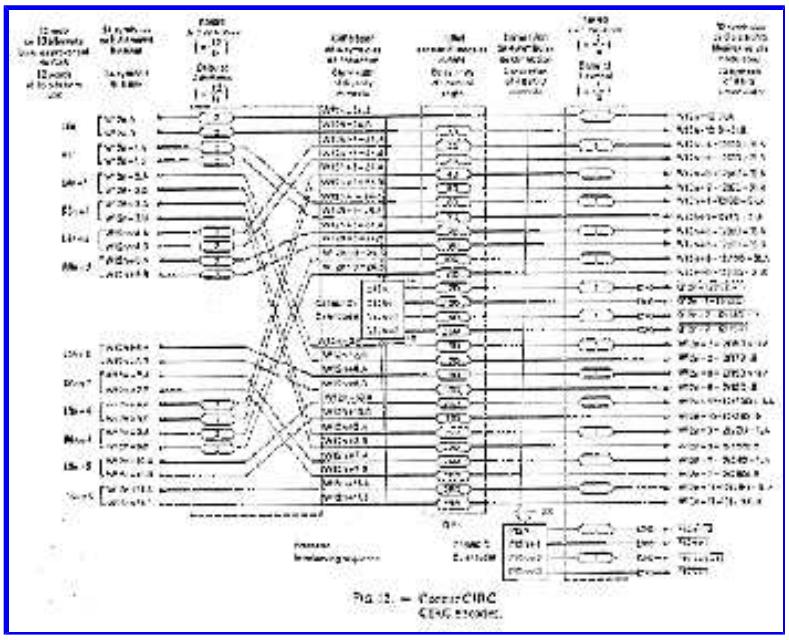
IEC-908...The BIG picture

The encoding of digital audio on CD player is governed by IEC 908. This standard is available in the library for your perusal. (Notice that every other page is missing -- this is because the standard is written in both French and English and I took out the French pages!) This information is also covered more generally in Chapter 3 of Ken Pohlman's book The Compact Disk Handbook, (A-R Editions, 1992).

CD players use parity and interleaving techniques to minimize the effects of an error on the disk. In theory, the combination of parity and interleaving in a CD player can detect and correct a burst error of up to 4000 bad bits -- or a physical defect 2.47 mm long. Interpolation can conceal errors up to 13,700 or physical defects up to 8.5 mm long.

The entire error detection and correction algorithm is summarized on the following table.



This is Figure 12 from the IEC 908 standard. This table will be described in more detail below.

The original musical signal is a waveform in time. A sample of this waveform in time is taken and "digitized" into two 16-bit words, one for the left channel and one for the right channel.

For example, a single sample of the musical signal might look like:

L1 = 0111 0000 1010 1000

R1 = 1100 0111 1010 1000

Six samples (six of the left and six of the right for a total of twelve) are taken to form a frame.

L1 R1 L2 R2 L3 R3 L4 R4 L5 R5 L6 R6

The frame is then encoded in the form of 8-bit words. Each 16-bit audio signal turns into two 8-bit words.

L1 LI R1 R1 L2 L2 R2 R2 L3 L3 R3 R3 L4 L4 R4 R4 L5 L5 R5 R5 L6 L6 R6 R6

This gives a grand total of 24 8-bit words. This is column two on the IEC 908 table.

The even words are then delayed by two blocks and the resulting "word" scrambled. This delay and scramble is the first part of the interleaving process.

The resulting 24 byte word (remember, it has an included two block delay -- so some symbols in this word are from blocks two blocks behind) has 4 bytes of parity added. This particular parity is called "Q" parity. Parity errors found in this part of the algorithm are called C1 errors. More on the Q parity later.

Now, the resulting 24 + 4Q = 28 bytes word is interleaved. Each of the 28 bytes is delayed by a different period. Each period is an integral multiple of 4 blocks. So the first byte might be delayed by 4 blocks, the second by 8 blocks, the third by 12 blocks and so on. The interleaving spreads the word over a total of 28 x 4 = 112 blocks.

The resulting 28 byte words are again subjected to a parity operation. This generates four more parity bytes called P bytes which are placed at the end of the 28 bit data word. The word is now a total of 28 + 4 = 32 bytes long. Parity errors found in this part of the algorithm are called C2 errors. More on the P parity later too.

Finally, the another odd-even delay is performed -- but this time by just a single block. Both the P and Q parity bits are inverted (turning the "1s" into "0s") to assist data readout during muting.

An 8-bit subcode is then added to the front end of the word. The subcode specifies such things as the total number of selections on the disk, their length, and so on. More on this later.

Next the data words are converted to EFM format. EFM means Eight to Fourteen Modulation and is an incredibly clever way of reducing errors. The idea is to minimize the number of 0 to 1 and 1-0 transitions -- thus avoiding small pits. In EFM only those combinations of bits are used in which more than two but less than 10 zeros appear continuously.

For example, a digital 10 given as a binary 0000 1010 is an EFM 1001 0001 0000 00

Each frame finally has a 24-bit synchronization word attached to the very front end -- (just for completeness the word is (100000000001000000000010) and each group of 14 symbols is then coupled by three merge bits.

These merge bits are chosen to meet two goals:

1. No adjacent 1’s from neighboring EFM encoded words

Remember that there are lots of EFM words which end in "1" -- as one example, all the eight-bit binary words from 128 to 152 end in "1". Similarly, there are EFM words that start in "1". Thus, it is relatively straightforward to have to have adjacent EFM words that create adjacent "1s".

For example -- binary 128 and binary 57

10000000 in EFM is 00111001 in EFM is

01001000100001 10000000001000

2. The digital sum value is kept near zero

Minimizing the digital sum value is just an attempt to keep the average number of "0's" and "1's" about the same. The value of +1 is assigned to the "1" states and the value of -1 is assigned to the "0" states. Then, the value of the merge bit is chosen to maintain the average near zero.

SO! The final frame (which started at 6*16*2 = 192 data bits) now contains:

```
   1 sync word                 24 bits
 1 subcode signal             14 bits
6*2*2*14 data bits            336 bits
 8*14 parity bits             112 bits
 34*3 merge bits              102 bits

   GRAND TOTAL                588 bits
```

[Lost in the maze of pages-Reload Imagemap]