About 'Long Coding'

The VAG 'long codings' are strings of hexadecimal numbers.

To understand what is going on with the long coding, you must first have a grasp of decimal and binary numbers.
Hexadecimal (hex) numbers may contain two digits per number - for example, 01, 02, 03 - and are based on 16 rather than 10 like the 'normal' decimal system. This means that the letters A through F are used as digits to expand from base-10 to base-16.

Binary numbers are written with only zeros and ones. Usually a 'byte' is a group of 8 digits (zeros and ones) which represents a number. That number can be converted to decimal or hex values. For example, the decimal numeral 79 (whose binary representation is 01001111) can be written as 4F in hexadecimal (4 = 01000000, F = 00001111 in binary).

“Zeros and Ones” can be thought of as “false and true”, or “no and yes” values, respectively

A VAG 'long code' is a string of hexadecimal numbers. For example: 9AC003382D08850FC88F447300
When you look at this code, you have to mentally break it up into 'bytes' of 2 digits each.
Therefore, think of it as 9A C0 03 38 2D 08 85 0F C8 8F 44 73 00
It is read LEFT --> RIGHT, and each byte is identified by it's position. The first position is 'byte 0', the second is 'byte 1', and so forth.
In the example above, '9A' is byte 0, 'C0' is byte 1, '03' is byte 2, '38' is byte 3, and so forth.
Another way to express it is to say the value of byte 0 is '9A', the value of byte 1 is 'C0', the value of byte 2 is '03'.

These hex values are then individually converted to binary. In our example above, the first byte (byte 0) has a value of '9A'.
“9A” (hex) converts to a binary value of '10011010' Each 'zero' or 'one' binary digit is called a 'bit', and they are read RIGHT --> LEFT.
As above, you count 'zero, one, two, three'. That means that bit 0 of '10011010' has a value of 'zero' (or 'false' or 'no').

Here's an example of how this all breaks out:

When converted to binary, each bit in each byte corresponds to an on-or-off choice for each setting. In our example here, “*This Setting*” is off, “*That Setting*” is on, “*Foo*” and “*Bar*” are both on, and “*Baz*” and “*Blah*” are both off. If you change one of those settings by changing a zero to a one, or a one to a zero, then that changes the 8-character binary number, which then converts to a different 2-character hex number, which obviously changes the 'long code'.
Writing that new 'long code' into the car's CANBUS computer system causes the desired settings changes to take effect.

Byte 00:	9A (hex)	10011010 (binary)	
Bit:			
0		0	*This Setting*
1		1	*That Setting*
2		0	*The Other Setting*
3		1	*Foo*
4		1	*Bar*
5		0	*Baz*
6		0	*Blah*
7		1	*Whatever*

Make sure you understand how each description is worded - '1' may mean ENABLE a feature, or it may mean DISABLE a feature.

Release 1, 04 February 2006

How the spreadsheet works:

There are sheets in this workbook for 'Comfort System' and for 'Central Electronics'.
Each sheet is set up specifically for that module.

The sheets are locked down except for the grey cells. Those are the only ones you should be able to enter data into.

This spreadsheet is set up for US Letter paper (8.5"x11") Landscape orientation. Do *File --> Page Preview* to verify before printing.

In the cells indicated at the top of the sheet, enter the original long code read from your car. This is for reference, so you can put it back if things go wrong, and so that you can compare it against the new long code that will be generated by your edits.

The values entered will be automatically copied into the 'Original' values in column B, and then converted into binary and broken out to correlate the individual bits with their respective settings.

Now, you can change the values for the individual bits in Column C corresponding to the settings that you want to change.
CAUTION: You must have all of the grey cells in Column C configured correctly! Do Not leave any blank! Do not set them to 'zero' unless you intend that to be the setting. Everything that you set there will determine the long code that is generated!

The spreadsheet automatically calculates the complete binary value from the individual bits, then converts that back to hexadecimal, then copies the hex values up to form the new long code that is shown at the top of the sheet.

NOTE: Central Electronics bytes 5 - 17, 20 and 21 are not entered as individual bits. Instructions are provided for each. Those bytes require you to enter a integer for percent (e.g. 18 or 100), or a time in milliseconds (e.g. 1000) or other decimal value (e.g. 0 to 6) into a field which will calculate the hex byte for you.

YOU, AND ONLY YOU, ARE RESPONSIBLE FOR THE CONSEQUENCES OF WHATEVER HAPPENS TO YOUR CAR AS A RESULT OF EDITING AND USING THE LONG CODING IN THESE MODULES! BY USING THIS SPREADSHEET YOU AGREE TO HOLD ME COMPLETELY HARMLESS FOR ANY DAMAGE OR INCONVENIENCE THAT YOU MAY INCUR. IF YOU DO NOT AGREE, THEN CLOSE AND DELETE THIS FILE IMMEDIATELY!

This spreadsheet was designed in and for OpenOffice 2.x only. Translation and redistribution in .xls format is not supported by the original author (Len Laughridge <oooo.a3@gmail.com>). OpenOffice may be obtained for free from http://www.openoffice.org

The sheets and most cells in this spreadsheet are locked to prevent accidental damage. The password is 'password'.