How to build an EEC Definition file – Part 2 (Finding Parameters)

By Jaysen Anderson & Jason Bolger

There are a couple of ways to go about building a definition file to suit your desired EEC. This document attempts to cover the steps required in actually editing a bin.

If you do not understand the relationship between binary, hexadecimal and decimal, then we strongly recommend that you look here http://tunerpro.net/tutorials/UnderstandingHex.htm. Hex really isn't hard, and as its short hand for binary, you dont really need to be fluent in binary, just have an understanding of how it all is inter-related. To summarise it: Binary is base 2, Decimal is base 10, and Hexadecimal is base 16. There is so much info on the net about this, so we need not go in to much detail.

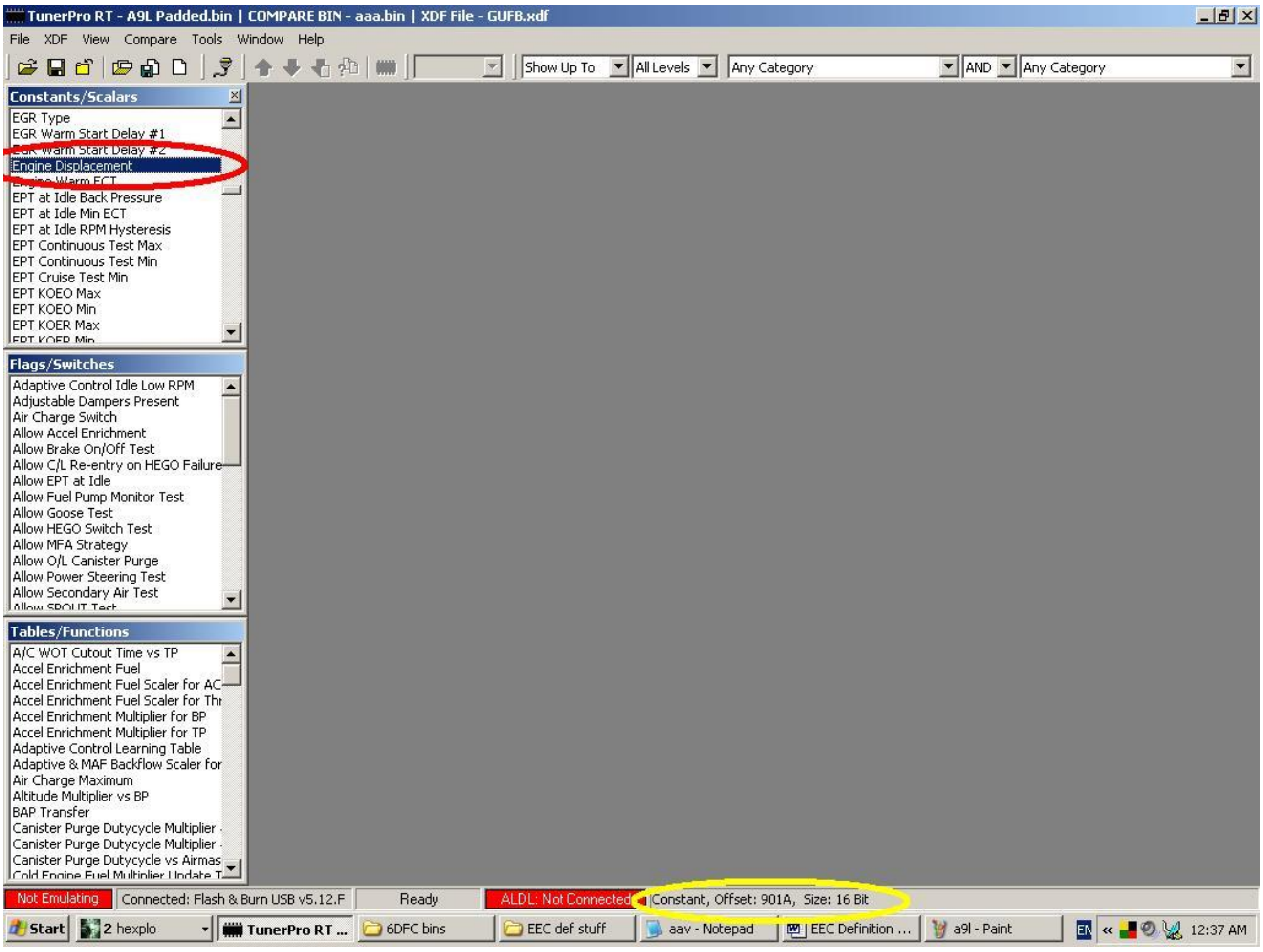
So now you know what is needed to read your EEC and program a chip with your edited parameters. Next is the software, which will be TunerPro RT. It's pretty easy to do, so we need not go in to much detail, however one thing to note is that you should set the start address to 0 and the end address to DFFF in an EECIV or 3FFFF in an EECV so that you read the whole bin. You can put your start address at 0x22000 in the EECV, but you will lose your EEC definition and bin code, and to run the xdf files that we have outlined above, you will need to repad the bin with the buffer you removed, so in summary, just start at 0.

Something else to add when dumping EEC bins is that you should have the ECU powered, but not in the operational mode, ie have it installed in the car but with the key in the off position. There are several 12V+ & Ground pins on the EEC connector,but you need to run power to the KAM 12V+ pin and the ground to the ECU ground (not the case ground). so check with your workshop manual or online to find which ones are the right ones if you choose to power up the ECU on the bench.

Next you need a GUI (graphical user interface) to tune parameters. Yes we know you can change things with a hex reader, but trust us, it gets old very quick. This is where TunerPro comes in, there are several others out there, however this is our preferred GUI. Tunerpro has very detailed instructions under the help menu, so again we need not go in to much detail. If you have already got your bin defined, you need not read any further, as you will already be able to read, manipulate and burn EEC binary to the necessary hardware. If you want to chase down other parameters to define for your EEC, or you need to convert a similar bin def file to the bin you plan on using, here is how you can do it.

On the next page is a screen shot of TunerPro. The Constant circled in red is the Engine Displacement, you can simply scroll down the lists at the side and choose any of these constants & scalars you wish to change. Below that is Flags and Switches which dictate operating conditions that occur (its not hard to figure out), and below that again is Maps & Tables. The information circled in yellow is where it gets important. The info there tells you that the selected item is a constant, that its hex address in the bin file is at 901A, and that the size of the constant is 16 Bit, or a word in programming lingo. This bin file is an A9L Mustang ECU and its def file is a GUFB.

Now you open up 2 applications of your hex editor, load the defined bin in to one of them, and the bin that your searching for date in the other. A couple of things to note, hex editors normally have the offset (hex location) down the left hand side, if it doesn't, we would advise against using it. Sometimes the editor will also give you the decimal equivalent to the highlighted data, in the screen shot on the following page for instance you can see that 901A & 901B are both highlighted, and down the bottom of the screen you can see "Word: 6986", this means you have selected a word in length (2 bytes) and that is the decimal equivalent with the LSB (Least significant byte) first (numbers are read backwards from left to right, reverse to how you would normally write it). If your editor doesn't have a search function, and a jump to offset function, its going to make life harder for you, and if you can, sort your columns in to 4 banks of 4 like it is here, it makes it easer to read.



TunerPro RT - A9L Padded.bin | COMPARE BIN - aaa.bin | XDF File - GUFB.xdf

Constants/Scalars

- EGR Type
- EGR Warm Start Delay #1
- EGR Warm Start Delay #2
- Engine Displacement
- Engine Warm ECT
- EPT at Idle Back Pressure
- EPT at Idle Min ECT
- EPT at Idle RPM Hysteresis
- EPT Continuous Test Max
- EPT Continuous Test Min
- EPT Cruise Test Min
- EPT KOEO Max
- EPT KOEO Min
- EPT KOER Max
- EPT KOER Min

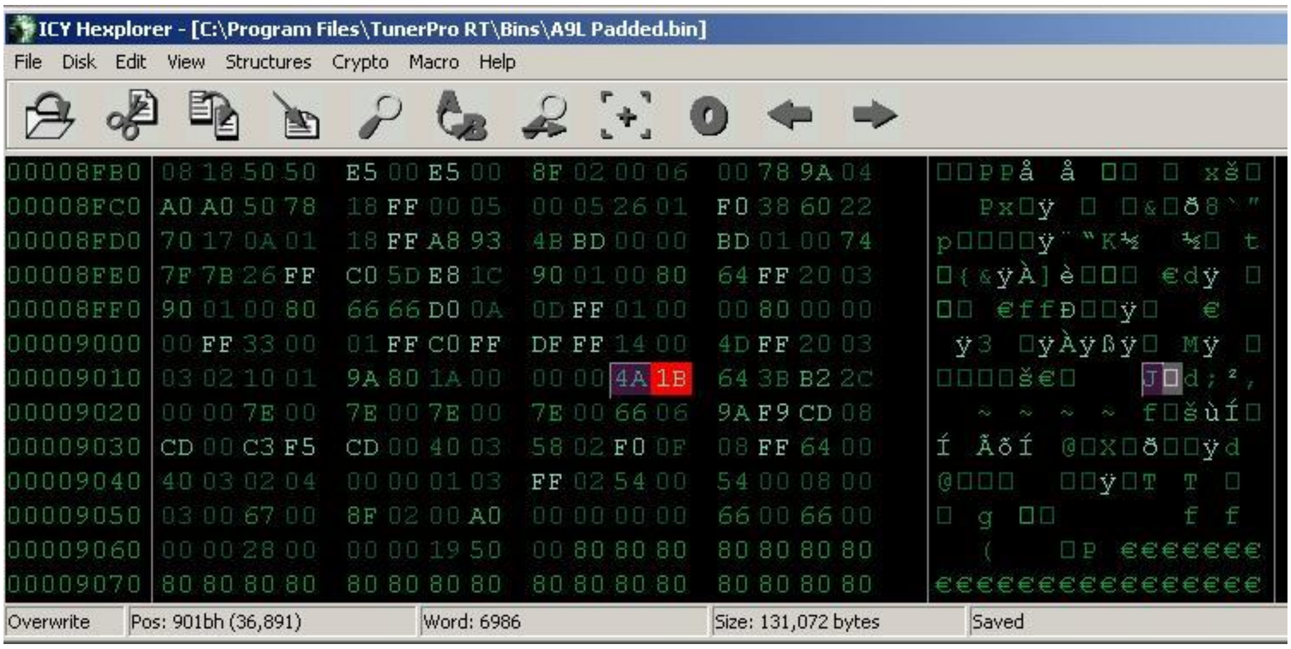
Flags/Switches

- Adaptive Control Idle Low RPM
- Adjustable Dampers Present
- Air Charge Switch
- Allow Accel Enrichment
- Allow Brake On/Off Test
- Allow C/L Re-entry on HEGO Failure
- Allow EPT at Idle
- Allow Fuel Pump Monitor Test
- Allow Goose Test
- Allow HEGO Switch Test
- Allow MFA Strategy
- Allow O/L Canister Purge
- Allow Power Steering Test
- Allow Secondary Air Test
- Allow SPOUT Test

Tables/Functions

- A/C WOT Cutout Time vs TP
- Accel Enrichment Fuel
- Accel Enrichment Fuel Scaler for AC
- Accel Enrichment Fuel Scaler for Thr
- Accel Enrichment Multiplier for BP
- Accel Enrichment Multiplier for TP
- Adaptive Control Learning Table
- Adaptive & MAF Backflow Scaler for
- Air Charge Maximum
- Altitude Multiplier vs BP
- BAP Transfer
- Canister Purge Dutycycle Multiplier
- Canister Purge Dutycycle Multiplier
- Canister Purge Dutycycle vs Airmas
- Cold Engine Fuel Multiplier Update T

Not Emulating | Connected: Flash & Burn USB v5.12.F | Ready | ALDL: Not Connected | Constant, Offset: 901A, Size: 16 Bit



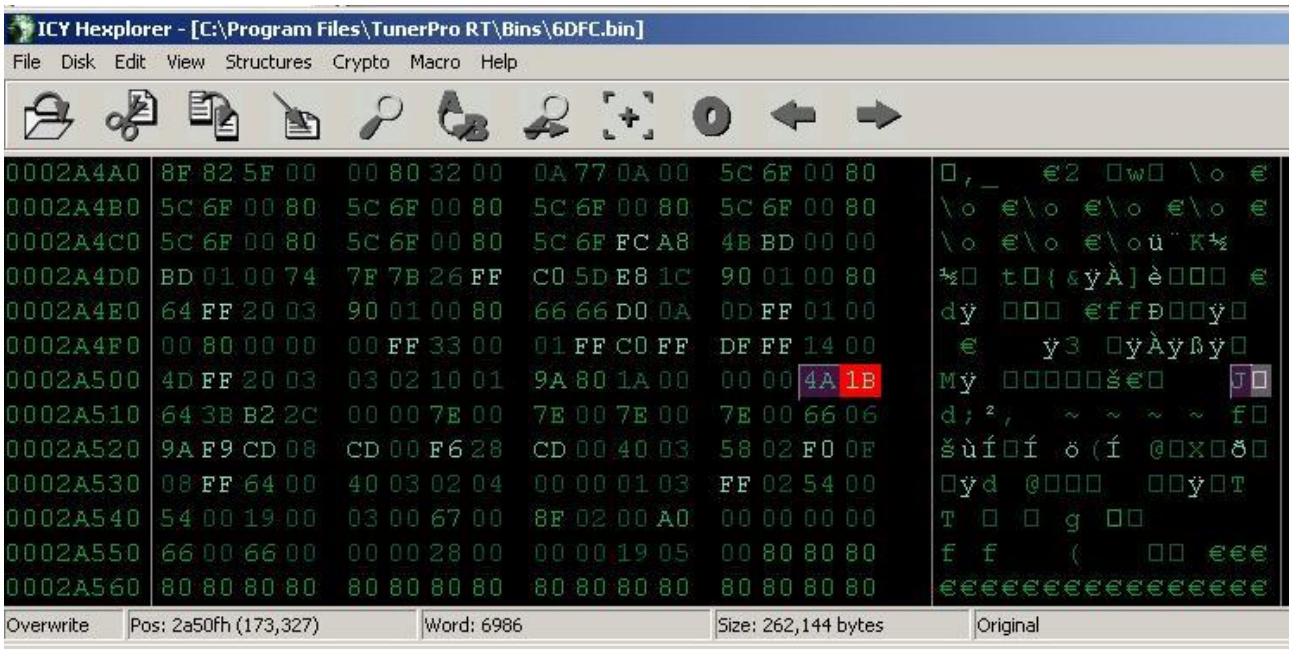
ICY Hexplorer - [C:\Program Files\TunerPro RT\Bins\A9L Padded.bin]

```
00008FB0 08 18 50 50  E5 00 E5 00  8F 02 00 06  00 78 9A 04
00008FC0 A0 A0 50 78  18 FF 00 05  00 05 26 01  F0 38 60 22
00008FD0 70 17 0A 01  18 FF A8 93  4B BD 00 00  BD 01 00 74
00008FE0 7F 7B 26 FF  C0 5D E8 1C  90 01 00 80  64 FF 20 03
00008FF0 90 01 00 80  66 66 D0 0A  0D FF 01 00  00 80 00 00
00009000 00 FF 33 00  01 FF C0 FF  DF FF 14 00  4D FF 20 03
00009010 03 02 10 01  9A 80 1A 00  00 00 4A 1B  64 3B B2 2C
00009020 00 00 7E 00  7E 00 7E 00  7E 00 66 06  9A F9 CD 08
00009030 CD 00 C3 F5  CD 00 40 03  58 02 F0 0F  08 FF 64 00
00009040 40 03 02 04  00 00 01 03  FF 02 54 00  54 00 08 00
00009050 03 00 67 00  8F 02 00 A0  00 00 00 00  66 00 66 00
00009060 00 00 28 00  00 00 19 50  00 80 80 80  80 80 80 80
00009070 80 80 80 80  80 80 80 80  80 80 80 80  80 80 80 80
```

Overwrite | Pos: 901bh (36,891) | Word: 6986 | Size: 131,072 bytes | Saved

This is the screen shot of what the above defined constant looks like in hex.

And on the following page it is found in the 6DFC bin. Notice the difference in location, however the surrounding data is very similar, and the word value is the same



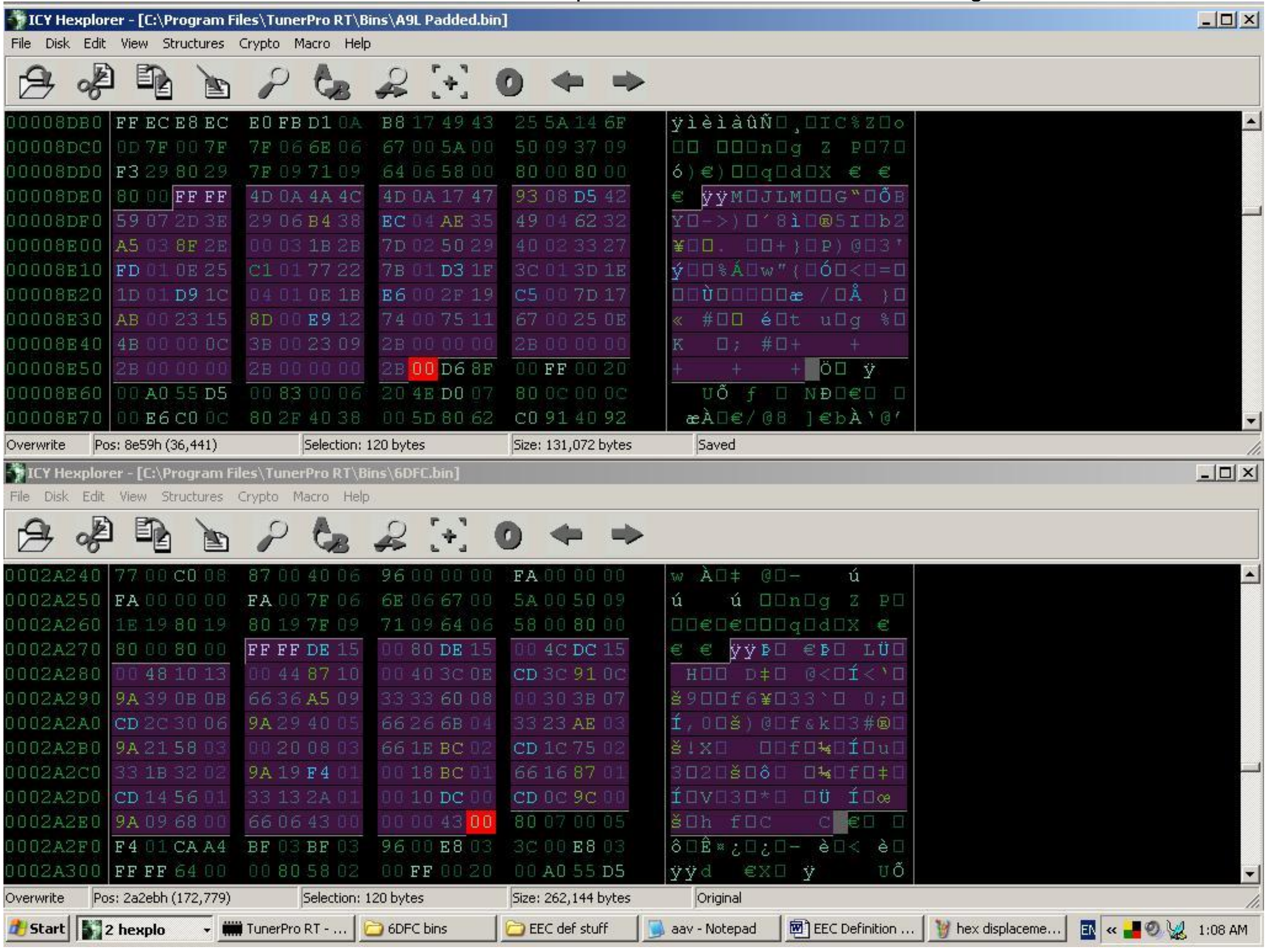
ICY Hexplorer - [C:\Program Files\TunerPro RT\Bins\6DFC.bin]

```
0002A4A0  8F 82 5F 00  00 80 32 00  0A 77 0A 00  5C 6F 00 80
0002A4B0  5C 6F 00 80  5C 6F 00 80  5C 6F 00 80  5C 6F 00 80
0002A4C0  5C 6F 00 80  5C 6F 00 80  5C 6F FC A8  4B BD 00 00
0002A4D0  BD 01 00 74  7F 7B 26 FF  C0 5D E8 1C  90 01 00 80
0002A4E0  64 FF 20 03  90 01 00 80  66 66 D0 0A  0D FF 01 00
0002A4F0  00 80 00 00  00 FF 33 00  01 FF C0 FF  DF FF 14 00
0002A500  4D FF 20 03  03 02 10 01  9A 80 1A 00  00 00 4A 1B
0002A510  64 3B B2 2C  00 00 7E 00  7E 00 7E 00  7E 00 66 06
0002A520  9A F9 CD 08  CD 00 F6 28  CD 00 40 03  58 02 F0 0F
0002A530  08 FF 64 00  40 03 02 04  00 00 01 03  FF 02 54 00
0002A540  54 00 19 00  03 00 67 00  8F 02 00 A0  00 00 00 00
0002A550  66 00 66 00  00 00 28 00  00 00 19 05  00 80 80 80
0002A560  80 80 80 80  80 80 80 80  80 80 80 80  80 80 80 80
```

Overwrite | Pos: 2a50fh (173,327) | Word: 6986 | Size: 262,144 bytes | Original

This parameter was simply found by highlighting a hex string (group of bytes) copying it and pasting it in to the find function of the hex editor. The key here is to get a string around a double word long (4 bytes) including no bytes with a value of zero (for some reason the editor will drop all data after the zero). In this example one would start with the string from 0x2A4F8 to 2A4FB, reading 01,FF,C0,FF, and chances are that you will get it nearly first time every time, but if at first you don't succeed, pick a different string and try again.

Once you have found the location, you can enter it in to the GUI, for TunerPro, you simply select the parameter, press F2 and you can then modify the xdf item. Put in the new location, apply and close the xdf editor window. Sometimes it can get tricky, for example the MAF Transfer table uses different calculations due to processor speeds, and as the A9L had the 8061, and the 6DFC had the 8065, the table multiplier is different. Back to the hex again.



ICY Hexplorer - [C:\Program Files\TunerPro RT\Bins\A9L Padded.bin]

```
00008DB0  FF EC E8 EC  E0 FB D1 0A  B8 17 49 43  25 5A 14 6F
00008DC0  0D 7F 00 7F  7F 06 6E 06  67 00 5A 00  50 09 37 09
00008DD0  F3 29 80 29  7F 09 71 09  64 06 58 00  80 00 80 00
00008DE0  80 00 FF FF  4D 0A 4A 4C  4D 0A 17 47  93 08 D5 42
00008DF0  59 07 2D 3E  29 06 B4 38  EC 04 AE 35  49 04 62 32
00008E00  A5 03 8F 2E  00 03 1B 2B  7D 02 50 29  40 02 33 27
00008E10  FD 01 0E 25  C1 01 77 22  7B 01 D3 1F  3C 01 3D 1E
00008E20  1D 01 D9 1C  04 01 0E 1B  E6 00 2F 19  C5 00 7D 17
00008E30  AB 00 23 15  8D 00 E9 12  74 00 75 11  67 00 25 0E
00008E40  4B 00 00 0C  3B 00 23 09  2B 00 00 00  2B 00 00 00
00008E50  2B 00 00 00  2B 00 00 00  2B 00 D6 8F  00 FF 00 20
00008E60  00 A0 55 D5  00 83 00 06  20 4E D0 07  80 0C 00 0C
00008E70  00 E6 C0 0C  80 2F 40 38  00 5D 80 62  C0 91 40 92
```

Overwrite | Pos: 8e59h (36,441) | Selection: 120 bytes | Size: 131,072 bytes | Saved

ICY Hexplorer - [C:\Program Files\TunerPro RT\Bins\6DFC.bin]

```
0002A240  77 00 C0 08  87 00 40 06  96 00 00 00  FA 00 00 00
0002A250  FA 00 00 00  FA 00 7F 06  6E 06 67 00  5A 00 50 09
0002A260  1E 19 80 19  80 19 7F 09  71 09 64 06  58 00 80 00
0002A270  80 00 80 00  FF FF DE 15  00 80 DE 15  00 4C DC 15
0002A280  00 48 10 13  00 44 87 10  00 40 3C 0E  CD 3C 91 0C
0002A290  9A 39 0B 0B  66 36 A5 09  33 33 60 08  00 30 3B 07
0002A2A0  CD 2C 30 06  9A 29 40 05  66 26 6B 04  33 23 AE 03
0002A2B0  9A 21 58 03  00 20 08 03  66 1E BC 02  CD 1C 75 02
0002A2C0  33 1B 32 02  9A 19 F4 01  00 18 BC 01  66 16 87 01
0002A2D0  CD 14 56 01  33 13 2A 01  00 10 DC 00  CD 0C 9C 00
0002A2E0  9A 09 68 00  66 06 43 00  00 00 43 00  80 07 00 05
0002A2F0  F4 01 CA A4  BF 03 BF 03  96 00 E8 03  3C 00 E8 03
0002A300  FF FF 64 00  00 80 58 02  00 FF 00 20  00 A0 55 D5
```

Overwrite | Pos: 2a2ebh (172,779) | Selection: 120 bytes | Size: 262,144 bytes | Original

Start | 2 hexplo | TunerPro RT - ... | 6DFC bins | EEC def stuff | aav - Notepad | EEC Definition ... | hex displaceme... | 1:08 AM

Notice that both transfers start with FF,FF, that's because both have the highest voltage set to 16V, and see how the second word in on the A9L is 4D,0A, where as the second word on the 6DFC is DE,15. LSB applies here also, which means that the second word on the A9L would equal 2637, and the 6DFC would be 5598 in decimal.

Note how the transfer table is laid out, the 120 bytes is separated in to words, and each word alternates what it means. The first word is FF,FF, and when divided by 4096 it equals to 16, and to give it meaning to humans, we put a V next to that to mean that is what the MAF output in volts is. The next byte is the Mass Air Flow in KG/Hr's, so in the A9L, if you divide 2637 by 3.1562283 you will get a number for the MAF of 835.49KG/Hr. So now we know that when the MAF voltage is 16, the MAF is 835.5KG/HR, and it keeps going until it reaches zero (negative in some parameters) If you get a new MAF meter, it is advised to adjust your MAF transfer to suit, as your MAF is one of the main factors to determine load calculations. Normally reputable companies supply you with a chart of MAF vs Voltage, which will help you enter in the required numbers.

Misc other stuff

Some other stuff of interest is that the checksum is usually located @ the 10th byte after the program code starts, ie 200A in the EECIV and 2200A in the EECV 1 & 2 bank bins. The bins are usually laid out in banks, for instance the A9L is a 2 bank bin, as is the 6DFC, however the 4DDG is a 1 bank. What this means is that there is usually a bank of null data at the beginning of the bin, then the code starts. If its a single bank, there will be a very short burst of null data beteen the operating code and the parameters, where as if its a multiple bank bin, then there will be a substantial amount of null data between the 2. In the later EECV's, ie AU falcons, they are 4 bank ECU's.

There is a linked list which can be found @ 0x2020 in the A9L and @ 22060 in the 6DFC. This should give you the pattern to find it in other bins. In the A9L the word is the actual jump to address of the linked list is 0x8c00, where as in the EECV 256k 6DFC, the 2 needs to be added to the front of the list due to the larger offset. The list is LSB, so in the A9L the first jump to is, and in the 6DFC its 0x2A000. When you jump to that location you will find that it has the next jump to address found in the linked list. We havnt discovered what to do with this yet, apart from the fact that the parameters start at the first jump to address, but if you can figure out the rest of them, we would appreciate it if you could share it with the rest of us.

More will be added to this document as we discover or figure out how to explain it in an easy manner.

JA & JB