of ones and zeroes.

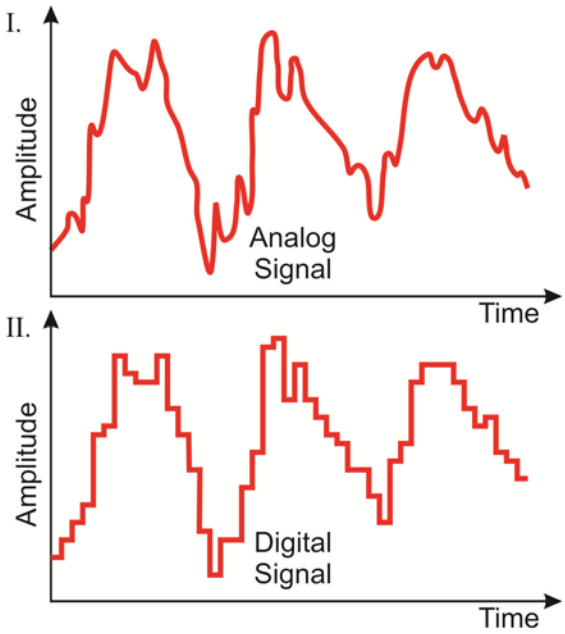

Figure 192: Analog and Digital Signals.

An **analog signal** is one that varies continuously with time. Nature abhors straight lines and sharp corners. Those things almost never happen naturally. For example, the sounds we send to our customers as audio tracks originate as pressure waves in the air that hit the microphones we use to record them. The pressure waves might vary as shown in Frame I of Figure 192. Notice that signal's amplitude varies smoothly, with every single instant in time having a distinct amplitude value associate with it, and that the signal's curve is smooth and curved, with no instantaneous shifts in value.

A **digital signal**, in contrast, is a signal that is only allowed to have one of a pre-determined number of distinct, separated values. Frame II shows a digital signal that resembles the analog one shown in Frame I. Notice that it has a lot of horizontal lines, representing a signal staying at the same amplitude for a set period, and then the line makes two square corners before settling on a new set amplitude for another period of time. Also notice that the vertical jumps between levels are all multiples of the same minimum jump. In the same vein,

* The two *P*s are not typos…this format is related to the YCbCr you've seen before but not quite the same. We'll discuss the differences in a little bit.

the horizontal distances between jumps are all the same. (A sharp eye might see a couple of places in the figure where those distance look a bit longer—that's just because the signal had the same level for several time periods.) Those characteristics—sharp corners between multiples of some minimum vertical and horizontal distances are common to all digital signals.

So why would we want to use a digital signal instead of an analog one for our broadcasts? First, we can use a lot less bandwidth to send a digital signal. Imagine we call the minimum allowable jump "1." That means the only allowable levels are 1, 2, 3… Our computers only have to be smart enough to know about whole numbers like that. There are also a set number of countable points we have to know about with the digital signal. For example, we need to know that at time 0, the signal has an amplitude of, say, five. We wait one time period and the amplitude now changes to a value of seven. After another time period, the amplitude changes to nine. In contrast, a computer dealing with the analog signal would have to know the value of the signal at *every possible* time and *every possible* amplitude. That's actually impossible for a computer to deal with, since there are an infinite number of times and amplitudes and there's not enough time in the universe for it to do that many calculations.

And calculations are the key to what we've done with our signal. Remember when we discussed MPEG compression? The brightness and color values for each pixel were given specific values so we could compare them on our way to compressing them (p. 16). We *calculated* intra-frame comparisons when we did run-length compression (p. 19). We *calculated* inter-frame comparisons when we tried to find similarities between frames that we could reduce to math instead of sending the same data more than once (p. 20). All that compression was doable because our computers were doing calculations, calculations that were only possible because we narrowed down the number of points from infinity to a countable number—we converted the real, natural analog signal to an unnatural but computationally-friendly digital one.

There are other benefits to using a digital signal. For one example, we can detect and correct errors in a digital signal when it's received thousands of miles from where it was sent. For error checking, we might insert a number every tenth number (or 20^{th}, or 100^{th}…the actual value is arbitrary) that tells what the sum of the previous nine numbers should have been. Such information is known as a

checksum.[*] The computer knows this code, and dutifully sums up every nine numbers and compares that sum with the next number, the checksum. If the sum and checksum aren't the same, we know there was an error in transmission. It could have been that one of our amplitudes was off, but it could also be that the checksum was transmitted incorrectly. What we do with this information and how we correct errors is an involved topic we won't discuss here, but it's worth being aware that error detection/correction is possible with a digital signal. It's *not* possible with an analog signal.

We hope you can see that there's a lot of goodness to using digital signals. However, if we're going from a signal with an infinite number of points to one with a number we can do math on, we've got to lose something, don't we? Yup. However, as we also discussed when we talked about MPEG compression, we can work hard to make sure the things we lose are things our eyes, ears, and brains won't notice anyway. We use physiology and psychology to determine the minimum number of points we need to keep that will allow a fantastic viewer experience while still remaining within the calculating power of our computers.

The way we transform an analog signal into a digital one is through the process of **digital sampling**. We have a computer with a clock in it and every time the clock ticks, we have a sensor that samples what the value of the thing we're measuring is—it might be the brightness of the red portion of a picture coming into a video camera or the loudness of the sound hitting a microphone. Whatever the natural analog signal is, our computer dutifully says "just saw a clock tick, time to take a reading."

That takes care of the minimum *horizontal* distance, the regular time step, but what about the minimum *vertical* distance? Ah, that one's easy, too. Remember, the measurer in the computer only knows about whole numbers (1, 2, 3…). It sees a reading from the real world where the values can take on an infinite range but it says, "Hmm, that value is larger than this one I know about, 42, but not larger than the next one, 43. I'll call it….42." In that way, we reduce the analog signal to a series of points that have the form [whole number time, whole number amplitude].

Figure 193 illustrates this concept. Look at Frame I. The red curve is the same analog signal we saw in the previous figure. Our goal is to make a digital copy that's so close our psychological and/or physiological responses won't be able to

[*] If you jump ahead to the next figure, the checksum of the blue signal calculated in the every-tenth-number way would be 163 (5+21+29+…+20).

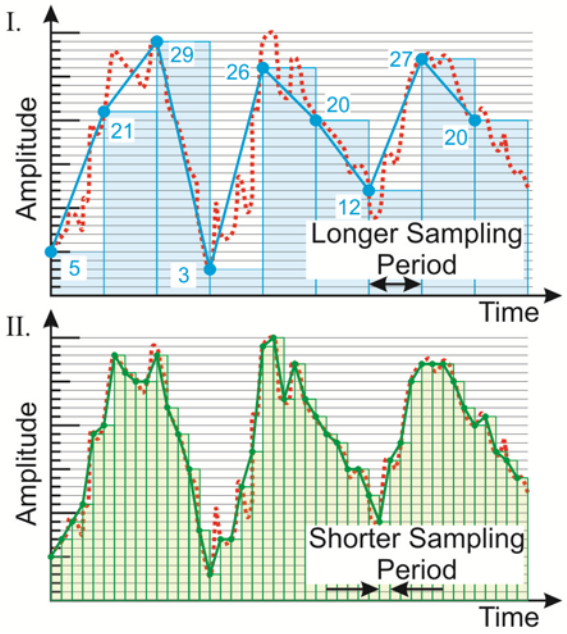

Figure 193: Digital Sampling.

detect a difference. In this frame, we've chosen a time step that's somewhat large. Large is a pretty subjective word, though, so we ought to compare it to something. In this case, the time step is large compared to many of the wiggles in the analog signal. Let's see what result we get with digital sampling with a large time step. The digitally-sampled signal is shown by the light blue rectangles. You can see the computer placed the first sample in the amplitude box it calls five. It waits for the next tick and puts the next signal in the amplitude box it calls 21. This process repeats seven more times until we run out of signal in this example.

Now, while the computer only knows rectangles (a fixed-level vertical jump after a set horizontal time step), our eyes see connect-the-dots better. That's why we drew the dots and connected them with the solid blue line. Can you see a difference between the blue line and the red curve? Of course you can! That's why we chose a large time step so you'd see that wasn't the right choice if we're trying to accurately match the rapidly-changing analog signal with a digital one. (We could also have used an amplitude step that was large with respect to the wiggles, but we think you can see the results would be similar.)

In Frame II we shrunk the time step to one-fifth what it was in Frame I. You can look at the same things we did before (rectangles, etc.), but to jump to the punch line let's compare the green line with the red curve. Much closer, huh? With this smaller time step, we really only miss capturing the parts of the red curve that change very quickly. When the time step is small when compared to the wiggles we're trying to measure, we get a much better digital copy of the signal.

So how good do we have to be? How close together do our time steps need to be in order to get a copy that's *good enough*? It turns out that answer is based on frequency, which makes sense because as we learned in *The Uplink Center* chapter, the rate at which the signal wiggles is the same thing as frequency. It's been shown that in order to get a *good enough* digital copy, we have to sample at

a rate at least twice the highest frequency we're trying to capture.[104] As an example, since the human ear is sensitive to frequencies up to 20,000 hertz, that means we need to sample sound at a minimum rate of 40,000 times per second to get an acceptable digital copy, one our ears won't be able to distinguish from nature's analog wave.

And with that, you now know a lot more about *why* we use digital signals (it's so our computers can manipulate a relatively small set of numbers) and *how* we get a digital signal from an analog one (we sample it at an appropriate rate).

Audio/Video Connectors.[106] And there ends our digression into digital sampling. As we said, the picture and sound are in uncompressed digital form when they leave the MPEG decoder. And we've seen that an appropriately-sampled digital form gives a pretty darned good reproduction of the original signals because we can make sure there are almost no errors between transmission and reception. However, the only output we have on the back of our receivers that will deliver true digital video signals to our televisions is the **HDMI** cable, shown in Figure 194. These cables are capable of delivering even higher than the 1080p resolution signals that are the best commercially available at the time of this writing.

Figure 194: An HDMI Connector.[105]

As we'll be citing screen resolution a few more times, it's probably worth yet another digression, this time into the subject of pixels. We just mentioned the best resolution—1080p—but what did that mean? The *1080* tells you the number of horizontal lines of pixels that are displayed on a television set. The *p* tells you that the scanning is *progressive*. Let's look at both of those bits of information separately.

We saw way back in Figure 5 (some of which is reproduced here in Figure 195) that on an HDTV screen there are 1080 horizontal lines and 1920 vertical columns of pixels. So why do we only

Figure 195: Rows and Columns of Pixels in an HDTV Display.

mention the 1080 part when we cite resolution? That's because the HD standard specifies that an HD display will have a ratio of 16:9, which means that for every 16 pixels across there will be 9 down. Doing a little math, $1080 \times 16 \div 9 = 1920$, we now see where the other number comes from. That's why we only have to say 1080. Knowing the aspect ratio gives us the other number. Thus, an HD display can show 1080 x 1920 = 2,073,600 pixels to the viewer in a single frame. Older, non-HD televisions had a ratio of 4:3, and the number of lines in the old-style display was only 480,[107] which means there were 640 columns for a total of 307,200 pixels that could be displayed at any one time. That's less than a sixth of what shows up on an HD screen.

To give you an idea of how good the best HD picture is compared to the best SD picture, look at Figure 196. The blue rectangle represents the shape and size of an old SD screen. The red rectangle shows an HD screen shape of the same height. If you look closely, there's a tiny green rectangle near the top left corner of these screens. We've blown that rectangle up to show the pixels an area that size would contain. The HD pixels are significantly smaller than the SD pixels, and you can see that there are about four of them in the space of every SD pixel. Knowing what you do about digital sampling, what do you think that sampling the picture captured by a camera four times as frequently (in distance along the screen, not time between samples) will do for our ability to match the analog picture? Well, the difference between the two digital sampling curves in Figure 193 was only 5 times, and look how much better a match we got there! If you've not had the opportunity to watch an SD and an HD picture of the same show side by side, we highly recommend it as a real-world example of how higher sampling rates improves the quality of the digital match to an original analog picture.

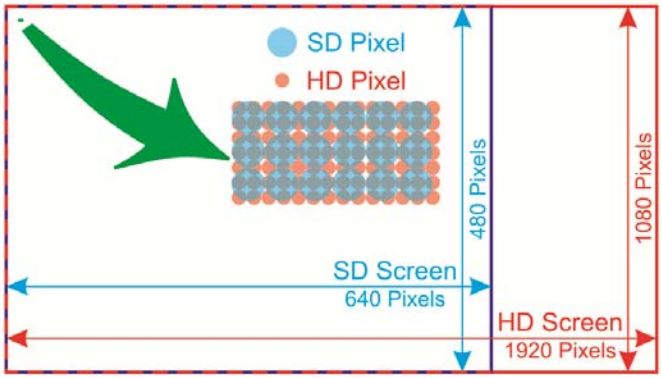

Figure 196: HD vs. SD Pixel Resolution.

So that's a look at what the *number* means when we hear television manufacturers brag about their resolution. But what about that pesky letter *p*? Why is sometimes the letter an *i* instead? Well, they stand for *progressive* and *interlaced*, respectively, and they describe how the picture is drawn on the screen.

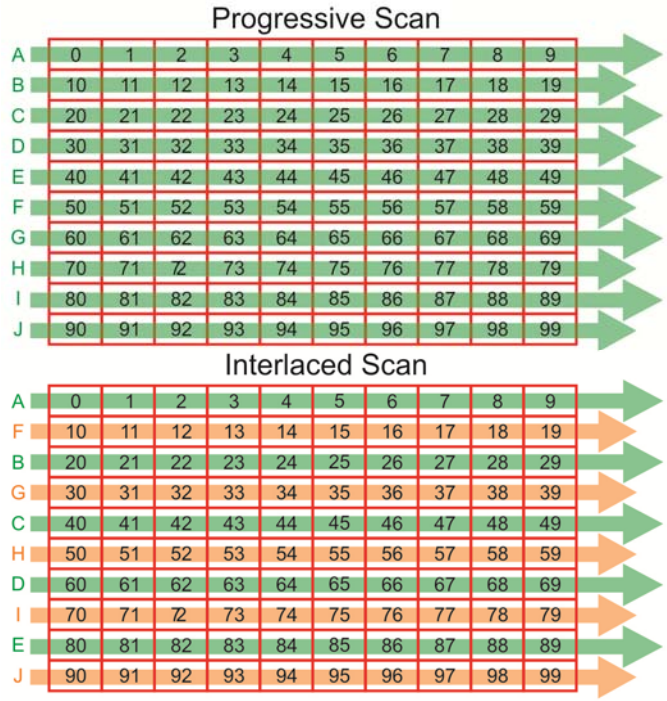

Figure 197: Progressive and Interlaced Scans.

Let's look at the highly simplified screens in Figure 197 to illustrate the two concepts. These screens both consist of ten rows by ten columns of pixels. The top screen, representing a **progressive scan**, displays the pixels in order, from zero to 99. Row A is sent to the screen from left to right—pixel 0 is displayed first, pixel 1 next, until pixel 9 is displayed—followed by row B—pixel 10, pixel 11, etc.—and so on until row J is completely displayed. Let's say the frames are captured by the camera and are delivered to the screen thirty times a second. That means that one-thirtieth of a second elapses between the display of pixel zero and pixel 99.

The **interlaced scan** image is drawn on the screen differently. In this case, pixels zero through 9 are drawn, then 20 to 29, 40 to 49, and so on until pixel 89 is drawn. The scan still uses the row-A-through-row-E pattern following the green arrows, but rows A and B are separated from each other by row F. If we use the same pixel-display rate, this pattern takes half the time—one sixtieth of a second—it took to display the progressive image since we've only displayed half the total number of pixels. Next, the camera captures *another* image but only draws the pixels starting at 10 to 19, then 30 to 39, until it gets to pixel 99 (rows F through J shown with the red arrows). Again, this second half-frame takes another sixtieth of a second to display. We highlighted the word *another* in the previous sentence because It's important to note that the second half-frame is actually taken from a *different* image than the one that was displayed in the first half-frame. Pictures are being delivered to the screen sixty times a second, but only half the information in a picture is ever actually seen by the viewer.

Basically, an interlaced image is drawn like two hands with their fingers interlaced—half the fingers belong to one hand and half to another as shown in Figure 198. One set of "fingers" gets drawn by the television followed by the other set of fingers from a different hand, so alternating rows of pixels are displayed in each successive frame.

The amount of time it takes between sequential frames in a video sequence is called the **refresh rate**. Just as we've learned from digital sampling of analog signals and sampling of a screen with smaller pixels, sampling a real moving image more often—with a higher refresh rate—will better approximate the movement in the real scene. However, as we've seen many times, there are limits on just how good things need to be before they exceed the human eye's ability to perceive them. Our eyes can only see differences in images that occur at a rate less than about 50 times per second.[109] When images are displayed at less than that rate, they appear to flicker. The motion they're trying to portray appears somewhat jerky, with the jerkiness becoming ever more noticeable as the refresh rate goes down.

Figure 198: Interlaced Fingers.[108]

The advantage of interlaced scan is that half the screen is refreshed at twice the rate as the progressive scan. Instead of 30 frames per second, you can get 60 half-frames per second using the same bandwidth. The flicker on a thirty-frames-per-second progressive image is very noticeable while the interlaced display with the pseudo-sixty-frames-per-second display appears less jerky. With interlace, you end up with the illusion of smoother motion—up to a point.

Let's look at an example of how interlacing can cause problems. Figure 199 shows two individual frames of everyone's favorite smiley face. However, at time 2 the smiley face winks one eye. The camera captures the non-grayed "pixels" from time 1 and interlaces them with the non-grayed pixels from time 2 to get the interlaced image displayed in Frame III. Notice that the wink, since it occurred so quickly, is only captured by two of the six pixels that depict the winking eye. The other four pixels still display the eye open, as it was at time 1. While this is a very simplistic example of problems resulting from interlacing, there are a number of very good animated illustrations of this phenomenon, known as *interline twitter*, on the web.[110]

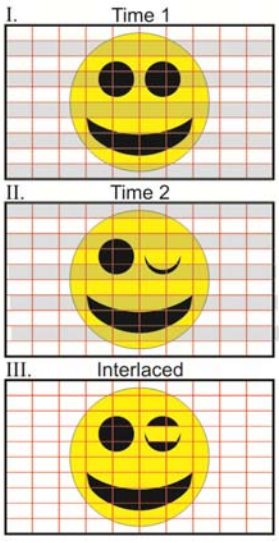

Figure 199: Interlace Blurring.

Progressively scanned pictures don't show this twitter at all, since each individual frame is shown all

at once. Another huge advantage, at least since everything DISH does is compressed using MPEG, is that MPEG works much, much better with complete frames. Can you imagine trying to calculate motion vectors or to accurately redraw the Ferrari in front of the building (p. 20) with only half the pixels available to you? Using MPEG with interlacing defeats the purpose, as the bandwidth reductions we gain from MPEG are all but wiped out because of the computational difficulty of dealing with interlacing.[111]

HD DISH signals are broadcast at up to 1080p (only 480i for SD).[112] So, if available on your TV, setting it up to display progressive scan is definitely the way to go. The downside is that the bandwidth requirements for progressive formats are twice as high as interlaced formats since we have to send an *entire* frame every sixtieth of a second.

OK, we now know what 1080p means and why it's the best picture possible at the present time. You now know to choose the settings on your television that have the biggest number possible, and then choose the *p* when it's available. Enough about TV resolution. Let's get back to a discussion of the cables coming out the back of the receiver.

Like coax cables, HDMI signals are carried by fairly high frequency waves in the cable—up to 340 MHz for the newer Category II standard[113]—so they suffer from the same skin effect-related decrease in signal strength as the cables get longer (p. 182). Most cables can be about 15 feet in length before the signal degrades enough to be noticeable, but cables manufactured to higher standards can reach up to about 50 feet.[114]

Except for the coax home distribution cable, HDMI is the only DISH output that carries both audio and video signals. However, with HDMI both signals are in digital format while the coax signals are both analog. The bottom line: if you've got a digital, HDMI-ready television then using an HDMI cable is the only rational way to connect it. With any other method, you're just losing the HD quality that you've spent so much money acquiring.

All of the other ways to get video to a television are analog, based on continuously oscillating waves instead of coded series of ones and zeroes. That means we've got to convert the digital signal to the lower-quality analog one, and we use a digital-to-analog converter inside the receiver to do that.

The first set of analog connectors we'll discuss are the **YPbPr** (commonly pronounced *yipper)* connectors, sometimes called *green-blue-red* because of their standard color coding—or *component video* because they carry different components of the video signal. These connectors deliver the highest quality of all our analog cables, but they're still far inferior to digital HDMI. All three of these cables are devoted to video, a distinction that will become important later when we discuss the other analog cables. Figure 200 shows these connectors. YPbPr is the highest-quality way to get a video signal to an analog television set, but it still is analog. That's the big difference between the YCbCr format we discussed when explaining MPEG (p. 16)—the "C" version is digital while the "P" version is analog.[116]

Figure 200: YPbPr Connectors.[115]

These cables can theoretically deliver full 1080p resolution but they don't do it as well as HDMI because we've got to convert the digital signal inside the receiver to an analog one, and that conversion inherently adds noise and other sources of signal error that are visible on the screen. While you do get the right number of pixels—1080x1920—they may or may not contain exactly the right color/luminance information. The error correction that's possible with digital signals isn't available on an analog system.

Those problems—no error correction and digital-to-analog conversion are common to all analog cables, but that's pretty much the end of the drawbacks for YPbPr cables. As you may remember from our discussion of YCbCr, there are only three signals needed to send a color television signal: the Y gives brightness or luminance information; the Cb and Cr give the blue and red color information; and the green color information can be reconstructed from the other three (see p. 16 for a review). Three signals, three cables in the YPbPr format. We don't have to do any additional encoding to get the signal to the TV. The more encoding we have to do, the lower the signal quality is going to be.[118]

Figure 201: An S-Video Connector.[117]

So why would we want to encode the signal more? Well, one reason is that we have to use fewer connectors to send the signal. Our next connector example—the **S-Video** connector (*super video* or *separate video*[119]) shown in Figure 201—only uses

two connections inside a single cable to get the signal to the TV. S-Video is another analog method of video transfer, and is also known as a Y/C connector because instead of three signals like YPbPr, it combines the two chroma (Pb and Pr) signals into a single *C* signal. We do lose some signal quality, however, when we combine the chroma signals. S-video is typically not used to deliver greater than 480i pictures—the best that an SD television can display.

Next down in analog video quality is **composite video**. Notice that in Figure 191 there are two sets of connectors labeled composite, one for TV 1 and another to feed TV 2. Each set consists of three connectors: yellow, white, and red. However, as shown in Figure 202, only one of these connectors actually deals with video: the yellow one. The other two supply audio signals, and we'll discuss them later.

Figure 202: A Composite Video Connector.[120]

The fact that with this system only one wire is handling all of the video should be a big clue to you that this method is far inferior to the other analog cables we've discussed. There's a whole lot of encoding that goes into putting the Y, Cb, and Cr signals into a single signal that can be transmitted over a single cable, and that encoding necessarily involves loss of quality. While, like S-Video, the cable can support resolutions up to 480i, the quality on this one-wire solution is definitely worse.

The lowest-quality video signal is delivered by coax, and an example of how DISH uses coax is shown in Figure 203. Now how, you may wonder, can a single coax cable that handles the huge amount of data making up three simultaneous transponder bands between the LNB and the receiver be unable to deliver higher-quality video when it's only sending a single channel from the receiver to the TV?

Figure 203: A Coaxial Output.

Well, the answer is that it's not the poor cable's fault; the problem again lies in the encoding. When we put signals from the LNB on the coax, they're digitally encoded with all the audio and video signals properly separated out into different packets. When we put just one of those video signals back onto a similar cable, it's now an analog signal that's been reconstructed from a digital one, and all three video signal components are now being transmitted as a single signal. The analog encoding and combining is why the signal's not up to snuff when compared to other delivery methods. Coax will

also deliver up to 480i, but again the quality is comparatively poor. Like HDMI, coax delivers both audio and video signals, but the coax audio is analog.

As an aside, you may be wondering how our Joeys can deliver HD-quality signals even though they're talking to their Hoppers across a coax connection. When we discussed MoCA in the *Switches, Nodes, and Other Signal Flow Devices* section (p. 170), we mentioned MoCA was a digital signal, using Internet protocol packets to move information. It's the digital/analog difference that enables very high quality information to flow to the Joey.

So, that exhausts the set of video connectors DISH uses on our receivers. The list of *stand-alone* audio connectors we use is quite a bit smaller, consisting of one digital and one analog version (remember that the HDMI and coax cables send both video and audio together). The digital connector we use is the **S/PDIF** (Sony/Phillips Data Interchange Format) connector. It's essentially a fiber optic cable, or a very high-quality thread of glass. Figure 204 shows one of these connectors, and if you look closely you can see the tiny thread coming out the end of it. The ones and zeroes that make up the audio stream of a program are converted in the receiver to pulses of light and are sent over this cable to suitably-equipped audio devices.

Figure 204: An S/PDIF Connector.[121]

When you go into the settings on your DISH receiver, you'll likely see two options for digital audio: PCM and Dolby® Digital. PCM stands for **Pulse Code Modulation**, and it means nothing more than a digitally sampled wave like we saw in Figure 193 (p. 227). Technically, we use *Linear* PCM (LPCM), which only means that the amplitude steps are all the same, as was shown in the figure. (There are other PCM schemes that use steps of different sizes for various reasons, but that's beyond the scope of this document.) In fact, we discussed Morse code as a special case of PCM way back in *The Uplink Center* chapter (p. 69). With PCM, the audio is sent to the television (or a high-end audio receiver if the customer has one) as an *uncompressed*, digitally-sampled wave. As we've seen before numerous times, compression generally adds errors, so PCM is the best sound quality we can deliver.

When complying with the most stringent Blu-Ray® standards, PCM can provide up to 8 channels of 24-bit audio at 96 kHz.[122] The "channels" number means how many different speakers the format supports (left front, center, right rear, etc.).

The number of bits just means how many amplitude steps there are (24-bit means 2 raised to the 24th power, or almost 170,000 steps; we only used 30 steps in Figure 193 and we got a reasonably good representation of the analog signal). The 96 kHz frequency indicates how often the signal is sampled (remember that we have "20/20" hearing, which means our ears can hear between 20 Hz and 20 kHz; we also need to sample at a minimum of twice the maximum frequency we want to hear to get an acceptable digital representation of the signal; 96kHz, then, is over twice as fast as the minimum required to capture the absolute highest frequency a human ear can hear). The bottom line is that the 8/24/96 PCM will deliver pretty darned good audio! Currently, DISH only provides up to 5.1 channels of audio (where 5.1 means six speakers; five of the channels are full bandwidth and the sixth is a limited-bandwidth subwoofer) but the actual audio quality of individual programs we deliver is determined primarily by what our content providers send us. The other audio option, **Dolby® Digital** (DD), is a compressed audio format. Compression means losses, so DD isn't as high a quality as PCM. Additionally, the best DD delivers only 5.1 channel, 16-bit, 48 kHz audio.

Figure 205: Composite Audio Cables.[123]

Besides S/PDIF, the other stand-alone audio connector we use is composite audio, shown in Figure 205. While commonly used together with the yellow composite video connector, they're also used in conjunction with YPbPr video cables when no digital input is available. Composite audio is analog, not digital, so the encoding required to convert our digital DISH signal to analog before sending it to the television introduces errors which degrade the audio quality. Obviously from this discussion, you'd want to use HDMI or S/PDIF if they are available before choosing the composite audio option.

Advanced Receivers

So far in this chapter, we've discussed the signal flow through a very basic receiver with only one tuner and no hard drive. More advanced tuners do almost the same thing as the basic example we've looked at in detail, but they also have a few more tricks up their sleeves.

Multi-Tuner Receivers. The first advancement we'll discuss is adding additional tuners to the receiver. Remember, the function of a tuner is to filter all but the channel requested by the customer watching the program. However, as we're all

aware, most households have more than one television set. One solution to getting DISH programming to all of those sets is to pair a receiver with every one of them, but that requires punching a new hole in our customers' homes near every TV. It also involves deploying a lot of expensive hardware, much of which is highly redundant.

Early in DISH's history, we made the business decision to reduce some of these redundancies by putting two tuners in some of our receiver models. What that innovation did was to allow a single wall penetration to serve more than one television set. It also allowed customers to watch one program and record another at the same time, either with their own recording device or one built into the receiver. It also saved us a lot of money up front because it's a lot cheaper to produce a single two-tuner receiver than it is to build two single-tuner receivers. However, having two tuners required us to run two cables to the receiver, at least until we developed the technology for using upper and lower bands on the same cable (p. 164). With that innovation, we can now use a single wall penetration to get two completely different channels to a single tuner, and then pipe those signals to two different television sets.

Now, we've jumped even deeper into the multiple-tuner creek with the development of the Hopper, which has three tuners (and a corresponding need for three LNB bands on the same coax, as previously discussed on p. 174). The three bands come into the Hopper on a single coax from a node. The Hopper then converts them to Internet-like MoCA packets and sends them back out to the node and on to the associated Joeys.

It's currently DISH business practice to only allow three Joeys to be attached to a single Hopper, but that limitation is not due to technology. The node is smart enough to handle routing data to many more Joeys. We're just worried about all the customer service calls that hooking up lots of Joeys could bring—remember, no matter how many Joeys you have, there are still only three tuners so only three different live programs can be viewed at any one time. Once all three tuners are in use, the other Joeys could watch pre-recorded content from the Hopper or watch a channel already being tuned, but they couldn't choose a new live program without cancelling one that's already being watched. We think that would be very confusing to the average customer who hasn't read this chapter on how tuners work.

However, using exactly this same equipment, we advertise the ability to record or watch up to six channels at once using Prime Time Anytime (PTAT). That

capability should bug you, now that you know how a receiver works. The Hopper has only three tuners—meaning it should be able to give us only three distinct channels to watch and/or record—so how can we record *six* different programs?

What a great question! We're glad we have smart people like you as our reader! We do it with some smart transponder management, that's how, and the answer to why transponder management helps will come directly from the understanding of what a tuner actually does that you've gained in this very chapter. Look back at Figure 178 (p. 211). It shows that the input to the tuner is the signal that makes up the 16 transponders on either an odd or even band of a single satellite, and there can be over 300 channels in that band. Now, take a look at the output: it's a single *transponder's* worth of information, with up to about 20 channels. The tuner doesn't filter out a single *channel*…it filters out a single *transponder*. DISH makes sure that the four major broadcast networks—ABC, CBS, Fox, and NBC—are all on the *same* spot beam transponder for *every* market area.

What PTAT does, then, is to take the signal from that transponder and separate out all the streams associated with the local ABC, CBS, Fox, and NBC broadcasts. In essence, when using PTAT the demultiplexer is told to throw away all but those *four* channels' information instead of throwing away all but the normal single channel. The streams are then decrypted and immediately sent to the hard drive for storage. So one tuner is able to record four channels at once, and the other two tuners, behaving normally, add the other two channels for a total of six.

One thing we've assumed in our discussion of PTAT is that we're able to *record* these four channels, and the capability to record programming leads us into the next topic under advanced receivers, the digital video recorder (DVR).

Integrated Digital Video Recorders. Here's one of those hard-to-believe facts, especially for people born after the Ford presidency: before 1976, if you wanted to watch a TV show you had to watch it on one of the three networks' time schedule. If you missed the show, you'd better hope the network decided to replay it! Want to watch a movie without commercials? You'd better go to the theater. Only since that time, when the consumer videocassette recorder (VCR) was introduced, have we been able to time-shift programs, choosing to watch them at times of our choosing and to watch them multiple times. Only since then could we watch movies in the comfort of our own home. Only since then have we been able to fast-forward through commercials on shows we recorded.

Videotape had a number of problems, but chief among them was that it was an analog recording mechanism. The copies it made of programs were highly

imperfect, showing many flaws and delivering resolution far below what the original had delivered. However, in 1999 many of those problems were corrected with the advent of the first consumer-grade **digital video recorder**, or DVR. While, as we've seen, digital copies aren't actually perfect, they can be made perfect enough that even the most astute human can barely discern the difference between copy and original.

DISH was a leader in delivering consumer DVRs, rolling out a prototype Microsoft®-based DVR—*DISHPlayer*—at the 1999 Consumer Electronics Show in Las Vegas. While two competitors did release consumer-ready devices at the show, DISH was definitely in the vanguard of that game-changing technology.[124] DVR technology has advanced to allow us to not only time-shift, but to pause and rewind live TV, watch a recording before it has finished recording, and to even automatically skip commercials. Without a DVR, none of these features that many of us take for granted would be possible.

So how do the concepts of live TV and DVRs get jumbled together? Those concepts involve live and delayed things, ideas that don't normally go together. However, in this case they work hand-in-glove. Unbeknownst to you (probably), *every* live show you watch on a DVR receiver is actually being shown to you from a recording off the receiver's internal hard drive. Whenever you change channels, the receiver starts recording all the program streams for that channel, writing the streams to the hard drive. Milliseconds later, it reads what it wrote and sends it to your screen. That's how you're able to rewind, pause, and do all of the other special features that manipulate "live" TV. That also explains why you can't go back in time to earlier than your decision to change channels—the receiver wasn't recording the appropriate streams before then. We don't record every stream—there's not enough hard drive space for that—we only record the ones you're currently watching.

Getting the streams to and from the hard drive sound pretty simple, but because of the contracts we have with some of our providers, it can add a few steps to the receiver process we saw in Figure 175 (p. 208). The additional steps all occur between the decryption step and the MPEG decoder step, and are shown in Figure 206. When we discussed PTAT above, we said the stream was put on the hard drive in an unencrypted MPEG format. That's possible because the content of the major networks is available over the air (OTA), and there are no digital

rights management (DRM) issues with OTA broadcasts. Similarly, many of the other networks we broadcast aren't worried about DRM, so they're sent directly to the hard drive unencrypted. That flow is shown with the green arrow in the figure, where the provider specifies no particular DRM requirement for their content. Programs like those can then be read from the drive and immediately sent to the MPEG decoder.

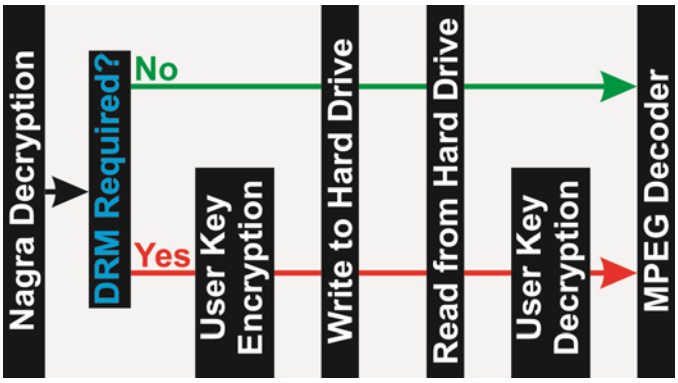

Figure 206: Signal Flow Modification Due to a Hard Drive in the Receiver.

However, for channels like HBO that broadcast movies in HD format, we send encrypted versions of the streams to the hard drive so the drive can't be taken out of one DVR and played on another, a capability that would enable video pirates to steal near-perfect digital copies. You've already become armchair experts on the Nagra encryption process we use for our transmissions to and from DISH satellites (*Smart Cards and Decryption* , p. 215). How our read-write process works is actually much simpler than that. The streams associated with the channel that requires DRM protection follow the normal receiver flow—LNB to tuner to demodulator to demultiplexer to decryption. But there they diverge a bit from the process we studied earlier, the green arrow. As shown with the red arrow, they go into a second scrambler that encrypts them using *only the user key* stored on the receiver's smart card. It's still 128-bit encryption, very difficult to crack, but it means that we don't have to store the Nagra-generated control words or service keys required to put the Nagra encryption scheme back together. To decrypt such a stream, all we need is the right smart card.

Once the stream is re-encrypted, it's written to the hard drive where it's available for replay upon demand. (Remember, this process doesn't only happen when you're watching content you specifically put on your DVR—it happens to *all* content, live or recorded, since all content goes to the hard drive so we can pause, rewind, etc.) By using the user key, we prevent protected content from being played on any other device besides the one on which it was recorded. No other player will have access to the right user key from the original smart card.

See, we told you hard drive encryption wasn't nearly as complicated as our previous foray into Nagra-land! And with that topic, we've come to the end of a

very long voyage, tracing the signal from provider to your television screen. However, you may have noticed that you're not quite at the end of this document. There are still pages to go before you come to the end, and we're not just talking about pages devoted to end notes and discussions of other phenomenal things the author has done. There's real, useful information left ahead of us. But we've come to the TV screen—the end of the line—what else can there possibly be left to discuss?

Beyond Just the TV [125]

See? You didn't have to wait long for the answer to that question. The TV is not the only end point of the DISH product. We're going beyond the TV. While at present DISH is primarily a provider of signals to television sets, we're aiming to be much more than that. More and more people are demanding to have their entertainment with them whenever they want it regardless of where in the world they are. Our first step towards satisfying these customer demands is to be able to deliver programming to smart phones, computers, and tablets wherever an Internet connection is available—whether wired, wireless, or even over a cell phone connection. And we're not just talking live programming. We're also giving our customers access to everything stored on their DVRs, either streaming via the Internet or by direct Bluetooth download for later viewing.

With the advent of VCRs and DVRs, customers became familiar with the concept of time-shifting. We're now giving them access to the phenomenon of **place-shifting**. The technology that allows us to place-shift is **Sling**, hardware and software developed by Sling Media, owned by our sister company, EchoStar. The big picture of what Sling does is to take the unencrypted video stream coming out of the receiver, reformat it to either Android or Apple standard, and deliver it to the Internet address of the customer's device. In the following discussion, we'll use the term *mobile device* to denote the place the customer wants to watch video, but realize it could also be a fixed location like a desktop computer.

While delivering video over the Internet doesn't sound too difficult, like most simple-sounding things the devil is in the details. To begin the process, the customer must establish a DISH Internet account and set up a username and password. Those credentials are used by the remote devices to gain access through Sling to the programming the customer has paid for. The customer then installs the appropriate application on their remote device—phone, tablet, computer, etc.—and uses their credentials to log on to the Sling server. Sling-type remote viewing is only enabled when the receiver is connected to the

Internet via a broadband connection. When a customer requests to view live or recorded content from his mobile device, that request sets off a complicated chain of events that can involve a large fraction of the six Sling servers used with DISH Anywhere.

Figure 207 graphically shows a version of this process; however the process has been simplified by conglomerating all the many Sling servers into one region of the diagram. In it, you can see that there are really just four two-way Internet links involved, but a lot of communications steps go on between the time a customer starts DISH Anywhere on his mobile device and the time streaming video actually starts being delivered. In the diagram, the reddish link connects the receiver and Sling, blues connect the mobile device and Sling, oranges connect the receiver and the mobile device, and greens connect Sling to DISH. Take a second to read through the color-coded steps in the figure and follow along with the numbers near the arrows making the links. We're not expecting you to become Sling experts with this diagram, but would like you to see that making the connection between receiver and mobile device isn't exceptionally straightforward.

As we stated, the diagram uses a generic "backend" server to represent what

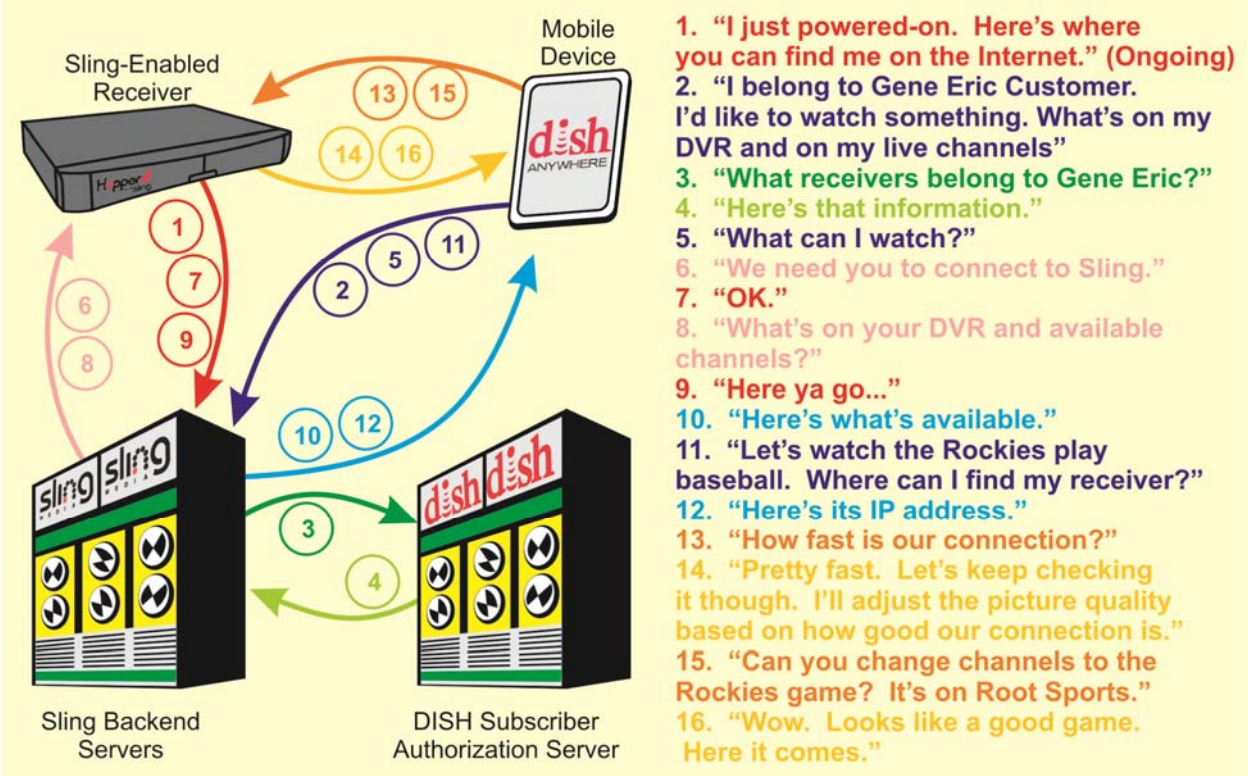

Figure 207: Delivering Content via Sling.

goes on at Sling. Let's take a little closer look at the different servers that make up this conglomeration to see what they really do. For ease of reference, we'll go through them in alphabetical order.

Enterprise Server. This server has three main functions. The first is to store and quickly return EPG information for use by the entire Sling system. Second, it receives requests from the mobile and web DISH Anywhere clients and responds with information about the customers' receivers, such as what channels they are authorized to view and what's on their DVR. Finally, it acts as the interface with the DISH Subscriber Authorization System to get user account information such as receiver identification numbers and viewing authorizations.

MAPI Server. The Messaging Application Programming Interface (MAPI) server isn't really a Sling server since it's owned, operated, and maintained by EchoStar. It contains the On Demand catalog used by DISH Anywhere.

Master Directory Server. The master directory holds app configuration information such as the URL of the servers the process uses. Without this information being available here, we'd have to update the apps on every user device whenever the addresses changed. The server also holds the IP addresses of all customer receivers that are powered on and connected to the Internet.

Mediation Server. For security reasons, many home routers are configured to not allow direct connections with outside servers. That means the normal Sling process would not work. The mediation server is used to establish a connection between the receiver and the relay server through an Internet connection just as if the mobile device were getting streaming video from a site like YouTube rather than through a direct peer-to-peer connection.

Message Server. This server establishes the always-on connection with Internet-connected receivers, and is Sling's way of keeping tabs on the IP address of the receivers. It sends commands to the receivers to contact the enterprise server when a mobile device wants to watch a program.

Relay Server. This server is used in conjunction with the mediation server when a direct connection through the customer's router is not allowed. It transports Sling video over the Internet instead of direct between receiver and mobile device.

In Figure 207 we were pretty folksy in how we describe the process, so let's look at it using a little more rigor. The process will be similar to the figure, so it may be a helpful reference during this discussion.

As soon as a receiver is powered on and connected to the Internet, it contacts the message server to let it know there's a live connection. This "I'm still awake" communication continues periodically for the entire time the receiver is Internet-connected. Simultaneously, the receiver contacts the master directory server to tell it what its IP address is. These things happen regardless of whether the customer ever uses DISH Anywhere or any of the other Sling-based applications.

Another key background task necessary to getting DISH content to a mobile device is the *mydish.com* website. Customers must establish an account on mydish.com before they're able to use DISH Anywhere, because their *mydish.com* credentials are used to ensure the mobile device connects to the correct receiver.

By logging in to DISH Anywhere using valid DISH Internet credentials, the mobile device connects with the Sling enterprise server. The enterprise server then connects to the DISH subscriber authorization server (p. 32 and p. 218) to get the receiver ID(s) of the customer's device(s). The enterprise server then returns the ID(s) to the mobile device. The DISH Anywhere app then asks the enterprise server to return the DVR recordings, channel list, and receiver ID(s) of the customer's receiver(s). The enterprise server then asks the message server to ask the receiver to connect to it. The receiver connects to the enterprise server after receiving the message server's request. The Enterprise server then directly asks the receiver for the three things the app asked for, the receiver answers, and the answers are passed back to the app. Whew! That's a lot of "hey, can you get so-and-so to call so-and-so" just to get information *about* the available content, and we haven't even discussed delivering the *actual* content yet!

Now that he knows what's available, the customer then chooses something to watch. The app asks the master directory for the IP address of the receiver using the current receiver ID. The master directory answers and the app connects directly to the receiver at that address. If live TV is what's desired, the app tells the receiver to change the channel to the one requested by the customer . Note this step requires an unused tuner, so there will be conflict management messages involved if all the receiver's tuners are in use. That's not a problem if DVR content is requested. The Sling chip on the receiver (or in the Sling adaptor) then reads the video from the hard drive and converts it to the appropriate format depending on the requesting device's operating system (Android, Apple, etc.). The app then pulls the converted video from the receiver and displays it on the mobile device's screen.

For the entire time a connection is made between the mobile device and the receiver, Sling monitors the speed of the connection to determine the optimal bit rate and resolution to deliver. That sounds simple, but being able to deliver watchable video at a wide variety of connection qualities is one of the discriminating strengths of the Sling product.

Sling can also provide On Demand viewing of the DISH Blockbuster library, manage the customer's DVR remotely (add/delete timers, etc.), and enables transfer of video files to a customer's mobile device for later viewing. Each of these processes requires similarly involved interactions with various servers and the receiver. We think you've seen enough to get an idea of what's involved, so we won't discuss them in detail here.

Chapter Summary

Who knew that so many things happened in such an innocuous looking box that sits near your television set? From probably a zero-based knowledge of those boxes, we hope you now have a reasonably good understanding of all the things that go on to bring our product to our customers. We saw that there were basically seven steps we use to take a modulated signal from the antenna and turn it into a program with audio and video: LNB → tuner → demodulator → demultiplexer → decryption → MPEG decoder → A/V module. If the tuner has a hard drive, we also need to insert that step between the decryption and MPEG sections. Each of these steps either filters out some of the signal or transforms it so it can be used further down the line. Let's revisit each of these steps briefly, summarizing this filtering and transforming.

- The LNB takes the full gamut of signals from the satellite it sees and filters out half of them, only delivering either even- or odd-numbered transponders
    - Additionally, the receiver selects which of the three LNBs it controls to look at, so after this step only one sixth of the possible signals are put into the coax and moved inside the receiver itself
- The tuner selects only one of the 16 transponders sent down by the LNB.
- The demodulator takes the 8PSK wave and translates it into binary code that computers use
- The demultiplexer reassembles the individual data streams that make up the single channel the customer has selected, discarding the rest of the data

- Decryption descrambles the data stream so a standard MPEG decoder can understand it
- If the receiver has a DVR, the signal is re-encrypted with a simpler coding scheme, written to the hard drive, read back from the hard drive, and then decrypted
- The MPEG decoder undoes all the math and compression the MPEG encoder did before the signal was uplinked
- Finally, the A/V module takes the decoded frames and sends them to the television set.

Well, that's the end of the line for our study of the signal that makes up the satellite television part of DISH. We're glad you stuck with us through this long, detailed journey. Hopefully you're now a much more informed operator of the DISH network!

Appendix I: DISH Receivers

Model	First Produced	Last Produced	Number Active (000)	Definition	Satellite Tuners	DVR	Modulation	Processed at Service?	Notes
Hopper with Sling	2013	Present	387	HD	3	Yes	8PSK	Yes	
Hopper	2011	2013	909	HD	3	Yes	8PSK	Yes	
Joey	2011	Present	2116	HD	N/A	N/A	8PSK	Yes	
922	2008	2012	58	HD	2	Yes	8PSK	Yes	
722	2007	2010	1051	HD	2	Yes	8PSK	Yes	
722k	2008	2012	3533	HD	2	Yes	8PSK	Yes	Reduced cost version of 722
622	2006	2008	396	HD	2	Yes	8PSK	Yes	
612	2008	2012	790	HD	2	Yes	8PSK	Yes	
222	2007	2009	188	HD	2	No	8PSK	Yes	
222k	2008	2012	2640	HD	2	No	8PSK	Yes	Reduced cost version of 222
211	2005	2009	528	HD	1	No	8PSK	Yes	
211k	2008	2012	2295	HD	1	No	8PSK	Yes	Reduced cost version of 211
211z	2012	Present	87	HD	1	No	8PSK	Yes	Reduced cost version of 211k
411	2005	2007	9	HD	1	No	8PSK	Yes	211 without Ethernet
811	2003	2005	8	HD	1	No	QPSK	No	
322G	2003	2008	689	SD	2	No	8PSK	Yes	
322GRC	2004	2009	1879	SD	2	No	8PSK	Yes	Reduced cost version of 322G
512	2003	2009	132	SD	2	Yes	8PSK	Yes	Software variant of the 522 or 625 with disabled TV2 outputs
522	2003	2007	237	SD	2	Yes	8PSK	Yes	
625	2004	2009	2087	SD	2	Yes	8PSK	Yes	Essentially the same as a 522 with a larger hard drive
311	2003	2009	1954	SD	1	No	8PSK	Yes	
311k	2003	2009	327	SD	1	No	8PSK	Yes	322 programmed to look like a single-tuner product
111	2003	2005	18	SD	1	No	QPSK	No	
301	2001	2005	777	SD	1	No	QPSK	No	
381	2003	2007	76	SD	1	No	QPSK	Yes	811 with HD outputs disabled
50x	2001	2005	119	SD	1	Yes	QPSK	No	501, 508, and 510; First Generation DVR products; the larger the "x" the larger the hard drive
1000	1997	2000	7	SD	1	No	QPSK	No	
2700	1998	2001	66	SD	1	No	QPSK	No	
2800	1998	2001	108	SD	1	No	QPSK	No	
3000	1996	1999	12	SD	1	No	QPSK	No	
3700/ 3800	1998	2001	49	SD	1	No	QPSK	No	

Figure 208: Receiver Models and Capabilities.[126]

Figure 208 shows the DISH receiver models in the field as of July 2013 as well as some of their distinguishing characteristics. Looking at the numbers of receivers associate with the various flavors of modulation and definition, you can begin to appreciate the magnitude of the problem of going to an all-8PSK or an all MPEG-4 (HD) business model. Of the nearly 24 million active DISH receivers, there are still almost 9 million standard definition/MPEG-2 boxes and 1.3 million that don't understand 8PSK modulation. The boxes that are indicated as not processed at Service are discarded when they come back to us. Red font indicates receivers

that do not incorporate the most modern modulation and/or compression techniques.

Figure 209 graphically shows the breakdown of receiver numbers in a variety of categories. Note that the bottom right pie chart is a combination of the middle and bottom left charts.

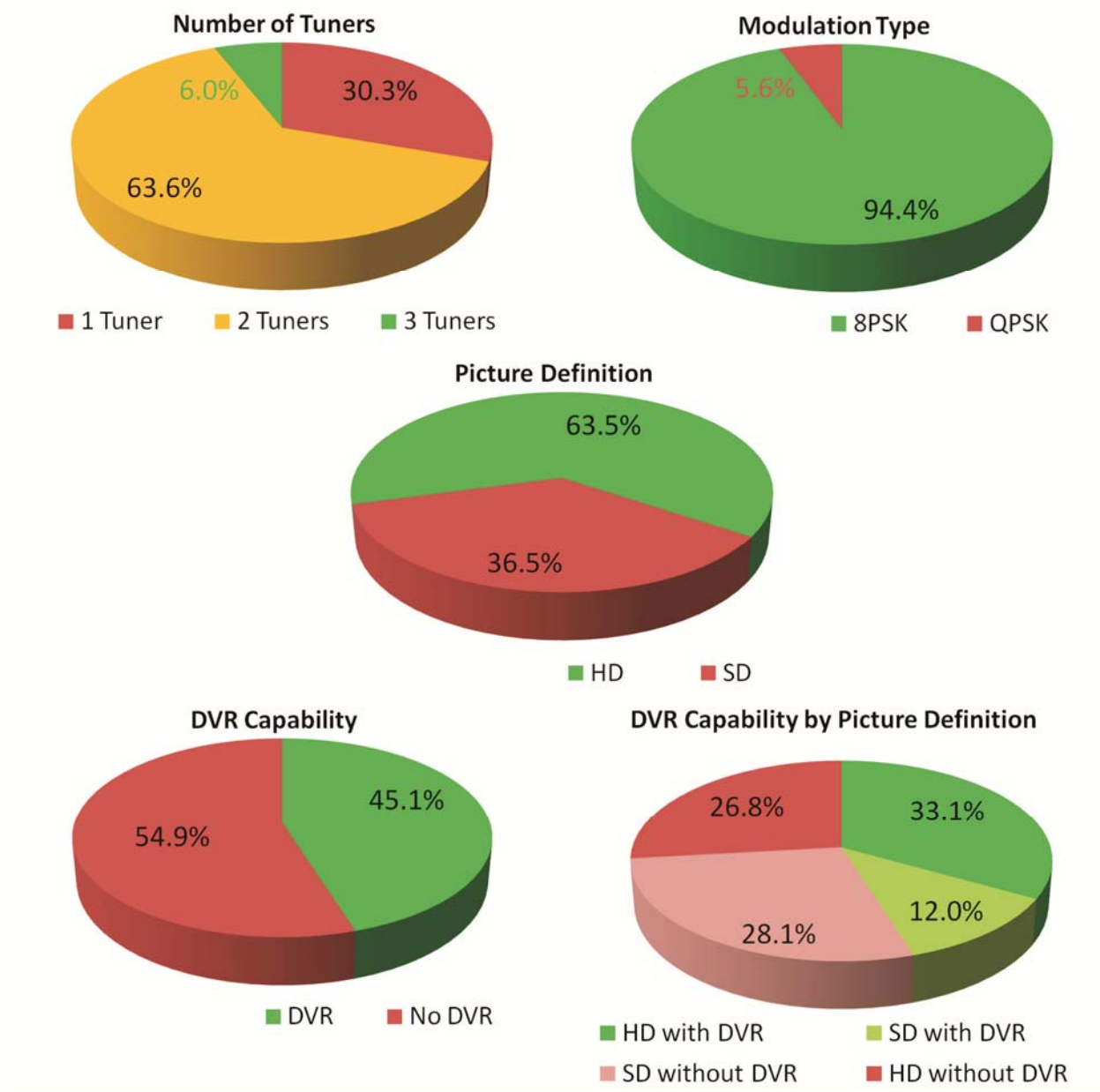

Figure 209: Receiver Statistics.

Finally, Figure 210[127] shows the ages of the receivers we currently have in service. The length of the line shows the production run and the thickness of the lines indicates the relative number of receivers still in the field. Of note, as of this

writing we still have a significant number of receivers in operation that are over a decade old and some that could be as old as 17 years. Also worthy of noting, DISH has gone through periods of fielding as many as *15 models simultaneously*, a practice that seriously complicates troubleshooting at our call centers and repairs at our remanufacturing centers. The counts of receivers manufactured in a particular year may be greater than indicated in the table due to the inclusion of receiver models that no longer are fielded.

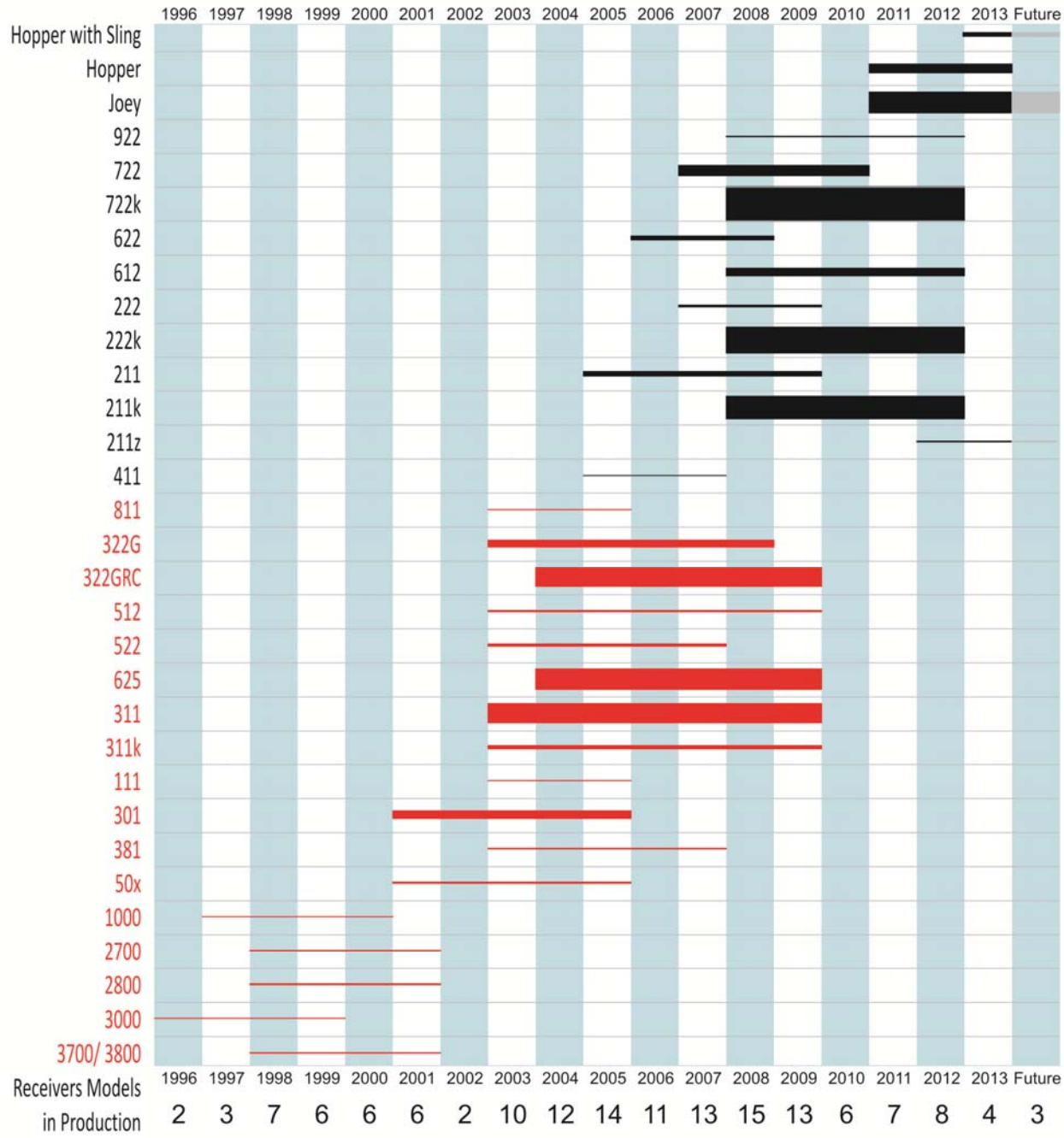

Figure 210: Fielded Receivers by Age and Number Active.

Appendix II: Satellites Actively Used by DISH

This information is from the Satbeams website.[128] The names of the beams DISH uses[129] are the ones used on that site.

Satellite TV

Satellite Name: Echostar 16
Status: active
Position: 61.5° W
NORAD: 39008
COSPAR number: 2012-065A
Operator: *EchoStar Corporation*
Launch date: 20-Nov-2012
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 6650
Dry mass (kg): 3228
Manufacturer: *Space Systems Loral (SSL)*
Model (bus): *LS-1300*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: QuetzSat 1
Status: active
Position: 77° W
NORAD: 37826
COSPAR number: 2011-054A
Operator: *SES S.A.*
Launch date: 29-Sep-2011
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 5514
Dry mass (kg): 2346
Manufacturer: *Space Systems Loral (SSL)*
Model (bus): *LS-1300*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Nimiq 5
Status: active
Position: 73° W
NORAD: 35873
COSPAR number: 2009-050A
Operator: *Telesat Canada Ltd.*
Launch date: 18-Sep-2009
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 4745
Dry mass (kg):
Manufacturer: *Space Systems Loral (SSL)*
Model (bus): *LS-1300*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Echostar 10
Status: active
Position: 110° W
NORAD: 28935
COSPAR number: 2006-003A
Operator: *EchoStar Corporation*
Launch date: 15-Feb-2006
Launch site: *Sea Launch (Odyssey)*
Launch vehicle: *Zenit 3SL*
Launch mass (kg): 4333
Dry mass (kg): 2165
Manufacturer: *Lockheed Martin*
Model (bus): *A2100AXS*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Echostar 11
Status: active
Position: 110° W
NORAD: 33207
COSPAR number: 2008-035A
Operator: *EchoStar Corporation*
Launch date: 16-Jul-2008
Launch site: *Sea Launch (Odyssey)*
Launch vehicle: *Zenit 3SL*
Launch mass (kg): 5511
Dry mass (kg): 2479
Manufacturer: *Space Systems Loral (SSL)*
Model (bus): *LS-1300*
Orbit: GEO
Expected lifetime: 15+ yrs.

Satellite Name: Echostar 14
Status: active
Position: 119° W
NORAD: 36499
COSPAR number: 2010-010A
Operator: *EchoStar Corporation*
Launch date: 21-Mar-2010
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 6379
Dry mass (kg): 3223
Manufacturer: *Space Systems Loral (SSL)*
Model (bus): *LS-1300*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Anik F3 (leased)
Status: active
Position: 118.7° W
NORAD: 31102
COSPAR number: 2007-009A
Operator: *Telesat Canada Ltd.*
Launch date: 9-Apr-2007
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 4715
Dry mass (kg): 2219
Manufacturer: *EADS Astrium*
Model (bus): *Eurostar-3000S*
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Ciel 2
Status: active
Position: 129° W
NORAD: 33453
COSPAR number: 2008-063A
Operator: *Ciel Satellite Group & Echostar*
Launch date: 10-Dec-2008
Launch site: *Baikonur Cosmodrome*
Launch vehicle: *Proton M*
Launch mass (kg): 5561
Dry mass (kg): 2620
Manufacturer: *Thales Alenia Space*
Model (bus): *Spacebus-4000C4*
Orbit: GEO
Expected lifetime: 16 yrs.

Broadband

Satellite Name: Spaceway 3
Status: active
Position: 95° W
NORAD: 32018
COSPAR number: 2007-036A
Operator: EchoStar Corporation
Launch date: 14-Aug-2007
Launch site: Guiana Space Center
Launch vehicle: Ariane 5 ECA
Launch mass (kg): 6075
Dry mass (kg): 3691
Manufacturer: Boeing (Hughes)
Model (bus): BSS-702
Orbit: GEO
Expected lifetime: 12 yrs.

Satellite Name: Echostar 17 (Jupiter 1, Spaceway 4)
Status: active
Position: 107.1° W
NORAD: 38551
COSPAR number: 2012-035A
Operator: EchoStar Corporation
Launch date: 5-Jul-2012
Launch site: Guiana Space Center
Launch vehicle: Ariane 5 ECA
Launch mass (kg): 6637
Dry mass (kg): 3497
Manufacturer: Space Systems Loral (SSL)
Model (bus): LS-1300
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: WildBlue 1 (KaStar 1, iSky 1, WB-1)
Status: active
Position: 111.1° W
NORAD: 29643
COSPAR number: 2006-054A
Operator: ViaSat, Inc.
Launch date: 8-Dec-2006
Launch site: Guiana Space Center
Launch vehicle: Ariane 5 ECA
Launch mass (kg): 4735
Dry mass (kg): 2100
Manufacturer: Space Systems Loral (SSL)
Model (bus): LS-1300
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: Anik F2 (CANSAT KA-4)
Status: active
Position: 111.1° W
NORAD: 28378
COSPAR number: 2004-027A
Operator: Telesat Canada, Ltd.
Launch date: 18-Jul-2004
Launch site: Guiana Space Center
Launch vehicle: Ariane 5 G+
Launch mass (kg): 5950
Dry mass (kg): 3489
Manufacturer: Boeing (Hughes)
Model (bus): BSS-702
Orbit: GEO
Expected lifetime: 15 yrs.

Satellite Name: ViaSat 1 (VIASAT-IOM)
Status: active
Position: 115° W
NORAD: 37843
COSPAR number: 2010-059A
Operator: ViaSat, Inc.
Launch date: 19-Oct-2011
Launch site: Baikonur Cosmodrome
Launch vehicle: Proton M
Launch mass (kg): 6740
Dry mass (kg): 3650
Manufacturer: Space Systems Loral (SSL)
Model (bus): LS-1300
Orbit: GEO
Expected lifetime: 12 yrs

Acknowledgements

This document could not have been produced without the help of some real subject matter experts within DISH, EchoStar, and EchoStar companies. From DISH, these experts include Wayne Kunze, Rex Povenmire, and Jason Valdez. At EchoStar, I was greatly assisted by Bill Moran, Scott Gebers, and Darren Hamilton. Dave Eyler at Sling and Mike Middeke and John Stein at Hughes were also very helpful.

Additionally, other non-technical-expert readers have made significant contributions to the readability and clarity of this book. They include Tracey Tomme, Boris Berman, and M.B. Tomme, Jr.

About the Author

Ed Tomme believes he has serious right-brain/left-brain issues, and that may be his greatest strength. He spent 25 years in the military, first graduating *summa cum laude* from the United States Air Force Academy (USAFA) with two degrees in physics, then moving on to become an Air Force pilot. He amassed over 2,000 hours in high performance jets, including the F-4G Wild Weasel where he led flights of four aircraft on missions playing cat-and-mouse against enemy surface-to-air missiles in the skies over Iraq, Kuwait, and Saudi Arabia. Playing the brash, ego-dominated games you may have seen in movies like *Top Gun* was the right brained part of his career.

The left-brained part began when he moved on to academia. He earned a Master's Degree in Physics from the University of Texas at Austin in preparation for an assignment teaching back at USAFA. He developed the first new course in the department of physics in over a decade, Combat Aviation Physics, which soon became the Academy's most popular technical elective. He was then selected to attend the University of Oxford in England, earning a D.Phil. (a European doctorate) in Plasma Physics. Returning to teach at USAFA, he became the only officer in that institution's history to win the top awards from both the military and academic commanders, one for leadership and one for teaching acumen.

He closed out his Air Force career leading an eighty-plus person unit which developed methods to enable tactical-level military personnel with low-level or no clearances to use information from our nation's highly-classified military and intelligence satellite systems. Managing over thirty million-dollar programs simultaneously, his organization delivered on average one critical, life-saving program a month during the height of the wars in Iraq and Afghanistan. One of his concepts became the pet project for the Air Force's highest-ranking officer, and he became a trusted counselor for numerous four-star generals.

Selected for early promotion to colonel, he chose to retire for family reasons. He worked in the defense industry for five years before coming to DISH. A respected consultant, his duties included writing speeches for the four-star commander of Air Force Space Command. He also earned an MBA during that time.

Ed believes his ability to move seamlessly between the right- and left-brained worlds is what helps him successfully explain highly technical subjects to non-technical audiences. He hopes after reading this document you'll agree with that self-assessment.

References

Bibliography

(Bolded words indicate how these works are cited in the end notes)

Bellamy, John C. *Digital Telephony*. New York: Wiley, 2000.

Connor, F. R. *Antennas*. Edward Arnold, 1989.

***IEEE** Standard for Letter Designations for Radar-frequency Bands*. New York, NY: Institute of Electrical and Electronics Engineers, 2003.

Jack, Keith. Video Demystified: A Handbook for the Digital Engineer. Amsterdam: Elsevier, 2005.

Lowe, Doug. *Networking: All-in-one Desk Reference for Dummies*. Hoboken, NJ: Wiley Pub., 2005. Print.

Module 12: Modulation. *United States Navy Electricity and Electronics Training Series (**NEETS**)*. Pensacola, FL: Naval Education and Training Program Development Center, 1985. NEETS. Web. 16 Jan. 2013. <http://jacquesricher.com/NEETS/>.

Reimers, Ulrich. *DVB: The Family of International Standards for Digital Video Broadcasting*. 2nd ed. Berlin: Springer, 2005

Sellers, Jerry Jon., William J. Astore, Robert B. Giffen, Wiley J. Larson, Douglas Kirkpatrick, Dale Gay, and Anita Shute. *Understanding Space: An Introduction to Astronautics*. New York: McGraw-Hill Companies, 2005.

Stimson, George W. *Introduction to Airborne Radar*. Mendham, NJ: SciTech Pub., 1998.

Watkinson, John. *The MPEG Handbook: MPEG-1, MPEG-2, MPEG-4*. Oxford: Elsevier/Focal, 2004.

End Notes

[1] The prefix *kilo* is lower case because the upper case K is reserved for the unit of temperature *Kelvin*.

[2] US Map from *Backroad Travel Consulting*. Web. 11 Mar. 2013. <http://www.travelconsulting.us/>.

[3] Image in Frame I courtesy of *Google Maps*. Web. 08 Mar. 2013. <http://maps.google.com/>.

[4] Were the red curve in Frame III of the figure actually parabolic, the torus would not look like the doughnut shown; that shape is for illustrative purposes only. However, the corresponding curves on the actual antenna really are parabolic.

[5] See Watkinson and Reimers for very thorough treatments of MPEG compression.

[6] Kuehni, Rolf G. *Color Vision and Technology*. Research Triangle Park, NC: American Association of Textile Chemists and Colorists, 2008. 79.

Judd, Deane Brewster, and Günter Wyszecki. *Color in Business, Science, and Industry*. New York: Wiley, 1975. 388.

[7] Adapted from Watkinson, p. 241.

[8] Luminance and chrominance are actually a little more involved than will be discussed below, involving the concept of gamma correction which is beyond the scope of this text. For an understandable general review of gamma correction, see "Understanding Gamma Correction." *Cambridge in Colour*. Web. 22 Feb. 2013. <http://www.cambridgeincolour.com/tutorials/gamma-correction.htm>.

[9] "BT.709: Parameter Values for the HDTV Standards for Production and International Programme Exchange." *International Telecommunications Union*. Apr. 2002. Web. 22 Feb. 2013. <http://www.itu.int/rec/R-REC-BT.709-5-200204-I/en>.

[10]

1.	$Y = R' + G' + B'$	(definition of Y)
2.	$G' = Y - R' - B'$	(rearrange equation 1)
3.	$Cr = R' - Y$	(definition of Cr)
4.	$R' = Cr + Y$	(rearrange equation 3)
5.	$Cb = B' - Y$	(definition of Cb)
6.	$B' = Cb + Y$	(rearrange equation 5)
7.	$G' = Y - (Cr + Y) - (Cb + Y)$	(substitute 4 and 6 into 2)

8. G’ = -(Y + Cr + Cb) (rearrange 7)

[11] Kerr, Douglas A. "Chominance Subsampling in Digital Images." *The Pumpkin* 1.3 (2012): 1-15. 19 Jan. 2012. Web. 22 Feb. 2013. <http://dougkerr.net/pumpkin/articles/Subsampling.pdf>.

[12] Lower flag image from "Diminished Value Claims and Appraisals in Colorado." *Collision Claim Associates*. Web. 05 Mar. 2013. <http://www.collisionclaims.com/colorado-diminished-value/>.

Car images from "The Sweet Chariot." *Endsleigh Insurance*. Web. 05 Mar. 2013. <http://www.endsleigh.co.uk/Pages/Sweet-Chariot-Pictures.aspx>.

DeGama, Ryan. "A Tale Of Nine Possessions." *Celtics Hub*. 3 Dec. 2010. Web. 22 Feb. 2013. <http://celticshub.com/2010/12/03/a-tale-of-nine-possessions/>.

[13] Frames II and III are adapted from Watkinson’s Figures 5.25 and 5.27.

[14] Images from Lopez, Linette. "Bloomberg TV's Betty Liu Reveals Her Cute Nickname." *Business Insider*. 06 Mar. 2012. Web. 22 Feb. 2013. <http://articles.businessinsider.com/2012-03-06/wall_street/31125650_1_favorite-book-business-insider-slow-drivers>.

DeGama, Ryan. "A Tale Of Nine Possessions." *Celtics Hub*. 3 Dec. 2010. Web. 22 Feb. 2013. <http://celticshub.com/2010/12/03/a-tale-of-nine-possessions/>.

[15] Figure adapted from "Inner Ear." *Funscrape.com*. Web. 13 Mar. 2013. <http://www.funscrape.com/Image/32512/Inner+Ear.html>.

[16] Figure adapted from Watkinson’s Figures 4.13 and 4.14, pp. 184-5.

[17] This section is a conglomeration of information from:

Povenmire, Rex. Vice President, DISH Corporate Initiatives. Personal communication. Feb-Mar 2013.

Ho, Tong. "Digital Video Broadcasting Conditional Access Architecture." *San Jose State University Department of Computer Science*. Fall 2002. Web. 13 Mar. 2013. <www.cs.sjsu.edu/faculty/stamp/CS265/projects/papers/dvbca.pdf>.

Numerous Bluebooks and Fact Sheets from the Digital Video Broadcasting Project Website, Web. 13 Mar 2013. http://www.dvb.org.

[18] "DVB CSA Algorithm." European Telecommunications Standards Institute. Web. 15 Apr. 2013. <http://www.etsi.org/services/security-algorithms/dvb-csa-algorithm>.

[19] For a detailed discussion see IEEE (2003).

[20] "C Band." *TechFAQ*. Web. 09 Jan. 2013. <http://www.tech-faq.com/c-band.html>.

[21] Absorption curve adapted from "Infrared Windows." *NASA Infrared Processing and Analysis Center*. Web. 07 Jan. 2013. <http://www.ipac.caltech.edu/outreach/Edu/Windows/irwindows.html>.

[22] Absorption curves adapted from "RF Atmospheric Absorption / Ducting." *Electronic Warfare and Radar Systems Engineering Handbook*. Point Mugu, Ca.: Naval Air Warfare Center., 1999. 5-1.1. Web. 04 Jan. 2013. <http://microwaves101.com/encyclopedia/Electronic_warfare.cfm>.

[23] Wheel/shock absorber picture from "Stock Illustration - Wheel, Shock Absorber and Brake." *CanStock Photo*. Web. 16 Jan. 2013. <http://www.canstockphoto.com/wheel-shock-absorber-and-brake-5432778.html>.

[24] "Overview of the ITU." *The International Telecommunications Union*. Web. 09 Jan. 2013. <http://www.itu.int/en/about/Pages/default.aspx>.

[25] "FCC Headlines." *Federal Communications Commission*. Web. 09 Jan. 2013. <http://www.fcc.gov/>.

[26] "United States Frequency Allocation Chart." *Home Page*. Web. 04 Jan. 2013. <http://www.ntia.doc.gov/page/2011/united-states-frequency-allocation-chart>.

[27] Image in Frames I, III, and V courtesy of *Google Maps*. Web. 08 Mar. 2013. <http://maps.google.com/>.

[28] Satellite dish photo from "Useful Images." *Technology for Secondary/College Mathematics*. Web. 15 Jan. 2013. <http://www.tsm-resources.com/images/index.html>.

[29] Frame I: "Single-Slit Diffraction with a Laser." Technical Services Group, MIT. Web. 15 Jan. 2013. <http://tsgphysics.mit.edu/front/?page=demo.php&letnum=Q%202>.

Frame II: "Lecture 35: Diffraction: Single Slit." *University of Alberta, Physics 130, Fall 2010*. Web. 15 Jan. 2013. <http://www.ualberta.ca/~pogosyan/teaching/PHYS_130/FALL_2010/lectures/lect35/lecture35.html>.

[30] "Single Slit Diffraction with Ripple Tank." *Technical Services Group, MIT*. Web. 15 Jan. 2013. < http://tsgphysics.mit.edu/front/?page=demo.php&letnum=Q1&show=0>.

[31] Image courtesy of Nist, Ken. "An HDTV Primer." HDTV Magazine Website. 07 Oct. 2008. Web. 16 Apr. 2013. <http://www.hdtvprimer.com/antennas/glossaryr.html>.

[32] To soothe the soul of the purist: yes, this is a two-dimensional example we're using for clarity of mental image. The real gain would need to be calculated using three-dimensional solid angles.

[33] The fundamental equation for gain in a circular antenna is $4\pi^2(D/2)^2\eta/\lambda$, where D is the antenna diameter, η is the antenna efficiency, and λ is the wavelength. We have chosen a relatively common efficiency factor, 0.65 to use for our gain calculations throughout this document. For more information, see Nelson, Robert A. "Antennas: The Interface with Space." *Satellite Today Website*. 10 Sept. 1999. Web. 29 Mar. 2013. <http://www.satellitetoday.com/via/antennas-the-interface-with-space _32505_p1.html>.

[34] Note that a Morse code dash should actually be three times as long as a dot; dashes of length two are used here for brevity.

[35] For a detailed discussion of phase shift keying, see Bellamy (2000), Ch. 6.

[36] E.g., Watkinson (2004), pp. 77-78.

[37] For more detail, see Bellamy Ch. 6 and "Differential Phase Shift Keying Tutorial." *Turbo Blog RSS*. Web. 21 Jan. 2013. <http://turboblogsite.com/differential-phase-shift-keying-dpsk-tutorial.html>.

[38] Hartley, Ralph V.L. "Transmission of Information." *Bell Systems Technical Journal* 7.3 (1928): 535-63. *Bell Systems Technical Journal*. Alcatel/Lucent. Web. 11 Jan. 2013. <http://www.alcatel-lucent.com/bstj/vol07-1928/bstj-vol07-issue03.html>.

[39] "Understanding Modern Digital Modulation Techniques." *Electronic Design*. Web. 18 Jan. 2013. <http://electronicdesign.com/communications/understanding-modern-digital-modulation-techniques>.

[40] Kaiser, M. Shamim. "Amplitude/Phase Modulation." *BRAC University*. Web. 18 Jan. 2013. <http://faculty.bracu.ac.bd/~kaiser/309l2.ppt>.

Blachman, Nelson. "A Comparison of the Informational Capacities of Amplitude- and Phase-Modulation Communication Systems." *Proceedings of the IRE* 41.6 (1953): 748-59. *IEEE Xplore*. Web. 18 Jan. 2013. <http://ieeexplore.ieee.org/xpl/login.jsp?reload=true&tp=&arnumber=4051383&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F10933%2F4051373%2F04051383.pdf%3Farnumber%3D4051383>.

[41] An excellent, easily understandable, non-mathematical treatment of this subject may be found in Stimson (1998), Ch. 5. However, the math may be interesting to some. If we start with a carrier wave $v_c = C \sin(\omega_c t)$ and a modulating signal $v_c = M \sin(\omega_m t + \phi)$, the effect of modulating the carrier amplitude with the modulating signal is $v_{AM} = (C + M \sin(\omega_m t + \phi)) \sin(\omega_c t)$. Expanding this equation using standard trigonometric identites, we get $v_{AM} = C \sin(\omega_c t) + \frac{M}{2}\sin[(\omega_c + \omega_m)t + \phi] + \frac{M}{2}\sin[(\omega_c - \omega_m)t - \phi]$. This equation clearly oscillates at three distinct frequencies. The carrier wave passes through unscathed while the modulation appears as equally-space sidebands surrounding the carrier, both having half the original amplitude of the modulation.

[42] To get two symmetrical sidebands like this, you must be modulating the carrier using pure amplitude, phase, or frequency modulation. There are ways to modulate a signal that create a single sideband.

[43] E.g., Young, Michael J. "The Last Mile Problem: Some Innovative Emerging Solutions." *MJY Online*. 1 Sept. 2002. Web. 21 Jan. 2013. <http://www.mjyonline.com/LastMileProblem.htm>.

"Cloudy Day Won't Rain on Laser Communications." *Phys.Org*. 10 Nov. 2006. Web. 11 Jan. 2013. <http://phys.org/news82381368.html>.

Sun, Jianfeng, Ya'nan Zhi, Wei Lu, Lijuan Wang, Enwen Dai, and Liren Liu. "High-data Rate Laser Communication Field Experiment in the Turbulence Channel." *Proc. of Proc. SPIE 8517, Laser Communication and Propagation*

through the Atmosphere and Oceans, San Diego, California. 24 Oct. 2012. Web. 11 Jan. 2013. <http://dx.doi.org/10.1117/12.928390>.

[44] Image of the Earth from “Am I at the North or South Pole?" *Earth Rangers*. Web. 17 Jan. 2013. <http://www.earthrangers.com/wildwire/this-just-in/am-i-at-the-north-or-south-pole/>.

[45] Image courtesy of Silas Cole, EchoStar.

[46] "Echo 1." NASA. Web. 19 Feb. 2013. <http://nssdc.gsfc.nasa.gov/nmc/masterCatalog.do?sc=1960-009A>.

[47] Image courtesy of NASA, “Project Echo." *Wikipedia*. 02 Oct. 2013. Web. 20 Feb. 2013. <http://en.wikipedia.org/wiki/Project_Echo>.

[48] The technical term for “real” weight is actually mass. It’s a measure of the amount of matter in an object. Weight is the product of the local force of gravity on mass. Since the force of gravity changes with position (as shown by Newton’s equation), the weight can change but the mass doesn’t.

[49] Astronaut image from Magnus, Sandra. "Sandra Magnus' Journal: Back in Space Again." *NASA: International Space Station*. Web. 24 Jan. 2013. <http://www.nasa.gov/mission_pages/station/expeditions/expedition18/journal_sandra_magnus_2.html>.

[50] Earth image from "An Ancient Universe - How Astronomers Know the Vast Scale of Cosmic Time." *The Universe in the Classroom*. American Astronomical Society. Web. 24 Jan. 2013. <http://astrosociety.org/edu/publications/tnl/56/changing1.html>.

[51] Geostationary satellite locations adapted from "Commercial Communications Satellites in Geosynchronous Orbits." *DocStock Images*. Boeing, 30 June 2012. Web. 24 Jan. 2013. <http://img.docstoccdn.com/thumb/orig/130165236.png>.

[52] Map background from "Satellite Coverage Maps." *SatBeams*. Web. 28 Feb. 2013. <http://www.satbeams.com/footprints>.

[53] An excellent, understandable reference for satellite design is Sellers (2005).

[54] Svitak, Amy. "SES-5, EchoStar 17 To Launch Despite Intelsat-19 Mishap." *Aviation Week*. 18 June 2012. Web. 28 Jan. 2013. <http://www.aviationweek.com/Article.aspx?id=/article-xml/AW_06_18_2012_p34-467825.xml>.

[55] Sellers (2005), Ch. 13.

[56] Diaz-Aguado, Millan F. et al. *Small Satellite Thermal Design Test and Analysis*. Austin, TX: Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, 2006. Web. 28 Jan 2013. <http://fastrac.ae.utexas.edu/documentation/Diaz_Aguando_10.1109.pdf>.

[57] Lafleur, Claude. "Spacecrafts Launched in 2010: EchoStar XIV." *Spacecraft Encyclopedia*. Web. 28 Jan. 2013. <http://claudelafleur.qc.ca/Spacecrafts-2010.html>.

"Arianespace...Fairing Fit (Launch)." *SatNews Daily*. Web. 28 Jan. 2013. <http://www.satnews.com/cgi-bin/story.cgi?number=988171498>.

"A Dedicated Ariane 5 to Launch Jules Verne." *European Space Agency*. 4 Mar. 2008. Web. 28 Jan. 2013.

[58] "Completion of Separation Tests on Ariane-5 Fairing." *European Space Agency Plain Text Press Releases*. 22 Nov. 1994. Web. 28 Jan. 2013. <http://www.esa.int/For_Media/Press_Releases/Completion_of_separation_tests_on_Ariane-5_fairing>.

[59] Figure adapted from a *Satellite Toolkit* (STK) plot graciously generated by Dr. T.S. Kelso, Senior Research Astrodynamicist, Center for Space Standards & Innovation, Analytical Graphics, Inc.

[60] The actual frequencies shown in the time-domain waves are 10 and 11 GHz. They were chosen to make the beat frequencies out of the mixer more easily visible in the figure.

[61] There's just a single trig identity behind what happens when the mixer multiplies the two signals. If the incoming signal is represented by $S_1 = A_1\sin(2\pi f_1 t + \phi)$ and the local oscillator signal is represented by $S_2 = A_2\sin(2\pi f_2 t)$, then, using the trig identity $\sin(\alpha)\sin(\beta) = {}^1/_2\cos(\alpha-\beta) - {}^1/_2\cos(\alpha+\beta)$, we get $S_1S_2 = \left(\frac{A_1A_2}{2}\right)\cos\left((2\pi f_1 t + \phi) - 2\pi f_2 t\right) - \left(\frac{A_1A_2}{2}\right)\cos\left((2\pi f_1 t + \phi) + 2\pi f_2 t\right)$, which simplifies a bit to $S_1S_2 = \left(\frac{A_1A_2}{2}\right)[\cos(2\pi(f_1 - f_2)t + \phi) - \cos(2\pi(f_1 + f_2)t + \phi)]$, showing that the mixed wave oscillates at the sum and difference frequencies (f_1+f_2) and (f_1-f_2).

[62] Adapted from "Satellite Coverage Maps." *SatBeams*. Web. 18 Feb. 2013. <http://www.satbeams.com/footprints>.

[63] Ahmad, Ghulam, and S. A. Mohsin. "Ch. 18, Modern Communication Satellite Antenna Techology." *Recent Advances in Technologies*. Ed. Maurizio A. Strangio. Vukovar: In-Tech, 2009. 309-326. Web. 19 Feb. 2013. <http://www.intechopen.com/books/recent-advances-in-technologies/modern-communication-satellite-antenna-technology>.

"Transponder Power." *Fixed Satellites and Telecommunications Services*. Web. 18 Feb. 2013. <http://www.intelsat.com/resources/tech-talk/transponder-power.asp>.

[64] Image courtesy of Scott Gebers, EchoStar.

[65] Data from "Satellite Coverage Maps." *SatBeams*. Web. 18 Feb. 2013. <http://www.satbeams.com/footprints>.

[66] Adapted from an image provided by Scott Gebers, EchoStar.

[67] Image courtesy of Scott Gebers, EchoStar.

[68] Bagad, V. S. *Satellite Communications*. Pune, India: Technical Publications Pune, 2009. 1.3. *Google Books*. Web. 20 Feb. 2013. <books.google.com/books?isbn=8184315929>.

[69] Svensson, Peter. "Dish Network Offering to Buy Sprint in $25.5B Deal." *Philly.Com Website*. 19 Apr. 2013. Web. 19 Apr. 2013. <http://www.philly.com/philly/business/technology/20130415_ap_dishnetworkofferingtobuysprintin255bdeal.html>.

[70] Image from "Satellite Dishes." Kansat. Web. 21 Mar. 2013. <http://www.kansat.com.au/dishes.php>.

[71] Connor, 66.

[72] Connor, 66.

[73] Vickers, R. *Feasibility Study for Enhanced C-130 Search and Rescue Capability with a Ground Penetrating Radar*. Tech. no. EMC G04011. Ottawa: Defence R&D Canada, 2004. Web. 07 Jan. 2013 <www.dtic.mil/dtic/tr/fulltext/u2/a436216.pdf>.

[74] Absorption curve adapted from "RF Atmospheric Absorption / Ducting." *Electronic Warfare and Radar Systems Engineering Handbook*. Point Mugu, Ca: Naval Air Warfare Center., 1999. 5-1.2. Web. 04 Jan. 2013. <http://microwaves101.com/encyclopedia/Electronic_warfare.cfm>.

[75] Distances and angles from "Satellite Coverage Maps." *SatBeams*. Web. 18 Feb. 2013. <http://www.satbeams.com/footprints>.

[76] Bagad, V. S. *Satellite Communications*. Pune, India: Technical Publications Pune, 2009. 313-335. *Google Books*. Web. 20 Feb. 2013. <books.google.com/books?isbn=8184315929>.

[77] Ahmad, Ghulam, and S. A. Mohsin. "Ch. 18, Modern Communication Satellite Antenna Techology." *Recent Advances in Technologies*. Ed. Maurizio A. Strangio. Vukovar: In-Tech, 2009. 304. Web. 19 Feb. 2013. <http://www.intechopen.com/books/recent-advances-in-technologies/modern-communication-satellite-antenna-technology>.

[78] The portion of the diagram for wavelengths less than 100 μm is adapted from Lean, Judith. "Solar Spectrum, Variability, and Atmospheric Absorption." *NASA Science*. Web. 18 Feb. 2013. <http://science.nasa.gov/science-news/science-at-nasa/images/sunbathing/ sunspectrum/>.

The portion of the diagram for wavelengths greater than 100 μm is adapted from Barron, W. R., E. W. Cliver, J. P. Cronin, and D. A. Guidice. "Ch. 11, Solar Radio Emission." *Handbook of Geophysics and the Space Environment*. Ed. Adolph S. Jursa. Washington, D.C.: Air Force Geophysics Laboratory, Air Force Systems Command, United States Air Force, 1985. Print.

[79] Diagram adapted from Kennewell, John A. "Solar Radio Interference to Satellite Downlinks." *Solar Satellite RFI*. Australian Space Academy. Web. 04 Feb. 2013. <http://www.spaceacademy.net.au/spacelink/solrfi/solrfi.htm>.

[80] E.g., Sætre, Jens T. "Sun Outage / Sun Interference Prediction for Geostationary Orbit Satellites." *Satellite Calculations.com*. 24 Oct. 2012. Web. 04 Feb. 2013. <http://www.satellite-calculations.com/Satellite/suninterference.php>.

[81] Calculations from Sætre.

[82] For a basic discussion of sidelobe suppression, see Chapter 8 of Stimson (1998).

[83] Sunspot image adapted from "The Sunspot Cycle." *National Center for Atmospheric Research*. Web. 19 Feb. 2013. <http://spark.ucar.edu/sunspot-cycle>.

Sunspot number plot adapted from Phillips, Tony. "Long Range Solar Forecast." *NASA Science*. 10 May 2006. Web. 19 Feb. 2013.

<http://science.nasa.gov/science-news/science-at-nasa/2006/10may_longrange/>.

Solar Flare image from "Solar Flares: Everything You Need to Know." *The (London) Telegraph*. 22 Feb. 2012. Web. 19 Feb. 2013. <http://www.telegraph.co.uk/ science/space/9097587/Solar-flares-everything-you-need-to-know.html>.

[84] Tascione, Thomas F. "Ch. 2, Solar Physics." *Introduction to the Space Environment*. Malabar, FL: Orbit Book, 1988. Print.

Tascione. "Ch. 9, Radiowave Propagation in the Ionosphere."

[85] Communications engineers may not immediately recognize the link budget equation as presented in this section. Our goal is to explain the concept to a non-technical audience, so we've chosen to put all things that detract from the budget, the losses, in the denominator of the equation. That way, we will be able to say things like, "if the atmospheric loss gets bigger, the power margin gets smaller," and be understandable based on our discussion of "easy math" in the introduction. In reality, the losses really are in the denominator, since they're numbers between 0 and 1 which means the fractions that they represent have a larger denominator component, but if you've made it this far in this endnote, you probably already knew that.

[86] This equation leaves out some of the more esoteric factors for clarity at this level. For a mathematical discussion of the link equation, see Dietrich, Fred J., and Richard S. Davies. "13. Communications Architecture." *Space Mission Analysis and Design*. Ed. Wiley J. Larson and James R. Wertz. Third ed. Torrance, CA: Microcosm, 1999. 533-70. Print, or Bagad, V. S. *Satellite Communications*. Pune, India: Technical Publications Pune, 2009. 313-335. *Google Books*. Web. 20 Feb. 2013. <books.google.com/ books?isbn=8184315929>

[87] For you purists, yes, we know waves don't really interact with slits like this illustration shows. However, for illustrative purposes we're going with it. If you'd like more information, see the section on diffraction.

[88] Diagrams and examples for these devices are based on information in DISH's internal online knowledge repository, *Community*.

[89] The physical phenomenon we're describing is called the skin effect, and many good explanations of it can be found on the Internet. However, most of the ones we found are fairly technical.

[90] "Resistivities for Common Metals." Brigham Young University Cleanroom Website. Web. 23 Apr. 2013. <http://www.cleanroom.byu.edu/Resistivities.phtml>.

[91] "United States Frequency Allocation Chart." *Home Page*. Web. 04 Jan. 2013. <http://www.ntia.doc.gov/page/2011/united-states-frequency-allocation-chart>.

[92] "Coaxial Cable." Summit Cable Website. Web. 18 Apr. 2013. <http://www.summit-cable.com/coaxial-cable.html>.

[93] Adapted from web calculations available at "Coaxial Cable Attenuation Calculator." *NetComber Website*. Web. 28 Mar. 2013. <http://www.net-comber.com/cable-loss.html>.

[94] The locations of specific channels on specific satellites and transponders are representative only.

[95] This section and the following section on tables are a conglomeration of information from:

Moran, William. Vice President, EchoStar Engineering. Personal communication. Apr-May 2013.

Povenmire, Rex. Vice President, DISH Corporate Initiatives. Personal communication. May 2013.

Kunze, Wayne. Director, Service Test Engineering. Personal communication. May 2013

Reimers, Ch. 5.

Watkinson, Ch. 6.

Digital Video Broadcasting; Guidelines on Implementation and Usage of Service Information. Rep. no. RTR/JTC-00DVB-40 (ETSI TR 101 211 V1.9.1). European Telecommunications Standards Institute, June 2009. Web. 30 Apr. 2013. <http://www.etsi.org/deliver/etsi_tr/101200_101299/101211/01.09.01_60/tr_101211v010901p.pdf>.

Morris, Steven. "An Introduction to Digital TV Technology." *TV Without Borders Website*. Web. 29 Apr. 2013. <http://www.mhp-interactive.org/tutorials/dtv_intro/dtv_intro>.

"MPEG-2 Transport Stream/Service Information in DVB Systems." *Digital Video Broadcasting*. Tektronix. Web. 29 Apr. 2013. <http://www.cascaderange.org/presentations/DVB_Poster.pdf>.

[96] Ibrahim, K. F., and Eugene Trundle. "Chapter 7: MPEG 2 Transport Stream." *Newnes Guide to Television and Video Technology*. Oxford: Newnes, 2007. 97-108. Google Books. Web. 2 May 2013. <http://books.google.com/books?id=90GZPA-hCuMC&dq=pes+packet+length+bits&source=gbs_navlinks_s>.

[97] For a quick, understandable overview of error correction, see Ibrahim, K. F., and Eugene Trundle. "Chapter 8: Channel Encoding." *Newnes Guide to Television and Video Technology*. Oxford: Newnes, 2007. 109-128. Google Books. Web. 2 May 2013. <http://books.google.com/books?id=90GZPA-hCuMC&dq=pes+packet+length+bits&source=gbs_navlinks_s>.

[98] Moran, William. Vice President, EchoStar Engineering. Personal communication. May 2013.

[99] E.g., IP Datacast: Program Specific Information/Service Information. Rep. no. A079 Rev 2. Digital Video Broadcasting Project, Geneva, Switzerland, Aug. 2008. Web. 9 May 2013. <http://www.dvb-h.org/PDF/a079r2-2.tm4098.dTS102470-2.V1.2.1.IPDC-over-SH_PSI-SI.pdf>.

[100] Table adapted from *Digital Video Broadcasting; Guidelines on Implementation and Usage of Service Information*. Rep. no. RTR/JTC-00DVB-40 (ETSI TR 101 211 V1.9.1). European Telecommunications Standards Institute, June 2009. Web. 30 Apr. 2013. <http://www.etsi.org/deliver/etsi_tr/101200_101299/101211/01.09.01_60/tr_101211v010901p.pdf>.

Reserved PIDs for the tables are from Reimers, p. 104.

[101] We must add a 180° phase delay to the positive sideband to make the signals add correctly.

[102] This section is a conglomeration of information from:

Moran, William. Vice President, EchoStar Engineering. Personal communication. Apr-May 2013.

Povenmire, Rex. Vice President, DISH Corporate Initiatives. Personal communication. Mar-May 2013.

Reimers, Ch. 8.

Chiariglione, Leonardo. "Protecting Content." *Riding the Media Bits*. 21 Aug. 2011. Web. 13 May 2013. <http://ride.chiariglione.org/protecting_content.php>.

Massel, Mark. *Digital Television, DVB-T COFDM and ATSC 8-VSB*. [S.l.]: DigitalTV.com, 2000. 115-22. *Google Books*. Web. 13 May 2013. <http://books.google.com/books?id=7oeJ_9wxUW8C>.

Morris, Steven. "The Conditional Access API." *TV Without Borders Website*. Web. 29 Apr. 2013. < http://www.interactivetvweb.org/tutorials/mhp/conditional_access_api>.

Ong, F. L. C., X. Liang, P. Pillai, P. M. L. Chan, G. Koltsidas, F. N. Pavlidou, E. Ferro, A. Gotta, H. Cruickshank, S. Iyengar, G. Fairhurst, and V. Mancuso. "Fusion of Digital Television, Broadband Internet and Mobile Communications—Part I: Enabling Technologies." *International Journal of Satellite Communications and Networking* **25**.4 (2007): 363-407. Print.

[103] A detailed rationale behind our best-to-worst ranking may be found on p. 7 and in Ch. 6 of Jack. Further information came from personal discussions with Jason Valdez, Engineering Manager in the Testing Directorate of DISH Service.

[104] Nyquist, H. "Certain Topics in Telegraph Transmission Theory." *Transactions of the American Institute of Electrical Engineers* 47.2 (1928): 617-44. Print.

[105] Image courtesy of "HDMI Connector." *Wikipedia*. 12 Aug. 2011. Web. 15 May 2013. <http://en.wikipedia.org/wiki/File:HDMI-Connector.jpg>.

[106] This section is a conglomeration of information from:

Valdez, Jason. Engineering Manager, DISH Service Testing. Personal communication. May 2013.

Kunze, Wayne. Director, Service Test Engineering. Personal communication. May 2013

Jack, Ch 6.

"Tutorial 734: Video Basics." *Maxim Integrated*. 8 May 2002. Web. 30 May 2013. <http://www.maximintegrated.com/app-notes/index.mvp/id/734>.

"Analog Video 101." *Developer Zone*. National Instruments, 15 Apr. 2011. Web. 30 May 2013. <http://www.ni.com/white-paper/4750/en>.

"List of Video Connectors." *Wikipedia*. Wikimedia Foundation, 16 May 2013. Web. 31 May 2013. <http://en.wikipedia.org/wiki/List_of_video_connectors>.

[107] The NTSC Standard uses 525 lines in its signal, but of those only 480 are used for the actual video image. "Analog Video 101." *Developer Zone*. National Instruments, 15 Apr. 2011. Web. 30 May 2013. <http://www.ni.com/white-paper/4750/en>.

[108] Image from "Businessman with Fingers Interlaced." *Corbis Images*. Web. 16 May 2013. <http://www.corbisimages.com/stock-photo/royalty-free/42-19720264/businessman-with-fingers-interlaced>.

[109] Watkinson, Ch.5.

[110] E.g., "Indian Head Interlace." *Wikipedia*. 16 Aug. 2009. Web. 16 May 2013. <http://en.wikipedia.org/wiki/File:Indian_Head_interlace.gif>.

[111] Watkinson, Ch.5.

[112] McSchooler, Jeff. Senior Vice President, Engineering, EchoStar. Personal communication. Jun 2013.

[113] "High Definition Multimedia Interface Specification Version 1.3a." *MCU Zone Website*. 10 Nov. 2006. Web. 15 May 2013. <http://www.mcuzone.com/AN/HDMISpecification13a.pdf>.

[114] Shackleford, Gordon. "Ask An Installer: HDMI 1.3 Cable Length Limit." *Sound and Vision Magazine.com*. 22 Feb. 2007. Web. 15 May 2013. <http://www.soundandvisionmag.com/article/ask-installer-hdmi-13-cable-length-limit>.

[115] Image courtesy of "Component Cables." *Wikipedia*. 30 Aug. 2010. Web. 15 May 2013. <http://en.wikipedia.org/wiki/File:Component-cables.jpg>.

[116] "YCbCr Definition." *PC Magazine Encyclopedia*. Web. 30 May 2013. <http://www.pcmag.com/encyclopedia/term/55147/ycbcr>.

[117] Image courtesy of "S-Video Connection." *Wikipedia*. 31 Aug. 2010. Web. 15 May 2013. <http://commons.wikimedia.org/wiki/File:S-video-connection.jpg>.

[118] "Tutorial 734: Video Basics." *Maxim Integrated*. 8 May 2002. Web. 30 May 2013. <http://www.maximintegrated.com/app-notes/index.mvp/id/734>.

[119] "S-video Definition." *PC Magazine Encyclopedia*. Web. 05 June 2013. <http://www.pcmag.com/encyclopedia/term/50751/s-video>.

[120] Image courtesy of "Composite Video Cable." *Wikipedia*. 31 Aug. 2010. Web. 15 May 2013. <http://en.wikipedia.org/wiki/File:Composite-video-cable.jpg>.

[121] Image courtesy of "Cinch RCA (Composite, Audio, HDTV Components)." *Tom's Hardware*. Web. 15 May 2013. <http://www.tomshardware.com/reviews/pc-interfaces-101,1177-5.html>.

[122] "Blu-Ray Disk Format 2b: Audio Visual Format Specifications for BD-ROM." *Blu-Ray*. Mar. 2005. Web. 31 May 2013. <http://www.blu-raydisc.com/Assets/Downloadablefile/2b_bdrom_audiovisualapplication_0305-12955-15269.pdf>.

[123] Image courtesy of "Composite Cables." *Wikipedia*. 31 Aug. 2010. Web. 15 May 2013. <http://en.wikipedia.org/wiki/File:Composite-cables.jpg>.

[124] "WebTV Networks and EchoStar Communications Introduce First Internet TV Satellite Product and Service." *Microsoft News Center*. 7 Jan. 1999. Web. 10 June 2013. <http://www.microsoft.com/en-us/news/press/1999/jan99/echostarpr.aspx>.

[125] Eyler, David, Director, Product Management, Sling Media. Personal communication. Jun 2013.

[126] Data from:

Valdez, Jason, Lead Manager, DISH Test Development. Personal communication. Jul 2013

Rink, Brian, Analyst, DISH Service. Personal communication. Jul 2013.

DISH Service Net

[127] Data from DISH Service Net.

[128] Data from "List of Satellites at Geostationary Orbit." *SatBeams*. Web. 25 Jun. 2013. <http://www.satbeams.com/satellites>.

[129] Gebers, Scott. EchoStar. Personal communication. Jul 2013.