

Chapter 17

T Carrier

The number of simultaneous conversations being carried by telephone companies from place to place is so huge, that telephone companies have for years been working to concentrate, or multiplex, large numbers of voice circuits onto a small number of wires. Early methods used frequency division multiplexing and analog methods, but digital methods have been common for almost 50 years.

The first of these digital methods was “T Carrier”, a group of synchronous multiplexing methods that are still widely used in the U.S. telephone industry. They come in various versions which run at various speeds, ranging from T-1 at the slow end to T-4 at the high speed end. Corresponding to these is a set of methods called DS-1 through DS-4. These are related like this:

- The DS- notation describes the basic multiplexing scheme, which specifies the speed, the format of the data, and various other characteristics
- The T- notation is the actual physical implementation of the corresponding DS-*x* system in a wire or cable.

For example, T-1 is the actual cable connection that carries a DS-1 data stream.

In addition to these, the term DS-0 is used (but without a T-0 to go with it) to designate a single digitized voice circuit. These methods can be summarized as follows:

DS-0	64K bps	1 voice channel
DS-1 / T-1	1.544 Mbps	24 voice channels
DS-1C/T-1C	3.152 Mbps	48 voice channels
DS-2 / T-2	6.312 Mbps	96 voice channels
DS-3 / T-3	44.736 Mbps	672 voice channels
DS-3C/T-3C	90.524 Mbps	1344 voice channels
DS-4 / T-4	274.176 Mbps	4032 voice channels

(The -1C and -3C versions don’t quite fit into the overall scheme: the -1C speed is two -1 channels plus a bit of overhead, while the -3C is two -3 channels. The letter C stands for “Combined”.)

There is a similar set of methods called E-1 through E-4, that are used in Europe and various other countries; these can be summarized as follows:

E-1	2.048 Mbps	30 voice channels
E-2	8.448 Mbps	120 voice channels
E-3	34.368 Mbps	480 voice channels
E-4	139.264 Mbps	1920 voice channels

The sequence of DS-0, DS-1, DS-2, DS-3, and DS-4 (or the corresponding set of E- methods), which build one upon each other, makes a hierarchical system, and so it (and other systems like it) are often called a *data hierarchy*.

These methods are excellent examples of synchronous multiplexing, and show a variety of interesting techniques. DS-0 and DS-1/T-1 especially make a good introduction to synchronous multiplexing.

DS-0

DS-0 is the 64K bps digital signal which carries a single voice channel. Since it isn’t specifically implemented as a corresponding T-0 signal, the name DS-0 is more of a general term which people use to describe several slightly different ideas. Without calling it DS-0, we actually introduced the idea in two previous chapters:

- In Chapter 14, we said that analog telephone voice signals are sampled 8000 times per second, and encoded in 8-bits using μ -law encoding. The data rate of a DS-0 connection is therefore 8000×8 , or 64,000 bps. (Though the letter K usually means 1024 in digital computer circuits, in communications it generally means 1000. Hence 64,000 bps is also called 64K bps.) Although we did not specifically say so, the implication is that these 64K bits per second are evenly spaced, $1/64000$ second apart.
- In Chapter 16, on the other hand, we described the analog line card in a switch. We pointed out that the line card contains a codec, which outputs a 64K bps data stream, but in a series of bursts. Each 8-bit burst lasts about 5 microseconds, and then there is a wait of about 120 microseconds before the next burst. Outputting 8 bits every 125 microseconds (i.e., 8000 times per second) also makes a total of 64K bits per second.

While technically both of these occur at 64K bps and so can be called a DS-0 data stream, the latter are actually part of a DS-1 data stream, so let us look at that next.

DS-1 Frame

Whereas DS-0 is one voice channel (and thus 64K bits per second), DS-1 contains 24 voice channels, synchronously multiplexed together. The system was developed by AT&T in the late 1950’s as the first step to digitizing the telephone network.

Since each voice channel needs 8 bits every $1/8000$ second, in 24 voice channels there is a total of 8×24 , or 192 data bits every $1/8000$ second. This group of 192 bits, plus one extra bit that marks the beginning of the group, makes up a DS-1 frame. The total frame therefore con-

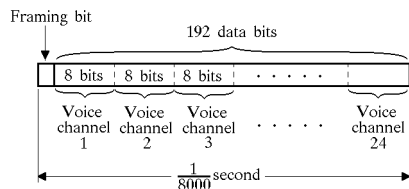


Fig. 17-1. One DS-1 Frame = 193 bits

sists of one *framing bit*, followed by 192 data bits, for a total of 193 bits per frame. This is shown in Fig. 17-1.

Since there are 8000 frames per second, there is a total of 8000×193 , or 1,544,000 bits per second.

Let's now look at one channel worth of bits. At 1.544 Mbps (megabits per second or 1.544 million bits per second), the time for each bit is

$$\frac{1}{1,544,000} = 0.65 \mu\text{sec}$$

and so the eight bits from one channel require $8 \times 0.65 \mu\text{sec}$, or about 5 microseconds. Since the entire frame lasts $1/8000$ second, which is 125 microseconds, that leaves about 120 microseconds between bursts. Just enough time to squeeze in 23 other voice channels plus a framing bit.

Now imagine that you are sitting by the side of a DS-1 data stream (or T-1 cable), watching 1,544,000 bits go by every second, looking for a particular voice channel. How do you identify the particular group of eight bits that correspond to your channel?

Simple — find the framing bit, and count from there. For example, the first voice channel is the first eight bits after a framing bit; the 13th voice channel is the 13th group of eight bits after a framing bit, and so on.

But how do you find the framing bit? After all, framing bits are just ONES and ZEROS, and so they look just like any other bit!

The D-1 Channel Bank

The very first DS-1 implementations used multiplexers called *D1 Channel Banks*. These multiplexers alternated the framing bits 1, 0, 1, 0, ... in a continuous repetition. The idea was that there would be a ONE framing bit, then exactly 193 bits later there would be a ZERO, then exactly 193 bits later another ONE, and so on.



Since these framing bits look just like any other bit, you would think that it would be almost impossible to

identify them. Yet finding such a pattern buried in the data is actually quite simple. Let's look at an example:

Suppose the frame length is 3 bits, not 193, and we have the following string of bits. Can you find the framing bits that are buried in it?

11010011100111010111010101110111100011000

Imagine that you have a *mask* — a piece of paper with cutout windows, through which you can only see every third bit. Position it on top of the line of numbers, like this (we'll show the hidden bits, but make them smaller):

1101001110011101011101010111011100011000

Hmmm ... not quite. Try it like this:

1101001110011101011101010111011100011000

That looks good! Just in case, try it this way:

1101001110011101011101010111011100011000

No — it looks like the second view was right.

The point is that it is possible to find the framing bits in the data if you look hard enough and long enough. But is it possible that the data *between* the framing bits also has the same pattern? For example, what if the data looked like this:

111000111000111000111000111000111000111

Now it would be quite impossible to tell exactly which bit is the framing bit because the pattern $1_{xx}0_{xx}1_{xx}0_{xx}$ appears in several different ways.

While it is possible that someone has purposely jury-rigged the data so it looks like framing bits, AT&T initially assumed that this is not likely to happen in real, meaningful data. Besides, they reasoned, once you do identify the real framing bits, it no longer matters what data is between them, because you've already found the framing bits, so you are no longer looking for them.

You can look at it as a probability problem. Suppose you have a large string of data bits that came from some unknown, and largely random process. What's the probability that the 193 bit after a ONE in this random data stream will be a ZERO? The probability is $1/2$, sort of like tossing a coin: the probability of a specific head or tail is $1/2$. What is the probability of having both this ZERO, and another ONE 193 bits later? It is $1/4$. Each time you get another framing bit, the probability of a random data string providing the right pattern to look like framing

bits goes down by a half. At the rate of 8000 framing bits per second, it doesn't take many seconds for the probability of being fooled by random data going very, very, very close to zero.

The problem with this reasoning is that it only applies to random data. Unfortunately, one of the test tones that was widely used at the time was a 1000 Hz sine wave. Sampling this tone 8000 times per second will (if the frequencies are exact) result in the tone always being sampled at exactly the same points. In that case, it is possible to get a repeating bit pattern which has ONES and ZEROS alternating every 193 bits, thus fooling the circuitry that looks for framing bits.

The solution back then was twofold: use 1004 Hz for testing instead of 1000 Hz, and modify the channel banks to use a different framing system with superframes and extended superframes, as we will see in a moment.



The D1 channel bank was used to carry 24 voice channels over one set of wires. To do that, however, it needed to also carry some signaling information. For example, consider a new building with 24 apartments. Rather than bring in 24 separate wire pairs for telephones, the telephone company might bring in one DS-1/T-1 line, and install a channel bank in the basement to split it up into 24 separate lines. They now needed some way to tell the central office switch when one of the telephones goes off hook, and also to tell the channel bank when to ring a phone.

Their solution was simple — instead of letting each channel use the full eight bits for digitized voice, use only seven bits for voice and take one (the least significant bit) and use it for signalling. The process of “stealing” one bit is called *bit robbing*.

Since bit robbing only leaves 7 actual data bits out of the 8, it means that the total number of usable data bits per second is only 56,000 rather than 64,000. When such a channel is sold to a customer for data, rather than voice applications, it only provides a 56K bps data channel.

When used for voice, the disadvantage of robbing a bit from every 8-bit sample in every frame was that this reduced the voice quality. This was acceptable for local calling, where only a small number of channel banks might be involved. But in long-distance calling, where the voice signal might be multiplexed and demultiplexed several times, the reduction in quality was serious. And so later model channel banks used a more sophisticated system to selectively rob only some

bits, by using *superframes* and later *extended superframes*.

The Superframe

The *superframe* was introduced with the D-2 channel bank; it is no longer designed into new equipment, but it still is used in plenty of older gear. It is a group of 12 frames, so it consists of 12×193 bits, or a total of 2316 bits. As before, it has a pattern that goes like this

- a framing bit
- 192 data bits, containing 24 channels
- a framing bit
- 192 data bits
- a framing bit
- 192 data bits

and so on, for a total of 12 framing bits, always separated by 192 data bits between them.

The difference is that the framing bits have a different pattern to them. Instead of being 101010101010, as in the older D-1 channel banks, the 12 framing bits within a D-2 superframe go like this:

100011011100

(and then the pattern repeats 100011011100 in the next frame again.) That is, the first framing bit in the superframe is always a 1; the second is always a 0; the third is also always a 0; and so on. To find the beginning of the frame, the receiver simply looks for this repeating pattern in the bits as they come streaming by.

Looking for a pattern of 100011011100 is not much different from looking for the 101010101010 pattern in a D1 channel bank. Just as before, there is, of course, the possibility that this same pattern might also occur in the data between the framing bits. But, although that may happen now and then, the chance of its happening continuously is zero. And since the receiver is continuously checking the framing bits, it is not likely to get fooled. Once it latches on to the correct pattern (which takes an average of 50 milliseconds, but may be longer), it expects to see that same pattern repeated over and over and over.

The advantage of using a 100011011100 pattern is that it allows the equipment to identify not just the beginning and end of a frame, but also to identify a specific frame out of a group of 12 in the superframe. This allows selective bit robbing in only every sixth frame, rather than *every* frame.

The normal method is to steal the least significant bit from each channel in the sixth frame and the twelfth frame of the superframe, as shown in Fig. 17-2. Each of the 24 voice channels therefore may have a slight error

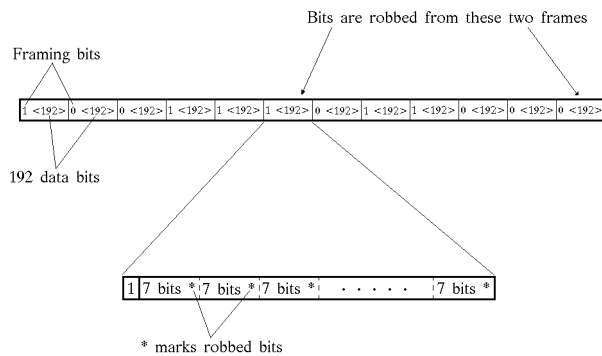


Fig. 17-2. Bit robbing in superframes

every six frames, or every $\frac{1}{8000}$ of a second. But this error is not audible because

- it is only in the least significant bit, and so represents only a tiny voltage error — one quantizing step,
- it only happens every sixth frame, and
- it is not continuous. On the average, half the time the replaced bit (representing signalling) happens to be the same as the 1 or 0 that would have been there in the voice data anyway, so in that case there is no error at all. Only if the replaced bit is different from the one that would have been there is there an error.

In each channel, the two stolen bits are called A and B: The A bit is the one stolen from frame 6, and B is the one stolen from frame 12. With two bits, the AB bits could have four different meanings in each direction.

Extended Superframes

A more recent, and better solution, was introduced with D-5 channel banks, which use *extended superframes* (ESF) rather than regular superframes (SF). Whereas a regular superframe is 12 frames, the extended superframe contains 24 frames. It therefore also has 24 framing bits.

Remember that the equipment recognizes the framing bits, as well as the beginning and end of a superframe, by looking at the framing bit pattern. As it turns out, it doesn't need to look at every single framing bit to recognize the framing bit pattern — being able to recognize just every fourth framing bit is good enough. In the extended superframe, it looks at framing bits 4, 8, 12, 16, 20, and 24. These six framing bits have the pattern 001011. That is, the 24 framing bits in the extended superframe look like

... 0 ... 0 ... 1 ... 0 ... 1 ... 1

where the 18 other framing bits are shown as dots. The remaining 18 framing bits are used as follows:

- The framing bits in frames 2, 6, 10, 14, 18, and 22 hold a 6-bit CRC (see Appendix C for an explanation of the CRC or Cyclic Redundancy Check). The CRC is used to check whether the *previous* extended superframe was received correctly.
- The remaining framing bits — those in the odd-numbered frames — add up to 4000 bits per second, and provide a 4K bps additional channel, called a Facility Data Link or FDL, which can be used by the equipment manufacturer for maintenance, testing, or other purposes.

If we use the letter C for CRC bits and F for this additional 4K bps FDL channel, the 24 framing bits in the extended superframe look like this:

FCF0FCF0FCF1FCF0FCF1FCF1

What about bit-robbing? Extended superframe also supports bit robbing in every sixth frame, so the least significant bit is again robbed, this time in frames 6, 12, 18, and 24 of every superframe. But in this case, the four bits robbed from each channel in each extended superframe are called A, B, C, and D, so these four bits could signify up to 2^4 or 16 different conditions.

These extra framing bits in extended superframe add quite a bit to the system. The CRC allows maintenance people to detect errors long before they become serious enough to disrupt operation, while the extra FDL bits provide other testing and maintenance options. In general, ESF systems are much more reliable than SF systems.

T-1 Facility

The term DS-1 describes the data pattern and general concept of the 24-voice-channel system. The term T-1 describes the actual physical connection that carries the DS-1 data.

The T-1 system was designed to use the same physical 24-gauge or 26-gauge unshielded twisted pair wire as is used for an analog subscriber loop connection. But it uses two pairs, one in each direction.

When used for an analog subscriber loop connection, long cables (over about 6000 feet long) generally use loading coils to improve the high frequency response. These loading coils are installed at 6000-foot intervals. When such a cable is used for a T-1 connection, the loading coils must be removed (since they prevent any high frequency digital data from getting through), and replaced by T-1 regenerators, which amplify and regenerate the T-1 digital data signal. So T-1 is designed

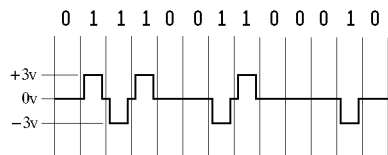


Fig. 17-3. AMI signal used in T-1 spans

to go up to 6000 feet in one hop; a single such hop is called a *span*.

These T-1 regenerators need electrical power to run. To avoid having to be plugged into a power outlet (which might be difficult to find on top of a telephone pole or in a manhole), the T-1 line often carries DC power in addition to the digital signal.

The T-1 line uses a digital AMI signal of ± 3 volts, as shown in Fig. 17-3. A ZERO is shown as 0 volts while ONES are shown as alternating +3 and -3 volt RZ (Return-to-Zero) pulses.

AMI stands for Alternate Mark Inversion (see Chapter 13 for a discussion of various digital signal formats.) The word “mark” means a ONE, so ONES alternate polarity. If one ONE is positive, the next ONE is supposed to be negative. If it ever occurs that two ONES are the same polarity, this is called a bipolar violation and is considered an error (except in one special case.)

The “Too Many ZEROS” problem

AMI is not what we called a “self-clocking” code in Chapter 13. That is, it does not have some sort of an edge or pulse for every bit, something which would make it easy for a signal receiver to synchronize its clock to the incoming data. Only the ONES have a pulse, the ZEROS don’t.

Nevertheless, a receiver can still synchronize its internal clock to an AMI signal as long as there are enough ONES to give the clock an occasional nudge to keep it in step. A long string of ZEROS, on the other hand, would cause the clock to fall out of step. Modern T-1 receivers are able to stay synchronized as long as there aren’t more than 15 ZEROS in a row.

When a T-1 line is used to carry 24 voice channels, this is not a problem. When discussing μ -law A-to-D conversion and compression in Chapter 14, we made a point of explaining that the converter would never output a code of 00000000 — eight ZEROS. The longest string of ZEROS that it can output is either the number 00000001 or the number 10000000, both of which have only seven ZEROS.

When two adjacent channels in a frame have the combination

....10000000 00000001...

that is still only 14 ZEROS in a row. No problem.

The worst case situation would be if the last voice channel in a frame is 10000000, then there is a framing bit of 0, and then the very first channel in the next frame is 00000001. The resulting bit pattern would then be

....10000000 0 00000001.....

which gives precisely 15 ZEROS in a row. And that is still within the acceptable range.

So 24 voice channels in a T-1 line is no problem. But there could be a problem if that T-1 line is carrying non-voice digital data.

Non-Channelized T-1

So far, we have been assuming that the DS-1 / T-1 carrier system carries 24 DS-0 voice channels. That was the original intent, but that has changed.

Today, some of those 24 channels might be carrying digitized voice, while others might be carrying pure data. For example, the B channel in an ISDN is a 64K bps channel which might carry voice or data; somewhere along the line, it might appear as just one channel among many on a T-1 span.

Furthermore, sometimes a DS-1 / T1 system might not carry any voice channels at all. In fact, it might not even be divided into 24 channels. It would still carry framing bits, but the remaining 192 data bits in each frame might be pure data for just one customer. Such a system would be called *non-channelized*. It would provide a 1.536 megabit-per-second channel (often called a *data pipe* or *digital pipe*) which the customer can use for any purpose.

You can’t very well tell a customer that he is not allowed to send a lot of ZEROS in his digital data, and so either of these systems might easily violate the rule of no more than 15 ZEROS in a row.

B8ZS

B8ZS or *Binary Eight Zero Substitution* is one answer to the problem of too many ZEROS in a row.

When a T-1 transmitter is set to use B8ZS, it continuously monitors the data it is sending. Any time it sees a 00000000 byte, it changes it to 00011011. When the T-1 receiver at the far end of the line receives the 00011011 code, it automatically changes it back to 00000000.

The catch, of course, is that the byte 00011011 could also appear inside normal data, and so the receiver has to know which 00011011 to change, and which 00011011 to change. This is done by marking the sub-

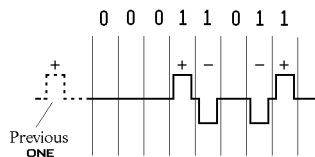


Fig. 17-4. Bipolar violation in B8ZS

stituted 00011011 by giving it a double bipolar violation.

As Fig. 17-3 showed, in AMI alternate ONES are supposed to be opposite polarity pulses. If two consecutive ONES come with the same polarity — either both plus, or both minus — this is a *bipolar violation* and normally considered an error. But in B8ZS this is done on purpose to mark the byte as one which should be changed back to 00000000.

Fig. 17-4 shows how this is done. If the last ONE before this byte was +, then the four ONES in 00011011 should start with a minus pulse, that is, their polarity should be - + - +. Instead, B8ZS puts a bipolar violation into bits 4 and 7, so that the polarity is + - - + as shown. (If the last ONE before this had been negative, then everything would be upside down, the four ONES would be - + + -, and there would still be a bipolar violation in bits 4 and 7.)

The T-1 receiver therefore monitors its input, and signals an error on all bipolar violations *except this one special case*. When it sees this particular bit combination with its double bipolar violation, it changes the 00011011 code back to 00000000.

Obviously, when using B8ZS, both the transmitter and the receiver have to know it. Most T-1 systems banks have a switch which has two positions labelled “B8ZS” and “AMI”. This is sometimes confusing to users, because a better description would be “AMI with B8ZS” and “AMI without B8ZS”.

T-1 equipment

The DS-1 / T-1 system has been quite standardized, and the equipment at the two ends of a span can come from different manufacturers — the fact that two different manufacturer’s equipments can be at opposite ends of a span is sometimes called a *mid-span meet*. But there are some variations, some of which are not compatible with each other, and there is a variety of equipment with often confusing names.

Channel banks

Let’s first summarize the various channel banks:

- The D-1 channel bank was first. It robbed a bit in every frame and did not use superframes. As a

result, only 7-bit conversion was used, and distortion was high.

- The D-2 channel bank introduced superframes, and robbed bits only in every sixth frame.
- The D-3 channel banks were more compact, and numbered the channels within a frame slightly differently.
- D-4 channel banks were even more compact, and also provided more maintenance functions.
- D-5 channel banks introduced the extended superframe.

Channel banks were more designed for connections between and inside central offices. In addition, telephone companies also used a variation of channel banks called Subscriber Loop Carrier, which were specifically designed to multiplex subscriber loops. SLC-96, for example, used four T-1 spans to provide connections for up to 96 subscribers in a building or group of houses.

Early versions of such multiplexers assigned the 96 channels to 96 subscribers, whether they were using the phone or not. Later versions would allow these channels to be shared as needed among more subscribers, on the theory that not all subscribers need to make a call at the same time. But if more subscribers try to call than there are channels, they may get no dial tone, or get a busy signal.

CSU and DSU

When they lease a T-1 line from the telco, customers need to install special equipment to “talk” to the telco equipment. Two distinct functions (often combined in one piece of equipment called a CSU/DSU) are needed:

- Connected to the telco T-1 line, a CSU or *Channel Service Unit* conditions the AMI signal to and from the T-1 line, performs B8ZS conversion (if needed), and allows the telco to do some testing.
- Between the customer and the CSU, a DSU or *Data Service Unit* does the conversion to and from AMI, adds or checks framing bits, and also allows the customer to do testing.

Repeaters and phantom power

As mentioned before, normal T-1 lines are designed to reach 6000 feet without repeaters. When a longer distance is needed, repeaters are inserted into the line at 6000-foot intervals (but the distance from the end of the line to the first repeater is generally only 3000 feet). As we mentioned earlier, this is the same distance that loading coils were originally used in analog subscriber loops, so removing loading coils and replacing them with repeaters combined two jobs with one trip.

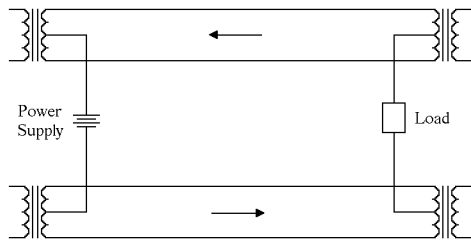


Fig. 17-5. Power transmission in a T-1 span

To provide power for these repeaters, and possibly other equipment along a T-1 span, DC power is sometimes sent along the cable. A T-1 cable without DC power is called a *dry span*, while one with power is called a *wet span*. Fig. 17-5 shows how the power is sent along the two pairs of cables used for the T-1 span. When there is more than one repeater on the line, then a somewhat more complex circuit is used to allow all of them to received power from the one supply.

Digital Cross Connects

Imagine this: a T-1 coming in from the east carries (among others) a signal on channel 3, and you want to send it out on channel 17 of another T-1 span going further west (and carrying other channels as well.) How do you transfer the signal from one line and channel to another?

This process of separating and recombining channels is called *grooming*. In the early days of T carrier systems, it would have required a demultiplexer with 24 codecs to separate the T-1 signals into 24 analog voice channels, then either a direct analog connection or perhaps a switch to direct the desired signal into another multiplexer, which would then take 24 new channels and combine them into one T-1 signal using another 24 codecs. (This process of converting from digital back to analog and then back to digital explains why the 7-bit coding of a D-1 channel bank was adequate for a short-distance connection, but not for long-distance circuits which might involve many such conversions.)

The modern approach is to do the entire process digitally. An *add-drop multiplexer* can extract a particular set of bits from a high-speed data stream, and substitute another set of bits instead of it. The entire system, which uses multiple such add-drop multiplexers to switch signals digitally between multiple inputs and outputs is called a *Digital Cross Connect* or *Digital Access Cross Connect*.

T-1 Variations

T-carrier is an old technology, dating back to the 1960's. There have been a number of extensions to the technique. Some of these have been adopted by many vendors, while others are proprietary.

HDSL and HDSL-2

Being able to go longer distances without repeaters would be nice, of course. In the early 1990's, work was being done on various kinds of DSL or *digital subscriber lines*. One of these was HDSL or *high-bit-rate digital subscriber line*. HDSL used the same speed as T-1, but had several advantages over T-1. It could reach as far as 12,000 feet, twice the distance of T-1, without repeaters; with repeaters, it could be extended to 24,000 or 36,000 feet. Like T-1, it used two pairs of wires (one pair in each direction), but there was a major difference. In T-1, the two pairs had to be separated from each other, and sometimes even shielded, to prevent interference between them, and between them and other circuits; in HDSL they did not. The longer distance and lack of interference was partially due to a different signal format: T-1 used AMI, whereas HDSL used 2B1Q like BRI ISDN.

In the late 1990's came an improvement called HDSL-2. Using Trellis coded pulse amplitude modulation instead of 2B1Q, and an echo-cancelling system called POET or *Partially Overlapped Echo Cancelled Transmission*, HDSL-2 achieved full duplex transmission over just one pair of wires.

Because of its longer distance capabilities, and the fact that it needs just one pair of wires instead of two, HDSL-2 is a significant improvement over T-1. As of the year 2000, commercial equipment is slowly becoming available.

Fractional T-1

Computer users sometimes say they have a *Fractional T-1* line. This terminology simply means a T-1 line in which some of the 24 channels are empty. For example, a user who does not need the full 1.536 Mbps of a T-1 line (although the data rate of T-1 is 1.544 Mbps, 8000 of those bits are framing bits, and they do not carry user data) might use only half the channels for a capacity of 758,000 bps, and refer to this as a fractional T-1. In terms of hardware, however, the fractional T-1 line requires the same two pairs as a full T-1 line.

Voice compression

The normal DS-0 scheme for converting a voice signal from analog to digital uses 8-bit pulse-code modulation, using μ -law encoding, to give a 64K bps signal. This allows 24 voice channels to fit into one DS-1 / T-1 signal.

More channels can be squeezed in, however, with voice compression.

One compression method combines a technique called linear predictive coding (LPC) with adaptive differential pulse code modulation (ADPCM). LPC tries to predict what the next sample will be, based on previous samples. By itself, LPC is not perfect, and so it produces an error. But that error is often small, and can be expressed with just a few bits using ADPCM, which can then correct the error. If each sample thus requires only 4 bits, instead of the standard 8, then the capacity of a T-1 line can be almost doubled. In practice, a T-1 line can produce 44 voice channels, rather than 48, because the extra bits are used for signalling.

Another compression method looks for periods of silence in your speech, and replaces your data with someone else's voice channel data while you are quiet. Since silent periods are fairly common, this technique can more than double the capacity of a channel. This technique is especially popular on overseas links.

E-1

T-1 is the slowest, basic T-carrier method used in the U.S. while E-1 is the slowest and basic method used in Europe and a few other countries. Whereas T-1 works at 1.544 Mbps and carries 24 voice channels (plus framing bits), E-1 works at 2.048 Mbps and carries 30 voice channels.

The basic E-1 frame consists of 32 eight-bit channels, for a total of 256 bits. At 8000 frames per second, this multiplies out to

$$32 \times 8 \times 8000 = 2,048,000 \text{ bps}$$

But of the 32 possible channels, one channel is used entirely for framing (since there are no framing bits), while another is used entirely for signalling (since E-1 does not use robbed bit signalling), leaving 30 channels for voice.

Timing problems

When a T-carrier span is used as a simple point-to-point connection, and is not connected to any other carrier system, then clock frequency and synchronization is not a major problem. For example, a T-1 receiver simply synchronizes its own clock to the received T-1 data (remember that there is a requirement that there not be more than 15 ZEROS in a row to guarantee that the receiver can synchronize.) The transmitter clock can be slightly off, and the receiver will adapt.

But two timing problems surface when the T-carrier is linked to other carrier systems, such as with add-drop multiplexers and digital cross connects:

- The incoming data rate may be different from the outgoing data rate. If the incoming data rate is higher, bits pile up and have nowhere to go. If the incoming data rate is too low, there is room for bits, but nothing to put into it. The solution here is to try to run everything at the same data rate, and also to provide some space into which extra bits can be stuffed.
- Even if the incoming and outgoing data rate are the same, the timing may be off — a frame may come in while the output is in the middle of sending another frame. The solution here is a buffer memory, but this is difficult to implement at very high speeds.

Recognizing the problem, the telephone companies decided to build an extremely accurate atomic clock located near the geographic center of the United States. At the time it was called the BSRF or Bell System Reference Frequency; with the breakup of the Bell System, it is now called the Basic System Reference Frequency.

The BSRF is accurate to 1 part out of 10^{11} , which translates to a maximum error of 1 second in about 3170 years. More important, it translates to a maximum error of $1/8000$ of a second — the time for one T-1 frame or one sample of digitized voice — in about 4.7 months, or an error of $1/2$ of a frame (one-half of $1/8000$ of a second) in about 72 days. This latter time is called the *time to the first frame slip*.

At the time of its original design, atomic clocks were extremely expensive, and so one master clock to run the entire telephone system was an economical solution, especially since the entire long-distance system was run by one company — AT&T. Today, however, system reference clocks are run by other IXC's such as Sprint and MCI. Moreover, the same kind of accuracy is available from other sources, and at much lower cost.

The original idea was to have a system of clocks as shown in Fig. 17-6, arranged in levels or strata. The BSRF atomic clock would be used for level 1 or *Stratum 1*. It would be the most accurate, and the reference for the entire system.

Underneath would be a set of Stratum 2 clocks, somewhat less accurate (and cheaper). These would normally be connected to and synchronized with the Stratum 1 clock, but if that connection failed, they would be able to run by themselves for a while (until the connection was fixed), though at lower accuracy. In the same way, there would be an even less accurate Stratum 3 layer underneath those, and a fourth Stratum 4 layer at the bottom. Each layer would be synchronized to the

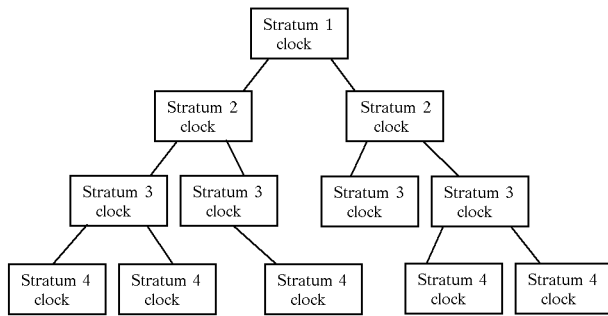


Fig. 17-6. System clock hierarchy

layer above, but able to run by itself for a while, if need be. These stratum clocks can be summarized like this:

Stratum	Where used	Drift if free-running for 24 hours	Time to first frame slip when free-running
1	BSRF etc.	1×10^{-11}	72 days
2	Toll centers	1×10^{-10}	7 days
3E	SONET	1×10^{-8}	1.7 hours
3	End Offices	3.7×10^{-7}	3 minutes
4	Customers	32×10^{-6}	2 seconds

The table shows one new Stratum level, level 3E or 3 Enhanced, which was inserted when it was found that Stratum 3 was not quite good enough for SONET systems (see the next chapter.)

Note the structure in the levels:

- Stratum 1 clocks are the system reference, primarily for interexchange carriers (IXC's). They provide the timing synchronization signals to
- Stratum 2 clocks, which would normally be just as accurate as the Stratum 1 clocks as long as they are connected to Stratum 1 reference. These Stratum 2 clocks would generally be used at the IXC toll centers. At each connection point to a Local Exchange Carrier (LEC), this clock would also send sync signals to the
- Stratum 3 clocks in LEC end offices, so these would normally also run at the Stratum 1 accuracy (unless there was a break in the chain somewhere above them.) In turn, the Stratum 3 clocks would send sync signals down the chain to the
- Stratum 4 clocks at individual customers. Although these have pretty bad free-running stability by themselves, they stay synchronized to the EO clocks above them in normal use. Only if the connection to the EO above them breaks do they go into their free-running, and inaccurate, mode, but at that point it doesn't matter.

The result would be a sort of master-slave structure, where higher stratum levels would be the master, and the lower levels, which copy the frequency of the master, would be the slaves. Furthermore, in every link, one end would be designated as the master and the other the slave.

This simplified structure has changed a bit over the years as accurate clocks have become cheaper. The U.S. Department of Defense runs a system of Global Positioning Satellites (GPS) which are used for navigational purposes, but which also have very accurate atomic clocks. Fairly inexpensive GPS receivers can receive the signals from these satellites and provide a clock accurate to Stratum 1 levels, but at minimal cost. Hence many offices which used to have Stratum 3 clocks in the past can now have their own Stratum 1 clock system.

DETOUR

Don't confuse the stratum concept described above with the stratum clocks on the Internet.

When an e-mail message is sent on the Internet, it includes a date and time. In order to make sure that the date and time on all systems is set more-or-less correctly, there is an Internet protocol for distributing the time between systems. There are systems which claim to have accurate clocks, and they call themselves Stratum 1 clocks. Computers which in turn get their time from Stratum 1 systems call themselves Stratum 2, and so on. But this scheme is nowhere as accurate as the system used in the communications field. Even aside from the fact that it takes milliseconds just to transfer the time from one system to another over the Internet, some of these self-proclaimed Stratum 1 Internet clocks are off by hours, weeks or even years.

END OF DETOUR

DS-2 / T-2

The next step up in the T-carrier hierarchy is DS-2 / T-2, which runs at 6.312 Mbps. It carries four DS-1 signals, or a total of 96 voice channels.

DS-2 is obtained by multiplexing four DS-1 signals together; such a multiplexer is referred to by the name M12 — a Multiplexer that goes from level 1 to level 2.

Stand-alone M12 multiplexers are actually very rare. There are some which take four T-1 signals and combine them into a DS-2 signal, and then send the resultant signal on an optical fiber. More common, however, is a combination which consists of seven M12 multiplexers, each of which takes four T-1 signals (96 voice channels) and produces a DS-2 signal, and then an M23 multiplexer which combines the seven DS-2 signals to pro-

duce one DS-3 / T-3 signal with 672 voice channels. The overall package is then called an M13 mux.

Since T-2 systems are not that common, we will skip over some of its details and go straight on to DS-3 and T-3.

DS-3

The DS-3 format was designed as a direct descendant of DS-1, but 28 times faster. Although originally designed for microwave links, it is also available in a cable version that uses coax cable, but with a maximum range of 600 feet in one span.

DS-3 runs at 44,736,000 bits per second. At this rate, each bit lasts about 22 nanoseconds; a byte lasts about 170 nanoseconds. This high speed creates a number of new problems.

- A buffer to store any overflow would have to be very fast — 170 nanoseconds per byte — and also potentially very large — a delay of, say, $\frac{1}{10}$ of a second would require storing 4,473,600 bits.
- Regardless of how well you try to time things, incoming data is almost certain to come in at the wrong time.

Here's why: the speed of light in a vacuum is about 186,000 miles per second, which translates to about 1 foot per nanosecond. Since signals in cables or optical fibers travel at about $\frac{2}{3}$ the speed of light in a vacuum, they would travel about 8 inches in one nanosecond. Hence about 14 feet of cable or fiber will delay a signal 22 nanoseconds, or one bit time at DS-3 speeds. Send the signal through a mile of fiber or cable, and the delay is about 3500 nanoseconds or about 160 bit times.

Hence any synchronous digital system (which includes T-2, T-3, SONET, and others) must have some arrangement for fixing or at least accepting timing problems. And that greatly complicates the picture.

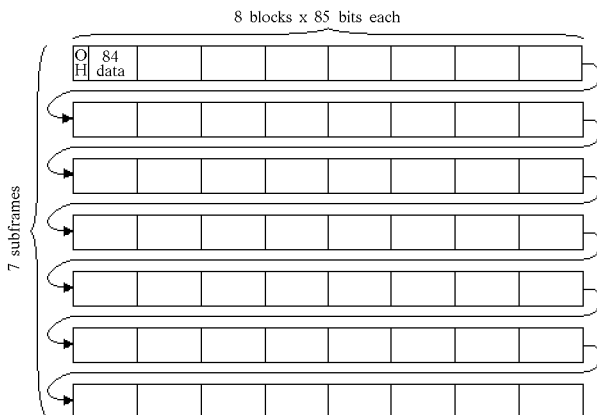


Fig. 17-7. DS-3 / T-3 Multiframe format

With that in mind, let us look at Fig 17-7, which shows a DS-3 frame, called a *Multiframe* or *M-frame*. It is much too long to show all in a straight line across the page, and so Fig. 17-7 shows it in seven pieces called *subframes*; the arrows show how the subframes follow each other. The overall format is like this:

- The M-frame is divided into seven subframes. There are 7, because the overhead bits in each subframe tell something about one of the seven DS-2 data inputs which make up the DS-3 data.
- Each subframe is divided into eight blocks.
- Each block has 85 bits, consisting of one overhead bit (OH) plus 84 data bits. (We only show this in the top left block, but they all have the same structure.)

The frame thus has a total of

$$7 \times 8 \times 85 = 4760 \text{ bits.}$$

Within the 84 data bits of each block are 12 bits from each of the seven DS-2 inputs; these are bit interleaved, meaning that there is one bit from the first DS-2 input, then a bit from the second DS-2, then a bit from the third, and so on.

DETOUR →

Incidentally, don't try to multiply out some of the other numbers to make sense of this, because it will only confuse you. For example, if you try to divide the data rate of 44,736,000 bits per second by 4760 bits per frame, you will get 9398.319 frames per second, etc. The strange result is due to the fact that there are many extra bits in both DS-2 and DS-3 to take care of timing problems. Things will be much neater when we get to SONET in the next chapter.

← **END OF DETOUR**

There is a total of 56 blocks (8 across by 7 down), so there are 56 overhead bits in each M-frame. Let's ignore the 84 data bits in each block, and just look at the overhead bits in the form of a table, and label them like this:

X	1	C	0	C	0	C	1
X	1	C	0	C	0	C	1
P	1	C	0	C	0	C	1
P	1	C	0	C	0	C	1
0	1	C	0	C	0	C	1
1	1	C	0	C	0	C	1
0	1	C	0	C	0	C	1

This table shows that there are four kinds of overhead bits in the frame:

- About half of the bits are shown as 0 or 1. These are permanently set in that format, and are similar to the framing bits we've seen in T-1. They help the receiver recognize the beginning of each block, and also the beginning of a frame.
- The two bits in the top left corner labelled X are used to send an alarm signal to the receiver. They would normally be 11; if they are changed to 00, that is an alarm message.
- The two bits labelled P are parity bits, similar to even parity in plain asynchronous serial data. Both P bits would normally be the same.
- The remaining 21 bits labelled C show whether, and where, bit stuffing is done.

As we've discussed a few pages ago, the high speeds of DS-2 and DS-3 introduce timing errors. To compensate for them, DS-2 and DS-3 both have extra space into which they can insert so-called *stuff bits*. These allow the input and output of a multiplexer to keep in step. The job of the C bits is to keep track of these, and tell the receiver where the transmitter put them.

A fuller discussion of the C bits would take up many pages, and so we will stop right here, especially since there is a later T-3 format, called the *C Bit Parity* format, which greatly extends the functions of the C bits to provide better parity checking, a 28K bps data channel, and more alarm and maintenance information.

T-3

T-3 is the U.S. implementation of DS-3. Although DS-3 is often used at microwaves (both ground-based and satellite), it is now also a fairly economical choice for coax-based services.

Like T-1, T-3 uses AMI as its signal format, but on a coax rather than a balanced line. But its voltage level is much smaller — whereas T-1 uses ± 3 volts, T-3 uses just ± 0.3 to ± 0.8 volts.

Because of the higher speed, it is harder to keep a receiver precisely synchronized with the incoming signal. As a result, no more than three ZEROS in a row are allowed. T-3 therefore always uses a full-time method to eliminate cases of 3 or more zeroes, called B3ZS.

B3ZS is similar to B8ZS, but it works on strings of 3 ZEROS rather than 8 ZEROS as in B8ZS, and also its bipolar violation (BPV) pattern is different: whenever B3ZS sees a pattern of three consecutive ZERO bits, it replaces them with either a 001 or 101 bit pattern, with the right-hand 1 being a bipolar violation. The choice of whether to use a 001 or 101 replacement depends on what came before — the aim is to make this BPV the opposite polarity of the *previous* B3ZS bipolar violation. For example, consider the data 100011000, assuming the first bipolar violation was +, like this:

```

1 0 0 0 1 1 0 0 0
+ 0 0 + - + 0 0
          BPV          X          BPV

```

In AMI, ONES are supposed to alternate polarity, but the spots marked with a BPV show where there should now be BPVs because of B3ZS. As shown, the last bipolar violation should also be a + pulse, but this would make it the same polarity as the previous BPV, which was also +. This can be changed by substituting a 101 pattern rather than 001 at the end, which makes the X bit a 1, like this:

```

1 0 0 0 1 1 0 0 0
+ 0 0 + - + - 0 -
          BPV          X          BPV

```

Now the two BPVs are opposite polarity.

The reason for this seemingly strange practice is that it makes the average number of positive bipolar violations the same as the number of negative ones. In other words, there is an equal number of positive pulses and negative pulses, which keeps the *average* DC level on the line at zero volts.



There is a similar scheme used in some foreign countries called HDB3 or *High-Density Bipolar 3 Zeroes*. It is identical to B3ZS except that it works on four ZEROS rather than three. That is, the combination 0000 is replaced by either 0001 or 1001, with the last 1 a BPV. The rule for choosing 0001 or 1001 is the same as in B3ZS.



DS-4 and T-4

Even faster than DS-3 and T-3 is DS-4 and T-4, which operates at 274.176 megabits per second, and carries 4032 voice channels (or equivalent data). But this system has not achieved anywhere near the popularity of T-1 and T-3. This is partially because there is no firm standard to specify how equipment will be designed, and so there is no "mid-span-meet" — it is not possible to use equipment from two different vendors at opposite ends of the same connection. Furthermore, such high speeds require fiber rather than copper cables to achieve any kind of distance, and SONET (which we will discuss in the next chapter) is a much more standardized, versatile, and robust fiber optic network. Hence we will ignore T-4 in this chapter as well.