Ethertech Digest is published by Ethertech Global Industries for the good of the public. We don't give a shit about corporate interest; this is for the people that care about the world around them.

No part of this magazine may be reprinted, in part or whole, without express permission of the authors of their respective works. It's their intellectual property. Don't steal. The views and opinions that may be expressed by the authors does not necessrily represent those of the staff, but it might. However, the existance of views in the absense of of anyone to hold them is left as an exercise for the reader.

Etherech Digest is produced for educational purposes. Everything is hypothetical unless otherwise noted. When illicit behavior is mentioned it means we want you to go do these things; otherwise we wouldn't write about them. We condone and encourage illicit behaivor. There, I said it. Do something about it.

However, we're not responsible if you do something stupid and/or illegal as you are choosing to do so out of your own free will. Remember that thing called free will?

So there you have it; don't copy our shit lest you be mocked for all eternity, and don't whine if you think we're assholes. In other words, don't be a fuckin' wanker.

We publish on a (with luck) quarterly basis, although like all things, this is fallable. We are human after all.

Submissions go to lawg@ethertech.org, if you feel so inclined.

Ethertech Digest

'the china syndrome'

issue one
volume one

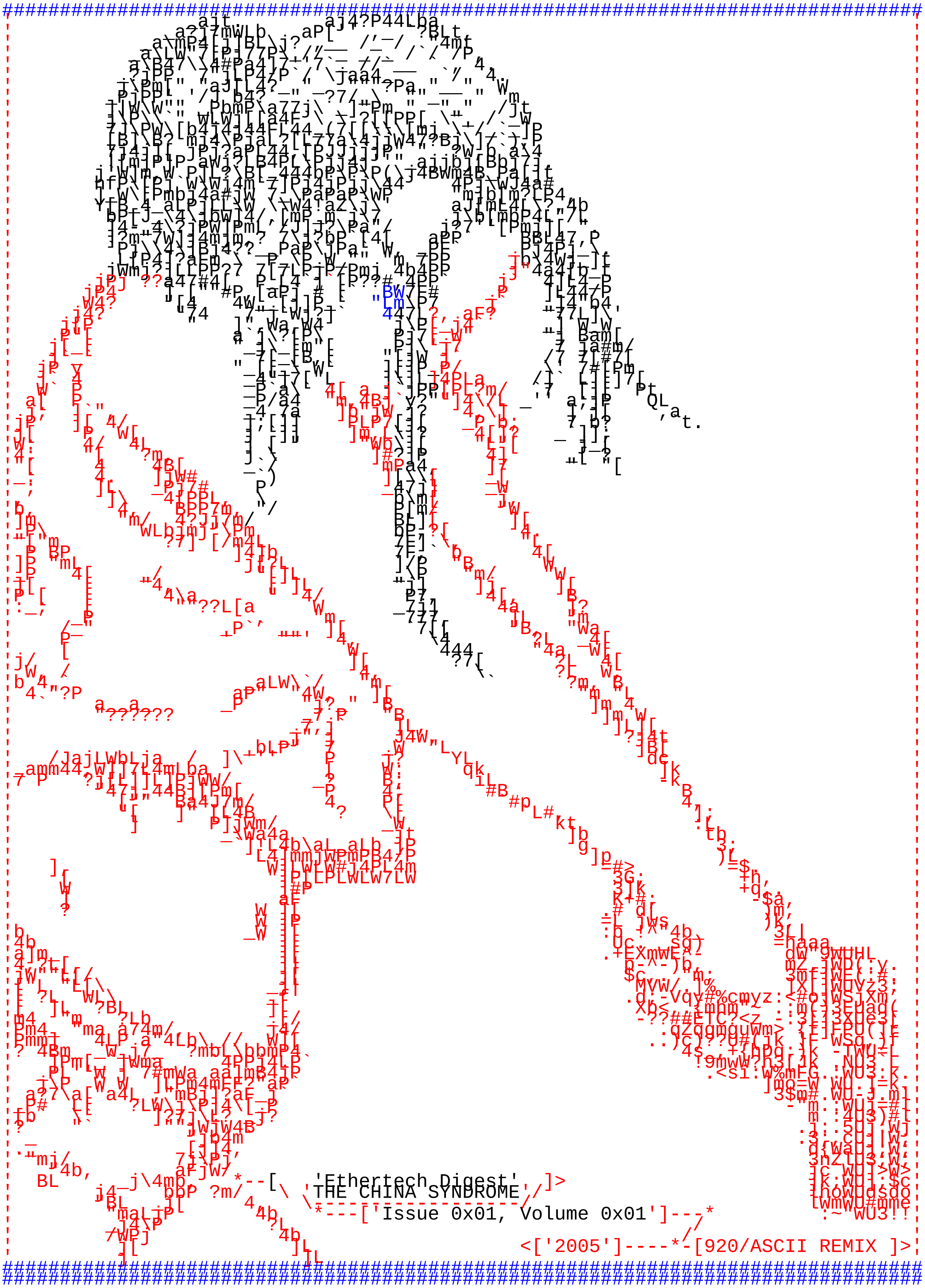
winter 2005

Table of Contents:

WWW: http://www.ethertech.org/
Contact: lawg@ethertech.org

The First Amendment to the Constitution of the United States of America:
"*Congress shall make no law respecting an establishment of religion, or prohibitting the free excercise thereof; or abbridging the freedom of speech or of the press; or of the right of the people peaceably to assemble, and to petition the Goverment for a redress of grievances*"

Exercise your rights and raise your fist.



'Ethertech Digest'

THE CHINA SYNDROME

'Issue 0x01, Volume 0x01'

<['2005']----*-[920/ASCII REMIX]>

Forward Error correction coding primer

Introduction

What is forward error correction coding? Well, kids, I'm going to let you know! It is the means by which to transmit data in a fashion that if there is say, a burst error that assrapes a couple bits, it can be detected and fixed. It's actually quite sexy, and writing algorithms to decode them is fun. Believe me. Would lawgie lie to you?

The algorithms I'm going to talk about in this article are most commonly used on FSK (or GMSK) and those systems are starting to be replaced by faster and more reliable systems implementing PSK. Systems are also starting to implement REAL encryption, rather than marketing a crappy bit scrambler as an encryption.

Hrm, what am covering exactly? Ohhhh... bit interleaving, and two simple forms of FECC used by motorola systems. If you're lucky enough to get a part II (I plan on writing it, but I'm a pretty busy guy, ok?), I plan on covering fano, goppa (pseudo-public-key-crypto!) and BCH coding (think TDMA).

Yeah, you might be thinking "what about viterbi and reed-solomon?!". Hey, shutup and unbunch your panties. It's a progressive series of articles, and those are a little more complex. If I get there, you'll see it.

All code will be in C. Just because I happen to like C. It's not the cleanest code ever written, and could be heavily improved. For our purposes here though, it should be easy enough to read.

```
Figure 1:

void diwabf(buf,offset,freq)
char *buf; /* our rcv. buffer */
int offset;
int freq;
{
      unsigned char *cpy;
      cpy = strdup(buf);
      memset(cpy, 0, MAX_BITS);
      int x,y,i,k,j=0;
      int l=1;
      for(x=0;x<offset;x++)
      {
            for(y=0;y<freq;y++)
            {
                  k=(y*offset)+x;
                  cpy[j] =
buf[k];
                  j++;
            }
      }
      for(i=0;i<MAX_BITS;i++)
      {
            if(cpy[i] & 0x01)
                  printf("1");
            else
                  printf("0");
      }
      printf("\n");
      free(cpy);
}
```

I guess this intro has lasted long enough, so go get a fresh cup of coffee, light a smoke (whatever you smoke) and read on... (because I'm doing the same thing.)

Bit Interleaving

Ok, so it's not really error detection and correction, but I figured I would start with it because it's easy. With this, you really have to have some info on the system you'll be receiving. An engineer may be able to provide you with the system specs (because you're testing/debugging it, right?), but beyond that.. the US Patent Office may be your best bet. There's a huge database online somewhere (LINK HERE), and you may be able to find exactly what you're looking for there.

Anyway, you receive some random data frame and disect it:
[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,...]

Now, when you receive them, they are all out of order (interleaved) and you need to fix this. You need to know the offset and the frequency (i don't know the real terms, but these work for me). Say you're offset is 19 and your frequency is 4. You're deinterleaved data would look like this:
[1,20,39,58,2,21,40,59,3,22,41,60,...]

Figure 1 shows a simple method for deinterleaving data stored in a char array, each char being a one or zero. I know it's not very optimized and it takes up a lot more memory than it should (each bit takes up an unsigned char, which is a full byte (8 bits)). That's ok, because it's just an example. =)

That's the deinterleave function. The calling routine would need to fill (or receive a transmission) to use it. It works pretty damned well, and you can specify your offset and frequency. Nifty, eh? If you were really going to use this, you might try replacing the printf() and free() sequence by passing that onto a second-stage decoder.

A simple Convolutional error correcting scheme

```
Figure 2:

Information. . . . . . :0 0 1 0 0 0 1 1 0 0 0 1 1 1 1 0
Parity . . . . . . . . : 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1
Transmitted bit Stream :00001101000011100100001110101001
```

```
Figure 3:

Received bit stream: 00001101000011100100001110101001
Strip out Info bits: 0 0 1 0 0 0 1 1 0 0 0 1 1 1 1 0
Calculate Parity   :  0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 1
Expected bit stream: 00001101000011100100001110101001
SYNDROME           : 00000000000000000000000000000000
```

```
Figure 4:
                           *       *          *
Received bit stream: 00001111000011000100001111101001
Strip out Info bits: 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 0
Calculate Parity   :  0 0 1 0 1 0 1 1 0 0 0 1 0 0 0 1
Expected bit stream: 00001110010011010000001110101001
SYNDROME CALCULATION:
Received bit stream: 00001111000011000100001111101001
Expected bit stream: 00001110010011010000001110101001
SYNDROME           : 00000001010000010100000001000000

Correct bit stream : 00001101000011100100001110101001
```

This is a fairly easy form of ECC to grasp. It's a 1/2 rate convolutional scheme. That means that half of the bits transmitted are parity bits used to check (and correct) the received data. Check out Figure 2 to see what I mean. It essentially uses a
X + X^-1 + X^-3 polynomial.

Let's get down to the nitty-gritty than, shall we...First we strip out the actual data bits {0, 2, 4...} and put them into a new buffer, and calculate the parity bits. Wait? You thought we were decoding, eh? Well, that's the fun part. You encode it again to decode it. Check it out... you're generating what the parity bits *should* be. This is your expected bit stream. In order to check it for problems, it's really easy. You XOR it against the received bitstream. This is called the 'syndrome'. It should be all zeros. To show a couple of examples, I'm going to borrow from a USENET post that I think shows it in a most happy way. Figure three shows how it works with no errors, and figure four shows it with a couple errors (denoted in red).

Notice in figure 4 that an error in an info bit causes the next two parity bits to be flipped, and an incorrect parity bit causes a single bit to show up in the syndrome. I'm sure you get it, but if not.. you should be able to catch my drift if you study the code in figure 5.

```
Figure 5b:

void decode()
{
      int i=0;
      unsigned int proc[BITLEN/2];

      for(i=0;i<BITLEN;i+=2)
      {
            proc[i>>1] = buffer[i];
            if(buffer[i] == 0L)
            {
                  buffer[i  ] ^= 0x01;
                  buffer[i+1] ^= 0x01;
                  buffer[i+3] ^= 0x01;
            }
      }
      for(i=0;i<BITLEN;i+=2)
      {
            if((buffer[i+1] == 1L) && (buffer[i+3]
== 1L))
            {
                  proc[i>>1]  ^= 0x01;
                  buffer[i+1] ^= 0x01;
                  buffer[i+3] ^= 0x01;
            }
      }
      for(i=0;i<BITLEN/2;i++)
      {
            if(proc[i] & 0x01)
                  printf("1");
            else
                  printf("0");
      }
      printf("\n");
}
```

```
Figure 5a:

#define BITLEN 200
unsigned int buffer[BITLEN];

int main(void)
{
      int i, x, k=0;
      srand(42); /* yeah... */
      printf("Original Data: ");
      for(i=0;i<BITLEN/2;i++)
      {
            x = rand();
            if(x > k)
                  encode(1);
            else
                  encode(0);
            k = x;
      }
      printf("\nEncoded Data:  ");
      for(i=0;i<BITLEN;i++)
            printf("%d",
buffer[i]);
      printf("\nDecoded Data:  ");
      decode();
      return(0);
}
```

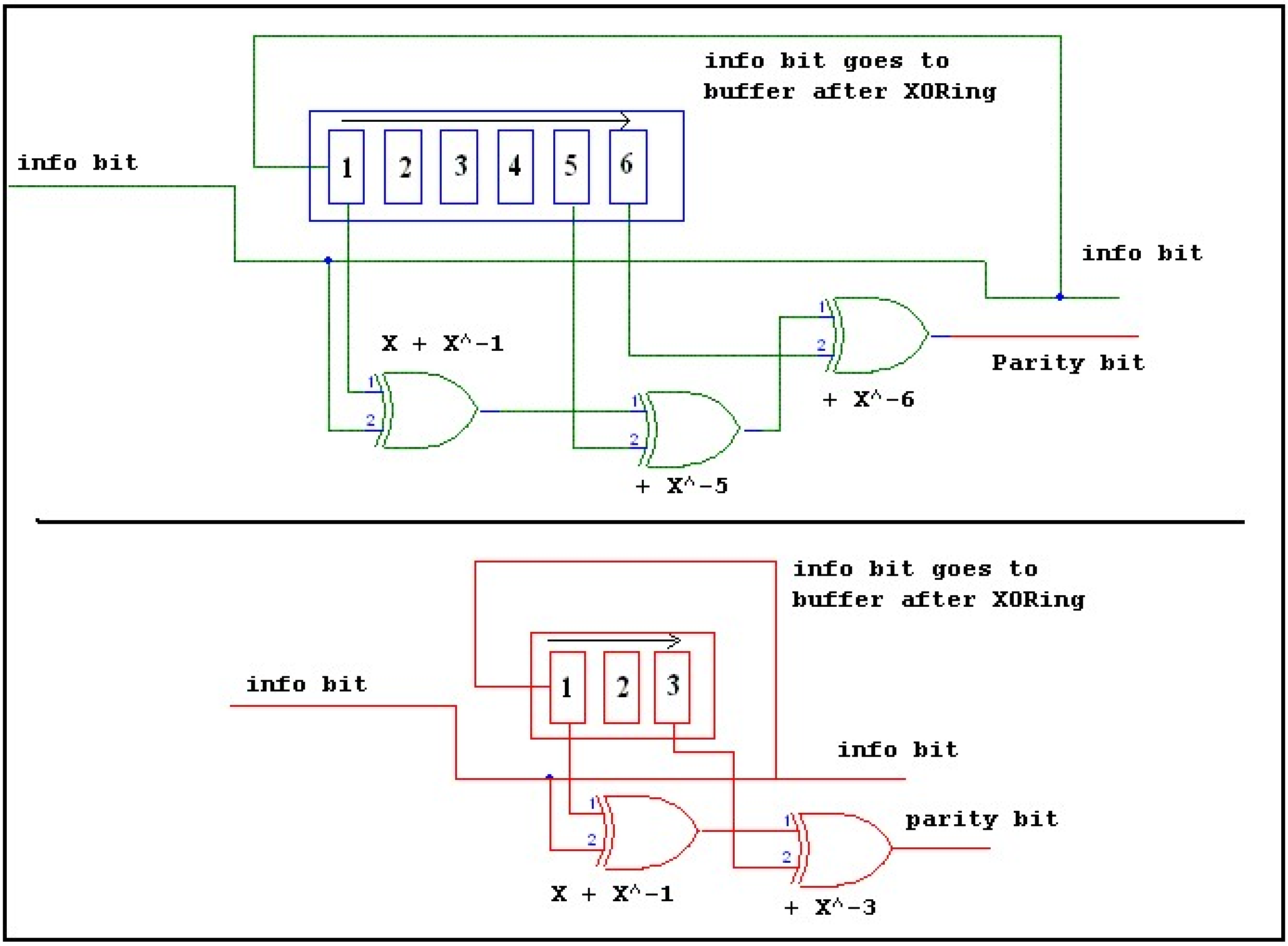
```
Figure 5c:

void encode(bit)
unsigned int bit;
{
      static unsigned int lead;
      unsigned int par;
      static int ct=0;

      if(ct ==0) lead=0L;

      par = bit^lead;

      buffer[ct] = bit;
      buffer[ct+1] = par;
      lead = bit;
      ct += 2;
      if(bit & 0x01)
            printf("1");
      else
            printf("0");
}
```

A slightly more complex convolutional system

This convolution is used by Motorola MDC4800, and is also pretty easy to get the idea of. It uses the poly 1 + X^-1 + X^-5 + X^-6. Werd. It pretty much follows the same principal as the previous section, except where it only relied on the first bit immediately previous to generate the parity bit, this one also uses the fifth and sixth bits back. Oh, and in case nobody has realized it yet: polynomial addition and subtraction are basically just XOR. Yeah.

The difference being that instead of XORing it against the previous and the third bit back, you XOR it against the previous bit and the fifth and sixth bits previous to get the parity bit. This can be accomplished by keeping a buffer of 6 bits wide and sliding it by one each time a bit is encoded/transmitted.

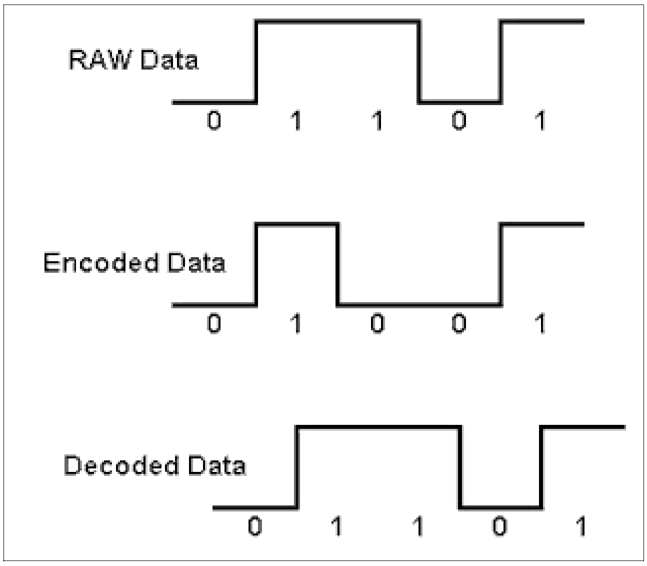
These logic charts should simplify things a bit, if you're still in the dark or confused about how it works. Yes, they're backwards. My bad. The two-stage encoder is on the bottom and the three-stage encoder is on top. It should be a snap to figure out anything else from here. ASM, anyone?



Differential Coding

This is one of the basic coding forms used. It simplifies logical state changes from absolute to being the difference between two states.

Encoding	**Decoding**
$d(t_0) = m(t_0)$ XOR **Initial condition** parameter value	$m(t_0) = d(t_0)$ XOR **Initial condition** parameter value
$d(t_k) = d(t_{k-1})$ XOR $m(t_k)$	$m(t_k) = d(t_k)$ XOR $d(t_{k-1})$
where	where
● m is the input message ● d is the differentially encoded output. ● t_k is the kth time step. ● XOR is the logical exclusive-or operator.	● d is the differentially encoded input. ● m is the output message. ● t_k is the kth time step. ● XOR is the logical exclusive-or operator.



The Data is padded with a leading zero to introduce a leading known element. A logical '1' is inserted only when there is a logical state change rather than when the state changes to '1'. Therefore you need to know what the previous state was in order to determine whether or not a state change should be introduced. This makes syncing a signal easier because if you're timing source (clock) is locked and the data is not then all you need to do is reverse the polarity and it will synchronize.

This can be accomplished (as seen in table above) with simple XOR arithmetic.

Resources

1. Motorola Bit interleaving: U.S. Patent #4159469
2. Motorola 1/2 Convolutional ECC: U.S. Patent #4055832
3. Google USENET
4. Digital Electronics by Roger L. Tokheim

Satellite Communications

Satcomm seems to be one of the most severely misunderstood forms of communication out there. Hopefully, these articles will shed some light upon the subject and help you understand what's going on in the sky above your head. In this article I'm going to cover modulation techniques, Eb/No and C/Kt (don't worry, I'll get into what those are later) and cover some definitions that are important. I've seen a couple of guides and primers on this subject but none of them explain it very well as far as I'm concerned.

The first misconception is that satellites 'change' the singal somehow (this may be true in some instances, but highly unlikely.) Their only real function is to relay/bounce signals back to earth, and overcome the problem of line-of-sight. Some 'smart' Satellites do ISL (Inter-Satellite Links), but they don't 'change' anything. Any modification to the data being tranceived is done by the Earth Terminals.

Most 'real' data being transmitted over satelittes is in a microwave band. Only some of the more expensive scanners can receive in these areas and most scanners will introduce a new problem: Bandwidth. Most scanners only have an rx bandwidth of maybe a hundered KHz. Most data is being transmitted at (least) a data rate of at least a T-1.

To receive these signals you'll need an antenna with a high enough gain, a downconverter and a PSK modem. Unless you plan on only receiving PSK31 data, a soundcard won't work (same problem as the scanners). This stuff can get expensive.

To make matters worse, satellites move. Yes, even geosynchronous satellites move. It is very little, to say the least. Depending on your antenna, you may or may not notice the receive level drop. This happens with older satellites when their orbits become inclined, and they begin to travel (looking from the ground) in a figure eight.

Most data transmissions (voice too, actually) you'll see on a satellite will be wide carriers modulated with PSK (Phase-Shifted Key). PSK was designed for deep-space communications and works extremely well with microwave satellite communications. The most common variations of this are BPSK and QPSK, although there is a good chance you'll see 8PSK, QAM or even higher symbol rates. The most important thing to remember about the modulation schemes is that as the symbol rate goes up the baud rate goes down. You'll see why later on.

BPSK

It's Binary PSK. You can encode either a logical one or zero into each symbol. A symbol refers to a change (in FSK or GMSK it's a frequency change and in a few other schemes use a change in amplitude to indicate a change in the logical state.) The starting state is always assumed to be logical '0'. One must know the Symbol rate in order to decode a modulated signal because no change in the carrier can also indicate a bit. The signal can either be absolute phase state or differential psk. Both are common and if you're interested in doing anything with PSK31, you'll be using Differential phase state.

Matlab Example:

```
function y = bpsk(t)
t = linspace(-1,1,101);
p(1:25) = 1;
p(26:50) = 2;
p(51:75) = 1;
p(76:101) = 2;
y = 2*cos(4*pi*(t+ (2*pi*(p - 1)/2)));
plot(t,y);
```

You know that a carrier is a radio wave and that a sinewave and is expressed mathematically as such:

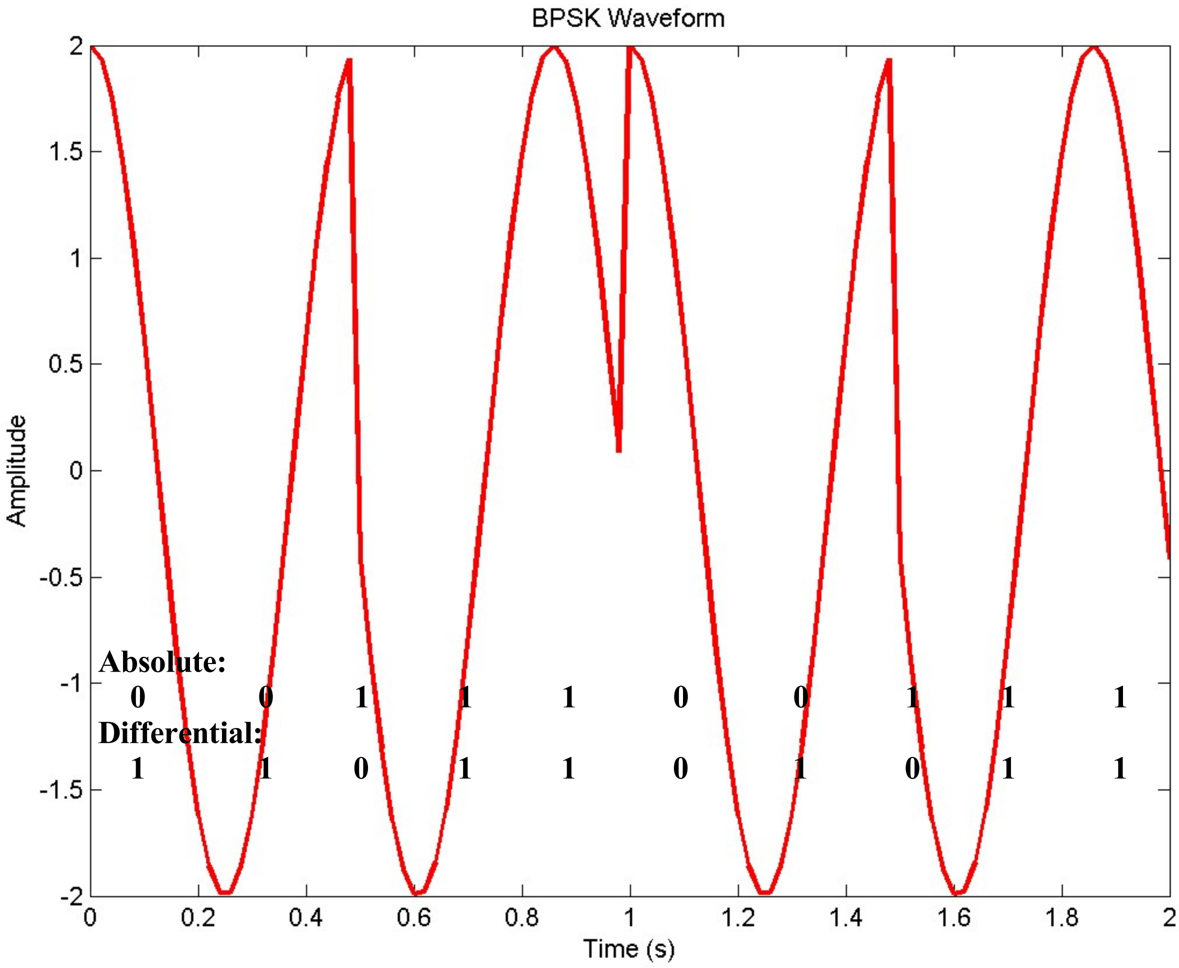
$$V_o(t) = A \cdot Sin \cdot w_o \cdot t$$

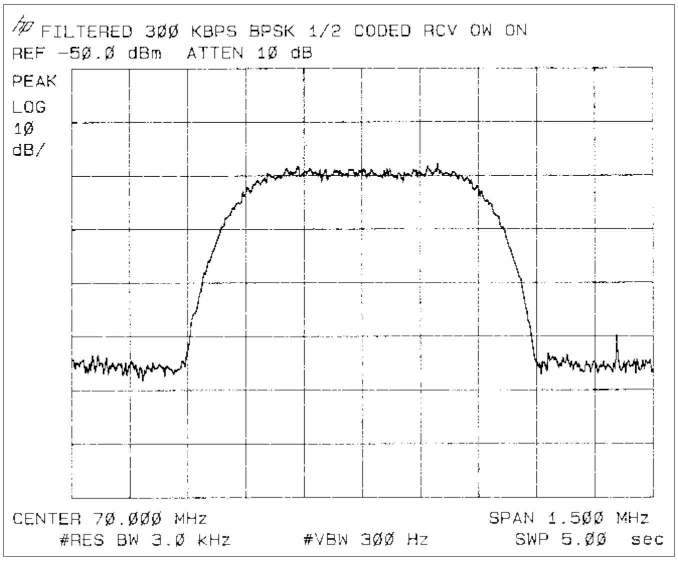
A BPSK symbol is a 180° shift in the carrier. The equation below shows how this carrier is generated. M is 2^n and in BPSK n = 2, therefore $2^1 = 2$.

$$V_o(t) = A \cdot Sin \cdot \left[w_o \cdot t + \frac{2\pi \cdot (i - 1)}{M} \right]$$

Where, A = Amplitude
w = frequency
t = time
i = 1,2...M
M = Number of symbols

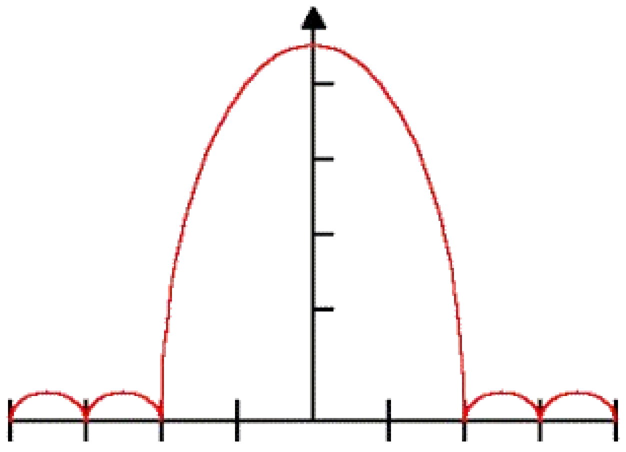
The Matlab example above will generate the BPSK wave shown on the next page. I've superimposed the relative data bits being modulated in the two forms over the waveform to help explain how the changes occurr. The example wave below has a Symbol Rate of 5 symbols per second.



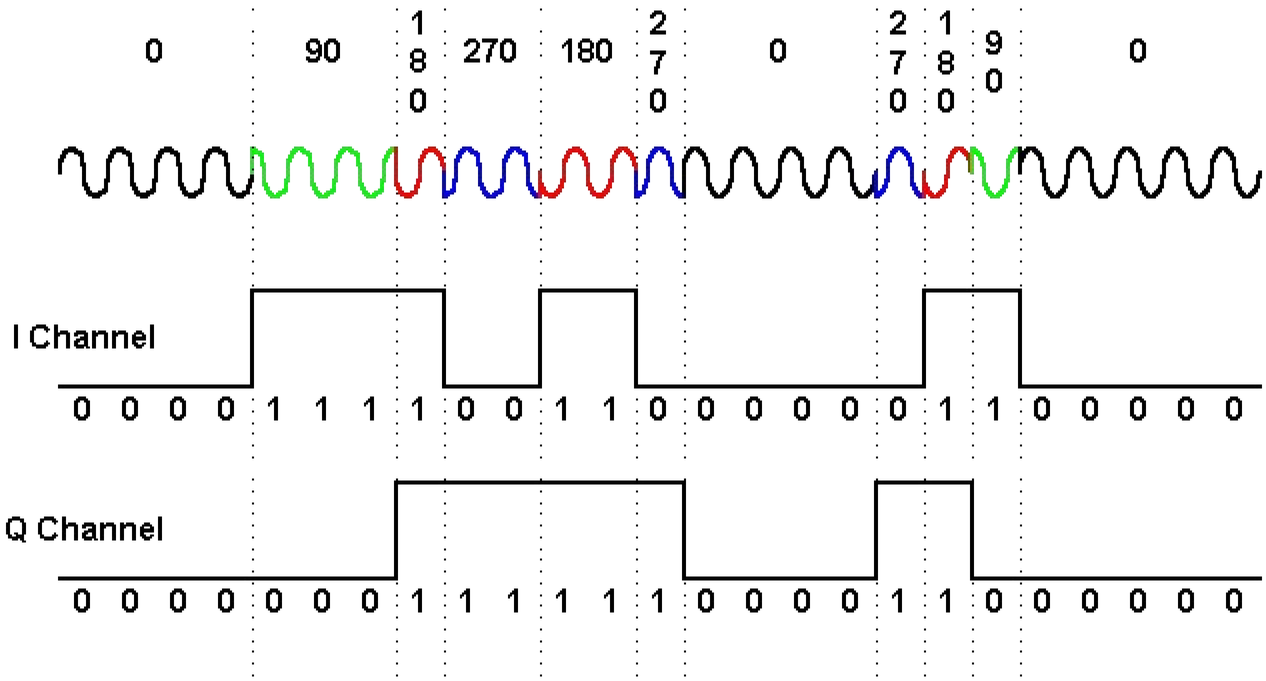


FILTERED 3ØØ KBPS BPSK 1/2 CODED RCV OW ON
REF -5Ø.Ø dBm ATTEN 1Ø dB
PEAK
LOG
1Ø
dB/

CENTER 7Ø.ØØØ MHz SPAN 1.5ØØ MHz
#RES BW 3.Ø kHz #VBN 3ØØ Hz SWP 5.ØØ sec

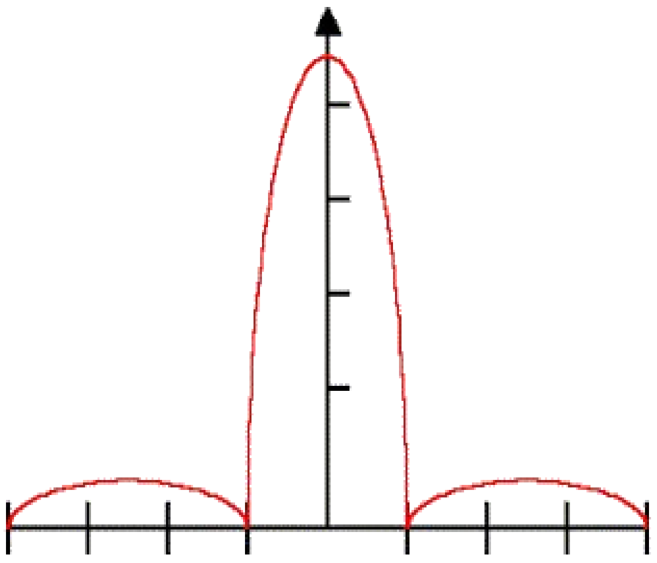


BPSK Waveform

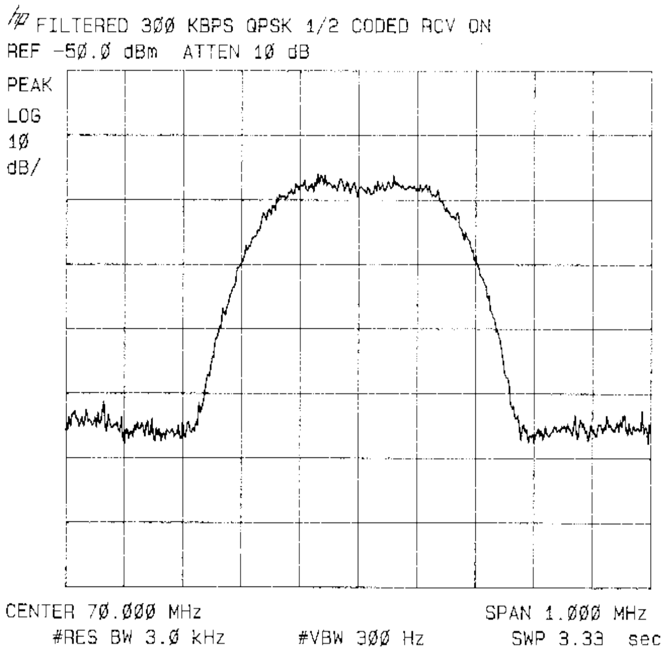


QPSK

Quadrature PSK is obviously a four-state modulation. Each state change represents two data bits (I and Q channels). One can effectively transmit twice as much data with the same baud rate. Each Symbol is represented by a different 90 degree phase shift. The same equation used for BPSK is used except M is a value 1 through 4.



QPSK Waveform



Tracking the 'Bird'

There are a couple of commonly used methods to maintain a quality signal while using a satellite. The most reliable is to use a signal strength comparator. We're working on a simple computer based solution for this at the moment, so once again, keep an eye out.

State 0 is the starting point where the signal is known to be strong. States 1-4 are the four directions (up,down,left,right) that will be compared for a greater signal strength. The antenna is moved from 0 to 1 and then back. Repeat with the other directions.

If the signal is detected to be stronger, then that state becomes state 0. The antenna is only moved by a tenth to a quarter of a degree (depending on how much the satellite moves.) This can really run your system hard if the satellite has an inclined orbit (older ones tend to move a lot).

Finding a Satellite

This shouldn't be too difficult for most people. Quite a few satellite locations are available online, including what's on them (predominately television). The best method is to use Predict, which comes furnished with a rather large database of ephemeris data and can calculate an elevation angle and azimuth (direction, in degrees) for you to point at. You'll more than likely have to scan around a bit to fine-tune.

If you have a spectrum analyzer (you probably don't, though), you can just set it for a wide span and move your antenna around until you see a shit-ton of carriers pop up. C-band is what you'll more than likely find, but Ku is pretty popular too.

If you see any dishes that look out of place (such as the location doesn't have any televisions) it could be data. Run out there with a compass and a protractor (or something, I dunno) and stick your frequency counter in front of it, until something comes up. I've heard that a lot of remote gas stations and truck stops use one of these to perform transactions and what have you. My guess is they have extremely small tx datarates (9.6KB) and should be no problem to pick up. Check it out and see what you find.

BER, Eb/Nø and C/Kt

Formulas used in calculating Rx Efficiency

C/Kt = EbNo + 10log(DR)

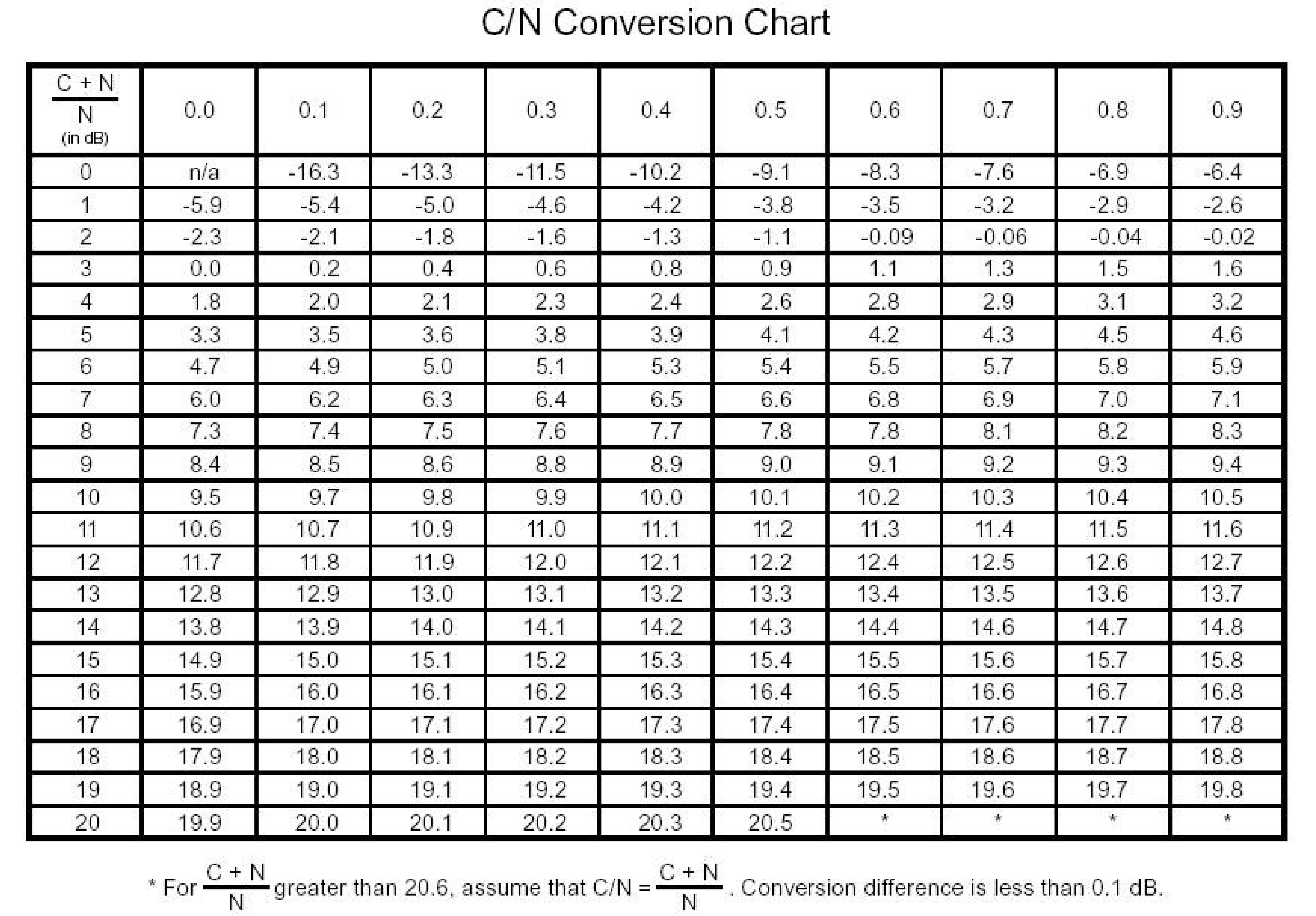
C/Kt = C/N + 10logSR

C/Kt = C/N + 10log(1/2BW)

C/N = 10log(log^{-1}((C+N/N)/10)-1)

EbNo = C/N + (10logSR – 10logDR)

Modulation	*Coding*	*Symbol Rate*	*Carrier Bandwidth*	*Eb/No*
BPSK	None	SR = DR	BW = 2 x DR	Eb/No = C/N
BPSK	½	SR = 2 x DR	BW = 4 x DR	Eb/No = C/N + 3.01
BPSK	¾	SR = 4/3 x DR	BW = 8/3 x DR	Eb/No = C/N + 1.25
QPSK	None	SR = ½ x DR	BW = DR	Eb/No = C/N – 3.01
QPSK	½	SR = DR	BW = 2 x DR	Eb/No = C/N
QPSK	¾	SR = 2/3 x DR	BW = 4/3 x DR	Eb/No = C/N – 1.76

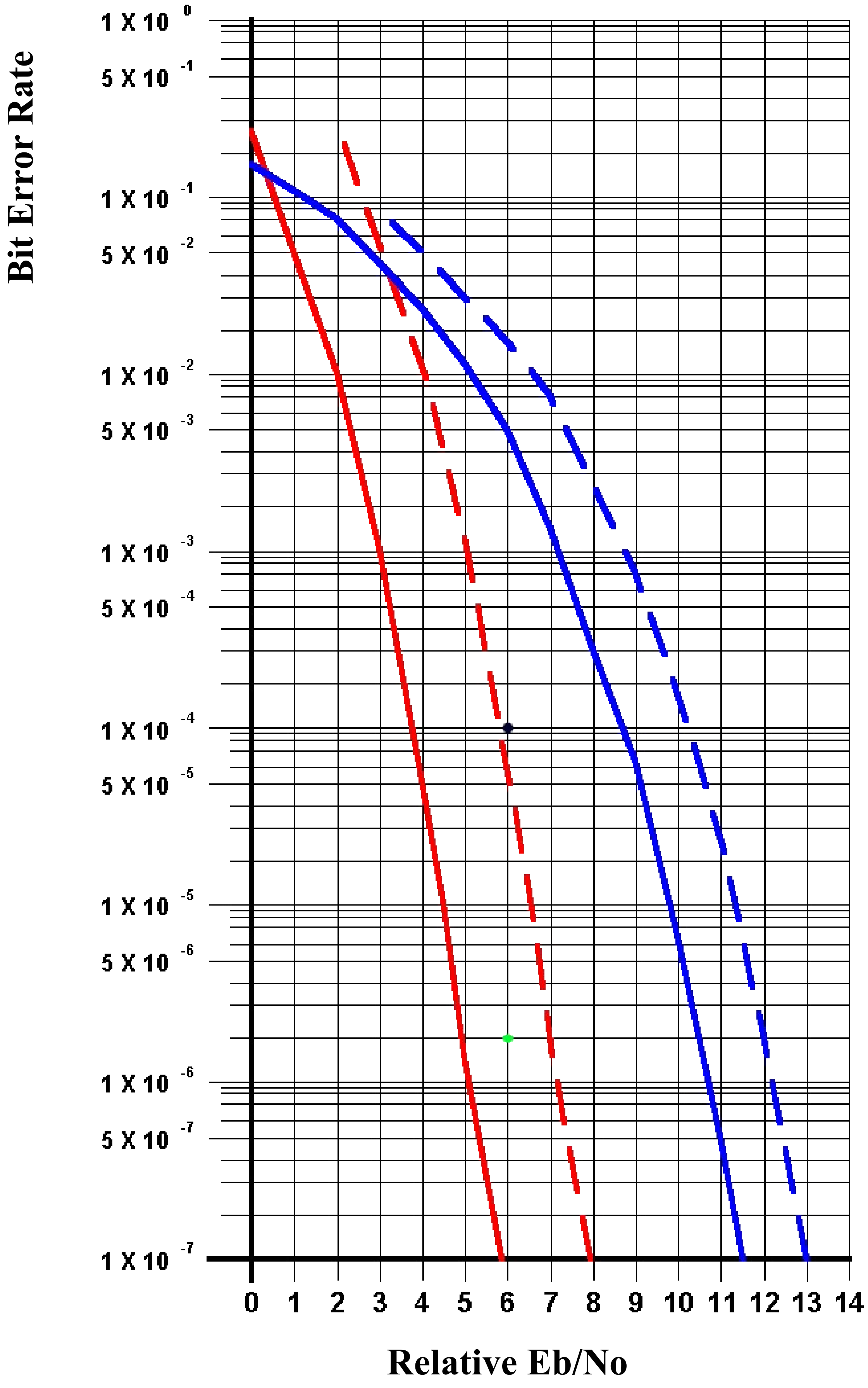


C/N Conversion Chart

$\frac{C+N}{N}$ (in dB)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	n/a	-16.3	-13.3	-11.5	-10.2	-9.1	-8.3	-7.6	-6.9	-6.4
1	-5.9	-5.4	-5.0	-4.6	-4.2	-3.8	-3.5	-3.2	-2.9	-2.6
2	-2.3	-2.1	-1.8	-1.6	-1.3	-1.1	-0.09	-0.06	-0.04	-0.02
3	0.0	0.2	0.4	0.6	0.8	0.9	1.1	1.3	1.5	1.6
4	1.8	2.0	2.1	2.3	2.4	2.6	2.8	2.9	3.1	3.2
5	3.3	3.5	3.6	3.8	3.9	4.1	4.2	4.3	4.5	4.6
6	4.7	4.9	5.0	5.1	5.3	5.4	5.5	5.7	5.8	5.9
7	6.0	6.2	6.3	6.4	6.5	6.6	6.8	6.9	7.0	7.1
8	7.3	7.4	7.5	7.6	7.7	7.8	7.8	8.1	8.2	8.3
9	8.4	8.5	8.6	8.8	8.9	9.0	9.1	9.2	9.3	9.4
10	9.5	9.7	9.8	9.9	10.0	10.1	10.2	10.3	10.4	10.5
11	10.6	10.7	10.9	11.0	11.1	11.2	11.3	11.4	11.5	11.6
12	11.7	11.8	11.9	12.0	12.1	12.2	12.4	12.5	12.6	12.7
13	12.8	12.9	13.0	13.1	13.2	13.3	13.4	13.5	13.6	13.7
14	13.8	13.9	14.0	14.1	14.2	14.3	14.4	14.6	14.7	14.8
15	14.9	15.0	15.1	15.2	15.3	15.4	15.5	15.6	15.7	15.8
16	15.9	16.0	16.1	16.2	16.3	16.4	16.5	16.6	16.7	16.8
17	16.9	17.0	17.1	17.2	17.3	17.4	17.5	17.6	17.7	17.8
18	17.9	18.0	18.1	18.2	18.3	18.4	18.5	18.6	18.7	18.8
19	18.9	19.0	19.1	19.2	19.3	19.4	19.5	19.6	19.7	19.8
20	19.9	20.0	20.1	20.2	20.3	20.5	*	*	*	*

* For $\frac{C+N}{N}$ greater than 20.6, assume that C/N = $\frac{C+N}{N}$. Conversion difference is less than 0.1 dB.

1. **Ensure that the Spectrum Analyzer noise floor to signal noise floor difference is 19.5dB or greater.**
2. **Select a resolution bandwidth that is less than or equal to 1/10th of the Carrier Symbol Rate.**


Bit Error Rate
1 X 10 0
5 X 10 -1
1 X 10 -1
5 X 10 -2
1 X 10 -2
5 X 10 -3
1 X 10 -3
5 X 10 -4
1 X 10 -4
5 X 10 -5
1 X 10 -5
5 X 10 -6
1 X 10 -6
5 X 10 -7
1 X 10 -7
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Relative Eb/No

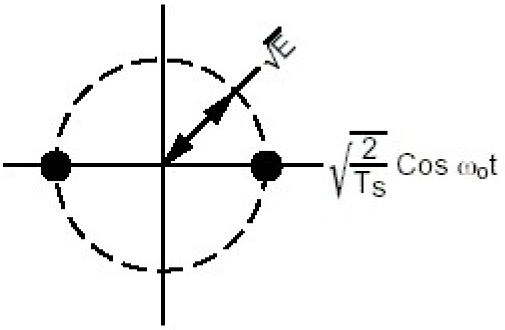


Illustration 4: BPSK Constellation

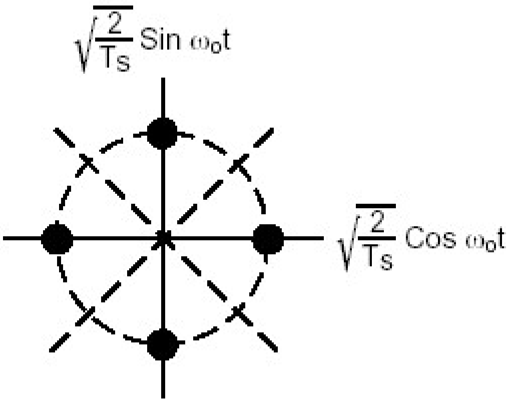

$\sqrt{\frac{2}{T_s}}$ Sin $\omega_0 t$
$\sqrt{\frac{2}{T_s}}$ Cos $\omega_0 t$

Illustration 2: QPSK Constellation

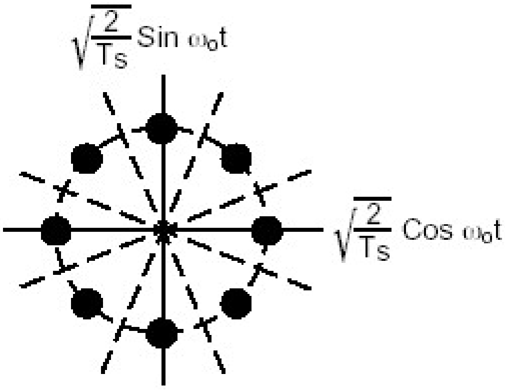

$\sqrt{\frac{2}{T_s}}$ Sin $\omega_0 t$
$\sqrt{\frac{2}{T_s}}$ Cos $\omega_0 t$

Illustration 5: 8PSK Constellation

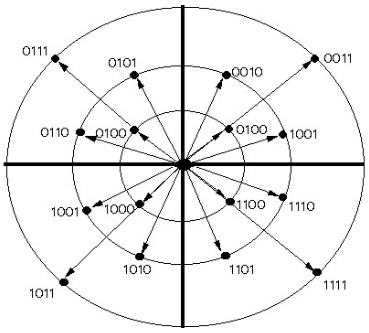


Illustration 6: QAM Constellation

Glossary

BANDWIDTH: A measure of radio frequency (RF) use or capacity. A terrestrial broadcast television channel, for example occupies a RF bandwidth of 6 MHZ or six million cycles per second while a telephone voice transmission requires a RF bandwidth of only 3 KHz or 3,000 cycles per second.

BAUD: A description of the rate of data transmission usually expressed in bit rate measurements of thousand bits per second (Kb/s). A 14.4 Baud modem for example will accommodate a bit rate of 14,400 bits per second.

C-BAND: A portion of the Radio Frequency (RF) spectrum located between 4 GHz and 8 GHz, a part of which is dedicated to satellite communications. Satellite downlink frequencies are located between 3.7 GHz and 4.2 GHz and uplink frequencies are located between 5.925 GHz and 6.425 GHz.

CLIFF EFFECT: A characteristic of the digital transmission of RF signals where there is a radical change in reception quality which results from a small change in reception power. By comparison, when an analog RF signal approaches the fringes of acceptable reception power, the television picture begins to experience gradual degradation with increasing sparkles or snow. As a digital RF signal reaches the fringes of acceptable reception power, there is no discernible degradation of picture quality until the level of reduced power reaches a threshold. At that point, picture quality changes from perfect to no picture.

COMMON CARRIER: An entity which provides communication transmission facilities for use by other entities or which carries other entity's communications signals. Examples of common carriers include the telephone or telegraph companies and the companies which own communications satellites. Common carriers are subject to tariff regulation and must file rates for specific services with appropriate regulatory agencies such as the FCC or state regulatory agencies.

DAMA (DEMAND-ASSIGNED MULTIPLE ACCESS): A process whereby satellite transponder channels are assigned for telephony transmission on the basis of immediate traffic demands.

DECIBEL: The difference (or ratio) between two signal levels; used to describe the effect of system devices on signal strength. For example, a cable has 6 dB signal loss or an amplifier has 15 dB of gain. This is useful since signal strengths vary logarithmically, not linearly. Since the dB scale is a logarithmic measure, it produces simple numbers for large-scale variations in signals. It is very useful because adding and subtracting whole numbers can calculate system gains and losses. Every time you double (or halve) the power level, you add (or subtract) 3 dB to the power level. This corresponds to a 50 percent gain or reduction. 10 dB gain/loss corresponds to a tenfold increase/decrease in signal level. A 20 dB gain/loss corresponds to a hundred-fold increase/decrease in signal level. In other words, a device (like a cable) that has 20 dB loss though it will lose lots of its signal by the time it gets to the other side. Thus, big variations in signal levels are easily handled with simple digits.

DECIBEL MILLIWATT: A signal strength or power level; 0 dBm is defined as 1 mW (milliWatt) of power into a terminating load such as an antenna or power meter. Small signals are negative numbers (e.g. -83 dBm).

FCC (FEDERAL COMMUNICATIONS COMMISSION): This is the national regulatory body for interstate telecommunications in the United States. The commission consists of five members all nominated to a specific term by the President of the United States and confirmed by the Senate. The current chairperson is Reed Hundt. The authority of the commission is contained within the Communications Act of 1934 as amended (most notably by the Telecommunications Act of 1996).

FSS (FIXED SATELLITE SERVICE): The segment of Ku-band satellite service established by the FCC to be provided from medium power satellites. These satellites are separated in orbit by at least 2 degrees. RF signals are transmitted to FSS satellites in the 14GHz to 14.5GHz range and received from the satellite in the 11.7GHz to 12.2GHz range.

GEO (GEOSYNCHRONOUS EARTH ORBIT): This is the orbital altitude of 35,580 km (22,237 miles) above the earth's surface where a satellite's velocity matches with the rotation of the earth. A satellite which is in a GAO position above the earth's equator (geostationary) will appear from the earth to be occupying a stationary position. The geosynchronous earth orbit is also referred to as the Clarke orbit (named in honor of Arthur C. Clarke, a science fiction writer who first postulated the characteristics of this orbit in 1945).

INCLINED ORBIT: A condition in which a satellite is unable to maintain a geostationary position above the earth's equator. Almost all satellites generate electrical power to operate their transponders by converting the sun's energy to electricity. Energy used for station keeping, that is keeping the satellite within a very narrow range of movement north or south of the equator, is stored on board the spacecraft and over time is dissipated, usually after passage of the design life of between 8-15 years. When this fuel runs out, the satellite is no longer able to keep station above the equator. Since satellite uplink and downlink antenna are aimed at a particular point above the equator, a satellite in inclined orbit will move in and out of the antenna's "range of vision" as the satellite's orbit fluctuates north and south of the equator.

ISL (INTER-SATELLITE LINK): A satellite architecture whereby two or more satellites are configured in such a way that they may communicate directly with one another.

ITU (INTERNATIONAL TELECOMMUNICATIONS UNION): A United Nations treaty organization which supports procedures for the international allocation of the radio frequency (RF) spectrum and provides the platform for the World Radio Conference (WRC), a biannual meeting of world communication leaders. The WRC publishes 'International Radio Regulations' for the RF spectrum. The ITU conducts ongoing policy and study group sessions.

KA-BAND: A portion of the RF spectrum located between 18 GHz and 31 GHz. Downlink frequencies for satellite communications are located in the 20 GHz range and uplink frequencies are located in the 30 GHz range.

KU-BAND: A portion of the RF spectrum located between 10.9 GHz and 17 GHz, a part of which is dedicated to satellite communications. Satellite downlink frequencies are located between 11.7 GHz and 12.2 GHz and uplink frequencies are located between 14 GHz and 14.5 Ghz.

L-BAND: A portion of the RF spectrum located between 500 MHZ and 1500 MHZ. The RF frequencies between 950 MHZ and 1450 MHZ are dedicated to mobile communications.

LEO (LOW EARTH ORBIT): Until recently, a distinction was made between LEO and MEO (Medium Earth Orbit) orbital classifications. LEO was classified as an earth orbit with an altitudeof between 200-300 kms. The MEO designation classified orbits between 300 kms and orbits approaching the geostationary orbit. Current literature frequently refers to any orbital distances from earth of less than that for geostationary orbit as being LEO. Satellites which are placed in LEO orbits move rather swiftly in relation to the earth, generally from a westerly to an easterly direction. The relative motion of the satellite in relation to the earth slows as the altitude of the satellite increases. The rapid relative movement in low earth orbit enables data gathering and communication satellites to cover large areas of the earth's surface in short periods of time. The space shuttle is injected into a low earth orbit, for example.

BIG LEO: A space segment architecture which consists of a constellation of many satellites in Low Earth Orbit in a configuration which will permit the delivery of global mobile telephony and data services. The system is served by a network of ground stations which provides gateway access from terrestrial networks as well as management, control and orbital correction functions. This system is intended to provide telecommunication service to remote sites, high latitude geographic locations which are not accessible by GEO satellites and to omnidirectional antenna on handset transceivers. Examples of Big LEO systems include Teledesic and Iridium.

LITTLE LEO: A space segment architecture which is similar to that of Big LEO system but which carries data only. The remote collection and transmission of utility meter data would be an example of a service provided by a Little LEO satellite system. Often, store and forward technology is used to facilitate communication with the ground segment.

LOW-POWER SATELLITE: A satellite with less than 30 Watts of transponder radio frequency (RF) transmitting power.

MEDIUM-POWER SATELLITE: A satellite with greater than 30 Watts but less than 100 Watts of transponder radio frequency (RF) transmitting power.

MEO (MEDIUM EARTH ORBIT): See LEO (LOW EARTH ORBIT).

MID BAND: A portion of the VHF (Very High Frequency) RF spectrum located between television channels 6 and 7 (88 MHZ to 174 MHZ) which has been reserved by the FCC for air, maritime and land mobile units, FM radio and aeronautical and maritime navigation. The frequencies between 108 MHZ and 174 MHZ can be used to provide additional channels on cable television systems.

CDMA (CODE DIVISION MULTIPLE ACCESS): A multiple access scheme whereby ground station uplinks access a satellite transponder using spread-spectrum modulations and orthogonal codes to avoid interfering with other transmissions using the same transponder. In contrast with the FDMA scheme which attempts to minimize the transmitted bandwidth, in this scheme all users transmit signals simultaneously across all of the dedicated multiple access channel. Receivers use a code corresponding to the transmission code to demodulate the signal or separate it from other signals on the channel.

FDMA (FREQUENCY DIVISION MULTIPLE ACCESS): A multiple access scheme whereby each ground station uplink is assigned a specific frequency slot and bandwidth for one of the multiple carriers within a specific satellite transponder. This scheme is usually used in conjunction with Frequency Modulation. The FDMA scheme may be divided into two categories, Multiple Channel Per Carrier and Single Channel Per Carrier.

TDMA(TIME DIVISION MULTIPLE ACCESS): A multiple access scheme whereby many users may access a single carrier by time sharing. A digital signal is compressed in packets which are transmitted to thecarrier in bursts. These packets are processed into consecutive time segments which do not overlap. Instructions are sent to a receiver which identify the packets representing a particular transmission (ie. Every 4th packet).

SPECTRUM: The entire range of frequencies which measure electromagnetic energy is called the spectrum. The Radio Spectrum (less than 300 GHz bandwidth) is divided into subsets of various bandwidths which are identified by their frequency characteristics. For example, the Ku-band subset includes those frequencies of the radio frequency (RF) spectrum between 11.7 Ghz and 14.5 GHz while the C-band subset includes those frequencies between 3720 MHZ and 6405 MHZ.

TRANSPONDER: A combination receiving and transmitting antenna on a communications satellite. A frequency converter is also including in the transmit/receive package which converts the uplinked signal frequency to a transmission or downlink frequency.

X-BAND: A portion of the RF spectrum located between 7 GHz and 8 GHz which is dedicated to the United States Military for satellite communications.

NOTICE

This page intentionally left blank.

Field Notes

A few notes I've taken and keep around to well.. remember. Stupid little shit that doesn't deserve an entire article, really. Also included are stupid jokes and a few other random bits. No, I didn't want to put it into Lost Signals, because it's not rhetoric. Mostly, anyway.

1. MAU's - Harris-Dracon Division - Metallic Access Units (Model 24845-001)
   - dialup 300/1200 baud (8N1)
   - Factory-Provided Password is 4372266
   - Won't announce itself until password is entered.

   More information (anything you want to know) is available in the tech. manual. You can download it from harris.com. The menus are simple and intuitive.

2. Q: AHHH!!! Your horrible 'zine made me shit my pants.. what should I do, lawg?!
   A: EASY. Remove said pants. Throw them out the window. Remove your underwear. Also throw them out afformentioned window. Continue to strip until all of your clothes are outside. Now walk calmly to the bathroom and SHOWER THE FUCKING SHIT OFF OF YOUR DIRTY BODY. Collect your clothes in a garbage bag and launder them immediately. Remember to stay calm throughout this process.

3. Use red lenses on your flashlight when dumpster diving as it's less visible at further distances. If you can use an LED light, and if you really want to use a head lamp. It looks funny.. but an extra free hand is just that.

4. Communism is funny. If you don't get it, you never will.

5. If an operator refuses to op-divert a call for you due to a 'new policy' or whatever, just say thanks anyway and hangup. Call back, get a new operator and inform them you are technincian (from the RBOC/IXC/CLEC providing service to the phone you are using) performing tests and need to place a toll-free call. Be quick with a fake/stolen employee number if need-be.

6. If you spoof the CND/ANI to the number your calling into Sprint PCS phones that has the passcode security disabled you can access their VMB. I'm told this works on T-Mobile as well.

Thoughts on Handscanning

Ah, one of the oldest trick in the bag: the handscan. It can be invaluable when you need to find numbers. This is just a quick list of tips to make your scanning go faster and be more effective.

1) Use a blacklist. You'll figure something out, or you'll abuse the shit out of grape.php because you're gay. OCR a damn phone book, or order one on CD for your local area (RBOC's have these available).

2) Dumpster Diving. Whenever you find numbers in the telco's dumpster, note the NXX and Suffix's of any numbers you come across. SBC has a habit of blocking off entire NXX-XX blocks, and so might your local telco. Note whether or not the NXX's you've found in the dumpster are listed as a valid (read: assigned) NXX. If it's not, then that means they could be running test numbers in an 'oddball' prefix, and you may find a great deal more.

3) NXX-99XX? I haven't really had all that much here, honestly. SBC likes to put their stuff in other places, from what I've found. Verizon and Bellsouth, do however (from what I'm told). Find what works for your RBOC and rock with it.

4) Randomize. Yeah. Not entirely important if you've blacklisted enough numbers out of your list.

5) Use a speed dial. Program your speed-dial to dial the first X number of digits for you (or the remainder if you're doing a prefix scan). This should speed things up a bit.

6) Don't make your note about the current number until you've dialed the next one. This will save you a few seconds per call (which adds up), and will make up for the fact that you need to wait at least 10 rings before moving on. Sometimes auto-attendants are programmed to wait that long, sometimes modems are too.. and sometimes people need to run back from the shitter to answer the phone.

Ok that's it. You should be a little more effective in getting those numbers you so desire. I spend a lot of time scanning, and these are just a few things I've learned over time. Use it if you like, don't if you don't. The most important thing (unless you're just stumbling around for random dialups, pbx's, etc) is to target your scans. See items one and two for more on targeting.

Code listing for grape.php

```
#!/usr/local/bin/php
<?
 function usage()
 {
   sprintf("Google Rape version 1.0.0 by lawg\n \
   Usage:\n%s NPA NXX START END (blacklist file)\n \
   START and END need to be values between 0-9999\n \
   with the start lower than the end, dipshit.\n",
   $argv[0]);
  exit;
 }

 if(empty($argv[1]) || empty($argv[2]) || empty($argv[3]) || empty($argv[4]))
   usage();
 $npa = $argv[1];
 $prefix = $argv[2];
 $st = intval($argv[3]);
 $end = intval($argv[4]);
 if($end < $st)
   usage();

 if(empty($argv[5]))
  $file = "blacklist.scn";
 else
  $file = $argv[5];

 $of = fopen($file, "a");

 for($k=$st;$k<$end;$k++)
 {
   $suff = sprintf("%d", $k);
   $suff = str_pad($suff, 4, "0", STR_PAD_LEFT);
   $scan = "$npa"."$prefix"."$suff";
   $buf = file_get_contents("http://www.google.com/search?q=".$scan);
   if(substr_count($buf, "Phonebook") > 0)
     fputs($of, $prefix.$suff."\n");
 }

 echo "\nDONE!\n";
 exit;
?>
```

Code listing for numgen.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <getopt.h>
#include <sys/time.h>

#define MAX_LENGTH 15
#define DEFAULT_BL "./blacklist.scn"

char **numbers;
char **blacklist;
char opfile[255];
char mask[MAX_LENGTH];
int mode;
char blfile[255];
const char def[11] = "0123456789";

int pw(x,y)
int x, y;
{
 int i;
 int ret = 1;
 for(i=0;i<y;i++)
  ret = ret * x;
 return(ret);
}
```

```
int gen(mask)
char *mask;
{
 int i=0, mask_loc, ndiv, div, x=0, y=0, pos, prevx=0, xpos[10],xcount=0, wbl=1, blcount=0,
cpy=0;
 FILE *bl;
 char tblh[MAX_LENGTH];
 char c[2];
 size_t size=1;

 memset(tblh, '\0', MAX_LENGTH);

 if(mode & 0x08) {
  if((bl = fopen(blfile, "r")) == NULL) {
   if(mode & 0x02) {
    if((strcmp(blfile, DEFAULT_BL)) != 0) {
     fprintf(stdout, "User-specified blacklist (%s) not found... attempting to open default
(%s).\n", blfile, DEFAULT_BL);
      if((bl = fopen(DEFAULT_BL, "r")) == NULL) {
       fprintf(stdout, "%s also not found... generating without a blacklist!", DEFAULT_BL);
       wbl = 0;
      }
    }
    else {
     fprintf(stdout, "%s not found... generating without a blacklist!\n", DEFAULT_BL);
     wbl = 0;
    }
   }
   else {
    if((strcmp(blfile, DEFAULT_BL)) != 0) {
     if((bl = fopen(DEFAULT_BL, "r")) == NULL)
      wbl = 0;
    }
    else
     wbl = 0;
   }
  }
 }
 else
  wbl = 0;

 if(wbl)
 {
  blacklist = (char **)malloc(100000 * sizeof(char *));
  /* read the blacklist into an array */
  while(size) {
   size = fread(c, 1, 1, bl);
   if(c[0] == '\n') {
    blacklist[i] = strdup((const char *)tblh);
    memset(tblh, '\0', MAX_LENGTH);
    x=0;
    i++;
   }
   else {
    tblh[x] = c[0];
    x++;
   }
  }
  blcount = i;
 }
 for(x=0;x<strlen(mask);x++) {
  if(mask[x] == 'x') {
   xpos[xcount] = x;
   mask[x] = '0';
   xcount++;
  }
 }
 div = pw(10, xcount);

 numbers = (char **)malloc(div * sizeof(char *));

 fprintf(stdout, "%d possible numbers, %d exist in blacklist.\n", div, blcount);

 x=y=0;
 while(x < div) {
  prevx=0;
  ndiv = div/10;
  mask_loc = 0;
  while(mask_loc < xcount) {
   pos = (x/ndiv)-prevx;
   mask[xpos[mask_loc]] = def[pos];
   prevx = (x/ndiv)*10;
   mask_loc++;
   ndiv = ndiv/10;
  }
  if(wbl) {
   i=0;
   while(i<blcount) {
    if(strncmp(mask, blacklist[i], strlen(mask)) == 0)
      goto notcopy;
    i++;
   }
  }
  numbers[y] = strdup((const char *)mask);
  y++;

notcopy:
  x++;
 }
 if(writefile(y) != y)
  return(-1);

 for(i=0;i<x;i++)
  free(numbers[i]);

 return(0);
}

int writefile(last)
int last;
{
 unsigned int i, rloc, x, k;
 FILE *of;

 struct timeval *bs;
 gettimeofday(bs, NULL);
 srandom((unsigned long)bs->tv_usec);
 x=last-1;

 if((of = fopen(opfile, "a")) == NULL) {
  fprintf(stderr, "Error opening %s for file output!", opfile);
  return(-1);
 }
 if(mode & 0x02)
  fprintf(stdout, "Writing %d entries into %s in ", last, opfile);

 if(mode & 0x01) {
  if(mode & 0x02)
   fprintf(stdout, "random order.\n\n");
  while(x > 0) {
   rloc = (int)random();
   /* rloc = rand() >> 8; */ /* heh. it works for 10-10000... */
   if((rloc <= x) && (rloc > 0)) {
    fprintf(of, "%s\tUNSCANNED\n", numbers[rloc]);
    if(mode & 0x02)
```

```
     fprintf(stdout, "%s\tUNSCANNED\n", numbers[rloc]);
    numbers[rloc] = numbers[x];
    x--;
   }
  }
  fprintf(of, "%s\tUNSCANNED\n", numbers[0]);
  if(mode & 0x02)
   fprintf(stdout, "%s\tUNSCANNED\n", numbers[0]);
 }
 else {
  if(mode & 0x02)
   fprintf(stdout, "sequential order.\n\n");

  for(i=0;i<last;i++) {
   fprintf(of, "%s\n", numbers[i]);
   if(mode & 0x02)
    fprintf(stdout, "%s\tUNSCANNED\n", numbers[i]);
  }
 }
 if(mode & 0x02)
  fprintf(stdout, "All Done!\n");

 return(i);
}


void usage(missing)
int missing;
{
 fprintf(stderr, "ROB (randomizing, optimizing, blacklisting) scan list generator v1.0.0\n");
 if(missing & 0x01)
   fprintf(stderr, "You MUST specify a mask!\n");
 if(missing & 0x02)
 {
   fprintf(stderr, "Available Options:\n");
   fprintf(stderr, "\t--mask/-m          The mask you want to generate from.\n");
   fprintf(stderr, "\t                    ex. 999xxx\n");
   fprintf(stderr, "\t--not-random       Do not randomize output list\n");
   fprintf(stderr, "\t--blfile           Specify a blacklist file\n", DEFAULT_BL);
   fprintf(stderr, "\t--file/-f          Output file (defaults to <mask>.scn)\n");
   fprintf(stderr, "\t--silent           Run silently\n");
   fprintf(stderr, "\t--no-blacklisting  Don't use a blacklist file\n");
   fprintf(stderr, "\t--help/-h          This. =)\n");
 }

}

int main(argc, argv)
int argc;
char **argv;
{
 int i=0, l=0, o=0, c, k=0;
 memset(opfile, '\0', sizeof(opfile));
 memset(mask, '\0', MAX_LENGTH);
 mode=0x00;
 mode ^= 0x01; /* randomize by default */
 mode ^= 0x02; /* verbose output by default */
 mode ^= 0x08; /* blacklist by default */

 while(1) {
  int option_index=0;
  static struct option long_options[] = {
   {"not-random", 0, 0, 'r'},
   {"mask", 1, 0, 'm'},
   {"blfile", 1, 0, 'b'},
   {"file", 1, 0, 'f'},
   {"silent", 0, 0, 's'},
   {"no-blacklisting", 0, 0, 'n'},
   {"help", 0, 0, 'h'},
   {0, 0, 0, 0}
  };

  c = getopt_long(argc, argv, "m:b:rsn:h", long_options, &option_index);
  if(c == -1)
   break;
  switch(c) {
   case 'm':
    strncpy(mask, optarg, MAX_LENGTH);
    i = 1;
    break;
   case 'r':
    mode ^= 0x01;
    break;
   case 'b':
    strncpy(blfile, optarg, 254);
    l = 1;
    break;
   case 'f':
    strncpy(opfile, optarg, 254);
    o = 1;
    break;
   case 's':
    mode ^= 0x02;
    break;
   case 'n':
    mode ^= 0x08;
    break;
   case 'h':
    usage(k ^ 0x02);
    return(0);
    break;
  }
 }

 if(!l)
  snprintf(blfile, 254, "%s", DEFAULT_BL);
 if(!o)
  snprintf(opfile, 254, "%s.scn", mask);
 if(i == 0)
  usage(k ^ 0x01);
 else
  gen(mask);

 return(0);
}
```

Thoughts on Social Engineering

So you're scanning along in some crazy telco reserved block o' numbers and hear a screeching.. damn, it's a fucking fax machine again. You jot it down and keep scanning... Hold on. This could be really useful. You look over at the piles of telco docs sitting in the corner of your room and see lying squarely on top of it is a faxed internal memo.

Hmm... TWO PLUS TWO IS FIVE!%!^%#$!$%!

I digress. To whom does this new fax number you've just found belong to? That's where you must put your standard SE skills into practice (unless you happen to have an internal telco phone directory. heh.) and find out. It really shouldn't be too hard.. "Hi, is XXX-XXXX your fax number? Yeah? great!..." Really, it should be that easy. Also, try asking various offices for their fax numbers (especially non-local) and you should have some extra fun. Having a fax line for a few provisioning offices should prove useful.

Now you basically need to use your trashed faxes as a basis for what your going to send into their office. Use photoshop or whatever you like... or just use a printer and a copy machine to get the results you want. However you do it, make it look at least passably legit. Fax machines tend to fuck quality all up and most people trust them if they have a proper cover sheet. You may find yourself chopping and editing (and possibly forging a signature or two) for a couple hours to get it looking just right.

Wording is important. Take note of how they word stuff on their own memos. Something like "You need to send me the docs on that spiffy xxxx along with your passcode..." won't cut the cake, bro. Presentation is key.

If you want them to fax you back something, just give them the number to kinko's and sit there and mack on the girl working the counter until your super-secret docs come rolling in. I'm sure if your going this far you have an idea of how to get the fax full'o'infoz without it being traced back to you.

Or use Efax (http://www.efax.com) and sign up for a free account. It's only a 30 day trial, and it's receive only... but it works for what you're going to do with it. If you card an account [Note: Ethertech.org does NOT condone evil credit-card fraud!] you can get a toll-free number and will be able to send faxes as well. There is also a GTK interface floating around on freshmeat somewhere, so check it out.

A few ideas that come to mind: asking various operators to send you pages out of their handbook; requesting that the engineering department send all relevant data on any current projects; requesting an updated phone directory. The list goes on. Be creative. Insist that the network in your office is down or your email got messed up. Computers are CRAZY and nobody's even sure how they work, so it could be anything!

And on that note, I dive into something a little more sketchy and possibly harder to pull off. Claim the email server is down, your email account is busted or whatever and convince them to send it to your gmail account. It may work, and you're not out that much if it doesn't. This would only really work if they have the docs on their computer (which is very likely). I'm not certain that their manuals would be, but they very well could be.

While I'm waxing poetic about Social Engineering, I suppose I'll mention this too: Get THEIR employee number, too. No matter who you're speaking with, why you're talking to them or where they work, make sure you insist your manager has a new policy to log employee numbers. They'll come in handy later, I'm sure.

Ok, that's it. Have fun, be creative... and get some info.

The Mindset: z0mbiedog

Richard, known as 'z0mbiedog' in some circles, awoke with a start. He had a feeling about today. A good feeling. It was saturday morning and of course he was pleased to be out of school for the weekend, but it was more than that. He grabbed his coffee mug and headed for the kitchen, barely acknowledging his mother's 'good morning' babble. He grunted and poured himself some coffee.

His parents thought he was weird, but figured it must be a phase all fifteen year olds went through.

After grabbing a package of pop-tarts out of the kitchen cabinet he headed back to his room and turned his main computer's monitor on. He was greeted with a dull black console sitting still.

Keying in a few commands while ripping into the pop-tarts yeilded what he was looking for. The wardial he had set into action the night before had proved to be successful.

He finished the breakfast snack and rose to his feet.

He reached for the phone line that was plugged into the back of his computer and unplugged it, gave it a good yank, and drew it back in through his window.

He would go close the termination box on his neighbor's house later, and he figured they wouldn't notice it hanging open seeing as how they were on vacation.

A few more commands at the keyboard and his screen filled with the capture file from the dialer, and it showed him exactly what he wanted.

Tonight he would be the fucking man.

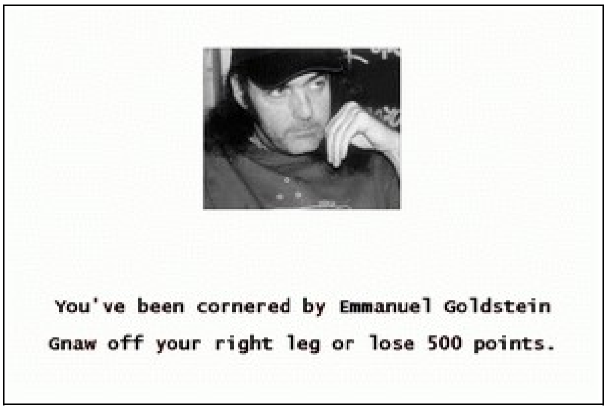
Tonight he would feel like a god.

He smiled and drank his coffee.

Code Listing for justcurio.sex

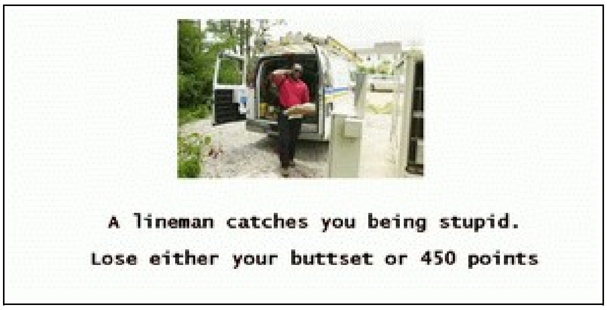
```
<?PHP
/* JUSTCURIO.SEX.PHP */
require("http.php"); /* http://www.phpclasses.org/httpclient */

$getqsreg = "/<a href='answer\/(\w*)' target='_parent'>/";
$ansfieldreg = "/<input type=\"hidden\" name=\"id\" id=\"id\" value=\"([\d]*)\">/";
$answerurl = "http://www.justcurio.us/answer.php";
$questionsource = "http://www.justcurio.us/questions.php";

$verbose = 0;
$quiet = 0;

while(1) {
  $randompage = file_get_contents($questionsource);
  preg_match_all($getqsreg,$randompage,$matches);

  foreach($matches[1] as $qs) {
    $tmp = file_get_contents("http://www.justcurio.us/answer/".$qs);
    preg_match($ansfieldreg,$tmp,$id);
    $qsid    = $id[1];
    $answer  = `sex`;
    $filename       = "";
    $attempted      = "1";
    set_time_limit(0);
    $http=new http_class;
    $http->timeout=0;
    $http->data_timeout=0;
    $http->debug=0;
    $http->html_debug=0;
    $http->user_agent="Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)";
    $error=$http->GetRequestArguments($answerurl,$arguments);
    $arguments["RequestMethod"]="POST";
    $arguments["PostValues"]=array(
             "answer"=> $answer,
             "attempted"=>1,
             "qs"=>$qs,
             "id"=>$qsid,
             "filename"=>$filename);
    $arguments["Referer"]="http://www.google.com/"; /* generic, pls advise */
    $error=$http->Open($arguments);

    if($error=="") {
      $error=$http->SendRequest($arguments);
      if($error=="") {
        if($verbose == 1) {
         echo "Request: ".$http->request."\n";
         echo "Request headers:\n";
       }
       for(Reset($http->request_headers),$header=0;
                $header<count($http->request_headers);
               Next($http->request_headers),$header++) {
         $header_name=Key($http->request_headers);
         if(GetType($http->request_headers[$header_name])=="array") {
           for($header_value=0;
               $header_value<count($http->request_headers[$header_name]);
             $header_value++) {
            if($verbose == 1) {
             echo $header_name.": ";
             echo $http->request_headers[$header_name][$header_value],"\n";
            }
           }
         }
         else {
           if($verbose == 1) {
             echo $header_name.": ".$http->request_headers[$header_name],"\n";
           }
         }
       }
       $headers=array();
       $error=$http->ReadReplyHeaders($headers);
       if($error=="") {
         if($verbose == 1) { echo "Response headers:\n"; }
         for(Reset($headers),$header=0;
             $header<count($headers);
             Next($headers),$header++) {
           $header_name=Key($headers);
           if(GetType($headers[$header_name])=="array") {
             for($header_value=0;
                 $header_value<count($headers[$header_name]);
                $header_value++) {
              if($verbose == 1) {
                echo $header_name.": ".$headers[$header_name][$header_value],"\n";
              }
             }
           }
           else {
             if($verbose == 1) {
              echo $header_name.": ".$headers[$header_name],"\n";
             }
           }
         }
         if($verbose == 1) { echo "\n\n"; }
         if($verbose == 1) { echo "Response body:\n"; }
         for(;;) {
            $error=$http->ReadReplyBody($body,1000);
            if($error!="" || strlen($body)==0)
              break;
            $accepted = substr_count($body, "can't accept that answer.");
         }
       }
      }
      $http->Close();
    }

    if($quiet == 0) {
      if(strlen($error)) { echo "Error!!\n"; }
      echo "http://www.justcurio.us/".$qs." -> ";
      if($accepted == 1) { echo "No Good!!\n"; }
      else { echo "Trolled!!\n"; }
      echo "---------> ".$answer."\n";
      unset($accepted);
    }
  }
}
?>
```

Phreak Cards for 1000 Blank White Cards



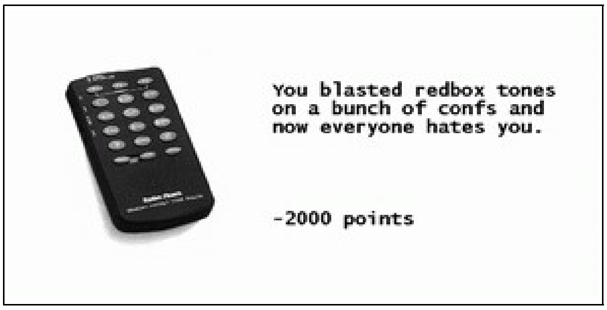
You've been cornered by Emmanuel Goldstein

Gnaw off your right leg or lose 500 points.



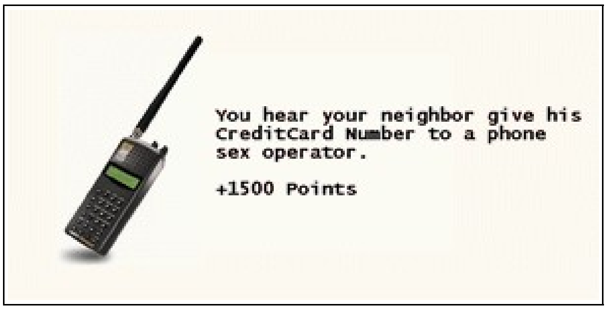
A lineman catches you being stupid.

Lose either your buttset or 450 points



You blasted redbox tones
on a bunch of confs and
now everyone hates you.

-2000 points



You hear your neighbor give his
CreditCard Number to a phone
sex operator.

+1500 Points

More information on 1000 Blank White Cards can be found at:
http://www.trouserarousal.nu/cards/

Rant and Editorial

AMERICAN TELEPHONE & TELEGRAPH CO.
LOCAL AND LONG DISTANCE TELEPHONE
BELL SYSTEM
AND ASSOCIATED COMPANIES

So there you have it. Issue One. I hope you learned something. There may or may not be an issue two as I'm usually too busy to do this stuff. We'll see about it.

All the code featured in this issue are available for download at http://ethertech.org/pub/code/ if you don't feel like chopping it up out of a pdf file.

This issue has been brought to you by the sounds of Jimmy Buffet, NOFX and the letter 'P'.

Shouts: #bantown (weev, sdf, oclet, other funny people), lanterndog, jackd, crow, pinguino, the clone, multiplx, ticom
Gayz: just about anyone else.