Making Connections Efficient: Multiplexing and Compression

MANY PEOPLE NOW HAVE a portable music playback system such as Apple Computer’s iPod. The full-sized iPod is capable of storing up to 5000 songs. How much storage is necessary to hold this much music? If we consider that a typical song taken from a compact disc is composed of approximately 32 million bytes (assuming that an average song length is 3 minutes, that the music is sampled 44,100 times per second, and that each sample is 16 bits, in both left and right channels), then storing 5000 songs of 32 million bytes each would require 160 billion bytes. Interestingly, Apple says its iPod contains only 20 Gigabytes of storage—in other words, roughly 20 billion bytes. How is it possible to squeeze 5000 songs (160 billion bytes) into a storage space of little over 20 billion bytes? The answer is through compression. While there are many types of compression techniques, the basic objective underlying them is the same— to squeeze as much data as possible into a limited amount of storage space.

Music compression is not the only type of data that can be compressed. iPods can also compress speech, thus allowing the user to record messages and memos to oneself or for later transmission to another person. Clearly, the iPod would not be the device it is today without a compression technique.

Does anything get lost when we compress data into a smaller form?

Are there multiple forms of compression?

Do certain compression techniques work better with certain types of applications?


After reading this chapter, you should be able to:
Under the simplest conditions, a medium can carry only one signal at any moment in time. For example, the twisted pair cable that connects a keyboard to a microcomputer carries a single digital signal. Likewise, the Category 5e twisted pair wire that connects a microcomputer to a local area network carries only one digital signal at a time. Many times, however, we want a medium to carry multiple signals at the same time. When watching television, for example, we want to receive multiple television channels in case we don’t like the program on the channel we are currently watching. We have the same expectations of broadcast radio. Additionally, when you walk or drive around town and see many people all talking on cellular telephones, something allows this simultaneous transmission of multiple cell phone signals to happen. This technique of transmitting multiple signals over a single medium is **multiplexing**. Multiplexing is a technique typically performed at the network access layer of the TCP/IP protocol suite.

For multiple signals to share one medium, the medium must somehow be “divided” to give each signal a portion of the total bandwidth. Presently, there are three basic ways to divide a medium: a division of frequencies, a division of time, and a division of transmission codes. Regardless of the technique, multiplexing can make a communications link, or connection, more efficient by combining the signals from multiple sources. We will examine the three ways a medium can be divided by describing in detail the multiplexing technique that corresponds to each division, and then follow with a discussion that compares the advantages and disadvantages of all the techniques.

Another way to make a connection between two devices more efficient is to compress the data that transfers over the connection. If a file is compressed to one half its normal size, it will take one half the time or one half the bandwidth to transfer that file. This compressed file will also take up less storage space, which is clearly another benefit. As we shall see, there are a number of compression techniques
currently used in communication (and entertainment) systems, some of which are capable of returning an exact copy of the original data (lossless), while others are not (lossy). But let’s start first with multiplexing.

**Frequency Division Multiplexing**

Frequency division multiplexing is the oldest multiplexing technique and is used in many fields of communications, including broadcast television and radio, cable television and cellular telephones. It is also one of the simplest multiplexing techniques. Frequency division multiplexing (FDM) is the assignment of non-overlapping frequency ranges to each “user” of a medium. A user may be a television station that transmits its television channel through the airwaves (the medium) and into homes and businesses. A user might also be the cellular telephone transmitting signals over the medium you are talking on, or it could be a computer terminal sending data over a wire to a mainframe computer. To allow multiple users to share a single medium, FDM assigns each user a separate channel. A channel is an assigned set of frequencies that is used to transmit the user’s signal. In frequency division multiplexing, this signal is analog.

There are many examples of frequency division multiplexing in business and everyday life. Cable television is still one of the more commonly found applications of frequency division multiplexing. Each cable television channel is assigned a unique range of frequencies, as shown in Table 5-1. Each cable television channel is assigned a range of frequencies by the Federal Communications Commission, and these frequency assignments are fixed, or static. Note from Table 5-1 that the frequencies of the various channels do not overlap. The television set, cable television box, or a videocassette recorder contains a tuner, or channel selector. The tuner separates one channel from the next and presents each as an individual data stream to you, the viewer.

[Note from Table 5-1 is off—the final horizontal line is misplaced. We weren’t able to make adjustments to it in this document without creating other formatting irregularities. Please adjust the table. For your reference, the table is unchanged from the text’s 3rd edition (ISBN 0-619-16035-7), where it appears on page 157.]

**Table 5-1**

*Assignment of frequencies for cable television channels*

<table>
<thead>
<tr>
<th>Channel</th>
<th>Frequency in MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-Band VHF and Cable</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>55–60</td>
</tr>
<tr>
<td>3</td>
<td>61–66</td>
</tr>
<tr>
<td>4</td>
<td>67–72</td>
</tr>
<tr>
<td>5</td>
<td>77–82</td>
</tr>
<tr>
<td>6</td>
<td>83–88</td>
</tr>
</tbody>
</table>
In the business world, some companies use frequency division multiplexing with broadband coaxial cable to deliver multiple audio and video channels to computer workstations. A user sitting at a workstation can download high-quality music and video files in analog format while performing other computer-related activities. Videoconferencing is another common application in which two or more users transmit frequency multiplexed signals, often over long distances. More and more companies are considering videoconferencing instead of having their employees make a lot of trips. A few companies also use frequency division multiplexing to interconnect multiple computer workstations or terminals to a mainframe computer. The data streams from the workstations are multiplexed together and transferred over some type of medium. On the receiving end, the multiplexed data streams are separated for delivery to the appropriate device.
Other common examples of frequency division multiplexing are the cellular telephone systems. These systems divide the bandwidth that is available to them into multiple channels. Thus, the telephone connection of one user is assigned one set of frequencies for transmission, while the telephone connection of a second user is assigned a second set of frequencies. As explained in Chapter Three, first-generation cellular telephone systems allocated channels using frequency ranges within the 800 to 900 megahertz (MHz) spectrum. To be more precise, the 824 to 849 MHz range was used for receiving signals from cellular telephones (the uplink), while the 869 to 894 MHz range was used for transmitting to cellular telephones (the downlink). To carry on a two-way conversation, two channels were assigned to each telephone connection. The signals coming into the cellular telephone came in on one 30-kHz band (in the 869 to 894 MHz range), while the signals leaving the cellular telephone went out on a different 30-kHz band (in the 824 to 849 MHz range). Cellular telephones are an example of dynamically assigned channels. When a user enters a telephone number and presses the Send button, the cellular network assigns this connection a range of frequencies based on current network availability. As you might expect, the dynamic assignment of frequencies can be less wasteful than the static assignment of frequencies, which is found in terminal-to-mainframe computer multiplexed systems and television systems.

Generally speaking, in all frequency division multiplexing systems, the multiplexer is the device that accepts input from one or more users, converts the data streams to analog signals using either fixed or dynamically assigned frequencies, and transmits the combined analog signals over a medium that has a wide enough bandwidth to support the total range of all the assigned frequencies. A second multiplexer, or demultiplexer, is attached to the receiving end of the medium and splits off each signal, delivering it to the appropriate receiver. Figure 5-1 shows a simplified diagram of frequency division multiplexing.

![Simplified example of frequency division multiplexing](image)

To keep one signal from interfering with another signal, a set of unused frequencies called a guard band is usually inserted between the two signals, to provide a form of insulation. These guard bands take up frequencies which might be used for other data channels, thus introducing a certain level of wastefulness. This wastefulness is much like that produced in static assignment systems when a user that has been assigned to a channel does not transmit data, and is therefore considered to be an inefficiency in the FDM technique. In an effort to improve upon these deficiencies, another form of multiplexing—time division multiplexing—was developed.

**Time Division Multiplexing**

Frequency division multiplexing takes the available bandwidth on a medium and divides the frequencies among multiple channels, or users. Essentially, this division enables multiple users to transmit at the same time. In contrast, time division multiplexing
time division multiplexing (TDM) allows only one user at a time to transmit, and the sharing of the medium is accomplished by dividing available transmission time among users. Here, a user uses the entire bandwidth of the channel, but only for a brief moment.

How does time division multiplexing work? Suppose an instructor in a classroom poses a controversial question to students. In response, a number of hands shoot up, and the instructor calls on each student, one at a time. It is the instructor’s responsibility to make sure that only one student talks at any given moment, so that each individual’s response is heard. In a relatively crude way, the instructor is a time division multiplexor, giving each user (student) a moment in time to transmit data (express an opinion to the rest of the class). In a similar fashion, a time division multiplexor calls on one input device after another, giving each device a turn at transmitting its data over a high-speed line. Suppose two users, A and B, wish to transmit data over a shared medium to a distant computer. We can create a rather simple time division multiplexing scheme by allowing user A to transmit during the first second, then user B during the following second, followed again by user A during the third second, and so on. Since time division multiplexing was introduced (in the 1960s), it has split into two roughly parallel but separate technologies: synchronous time division multiplexing and statistical time division multiplexing.

Synchronous time division multiplexing

Synchronous time division multiplexing (Sync TDM) gives each incoming source signal a turn to be transmitted, proceeding through the sources in round-robin fashion. Given \( n \) inputs, a synchronous time division multiplexor accepts one piece of data, such as a byte, from the first device, transmits it over a high-speed link, accepts one byte from the second device, transmits it over the high-speed link, and continues this process until a byte is accepted from the \( n \)th device. After the \( n \)th device’s first byte is transmitted, the multiplexor returns to the first device and continues in round-robin fashion. Alternately, rather than accepting a byte at a time from each source, the multiplexor may accept single bits as the unit input from each device. Figure 5-2 shows an output stream produced by a synchronous time division multiplexor.

![Sample output stream generated by a synchronous time division multiplexor](image)

Note that the demultiplexor on the receiving end of the high-speed link must disassemble the incoming byte stream and deliver each byte to the appropriate destination. Since the high-speed output data stream generated by the multiplexor does not contain addressing information for individual bytes, a precise order must be maintained—this will allow the demultiplexor to disassemble and deliver the bytes to the respective owners in the same sequence as the bytes were input.

For a visual demonstration of synchronous time division multiplexing and statistical time division multiplexing, see the student online companion that accompanies this text.
Under normal circumstances, the synchronous time division multiplexor maintains a simple round-robin sampling order of the input devices, as depicted in Figure 5-2. What would happen if one input device sent data at a much faster rate than any of the others? An extensive buffer (such as a large section of random access memory) could hold the data from the faster device, but this buffer would provide only a temporary solution to the problem. A better solution is to sample the faster source multiple times during one round-robin pass. Figure 5-3 demonstrates how the input from device A is sampled twice for every one sample from the other input devices. As long as the demultiplexor understands this arrangement and this arrangement doesn’t change dynamically, there should, in theory, be no problems. In reality, however, there is one additional condition that must be met. This sampling technique will only work if the faster device is two, three, or four—an integer multiple—times faster than the other devices. If device A is, say, two and one-half times faster than the other devices, this technique will not work. In that case, device A’s input stream would have to be padded with additional “unusable” bytes to make its input stream seem a full three times faster than that of the other devices.

Figure 5-3

A synchronous time division multiplexing system that samples device A twice as fast as the other devices

What happens if a device has nothing to transmit? In this case, the multiplexor must still allocate a slot for that device in the high-speed output stream, but that time slot will, in essence, be empty. Since each time slot is statically fixed in synchronous time division multiplexing, the multiplexor cannot take advantage of the empty slot and reassign busy devices to it. If, for example, only one device is transmitting, the multiplexor must still go about sampling each input device (Figure 5-4). In addition, the high-speed link that connects the two multiplexors must always be capable of carrying the total of all possible incoming signals, even when none of the input sources is transmitting data.

Figure 5-4

Multiplexor transmission stream with only one input device transmitting data

As with a simple connection between one sending device and one receiving device, maintaining synchronization across a multiplexed link is important. To maintain synchronization between sending multiplexor and receiving demultiplexor, the data from the input sources is often packed into a simple frame, and synchronization bits are added somewhere within the frame (see Figure 5-5). Depending on the TDM technology used, anywhere from one bit to several bits can be added to a frame to provide synchronization. The synchronization bits act in a fashion similar to differential Manchester’s constantly changing signal—they provide a constantly reappearing bit sequence that the receiver can anticipate and lock onto.

Figure 5-5

Transmitted frame with added synchronization bits
Three types of synchronous time division multiplexing that are popular today are T-1 multiplexing, ISDN multiplexing, and SONET/SDH. Although the details of T-1, ISDN, and SONET/SDH are very technical, a brief examination of each technology will show how it multiplexes multiple channels of information together into a single stream of data.

**T-1 Multiplexing**

In the 1960s, AT&T created a service known as T-1, which multiplexed digital data and digitized voice onto a high-speed telephone line with a data rate of 1.544 megabits per second. The T-1’s original purpose was to provide a high-speed connection between AT&T’s switching centers. When businesses learned of this high-speed service, they began to request it to connect their computer and voice communications systems to the telephone network. In 1984, AT&T finally began offering this service to business customers.

In **T-1 multiplexing**, the frames of the T-1 multiplexor’s output stream are divided into 24 separate digitized voice/data channels of 64 kbps each (see Figure 5-6). Users who wish to use all 24 channels are using a full T-1, while other users who need to use only part of the 24 channels may request a fractional T-1. The T-1 multiplexed stream is a continuous repetition of frames. Each frame consists of 1 byte from each of the 24 channels (users) plus one synchronization bit. Thus, data from the first user is followed by the data from the second user, and so on, until data from the 24th user is once again followed by data from the first user. If one of the 24 input sources has no data to transmit, the space within the frame is still allocated to that input source. The input data from a maximum of 24 devices is assigned to fixed intervals. Each device can only transmit during that fixed interval. If a device has no significant data to transmit, the time slot is still assigned to that device, and data such as blanks or zeros is transmitted. The T-1 system is a classic application of synchronous time division multiplexing. Although non-TDM technologies such as frame relay and Asynchronous Transfer Mode (both of which will be discussed in Chapter Twelve) have grown in popularity, T-1 systems are still widely used.

*Figure 5-6*

**T-1 multiplexed data stream**

**ISDN Multiplexing**

Integrated Services Digital Network (ISDN) is a digital telephone service that provides voice and data transfer services over standard twisted pair wire to a home or small business. Although ISDN was designed to provide a number of services in addition to voice and data, data is the more popular use of most ISDN installations.

**ISDN multiplexing** is the synchronous time division multiplexing technique used to support ISDN; it comes in two basic forms: Primary Rate Interface (PRI) and Basic Rate Interface (BRI). PRI was designed for business applications and, like T-1, it multiplexes 24 input channels together onto one high-speed telephone line. BRI, the interface more often used by consumers to connect their home and small business computers to the Internet, multiplexes only three separate channels onto a single medium-speed telephone line. Two of the three channels—the B channels—carry either data or voice, while the third channel—the D channel—carries the signaling information that
controls the two data/voice channels. Since most consumers already have a standard telephone line, it is very common to use both of the B channels for data.

Figure 5-7 shows how the data from the two B channels (B1 and B2) plus the signaling information from the D channel are multiplexed together into a single frame. Note that 8 bits of data from the first B channel are followed by signaling control information, which is then followed by 8 bits of data from the second B channel and more signaling control information. These four groups of information repeat again to form a single frame.

Figure 5-7
ISDN frame layout showing B channel bits and signaling control information bits

SONET/SDH Multiplexing

Synchronous Optical Network (SONET) and Synchronous Digital Hierarchy (SDH) are very powerful standards for multiplexing data streams over a single medium. SONET (developed in the United States by ANSI) and SDH (developed in Europe by ITU-T) are two almost identical standards for the high-bandwidth transmission of a wide range of data types over fiber-optic cable. SONET and SDH have two features that are of particular interest in the context of multiplexing. First, they are both synchronous multiplexing techniques. A single clock controls the timing of all transmission and equipment across an entire SONET (or SDH) network. Using only a single clock to time all data transmissions yields a higher level of synchronization, because the system does not have to deal with two or more clocks having slightly different times. This high level of synchronization is necessary to achieve the high level of precision required when data is being transmitted at hundreds and thousands of megabits per second.

T-1 Multiplexing

T-1 communications lines are a popular technology for connecting businesses to high-speed sources such as Internet service providers and other wide area networks. Because T-1 multiplexing is a classic example of synchronous time division multiplexing, it merits further examination.

A T-1 telecommunications line uses a multiplexing technique termed DS-1 signaling, which provides for the multiplexing of up to 24 separate channels at a total speed of 1.544 Mbps. How does the T-1 line achieve the unique transmission speed of 1.544 Mbps? To answer this question, let’s consider an example in which the T-1 line supports the maximum 24 voice channels.

Since the average human voice occupies a relatively narrow range of frequencies (approximately 3000 to 4000 Hz), it is fairly simple to digitize voice. In fact, an analog-to-digital converter needs only 128 different quantization levels to achieve a fair digital
representation of the human voice. Since 128 equals $2^7$, each pulse code modulated voice sample can fit into a 7-bit value. Two hundred and fifty-six quantization levels would allow for an even more precise representation of the human voice. Since $256 = 2^8$, and eight bits is one byte, the telephone system uses 256 quantization levels to digitize the human voice. (If you need a refresher on this material, revisit pulse code modulation in Chapter Two.)

Recall that to create an accurate digital representation of an analog signal, you need to sample the analog signal at a rate that is twice the highest frequency. Given that the highest voice frequency is 4000 Hz, you need, when digitizing voice, to sample the analog voice signal 8000 times per second. Recall also, from Figure 5-6, the T-1 frame sequence. Since each T-1 frame contains 1 byte of voice data for 24 different channels, the system needs 8000 frames per second to maintain 24 simultaneous voice channels. Since each frame is 193 bits in length (24 channels x 8 bits per channel + 1 control bit = 193 bits), 8000 frames per second is multiplied by 193 bits per frame, which yields a rate of 1.544 Mbps.

T-1 can be used to transfer data as well as voice. If data is being transmitted, the 8-bit byte for each channel is broken into 7 bits of data and 1 bit of control information. Seven data bits per frame x 8000 frames per second = 56,000 bits per second per channel. Thus, when used for data, each of the 24 T-1 channels is capable of supporting a 56-kbps connection.

Second, SONET and SDH are able to multiplex varying speed streams of data onto one fiber connection. SONET defines a hierarchy of signaling levels, or data transmission rates, called synchronous transport signals (STS). Each STS level supports a particular data rate, as shown in Table 5-2, and is supported by a physical specification called an optical carrier (OC). Note that the data rate of OC-3 is exactly three times the rate of OC-1; this relationship carries through the entire table of values. SONET is designed with this data rate relationship so that multiplexing signals is relatively straightforward. For example, it is relatively simple to multiplex three STS-1 signals into one STS-3 signal. Likewise, four STS-12 signals can be multiplexed into one STS-48 signal. The STS multiplexor in a SONET network can accept electrical signals from copper-based media, convert those electrical signals into light pulses, and then multiplex the various sources onto one high-speed stream.

[GEX: Please note that the formatting of the bottom half of Table 5-2 is off—the final horizontal line is misplaced. We weren't able to make adjustments to it in this document without creating other formatting irregularities. Please adjust the table. For your reference, the table is unchanged from the text's 3rd edition (ISBN 0-619-16035-7), where it appears on page 165.]

<table>
<thead>
<tr>
<th>Table 5-2</th>
<th>STS signaling levels, corresponding OC levels, and data rates</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STS Level</strong></td>
<td><strong>OC Specification</strong></td>
</tr>
<tr>
<td>STS-1</td>
<td>OC-1</td>
</tr>
</tbody>
</table>
Each SONET frame contains the data that is being transmitted plus a number of control bits, which are scattered throughout the frame. Figure 5-8 shows the frame layout for the STS-1 signaling level. The STS-1 signaling level supports 8000 frames per second, and each frame contains 810 bytes (6480 bits). 8000 frames per second times 6480 bits per frame yields 51,840,000 bits per second, which is the OC-1 data rate. The other STS signaling levels are similar except for the layout of data and the placement and quantity of control bits.

**Figure 5-8**

*SONET STS-1 frame layout*

SONET and SDH are used in numerous applications in which very high data transfer rates over fiber-optic lines are necessary. For example, two common users of SONET are the telephone company and companies that provide an Internet backbone service. Both telephone companies and Internet backbone providers have very high-speed transmission lines that span parts of the country and must transmit hundreds and thousands of millions of bits per second over long distances. Installing fiber-optic lines that support SONET transmission technology is one of the best ways to meet the demands of such challenging applications.

**Statistical time division multiplexing**

As you’ve seen in the preceding discussions, both frequency division multiplexing and synchronous time division multiplexing can waste unused transmission space. One solution to this problem is statistical time division multiplexing. Sometimes called asynchronous time division multiplexing, statistical time division multiplexing (Stat TDM) transmits data only from active users and does not transmit empty time slots. To transmit data only from active users, the multiplexor creates a more complex frame that contains
data only from those input sources that have something to send. For example, consider the following simplified scenario. If four stations, A, B, C, and D, are connected to a statistical multiplexor, but only stations A and C are currently transmitting, the statistical multiplexor transmits only the data from stations A and C, as shown in Figure 5-9. Note that at any moment, the number of stations transmitting can change from two to zero, one, three, or four. If that happens, the statistical multiplexor must create a new frame containing data from the currently transmitting stations.

Figure 5-9
Two stations out of four transmitting via a statistical multiplexor

Since only two of the four stations are transmitting, how does the demultiplexor on the receiving end recognize the correct recipients of the data? Some type of address must be included with each byte of data, to identify who sent the data and for whom it is intended (see Figure 5-10). The address can be as simple as a binary number that uniquely identifies the station that is transmitting. For example, if the multiplexor is connected to four stations, then the addresses can simply be 0, 1, 2, and 3 for stations A, B, C, and D. In binary, the values would be 00, 01, 10, and 11, respectively.

Figure 5-10
Sample address and data in a statistical multiplexor output stream

If the multiplexor transmits more than one byte of data at a time from each source, then an alternate form of address and data is required. To transmit pieces of data of variable sizes, a length field defining the length of the data block is included along with the address and data. This packet of address/length/data/address/length/data is shown in Figure 5-11.

Figure 5-11
Packets of address, length, and data fields in a statistical multiplexor output stream

Finally, the sequence of address/length/data/address/length/data... is packaged into a larger unit by the statistical multiplexor. This larger unit, shown in Figure 5-12, is a more realistic example than Figure 5-11 and looks much like the frame that is transmitted using a synchronous connection. The flags at the beginning and end delimit the beginning and end of the frame. The control field provides information that is used by the sending and receiving multiplexors to control the flow of data between them. Last, the frame check sequence (FCS) provides information that the receiving multiplexor can use to detect transmission errors within the frame.

Figure 5-12
Frame layout for the information packet transferred between statistical multiplexors
Wavelength Division Multiplexing

Although frequency division and time division are two very common multiplexing techniques, another multiplexing technique—wavelength division multiplexing—emerged several years ago and has since become a powerful alternative. When transmission systems employing fiber-optic cable were first installed (in the 1980s), the explosive growth of the Internet and other data transmission networks had not even been imagined. Now that the twenty-first century has begun, it is painfully obvious that early growth forecasts were gross underestimates. With Internet access growing by more than 100 percent per year and individuals requesting multiple telephone lines for faxes and modems, video transmissions, and teleconferencing, a single fiber-optic line transmitting billions of bits per second is simply no longer sufficient. This inability of a single fiber-optic line to meet users’ needs is called fiber exhaust. For many years, technology specialists saw few ways to resolve fiber exhaust other than by installing additional fiber lines, sometimes at great expense. Now there appears to be an attractive solution that takes advantage of currently installed fiber-optic lines—wavelength division multiplexing.

Wavelength division multiplexing (WDM) multiplexes multiple data streams onto a single fiber-optic line. It is, in essence, a frequency division multiplexing technique that assigns input sources to separate sets of frequencies. Wavelength division multiplexing uses different wavelength (frequency) lasers to transmit multiple signals at the same time over a single medium. The wavelength of each differently colored laser is called the lambda. Thus, WDM supports multiple lambdas.

The technique assigns a uniquely colored laser to each input source and combines the multiple optical signals of the input sources so that they can be amplified as a group and transported over a single fiber. It is interesting to note that because of the properties of the signals and glass fiber, plus the nature of light itself, each signal carried on the fiber can be transmitted at a different rate from the other signals. This means that a single fiber-optic line can support simultaneous transmission speeds such as 51.84 Mbps, 155.52 Mbps, 622.08 Mbps, and 2.488 Gbps (which, incidentally, are multiples of T-1 speeds and are defined as OC-1, OC-3, OC-12, and OC-48, the optical carrier specifications for high-speed fiber-optic lines). In addition, a single fiber-optic line can support a number of different transmission formats such as SONET, Asynchronous Transfer Mode (ATM), and others, in various combinations (see Figure 5-13).

Figure 5-13
Fiber-optic line using wavelength division multiplexing and supporting multiple-speed transmissions

Wavelength division multiplexing is also scalable. As the demands on a system and its applications grow, it is possible to add additional wavelengths, or lambdas, onto the fiber, thus further multiplying the overall capacity of the original fiber-optic system. Most systems support less than 100 lambdas, while some of the ultra high-priced systems can handle more than 100 lambdas. When WDM can support a large number of lambdas, it is often called dense wavelength division multiplexing (DWDM). This additional power does not come without a price tag, however. Dense wavelength division multiplexing is an expensive way to transmit signals from multiple devices due to the high number of differently colored lasers required in one unit. One less expensive variation on dense
wavelength division multiplexing is coarse wavelength division multiplexing. Coarse wavelength division multiplexing (CWDM) is a less expensive technology because it is designed for short-distance connections and has only a few lambdas, with a greater space between lambdas. Because the wavelengths are farther apart and not packed as closely together as they are in DWDM, the lasers used for coarse wavelength division multiplexing can be less expensive and do not require extensive cooling. Despite its cost and complexity, many technology experts predict that wavelength division multiplexing will remain a powerful technology.

While wavelength division multiplexing is relatively uncommon and expensive, the next type of multiplexing that we will examine is very common and relatively inexpensive. Discrete multitone, the technology behind digital subscriber line, is no less impressive and our next subject.

**Discrete Multitone**

Discrete multitone (DMT) is a multiplexing technique commonly found in digital subscriber line (DSL) systems. DSL, as we have already seen, is a technology that allows a high-speed data signal to traverse a standard copper-based telephone line. We have also seen that the highest transmission speed we can achieve with a standard dial-up telephone line is 56 kbps. DSL, however, is capable of achieving speeds into the millions of bits per second. How is this possible? The answer is the multiplexing technique DMT. DMT essentially combines hundreds of different signals, or subchannels, into one stream; unlike the previous multiplexing techniques, however, all these subchannels are destined for a single user.

**Additional Multiplexing Techniques**

A number of new multiplexing techniques have appeared in the last several years, that are all interesting and might have great promise. Three of these multiplexing techniques are Optical Spatial Division Multiplexing (OSDM), Orthogonal Frequency Division Multiplexing (OFDM), and Optical Time Division Multiplexing (OTDM). The first, Optical Spatial Division Multiplexing, allows for the multiplexing of “bursty” traffic (that is, traffic that comes in bursts and is produced by numerous voice and Internet data sources) onto an optical transmission technology that has not supported this kind of traffic well in the past. An example of one such technology is SONET. Since most, if not all, telephone companies use SONET somewhere in their high-speed backbone networks, the use of OSDM creates systems that can carry more traffic and perhaps even provide it at a lower cost.
A second multiplexing technique is Orthogonal Frequency Division Multiplexing. OFDM is a discrete multitone technology (used in DSL systems) that combines multiple signals of different frequencies into a single, more complex signal. Before the multiple signals are combined, each is individually phase-modulated. The phase-modulated signals are then combined to create a compact, high-speed data stream. OFDM is used in applications such as wireless local area networks, digital television, digital radio, and home AC power-line transmissions.

The third multiplexing technique is Optical Time Division Multiplexing. OTDM is similar to wavelength division multiplexing, in that fiber-optic cables are used extensively. But where wavelength division multiplexing is a form of frequency division multiplexing, OTDM (as its name implies) is a form of time division multiplexing. An OTDM multiplexor combines the data from each input source into a high-speed time multiplexed stream. In the better systems, all input and output streams are optical, and the data, instead of changing to electrical form, remains in optical form throughout the multiplexing and demultiplexing phases. These all-optical systems are extremely fast (with speeds in the terabits-per-second range) and hold great promise for future applications.

The real power of DMT is the fact that each of the subchannels can perform its own quadrature amplitude modulation (QAM). (Recall from Chapter Two that QAM is a modulation technique that involves a four-bit code in which eight phase angles have a single amplitude, and four phase angles have double amplitudes.) For example, one form of DMT supports 256 subchannels, each of which is capable of a 60-kbps QAM modulated stream (Figure 5-14). Thus, 256 x 60 kbps yields a 15.36-million-bps system. Unfortunately, because of noise, not all 256 subchannels can transmit at a full 60-kbps rate. Those subchannels experiencing noise will modify their modulation technique and drop back to a slower speed. Thus, DSL systems that transmit data in the hundreds of thousands of bits per second are more the norm.

Imagine one technology that can support 256 independently modulated streams, many of them transmitting at different speeds. DMT is certainly a fascinating technology that have been developed in the quest to increase data transmission speeds for the average consumer. Let’s turn our attention to another multiplexing technique that is also pushing the limits of technology: code division multiplexing.
Also known as code division multiple access, code division multiplexing (CDM) is a relatively new technology that has been used extensively by both the military and cellular telephone companies. Whereas other multiplexing techniques differentiate one user from another by either assigning frequency ranges or interleaving bit sequences in time, code division multiplexing allows multiple users to share a common set of frequencies by assigning a unique digital code to each user.

More precisely, code division multiplexing is based upon spread spectrum technology (Chapter Two), which falls into two categories—frequency hopping and direct sequence. Code division multiplexing uses direct sequence spread spectrum technology, a technique that spreads the transmission of a signal over a wide range of frequencies, using mathematical values. As the original data is input into a direct sequence modulator, each binary 1 and 0 is replaced with a larger, unique bit sequence. For example, each device in a cell phone market that uses code division multiplexing to transmit its signal is assigned its own bit sequence. When the bit sequences arrive at the destination station, the code division multiplexer is capable of telling one mobile device’s bit sequence from another’s.

Despite the fact that this is a fairly complex procedure, code division multiplexing is one of the more fascinating technologies in data communications, and it merits a little closer examination. Let’s create an example using three mobile users: A, B, and C. Suppose mobile user A has been assigned the binary code 10010101, mobile user B the code 11100011, and mobile user C the code 00110011. These binary codes are called the chip spreading codes. In the real world, these codes are 64 bits in length. To keep our example simple, we’ll use 8-bit codes. If mobile user A wishes to transmit a binary 1, it transmits instead its code—10010101. If mobile user A wishes to transmit a binary 0, it transmits the inverse of its code—01101010. Actually, the mobile user transmits a series of positive and negative voltages—a positive voltage for a 1 and a negative voltage for a 0. For example, let’s say mobile user A transmits a binary 1, mobile user B transmits a binary 0, and mobile user C transmits a binary 1. The following is actually transmitted:

Mobile user A sends a binary 1 (10010101), or +---++++
Mobile user B sends a binary 0 (00011100), or ---+++--
Mobile user C sends a binary 1 (00110011), or --++-+++

The receiver receives all three signals at the same time and adds the voltages as shown below:

\[
\begin{align*}
+ & - - + - + - + \\
- & - - + + + - \\
- & + + - - + + \\
\end{align*}
\]

Sums:
-1 -3 -1 +3 -1 +1 -1 +1

Then, to determine what each mobile user transmitted, the receiver multiplies the sums by the original code of each mobile user, expressed as + and - values, then takes the sum of those products:

Sums: -1 -3 -1 +3 -1 +1 -1 +1
Mobile user A’s code: 

+1 -1 -1 +1 -1 +1 +1

Products: 

-1 +3 +1 +1 +1 +1 +1

Sum of Products: 

+10

Since the Sum of Products is greater than or equal to +8 (>= +8) in this 8-bit example, the value transmitted must have been a binary 1. In the real world, with the 64-bit system, the Sum of Products would have to be greater than or equal to +64 (>= +64). If the Sum of Products were <= -8 (or <= -64 using real codes), the value transmitted would have been a binary 0.

The same procedure would be performed to determine mobile user B’s transmitted value:

Sums: 

-1 -3 -1 +3 -1 -1 +1

Mobile user B’s code: 

+1 +1 +1 -1 -1 +1 +1

Products: 

-1 -3 -1 -3 +1 -1 +1

Sum of Products: 

-8

Since the Sum of Products is <= -8, the value transmitted must have been a binary 0.

Using a 64-bit code, it is theoretically possible to support $2^{64}$ (18,446,744,073,709,551,616) cellular telephones in the same metropolitan area at the same time (actually the value is one half of this, since transmitted binary 0s use the inverse codes). In reality, this number is not achievable, but as we can see, code division multiplexing is a complex yet powerful multiplexing technique. Techniques such as this one will allow data communications systems to grow in response to an ever-increasing demand for communications services.

Now that we’ve examined how the various multiplexing techniques work, let’s compare their advantages and disadvantages.

**Comparison of Multiplexing Techniques**

Frequency division multiplexing suffers from two major disadvantages. The first disadvantage is found in computer-based systems that multiplex multiple channels over a single medium. Since the frequencies are usually statically assigned, devices that do not have anything to transmit are still assigned frequencies, and thus bandwidth is wasted.

The second disadvantage of frequency division multiplexing is due to the fact that the technique uses analog signals, and analog signals are more susceptible to noise disruption than digital signals. Nonetheless, because of its simplicity, many different types of applications (such as television and radio) use frequency division multiplexing, and the technique is probably going to be with us for a long time.

Synchronous time division multiplexing is also relatively straightforward, but as in frequency division multiplexing, input devices that have nothing to transmit can waste transmission space. The big advantage of synchronous TDM over frequency division multiplexing is the
lower noise due to the use of digital signals during transmission. Statistical TDM is one variation of synchronous TDM that transmits data only from those input devices that have data to transmit. Thus, statistical TDM wastes less bandwidth on the transmission link.

Statistical multiplexors have another very good advantage over synchronous time division multiplexors. Although both types of time division multiplexing can transmit data over a high-speed link, statistical time division multiplexing does not require as high-speed a line as synchronous time division multiplexing does. Statistical time division multiplexing assumes that all devices do not transmit at the same time; therefore, it does not require a high-speed link that is the total of all the incoming data streams. Another consequence of this assumption is that the output line capacity coming from the statistical multiplexor can be less than the output line capacity from the synchronous multiplexor, which also allows for a slower-speed link between multiplexors. This slower-speed link usually translates into lower costs.

One disadvantage of statistical multiplexors is their increased level of complexity. Synchronous TDM simply accepts the data from each attached device and transmits that data in an unending cycle. The statistical multiplexor must collect and buffer data from active attached devices and, after creating a frame with necessary control information, transmit that frame to the receiving multiplexor. Although this slightly higher level of complexity translates into higher initial costs, those costs are usually offset by the statistical TDM’s ability to use a smaller-capacity interconnecting line.

Statistical time division multiplexing is a good choice for connecting a number of lower-speed devices that do not transmit data on a continuous basis to a remote computer system. Examples of these systems include data-entry systems, point-of-sale systems, and many other commercial applications in which users enter data at computer terminals.

Wavelength division multiplexing is a very good technique for transmitting multiple concurrent signals over a fiber-optic line. Wavelength division multiplexing is also scalable. As the demands on a system and its applications grow, more wavelengths, or lambdas, can be added onto the fiber, thus further multiplying the overall capacity of the original fiber-optic system. Wavelength division multiplexing systems that use a large number of lambdas are termed dense wavelength division multiplexing, while those systems that use only a few lambdas are termed coarse wavelength division multiplexing. While wavelength division multiplexing can be a costly alternative, it may be less expensive than trying to install additional fiber-optic lines.

Discrete multitone technology is a unique form of multiplexing in that all the subchannels multiplexed together are intended for one user. Thus, discrete multitone does not directly compare with the other multiplexing techniques, in which each subchannel or channel is destined for a different user. However, discrete multitone is a complex technology and can suffer greatly from too much noise.

Finally, code division multiplexing, while using a fairly wide bandwidth of frequencies and a complex technology, is scalable like WDM and can produce system capacities that are 8 to 10 times those of frequency division multiplexing systems.

The advantages and disadvantages of each multiplexing technique are summarized in Table 5-3.

[GEX: Please note that the formatting of the bottom half of Table 5-3 is off—the final horizontal line is misplaced. We weren’t able to make adjustments to it in this document without creating other formatting irregularities. Please adjust the table. For your reference, the table is unchanged from the text’s 3rd edition (ISBN 0-619-16035-7), where it appears on page 174.]
Table 5-3

Advantages and disadvantages of multiplexing techniques

<table>
<thead>
<tr>
<th>Multiplexing Technique</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Division Multiplexing</td>
<td>Simple</td>
<td>Noise problems due to</td>
</tr>
<tr>
<td></td>
<td>Popular with radio, TV, cable TV</td>
<td>analog signals</td>
</tr>
<tr>
<td></td>
<td>All the receivers, such as cellular</td>
<td>Wastes bandwidth</td>
</tr>
<tr>
<td></td>
<td>telephones, do not need to be at</td>
<td>Limited by frequency</td>
</tr>
<tr>
<td></td>
<td>the same location</td>
<td>ranges</td>
</tr>
<tr>
<td>Synchronous Time Division Multiplexing</td>
<td>Digital signals</td>
<td>Wastes bandwidth</td>
</tr>
<tr>
<td></td>
<td>Relatively simple</td>
<td></td>
</tr>
<tr>
<td>Statistical Time Division Multiplexing</td>
<td>More efficient use of bandwidth</td>
<td>More complex than</td>
</tr>
<tr>
<td></td>
<td>Frame can contain control</td>
<td>synchronous time</td>
</tr>
<tr>
<td></td>
<td>and error information</td>
<td>division multiplexing</td>
</tr>
<tr>
<td></td>
<td>Packets can be of varying size</td>
<td></td>
</tr>
<tr>
<td>Wavelength Division Multiplexing</td>
<td>Very high capacities over fiber</td>
<td>Cost</td>
</tr>
<tr>
<td></td>
<td>Signals can have varying speeds</td>
<td>Complexity</td>
</tr>
<tr>
<td>Discrete multitone</td>
<td>Capable of high transmission speeds</td>
<td>Complexity, noise</td>
</tr>
<tr>
<td></td>
<td></td>
<td>problems</td>
</tr>
<tr>
<td>Code Division Multiplexing</td>
<td>Large capacities</td>
<td>Complexity</td>
</tr>
<tr>
<td></td>
<td>Scalable</td>
<td>Primarily a wireless technology</td>
</tr>
</tbody>
</table>

So far, with multiplexing, we have examined how multiple data streams can be variously combined to maximize the number of them that can be transmitted through different types of media, thus yielding a more efficient connection. Let’s now examine another technique that can maximize the amount of data transmitted at a time or stored in a given space—the process is called compression.

Compression - Lossless versus Lossy
As we have already seen, **compression** is the process of taking data and somehow packing more of it into the same space, whether this is in the form of a storage device such as a hard drive or iPod or a medium such as a fiber-optic line. (When data is compressed for transmission, the data transfers more quickly, since there is less of it, which can lead to a more efficient connection.) The basic way to do this is to look for some common pattern in the data and replace each data pattern with a symbol or symbols that will consume less space during transmission or storage. For example, if a document contains a large number of occurrences of the word *snow*, the sender might want to replace the word *snow* with a symbol such as a percent sign, %. After the data is transmitted, the receiver then replaces the symbol % with the original word, *snow*. This replacement immediately raises two questions: How does the receiver know to replace the symbol % with *snow*? What happens if a percent sign (%) actually appears in the document as a percent sign? We don’t want the receiver replacing valid percent signs with the word *snow*. As we look at real examples of compression, you will see how these questions, and more like them, are addressed.

Before we examine some actual compression techniques, however, we should divide the compression process into two categories. If a compression technique compresses data and then de-compresses it back into the original data, then it is referred to as a lossless technique. With a **lossless compression** technique, no data is lost due to compression. If a compression technique does lose some of the data as a result of the compression process, then it is referred to as a **lossy compression** technique. Consider as an example a bank that wishes to compress all of its customer accounts in order to [Insert what might be the hypothetical purpose of compressing this data?]. Given the disaster that would ensue if the customer accounts were to lose data due to compression, the bank would obviously want to use a lossless compression technique to perform this task. On the other hand, if you wanted to take a song off a compact disc and copy it to an iPod, you would first need to compress the song. During the compression process, some of the data got lost, you might not even notice the loss. Especially if the compression algorithm “lost” only those sounds which most human ears are not likely to detect. Lossy compression algorithms are often used to compress music and video files, and thus are commonly used in technologies such as portable digital music devices. To investigate the process of compression in more detail, let’s start by examining the lossless techniques.

**Lossless Compression**

One of the more common and simpler examples of lossless compression is **run-length encoding**. This technique replaces any repetitions of the same bit or byte that occur in a sequence of data with a single occurrence of the bit/byte and a run count, or simply with a run count. For example, this technique works at the binary level by counting either long strings (or runs) of binary 0s or long strings of binary 1s. Let’s consider the following data string which is composed predominantly of binary 0s:

```
000001000000000110000000000000001000011000000000000000000001000000
```

A compression technique based on run-length encoding would compress the 0s by first counting the “runs” of 0s—that is, it would start by counting the 0s until a binary 1 is encountered. If there are no 0s between a pair of 1s, then that pair would be considered a run that contains zero 0s. Doing this to our data string, we find the following runs:
Thus, in the first run we encountered five 0s, while the second run had nine 0s. The third run had zero 0s because there was a 1 immediately following a 1. In the next run, we encountered fifteen 0s, followed by a run of four 0s, zero 0s, twenty 0s, and finally six 0s.

The next step in this compression technique would be to convert each of the decimal values (5, 9, 0, 15, and so on) into 4-bit binary values, or nibbles. The only unique rule to follow during this conversion comes into play when you encounter a decimal value of 15 or greater. Since the largest decimal number that a 4-bit binary nibble can represent is 15 (which corresponds to four binary 1s—1111), you must convert a run/decimal value that is greater than 15 into multiple 4-bit nibbles. For example, a run of 20 would be converted into 1111 1010, in which the first nibble is the value 15, and the second nibble is the value 5. A caveat to this rule is that if you are converting the value 15 itself, then you would also create two nibbles: 1111 followed by 0000. The reason for this is simply to be consistent—so that whenever a binary nibble of 1111 is encountered, the following nibble is added to the value of 15.

Thus, converting the above runs of 5, 9, 0, 15, 4, 0, 20, and 6, would produce the following nibbles:

0101  1001  0000  1111 0000  0100  0000  1111 0101  0110

In this example, note that the original bit string which consisted of 68 bits is compressed to 40 bits—which is a reduction of 42%—and that no data has been lost (hence, the name lossless). One disadvantage of this technique is that it is worthwhile only if the original data consists predominantly of binary 0s. As we will see a little later in this chapter, run-length encoding is used in compressing video images (due to many zero values), as well as compressing other documents that have repeated characters.

A second technique that can be used to compress data when a lossless compression is necessary is the Lempel-Ziv technique. This technique is quite popular and is used by programs such as pkzip, WinZip, gzip, UNIX compress, and Microsoft compress. While the actual algorithm is fairly complex, it is possible to get a basic understanding of how it works. As the string to be transmitted is processed, the sender of the data creates a “dictionary” of character strings and associated codes. This set of codes is transmitted and the receiver then recreates the dictionary and the original data string as the data codes are received.

The Lempel-Ziv algorithm can be fairly effective in compressing data. Studies have shown that computer program files can be reduced to 44% of the original size, text files can be reduced to 64% of the original size, and image files can be reduced to 88% of the original size.

Lossy Compression

All of the compression techniques described thus far have been examples of lossless compression. Lossless compression is necessary when the nature of the data is such that it is important that no data be lost during the compression / de-compression stages. Like program, text, and image files, video images and higher-quality audio files can also be compressed using lossless compression, but the percentage of reduction is usually not as significant. This is due to the nature of the data in video and audio files—there is not one symbol or set of symbols that occur frequently enough to produce a reasonable level of compression. For example, if you take some music and digitize it,
you will produce a long stream of binary 1s and 0s. You can either run-length encode the 1s, or the 0s. Unfortunately, there is usually not enough of either bit to produce a reasonable compression, thus some other compression technique is required.

Music and video have other properties, however, that can be tapped in order to perform compression. Consider first music. When one is listening to music, if two sounds play at the same time, the ear hears the louder one and usually ignores the softer one. Also, there are certain sounds at the extremes of the normal hearing range (recall from Chapter Two, that the average ear can actually hear sounds from 20 Hz to 20 kHz (20,000 Hz)) that the human ear cannot hear well or even at all. Audio engineers take advantage of these and other facts and use techniques called perceptual noise shaping, or perceptual encoding, to compress music. If the encoding is performed well, the compressed version of an audio stream sounds fairly close to the uncompressed version (that is, to CD-quality).

MP3, which is an abbreviation for MPEG (Moving Picture Experts Group) audio Layer-3, is a very common form of audio compression. (The Moving Picture Experts Group has developed compression standards for HDTV broadcasts, digital satellite systems (DSS), and DVD movies.) After employing these perceptual encoding tricks, the MP3 encoder produces a data stream that has a much slower rate than conventional CD-quality music. While a CD player is designed to reproduce music that has been encoded with 44,100 samples per second, which generates a data stream of 705,600 bits per second (44,100 samples per second times 16 bits per sample) or about 706 kbps, an MP3 encoder typically produces a data stream of 128 kbps to 192 kbps. This kind of reduction in data leads to a 10 to 1 compression ratio for a typical song.

Video files can also be compressed by removing small details in the image that the average human eye won’t notice are missing. JPEG, which stands for Joint Photographic Experts Group, is an [insert noun, e.g., algorithm, technique] that is very commonly used to compress video images. The process of converting an image to JPEG format involves three phases: discrete cosine transformation, quantization, and run-length encoding. To perform the discrete cosine transformation, the image is broken into multiple 8 by 8 blocks of pixels, where each pixel represents either a single dot of color in a color image or a single shade of black and white in a black and white image. Each 8 by 8 block (64 pixels) is then subjected to a fairly common/basic? mathematical routine called the discrete cosine transformation. Essentially, what this transformation does is produce a new 8 by 8 block of values. These values, however, are now called spatial frequencies, which are cosine calculations of how much each pixel value changes as a function of its position in the block. Rather than deal with the mathematics of this process, let's examine two fairly simple examples. If we have an image with fairly uniform color changes over the area of the image—in other words, not a lot of fine details, then one of its 8 by 8 blocks of pixels might look something like the following:

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>18</td>
<td>21</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
</tr>
<tr>
<td>19</td>
<td>22</td>
<td>25</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
<td>44</td>
</tr>
<tr>
<td>22</td>
<td>25</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
<td>44</td>
<td>48</td>
</tr>
<tr>
<td>26</td>
<td>29</td>
<td>32</td>
<td>35</td>
<td>39</td>
<td>43</td>
<td>47</td>
<td>51</td>
</tr>
</tbody>
</table>
where each decimal value represents a particular level of color.

After applying the discrete cosine transformation to these pixels, we would then have a set of spatial frequencies such as the following:

\[
\begin{array}{cccccccc}
628 & -123 & 12 & -8 & 0 & -2 & 0 & -1 \\
-185 & 23 & -5 & 0 & 0 & 0 & 0 & 0 \\
10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

Note how there are many zero entries, and the non-zero entries are clustered towards the upper-left corner of the block. This is because of the discrete cosine calculations, which in essence depicts the difference between one pixel’s color relative to that of another rather than the absolute value of a particular pixel’s color, and because this image, as noted earlier, is one with fairly uniform color—that is, not a lot of color variation.

Suppose, however, that we have an image that has lots of fine detail. It will have an 8 by 8 block of pixels that has widely different values and may look like the following:

\[
\begin{array}{cccccccc}
120 & 80 & 110 & 65 & 90 & 142 & 56 & 100 \\
40 & 136 & 93 & 188 & 90 & 210 & 220 & 56 \\
95 & 89 & 134 & 74 & 170 & 180 & 45 & 100 \\
9 & 110 & 145 & 93 & 221 & 194 & 83 & 110 \\
65 & 202 & 90 & 18 & 164 & 90 & 155 & 43 \\
\end{array}
\]
After applying the discrete cosine transformation to the pixels of this image, we would then have a set of spatial frequencies such as the following:

```
93  111  39  221  33  37  40  129
55  122  52  166  93  54  13  100
29  92  153  197  84  197  84  83
```

Notice how there are very few zero entries in this block of spatial frequencies.

The second phase in the conversion to JPEG is the quantization phase. The object of this phase is to try to generate more zero entries in the 8 by 8 block. To do this, we need to divide each value in the block by some pre-determined number and disregard the remainders. For example, if the pixel block contains a spatial frequency with the value 9, and we would divide this by 10 to get the result of 0. But we don’t want to divide all 64 spatial frequencies by the same value, because the values in the upper left hand corner of the block have more importance (due to the cosine transformation operation). So let’s divide the block of spatial frequencies with a block of values (shown below) in which the upper-left-hand corner of values are closer to 1 (thus reproducing the original number), such as the following:

```
1   4   7  10  13  16  19  22
4   7  10  13  16  19  22  25
7  10  13  16  19  22  25  28
10  13  16  19  22  25  28  31
```
Now when we divide the block of spatial frequencies with this block of weighted values, we should produce a new block of values with more zero entries.

<table>
<thead>
<tr>
<th>13</th>
<th>16</th>
<th>19</th>
<th>22</th>
<th>25</th>
<th>28</th>
<th>31</th>
<th>33</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>19</td>
<td>22</td>
<td>25</td>
<td>28</td>
<td>31</td>
<td>33</td>
<td>36</td>
</tr>
<tr>
<td>19</td>
<td>22</td>
<td>25</td>
<td>28</td>
<td>31</td>
<td>34</td>
<td>37</td>
<td>40</td>
</tr>
<tr>
<td>22</td>
<td>25</td>
<td>28</td>
<td>31</td>
<td>34</td>
<td>37</td>
<td>40</td>
<td>43</td>
</tr>
</tbody>
</table>

A question you should ask at this point is, if we perform 64 divisions and toss out the remainders, won’t we be losing something from the original image? The answer is yes, we will. But we hope to select an optimal set of values such that we do not lose too much of the original image. In other words, in striving to maximize the number of zeros in each block (so that we may successfully perform the run-length encoding of the final phase), we allow the data—i.e., the image—to change a little bit, but hopefully not so much that the human eye might detect gross differences between the original file and the one that has been compressed and de-compressed.

Finally, the third phase of the JPEG compression technique is to take the matrix of quantized values and perform run-length encoding on the zeros. But the trick here is that you don’t just run-length encode the zeros by going up and down the rows of the 8 by 8 block. Note that we would achieve longer runs of zeros if we encode on a diagonal, as is shown in Figure 5.15.

Figure 5-15

Run-length encoding of a JPEG image

What about moving video images, such as when watching digital television or a DVD? Is there a unique characteristic of this type of data that we can exploit in order to do compression? As it turns out, there is. A video is actually a series of images. When these images, or frames, are shown in rapid succession, it appears as if the objects in the video are moving. In order to make it seem as though the
movement (of characters, objects, scenery) in a movie is fluid, a movie projection device or television displays these images or frames at a rate of approximately 30 frames per second. But there is an interesting thing about these frames—which you might have noticed if you ever tried to create a cartoon by drawing images on multiple sheets of paper and then flipping through the pages. Unless there is a complete scene change, one image looks very similar to the next. In the context of compression, the question to consider is: If successive images are very similar, why transfer the full image of each frame? Why not just transfer the difference between the two frames? This sort of transfer is an example of differential encoding. MPEG-1 and MPEG-2 (or simply MPEG) are common examples of this form of compression.

Recall that a video display device displays multiple (typically 30) frames per second. In order to save space, not all of those 30 frames are complete images. MPEG actually creates a complete frame of information, followed by several partial, or difference frames, followed by a complete frame. More precisely, the following frames are created:

\[ I \ B \ B \ P \ B \ B \ I \ B \ B \ P \ldots \]

where the I-frame is a complete frame, the P-frame is the difference from the previous I frame (and is created using motion-compensated prediction), and the B-frames are the difference frames which contain the smaller differences between the I-frame and the P-frame and are inserted between I and P frames to smooth out the motion.

Since MPEG is computationally complex, processor chips have been designed, such as Intel’s MMX Technology, specifically for the compression and de-compression of MPEG images.

Business Multiplexing In Action

XYZ Corporation has two buildings, A and B, separated by a distance of 300 meters (see Figure 5-16). A 3-inch diameter tunnel runs underground between the two buildings. Building B contains 66 text-based terminals that need to be connected to a mainframe computer in Building A. The text-based terminals transmit data at 9600 bits per second. What are some good ways of connecting the terminals in Building B to the mainframe computer in Building A? Are there any that maximize the throughput through the connection?

Figure 5-16

Buildings A and B and the 3-inch-diameter tunnel connecting the buildings

Considering the technologies that have been introduced in the text thus far, there are four possible scenarios for connecting the terminals and mainframe computer:

1. Connect each terminal to the mainframe computer using separate point-to-point lines. Each line will be some form of conducted medium.

2. Connect all the terminals to the mainframe computer using one multipoint line. This one line will be some form of conducted medium.
3. Collect all the terminal outputs, and use microwave transmissions, free space optics, or Wi-MAX to send the data to the mainframe computer.

4. Collect all the terminal outputs using multiplexing, and send the data to the mainframe computer over a conducted-medium line.

Let’s examine the pros and cons of each solution.

The first solution of connecting each terminal to the mainframe computer using a point-to-point conducted medium has some advantages but also some serious drawbacks. The distance of 300 meters poses an immediate problem. When transmitting data in millions of bits per second, twisted pair typically has a maximum distance of 100 meters. XYZ Corporation’s data is not being transmitted at millions of bits per second, but at 9600 bits per second instead. At this slower rate, we may be able to successfully transmit across a distance farther than 100 meters, but doing so may not be a good idea. Electromagnetic noise is always a potential problem, and we may discover after the installation of the wires that there is too much noise. A more noise-resistant medium, such as coaxial cable or fiber-optic cable, might be a reasonable option. But 66 coaxial cables (one for each terminal) will probably not fit in a 3-inch-diameter tunnel. Fiber-optic cable has roughly the same dimensions as coaxial cable and a much higher cost if you factor in the 66 pairs of optical devices that would be needed at the ends of the cables. Even if we could run 66 wires of some medium through the tunnel, what if management decides, a month after the installation, to add 10 more terminals to Building B? Ten additional cables are not likely to fit through the tunnel, and if they were to fit, pulling 10 more cables through would be time-consuming and expensive.

The main advantages of the first solution include the lower cost of using a relatively inexpensive conducted medium, and the fact that multiple point-to-point lines eliminate the need for additional services such as polling or multiplexing.

The chief advantage of the second solution—using one multipoint line—is that it would require only a single wire running through the tunnel. There are two major problems, however. First, it may not be possible to create or purchase a system that polls 66 devices fast enough to maintain 9600-bps transmission on each terminal line. Second, even if you could poll 66 devices at 9600 bits per second, the solution is not scalable. We could not add more terminals in the future and still have the system work at an acceptable level.

The third solution, transmitting the data using microwave signals, free space optics, or Wi-MAX is interesting. All three technologies are very fast, and private ownership of [insert noun] between buildings can be attractive. The following concerns, however, are worth investigating:

- Is the line-of-sight between Building A and Building B unobstructed by trees or other buildings? If there is an obstruction, microwave and free space optics will not work.
- What is the cost of installing a microwave, free space optic, or Wi-MAX system between the two buildings? If the cost is high, there may be a more reasonable alternative.
- The system would still need some kind of device that collects data from the 66 terminals and prepares a single data stream for transmission. Will the system handle this collection, or will we need something like a multiplexor?
Thus, microwave and free space optics are possible solutions if there is a clear line-of-sight between the two buildings and the associated costs are not too high. Wi-MAX, however, is capable of penetrating buildings and does not pose the same line-of-sight problems. Unfortunately, Wi-MAX is too new a technology at the time of writing for us to consider further.

The fourth solution—to install multiplexors at each end of the tunnel and connect the multiplexors with some type of high-speed medium—also requires some forethought and investigation. The following issues are worth considering:

- Can one pair of multiplexors handle 66 terminals? If not, we may have to install two pairs of multiplexors.
- What does a pair of multiplexors cost? Will this cost be so high that we are forced to consider other alternatives?
- What kind of medium could we use to connect the multiplexors? How many wires would we need to run? Fiber-optic cable or even coaxial cable would be a good choice. Even if the system required multiple strands of fiber or coaxial cable, they would fit within the 3-inch-diameter tunnel since there should be much fewer than 66 sets of cables.
- Is the multiplexor solution scalable? Can the system expand to include additional terminals in the future? In the worst-case scenario, we would have to add an additional pair of multiplexors and another cable in the tunnel. We could plan ahead and pull several strands of fiber or coaxial cable through the tunnel so that we are prepared for future expansion.

In conclusion, it appears that a multiplexing scheme provides the most efficient use of a small number of cables running through the small tunnel. If a high-quality cable such as fiber-optic wire is used, it will minimize noise intrusion and allow for the greatest amount of future growth. The microwave/free space optic solution is also attractive, but may cost more than a pair of multiplexors and connecting cables. Wi-MAX might be a very interesting solution – one we will have to keep an eye on in the near future.

SUMMARY

- For multiple signals to share a single medium, the medium must be divided into multiple channels. There are three basic techniques for dividing a medium into multiple channels: a division of frequencies, a division of time, and a division of transmission codes.
- Frequency division multiplexing involves assigning non-overlapping frequency ranges to different signals. Frequency division multiplexing uses analog signals, while time division multiplexing uses digital signals.
- Time division multiplexing of a medium involves dividing the available transmission time on a medium among the users. Time division multiplexing has two basic forms: synchronous time division multiplexing and statistical time division multiplexing.
- Synchronous time division multiplexing accepts input from a fixed number of devices and transmits their data in an unending repetitious pattern. T-1, ISDN, and SONET/SDH telephone systems are common examples of systems that use synchronous time division
multiplexing. The static assignment of input devices to particular frequencies or time slots can be wasteful if the input devices are not constantly transmitting data.

- Statistical time division multiplexing accepts input from a set of devices that have data to transmit, creates a frame with data and control information, and transmits that frame. Input devices that do not have data to send are not included in the frame.

- Wavelength division multiplexing involves fiber-optic systems and the transfer of multiple streams of data over a single fiber using multiple, colored laser transmitters. Wavelength division multiplexing systems can be dense or coarse.

- Discrete multitone is a technology used in DSL systems. Multiple subchannels, each supporting a form of quadrature amplitude modulation, are multiplexed together to provide one data stream for a user.

- Code division multiplexing employs direct sequence spread spectrum concepts and allows multiple users to share the same set of frequencies by assigning a unique digital code to each user.

- Compression is a process that compacts data into a smaller package. When stored, compressed data saves space; when transmitted, it results in shorter transmission times.

- There are two basic forms of compression: lossless, in which no data is lost during the compression and de-compression stages; and lossy, in which some of the original data is lost.

- Two popular forms of lossless compression include run-length encoding and the Lempel-Ziv compression technique.

- Lossy compression is the basis of in a number of compression techniques, including MP3 for audio, JPEG for still images, and MPEG for moving video.

**KEY TERMS**

channel
chip spreading codes
course wavelength division multiplexing (CWDM)
code division multiplexing (CDM)
compression
demultiplexor
dense wavelength division multiplexing (DWDM)
discrete multitone (DMT)
DS-1 signaling
fiber exhaust
REVIEW QUESTIONS

1. List three common examples of frequency division multiplexing.

2. Frequency division multiplexing is associated with what type of signals?

3. In what order does synchronous time division multiplexing sample each of the incoming signals?

4. What would happen if a synchronous time division multiplexor sampled the incoming signals out of order?

5. How does a synchronous time division multiplexor stay synchronized with the demultiplexor on the receiving end?

6. How many separate channels does a T-1 multiplexor combine into one stream?

7. How many channels does Basic Rate Interface (BRI) ISDN support over a single connection?
8. What are the main differences between statistical time division multiplexing and synchronous time division multiplexing?

9. If a statistical multiplexer is connected to 20 devices, does it require a high-speed output line that is equivalent to the sum of the 20 transmission streams? Defend your response.

10. Why is addressing of the individual data streams necessary for statistical multiplexing?

11. What type of medium is required to support wavelength division multiplexing?

12. How many different wavelengths can dense wavelength division multiplexing place onto one connection?

13. What is the difference between dense wavelength division multiplexing and coarse wavelength division multiplexing?

14. How is discrete multitone different from the other multiplexing techniques? How is it similar?

15. How does code division multiplexing distinguish one signal from another?

16. What are the two basic forms of compression?

17. Run-length encoding can be used to compress what kind(s) of data?

18. What are the three phases of JPEG compression?

---

**EXERCISES**

1. Compared to the other multiplexing techniques, state two advantages and two disadvantages of each of the following:
   a. frequency division multiplexing
   b. synchronous time division multiplexing
   c. statistical time division multiplexing
   d. wavelength division multiplexing

2. A benefit of frequency division multiplexing and code division multiplexing is that all the receivers do not have to be at the same location. Explain the consequences of this benefit, and give an example.

3. Twenty-four voice signals are to be multiplexed and transmitted over twisted pair. What is the total bandwidth required if frequency division multiplexing is used?

4. Twenty voice signals are to be multiplexed and transmitted over twisted pair. What is the bandwidth required (in bps) if synchronous time division multiplexing is used, along with the standard analog-to-digital sampling rate, and each sample is converted into an 8-bit value?

5. If only four computers are transmitting digital data over a T-1 line, what is the maximum possible data rate for each computer?

6. What is the synchronization bit in a T-1 frame used for? Why is it necessary?
7. Ten computer workstations are connected to a synchronous time division multiplexor. Each workstation transmits at 128 kbps. At any point in time, 40 percent of the workstations are not transmitting. What is the minimum necessary speed of the line leaving the multiplexor? Will the answer be different if we use a statistical multiplexor instead? Explain your reasoning.

8. When data is transmitted using a statistical multiplexor, the individual units of data must have some form of address that tells the receiver the identity of the intended recipient of each piece of data. Instead of assigning absolute addresses to each piece of data, is it possible to incorporate relative addressing? If so, explain the benefits.

9. The telephone company has a fiber-optic line with time division multiplexing that runs from the United States to England and lies on the ocean floor. This fiber-optic line has reached capacity. What alternatives can the telephone company consider to increase capacity?

10. A discrete multitone system is using a modulation technique on its subchannels, each of which generates a 64-kbps stream. Assuming ideal conditions (no noise), what is the maximum data rate of the discrete multitone system?

11. The cell phone company in town uses code division multiplexing to transmit signals between its cell phones and the cell towers. You are using your cell phone while standing next to someone using her cell phone. How does the system distinguish the two signals?

12. Mobile user A is using code division multiplexing and has been assigned a binary code of 10010111. Mobile user B, also using code division multiplexing, has been assigned a binary code of 01001010. Mobile user A transmits a 1, while Mobile user B transmits a 0. Show the sum of products that results and your calculations.

13. A T-1 can transmit up to 1.544 Mbps, the same as primary rate ISDN. What are the key differences between the two technologies?

14. Why is wavelength division multiplexing more like frequency division multiplexing and less like time division multiplexing?

15. Which of the multiplexing techniques can be used on both conducted media and wireless media, which on only conducted media, and which on only wireless media?

16. In theory, code division multiplexing can have \(2^{64}\) different signals in the same area. In reality, this is not possible. Why not? Show an example.

17. Is the form of DSL that a company uses different from the form of DSL that a home user subscribes to? Explain.

18. If data has a large number of one type of symbol, which type of compression would be the most effective?

19. Given the following bit string, show the run-length encoding that would result:

   00000001000001100000000000000001000001110000000000


21. MP3, JPEG, and MPEG all rely on what characteristic in the data in order to perform compression?


THINKING OUTSIDE THE BOX
1. A company has two buildings that are 50 meters apart. Between the buildings is private land owned by the company. A large walk-through tunnel connects the two buildings. In one building is a collection of 30 computer workstations, and in the other building a mainframe computer. What is the best way to connect the workstations to the mainframe computer? Explain your reasoning and all the possible solutions you considered.

2. A company has two buildings that are 100 meters apart. Between the buildings is public land with no access tunnel. In one building is a collection of 30 computer workstations, and in the other building a mainframe computer. What is the best way to connect the workstations to the mainframe computer? Explain your reasoning and all the possible solutions you considered.

3. What is the relationship, if there is any, between synchronous time division multiplexing and the synchronous and asynchronous connections described in Chapter Four?

4. Compare and contrast the older multiplexing techniques such as frequency division and time division multiplexing with the newer techniques such as discrete multitone and orthogonal frequency division multiplexing. What appears to be the trend in these newer protocols?

5. You are receiving high-speed Internet access from a DSL provider. You also live down the street from a radio station’s AM broadcast antenna. Is your DSL affected by this antenna? Explain.

6. Consider a VGA screen which has 640 x 800 pixels per screen. Further assume that each pixel is 24-bits (8 for red, 8 for blue, and 8 for green). If a movie video presents 30 frames (images) per second, how many bytes will a 2-hour movie require for storage? How many bytes can a standard DVD hold? What then must be the compression ratio?

**HANDS-ON PROJECTS**

1. Locate advertising material that lists the maximum number of devices a frequency or time division multiplexor can handle. Is this number consistent with what was presented in the chapter?

2. Digital broadcast television will someday replace conventional analog television. What form of multiplexing is used to broadcast digital television signals?

3. Broadcast radio is one of the last forms of entertainment to go digital. Find the latest material describing the current state of digital broadcast radio, and write a two- or three-page report that includes the type of multiplexing envisioned, and the impact digital radio will have on the current radio market.

4. There is a set of frequencies created by the FCC for walkie-talkie radios. This set is called the family radio service and allows two radios to transmit up to a several-mile distance. What kind of multiplexing is used with these radios? How many concurrent channels are allowed? Is there a technology newer than family radio service? If so, describe its characteristics.
5. The local loop of the telephone circuit that enters your house uses multiplexing so that the people on the two ends of the connection can talk at the same time (if they wish). What kind of multiplexing is used? State the details of the multiplexing technique.

6. There are numerous forms of MPEG compression (such as MPEG-1, MPEG-2, etc.). List each of the forms with an accompanying sentence describing what type of data that compression form is designed for.

7. What other compression schemes exist besides those listed in this chapter? What is each compression scheme used for?