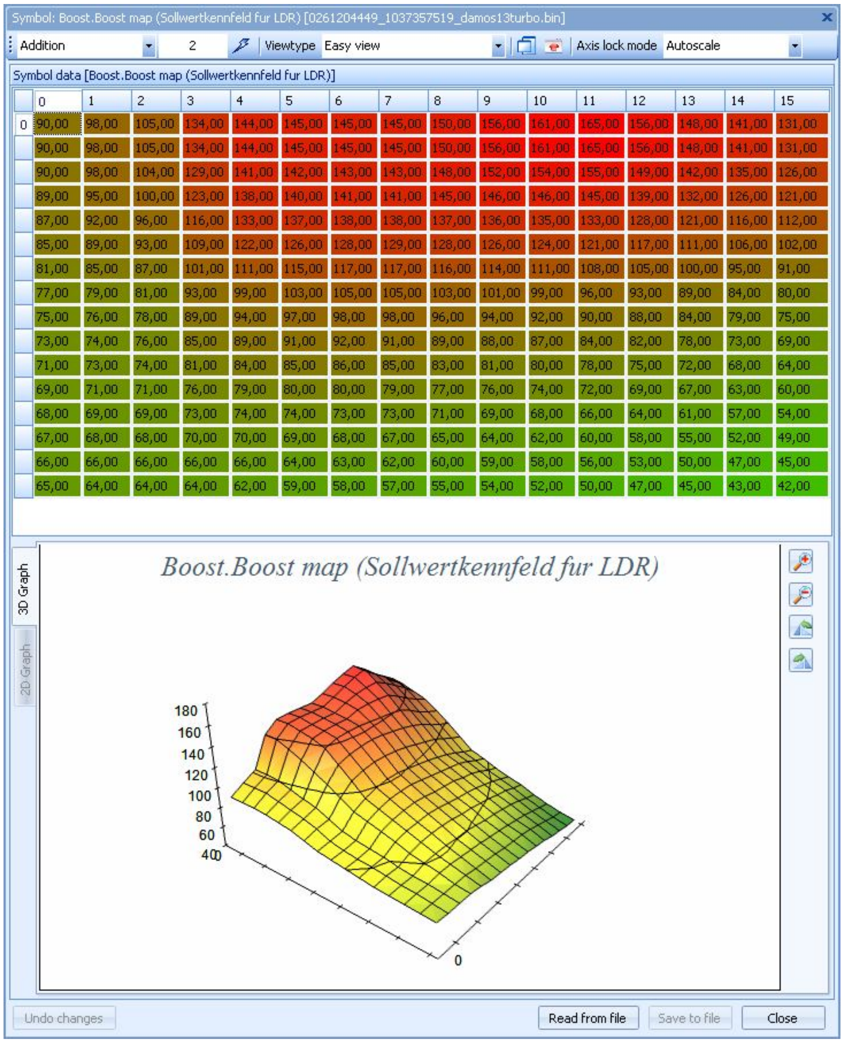
Volvo Motronic M4.4

A detailed description for Motronic 4.4 ECUs used in Volvo 850 T5(R) and S70 T5(R)



Symbol: Boost.Boost map (Sollwertkennfeld fur LDR) [0261204449_1037357519_damos13turbo.bin]

Addition 2 | Viewtype Easy view | Axis lock mode Autoscale

Symbol data [Boost.Boost map (Sollwertkennfeld fur LDR)]

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	90,00	98,00	105,00	134,00	144,00	145,00	145,00	145,00	150,00	156,00	161,00	165,00	156,00	148,00	141,00	131,00
	90,00	98,00	105,00	134,00	144,00	145,00	145,00	145,00	150,00	156,00	161,00	165,00	156,00	148,00	141,00	131,00
	90,00	98,00	104,00	129,00	141,00	142,00	143,00	143,00	148,00	152,00	154,00	155,00	149,00	142,00	135,00	126,00
	89,00	95,00	100,00	123,00	138,00	140,00	141,00	141,00	145,00	146,00	146,00	145,00	139,00	132,00	126,00	121,00
	87,00	92,00	96,00	116,00	133,00	137,00	138,00	138,00	137,00	136,00	135,00	133,00	128,00	121,00	116,00	112,00
	85,00	89,00	93,00	109,00	122,00	126,00	128,00	129,00	128,00	126,00	124,00	121,00	117,00	111,00	106,00	102,00
	81,00	85,00	87,00	101,00	111,00	115,00	117,00	117,00	116,00	114,00	111,00	108,00	105,00	100,00	95,00	91,00
	77,00	79,00	81,00	93,00	99,00	103,00	105,00	105,00	103,00	101,00	99,00	96,00	93,00	89,00	84,00	80,00
	75,00	76,00	78,00	89,00	94,00	97,00	98,00	98,00	96,00	94,00	92,00	90,00	88,00	84,00	79,00	75,00
	73,00	74,00	76,00	85,00	89,00	91,00	92,00	91,00	89,00	88,00	87,00	84,00	82,00	78,00	73,00	69,00
	71,00	73,00	74,00	81,00	84,00	85,00	86,00	85,00	83,00	81,00	80,00	78,00	75,00	72,00	68,00	64,00
	69,00	71,00	71,00	76,00	79,00	80,00	80,00	79,00	77,00	76,00	74,00	72,00	69,00	67,00	63,00	60,00
	68,00	69,00	69,00	73,00	74,00	74,00	73,00	73,00	71,00	69,00	68,00	66,00	64,00	61,00	57,00	54,00
	67,00	68,00	68,00	70,00	70,00	69,00	68,00	67,00	65,00	64,00	62,00	60,00	58,00	55,00	52,00	49,00
	66,00	66,00	66,00	66,00	66,00	64,00	63,00	62,00	60,00	59,00	58,00	56,00	53,00	50,00	47,00	45,00
	65,00	64,00	64,00	64,00	62,00	59,00	58,00	57,00	55,00	54,00	52,00	50,00	47,00	45,00	43,00	42,00

Document created by Dilemma 2011

Table of content

Introduction

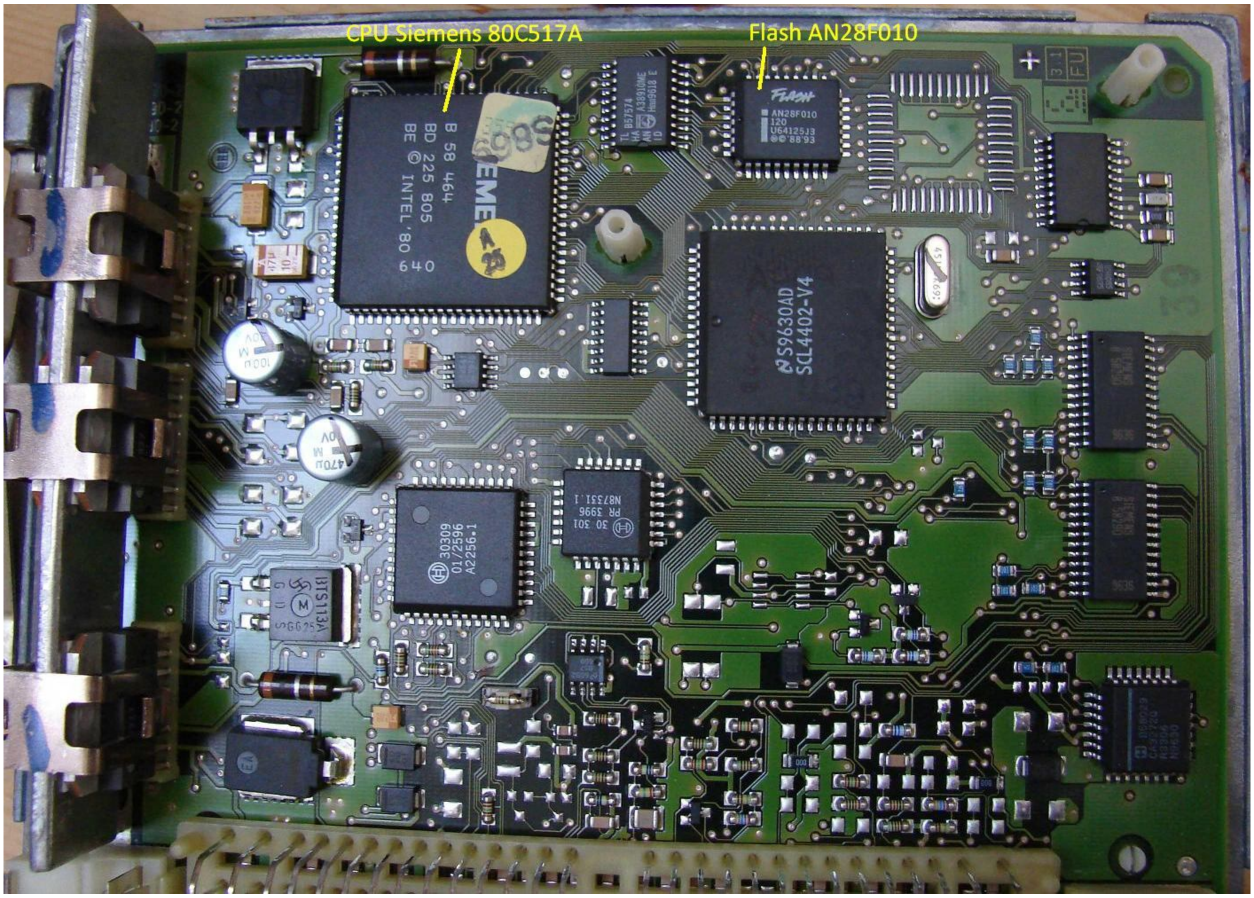
This document describes the Motronic M4.4 ECU in detail. It will first describe the hardware and proceed with a even more detailed description of the software that is running in the ECU so that we can learn how to tweak and tune the ECU to match the hardware – altered or not – that is on the car better.

Special thanks for getting all this together go out to rkam, T5_Germany, Steve Hayes and others on ecurproject.com, trionictuning.com and volvospeed.com.

Hardware

Overview of the board

The ECU contains a tri-layer printed ciruit board (PCB) which holds a lot of SMD components. The main component are – logically: Main CPU, Flash program storage, SRAM memory (working memory) and a lot of input/output (I/O).



Main CPU: SAB83C517A-5N18

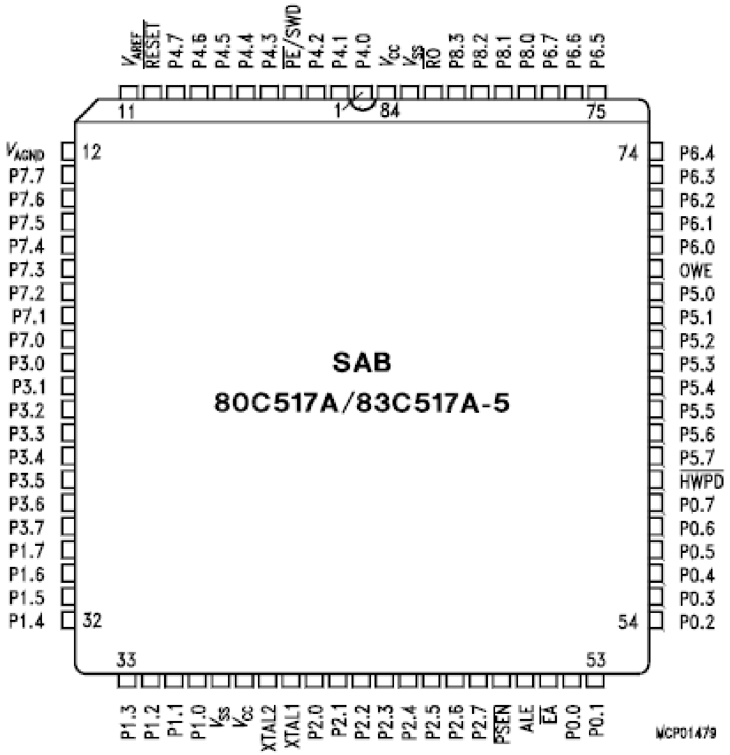
- ✓ Upto 18Mhz operation
- ✓ 32K x 8 ROM
- ✓ 256 x 8 on-chip RAM
- ✓ 2K x 8 on-chip RAM (XRAM)
- ✓ Four 16 bit timers/counters
- ✓ 10 bit A/D converter with 12 multiplexed inputs
- ✓ Two full duplex serial interfaces
- ✓ Nine ports: 56 I/O lines, 12 input lines
- ✓ Mask programmable ROM (Internal ROM protection)

Datasheet documents

http://trionic.mobixs.eu/Motronic/M4.4/80c535.pdf

http://trionic.mobixs.eu/Motronic/M4.4/80c517um.pdf

http://trionic.mobixs.eu/Motronic/M4.4/SAB83C517A-5N18.pdf



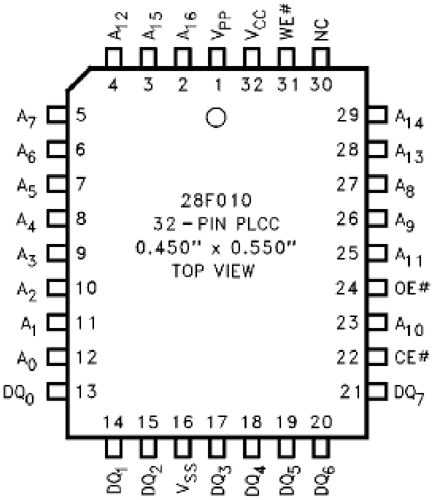
Flash eprom: AN28F010

This is an automotive specified 128Kb flash chip with a temperature range from -40 degrees upto 125 degrees celcius.

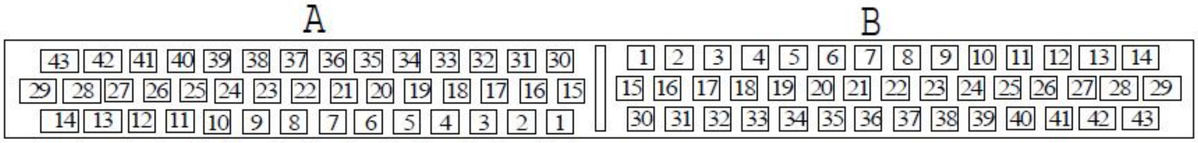
- Programming voltage: 12V
- Chip erase time: 1 second
- Byte program time: 10 uS
- Access time: 120 nS

Datasheet:

http://trionic.mobixs.eu/Motronic/M4.4/AN28F010.pdf



ECU Pinout



Pin number	Color	Description
A1		
A2		Signal (+) front knock sensor (KS) (measured to terminal #A17)
A3		Power ground mass air flow (MAF) sensor
A4		Signal mass air flow (MAF) sensor (measured to terminal #A5)
A5		Signal ground mass air flow (MAF) sensor
A6		Signal ground engine speed (RPM) sensor
A7		Control signal engine cooling fan (FC) low-speed
A8		
A9		Control signal injector 5
A10		Control signal injector 1
A11		Opening signa the idle air control (IAC) valve
A12		15 supply (power supply from the ignition switch) (+12V)
A13		Power ground
A14		Control signal front heated oxygen sensor (HO2S), preheating
A15	Y	Power supply the throttle position (TP) sensor
A16	G/W	Signal the throttle position (TP) sensor
A17		Knock sensor common ground
A18	BR/B	Ground (sensor)
A19		Signal (–) rear heated oxygen sensor (HO2S)
A20		Signal engine speed (RPM) sensor (measured to #A6)
A21	BL/Y	Signal camshaft position (CMP) sensor
A22		Control signal engine cooling fan (FC) high-speed
A23		Control signal injector 4
A24		Control signal injector 3
A25		Closing signal the idle air control (IAC) valve
A26		30– supply (power supply from the battery) (+12V)
A27		Power supply (from the main relay) (+12V)
A28		Power ground control module
A29		Control signal rear heated oxygen sensor (HO2S), preheating
A30		Signal (+) rear knock sensor (KS) (measured to terminal #A17)
A31	Y/GY	Signal the engine coolant temperature (ECT) sensor
A32		Signal (+) front heated oxygen sensor (HO2S)
A33		Signal (–) front heated oxygen sensor (HO2S)
A34		Signal (+) rear heated oxygen sensor (HO2S)
A35	G/Y	Outer temperature sensor
A36	Y/R	Power supply camshaft position (CMP) sensor
A37		Pulsed secondary air injection system (PAIR) pump valve, control signal
A38		Control signal injector 2
A39		Control signal Canister purge (CP) valve
A40		
A41		Control signal main relay
A42		Signal ground control module (measured to the battery negative terminal)
A43		

Pin number	Color	Description
B1		Power supply Accelerometer (vehicle speed)
B2		Signal torque limiting (from automatic gearbox)
B3		Signal torque limiting (from automatic gearbox)
B4		Signal torque limiting acknowledgement (to automatic gearbox)
B5		To diagnostics socket (also to Aut-transmission control module)
B6	GY	Signal air conditioning (A/C) compressor status
B7		Control signal malfunction indicator lamp (MIL) (to the combined instrument panel)
B8		Enable internal ECU ROM flashing when +12V is applied (results in +5V on EA pin on CPU)
B9		Signal A/C pressure sensor
B10		
B11		Control signal ignition discharge module (IDM)
B12		Signal load Tq (to automatic gearbox)
B13		
B14		
B15		Power supply fuel tank pressure sensor
B16		
B17		
B18		Signal speed (from combined instrument panel)
B19		Control signal fuel pump
B20		Signal throttle position (TP) sensor (to automatic gearbox)
B21		Signal tachometer (to combined instrument panel)
B22		Atmospheric pressure sensor
B23		Signal engine coolant temperature (ECT) (to ECC and combined instrument panel)
B24		Signal constant idle speed compensation P/N position (from automatic gearbox)
B25	BL/GY	Signal air conditioning (A/C) relay status
B26		Signal malfunction indicator lamp (MIL) request (from automatic gearbox)
B27		Control signal fuel pump
B28		Signal ground sensor (measured to the battery negative terminal)
B29		Power supply A/C pressure sensor
B30		
B31		Fuel tank pressure sensor (certain markets only)
B32		Signal Accelerometer (vehicle speed)
B33		
B34		
B35		
B36		Diagnostic lead K–link
B37		
B38		Control signal pulsed secondary air injection system (PAIR) pump relay
B39		Signal fuel consumption (to the trip computer)
B40		Control signal air conditioning (A/C) relay (allows A/C to start)
B41		Control signal, turbocharger (TC) control valve
B42		Turbocharger (TC) boost pressure limiting signal (from automatic gearbox)
B43		

M4.4 specific implementations

Since M4.4 has twice the flash size compared to M4.3 so it has 128Kb of ROM to work with and the CPU only has 16 address lines, Bosch needed to figure out a way to address the upper flash bank (ranging 0x10000-0x1FFFF) that needs the 17th address line. This is done by using a general purpose IO pin for switching the banks. P5.7 (port 5 highest bit) was chosen for this task. P5.7 high means the upper flash bank is selected, P5.7 low means the lower flash bank is selected. The latter being default.

P8.2 is the 10th analogue input on the CPU and this one is used in the bootloader code as well.

Software

Once we download the data from the ECU – e.g. with a flash programmer we can load the binary file into MotronicSuite (ref: http://trionic.mobixs.eu/Motronic/Motronic.msi)

Details on how the software works can be found in appendix I.

Software conversion factors

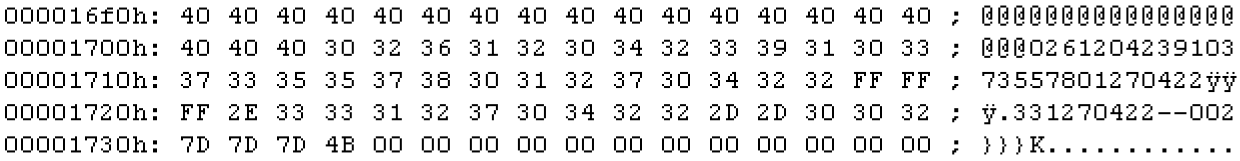
The software contains a lot of information for which we have to know the conversion factors to use to be able to translate them into units that we use on a daily basis. This is a shortlist of the most used factors.

36 Battery voltage *0.0704 Volt
38 Engine Coolant Temperature (ECT) *1-80 < -50 degrees Celsius
3B Engine speed *30 RPM
40 Internal load signal *0.05 msec
4B Mass Air Flow (MAF) sensor signal *0.0196 Volt
54 Ignition advance *(-0.75)+78 Degrees
BC Turbo duty cycle *0.391 (uncertain)

Complete document here: http://trionic.mobixs.eu/Motronic/M4.4/M44-scaling-RAM.pdf

Software information data

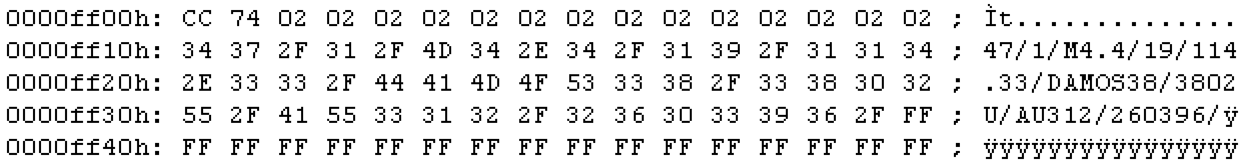
In the software, the identifiers are stored as well about HW revision, SW version, volvo partnumbers etc. This data is stored in ASCII in the binary file and looks something like this:



```
000016f0h: 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 ; @@@@@@@@@@@@@@@@
00001700h: 40 40 40 30 32 36 31 32 30 34 32 33 39 31 30 33 ; @@@0261204239103
00001710h: 37 33 35 35 37 38 30 31 32 37 30 34 32 32 FF FF ; 73557801270422ÿÿ
00001720h: FF 2E 33 33 31 32 37 30 34 32 32 2D 2D 30 30 32 ; ÿ.331270422--002
00001730h: 7D 7D 7D 4B 00 00 00 00 00 00 00 00 00 00 00 00 ; }}}K............
```

1270422 = P01270422
0261204239 is the hardware ID
1037355780 is the software ID
331270422—002 is some partnumber ID

And there’s more readable information in the file as well:



```
0000ff00h: CC 74 02 02 02 02 02 02 02 02 02 02 02 02 02 02 ; Ìt..............
0000ff10h: 34 37 2F 31 2F 4D 34 2E 34 2F 31 39 2F 31 31 34 ; 47/1/M4.4/19/114
0000ff20h: 2E 33 33 2F 44 41 4D 4F 53 33 38 2F 33 38 30 32 ; .33/DAMOS38/3802
0000ff30h: 55 2F 41 55 33 31 32 2F 32 36 30 33 39 36 2F FF ; U/AU312/260396/ÿ
0000ff40h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ; ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

47/1/M4.4/19/114.33/DAMOS38/3802U/AU312/260396
In which the latter string is the date of software build.

Reading the code

To be able to understand the software better we'll need to dive into the world of assembler language. This is a sort of intermediate between understandable human language and the operation codes used by the microprocessor. Once we can read the assembler language (assembly for short) we can track all the things the microprocessor is told to to when the program is running. This is very valuable information because we don't have first hand information from either Bosch or Volvo that can tell us in details what the ECU does.

We convert the binary file into assembly language we need to disassemble the file. We can do that by running the disassembler in Motronic Suite, by running the disassembler manually or by using a seperate program like IDAPro to do it for us. A seperate disassembler can be found here in the website.

Disassembler D52 http://trionic.mobixs.eu/Motronic/M4.3/d52.exe

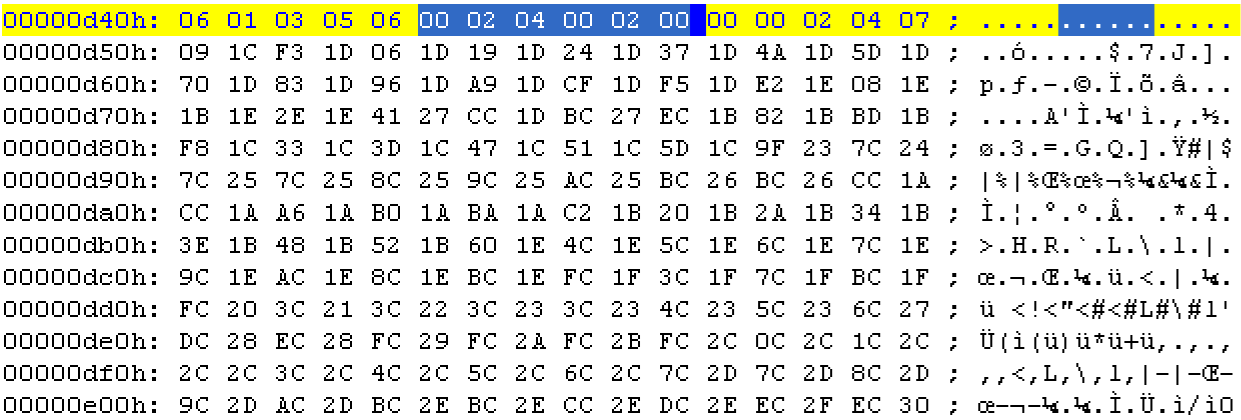
Once we disassemble the binary file we have an file containing the complete assembly listing in which we can start to explore and understand the internal workings of M4.4.

Maps and variables

Determining the location and type of maps and variables in the M4.4 binaries is quite a hassle. In contrast to M4.3 the map addresses are present in the file but they are *not linked directly* to the axis (support point) in the file. To be able to detect the available maps we have to do some tricks and make a couple of assumptions in the algorithm used. This chapter will describe – in detail – what the Motronic Suite software does to fetch the maplist from the file.

Getting the map lookup table index

First we need to lookup the index in the file at which the lookup table is located. This lookup table consists of addresses in the file which we can use to determine axis and map information. Do find the correct index we look for a certain byte sequence in the file. M4.3 has a leading sequence of 4 bytes that always seem to be the same, 0x00 0x02 0x04 0x00 0x02 0x00. The picture below shows the data found in a certain M4.4 file.



```
00000d40h: 06 01 03 05 06 00 02 04 00 02 00 00 00 02 04 07 ; ................
00000d50h: 09 1C F3 1D 06 1D 19 1D 24 1D 37 1D 4A 1D 5D 1D ; ..ó.....$.7.J.].
00000d60h: 70 1D 83 1D 96 1D A9 1D CF 1D F5 1D E2 1E 08 1E ; p.ƒ.–.©.Ï.õ.â...
00000d70h: 1B 1E 2E 1E 41 27 CC 1D BC 27 EC 1B 82 1B BD 1B ; ....A'Ì.¼'ì.‚.½.
00000d80h: F8 1C 33 1C 3D 1C 47 1C 51 1C 5D 1C 9F 23 7C 24 ; ø.3.=.G.Q.].Ÿ#|$
00000d90h: 7C 25 7C 25 8C 25 9C 25 AC 25 BC 26 BC 26 CC 1A ; |%|%Œ%œ%¬%¼&¼&Ì.
00000da0h: CC 1A A6 1A B0 1A BA 1A C2 1B 20 1B 2A 1B 34 1B ; Ì.¦.°.º.Â. .*.4.
00000db0h: 3E 1B 48 1B 52 1B 60 1E 4C 1E 5C 1E 6C 1E 7C 1E ; >.H.R.`.L.\.l.|.
00000dc0h: 9C 1E AC 1E 8C 1E BC 1E FC 1F 3C 1F 7C 1F BC 1F ; œ.¬.Œ.¼.ü.<.|.¼.
00000dd0h: FC 20 3C 21 3C 22 3C 23 3C 23 4C 23 5C 23 6C 27 ; ü <!<"<#<#L#\#l'
00000de0h: DC 28 EC 28 FC 29 FC 2A FC 2B FC 2C 0C 2C 1C 2C ; Ü(ì(ü)ü*ü+ü,.,.,
00000df0h: 2C 2C 3C 2C 4C 2C 5C 2C 6C 2C 7C 2D 7C 2D 8C 2D ; ,,<,L,\,l,|-|-Œ-
00000e00h: 9C 2D AC 2D BC 2E BC 2E CC 2E DC 2E EC 2F EC 30 ; œ-¬-¼.¼.Ì.Ü.ì/ì0
```

Reading the map lookup table

If we find a 0x00 directly after this sequence, we need to add 7 to the address found, so reading of map locations in the above file will commence at address 0x000D51, 2 bytes at a time. So, the first address we find is 0x1CF3 and the second one is 0x1D06. We keep reading addresses and storing them in a list until we reach a 2 byte value that is smaller than 0x1000 (so, in fact we assume that there are no maps located in the memory section < 0x1000).

How Motronic calculates theoretical engine load

Motronic needs three things to calculate the internal Load signal (which can be found as axis for several maps):

1. A signal from the airmass meter, normalized to airflow in kg/hr: Q
2. The current engine speed (rpm): n
3. The programmed injector constant: Ki

$$Q = f(\frac{Up}{Uv})$$

in which $\frac{Up}{Uv}$ is the ratio between MAF output and MAF reference voltages.

$$Tl = \frac{Q}{n * Ki}$$

Tl (LOAD)is not just a representation of cylinder filling, but the theoretical Injector Time Open (Ti) needed to reach stoich (Lambda= 1) with the current injector setup assuming that the motor has an efficiencey of 100% (VE), which it has not of course. Hence there are fueling tables which are used as multiplicative corrections to Tl to reach the actual Ti.

With Tl quantified, Motronic now takes into account the correction factors for the engine and the current operating conditions by introducing multiplicative factors to correct the THEORETICAL injector time to the ACTUAL time for injection (Ti) needed at that operating condition point. Finally an additive factor (Tv) is added to compensate for the fluctuating injector opening time under lower than nominal voltages (battery correction map).

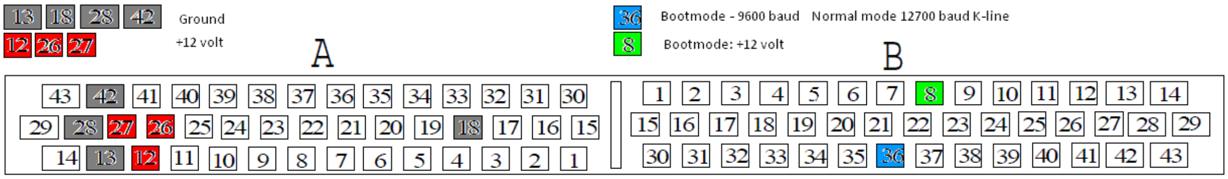
$$Ti = (Tl * [X, Y, Z \ldots]) + Tv$$

The final Ti is the injector open time that is applied to the injectors.

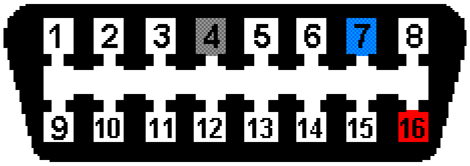
Informational credits to: *Jim Conforti*

Communication with the ECU

Connection diagram



ECU connector: This is looking at the connector on the ECU



This is the socket-part, connector part is mirrored

There are two methods of communication that can be used with a M4.4 ECU.

- Normal mode communication
- Boot mode communication

Normal mode communication

Normal mode is used for reading data from the ECU while it is in its operational state. Reading live data and the contents of the flash file are procedures that are carried out in normal mode.
To activate normal mode communication we need to connect a K-line interface to the ECU on pin B36 and after the wake-up procedure communication can commence at 12700 baud.

Wakeup procedure for normal mode

To be able to communicate in normal mode, the ECU needs to be aware of the fact that there is a diagnostics device connected to pin B5. To let the ECU know we need to send a 0x10 byte to the port at 5 baud (!). After a correct wakeup byte on the B5 pin we will receive a response from the ECU at 12700 baud. This response will be 0x55 0xAB 0x02 in which the 0x55 is the acknowledge and the 0xAB and 0x02 are the keywords used to communicate with a M4.3 ECU. After reception of this sequence we need to send an acknowledge message to the ECU which is the inverted last keyword which will be 0xFD.

Normal mode: KWP71

After the wakeup procedure, communication with the ECU takes place in the KWP71 protocol. This protocol is standarized and therefore it will not be discussed within this document.

Bootmode communication

Boot mode communication is only useful for flashing the ECU with a new firmware version. This is a special mode which is indicated to the ECU by pulling pin B8 to +12V before the ECU starts (boots). So, you will need to have the ECU powered down and apply +12V to B8 before the ECU is powered on. The ECU will now run a special boot loader program which resides in internal ROM and allows us to reprogram the ECU. After starting in bootmode communication can commence on pin B36 at 9600 baud with a K-line interface.

The M4.4 bootloader is quite a bit bigger than the M4.3 bootloader. There is additional code for (probably) hardware testing purposes or KWP communication in the bootloader as well. It also checks more stuff than the M4.3 bootloader and it has additional code for switching banks since M4.3 does not have that.

Bootmode is forced by applying +12V on pin B8 and commencing communication on pin B36 at 9600 baud. Pin B5 is not used in M4.4.

The programming sequence for M4.4 is as follows:

- Send erase command
- Wait for erase completed
- Send data to flash from 0x000000-0x00FFFF
- Send command to switch to upper flashbank
- Send data to flash from 0x010000-0x01FFFF
- Wait for acknowledge message
- Send checksum verification message (optional)
- Wait for checksum OK or FAIL message (optional)

Erase command

```
3AFF5544332211000000000000020000000000000000000000000000000000000000000000000000
```

FF as length results in FRAME_TYPE 0x55 (does not get programmed)

Wait for erase completed

```
02305930390363    START ERASE
02315930390362    START ERASE
02315930390362    START ERASE
0230593035036F    FINISHED ERASE
0231593035036E    FINISHED ERASE
0231593035036E    FINISHED ERASE
```

Send data for lower flash bank

```
3A20000000022EC1020A3B0202020202020A510202020202020A880202020202020AA70202E2
...
3A20FFE000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF21
```

```
3A20AHAL00XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXCS
AH = Address high
AL = Address low
XX = Data to program
```

Switch to upper flash bank

```
3A200000020000000000000000000000000000000000000000000000000000000000000000DE
```

Send data for upper flash bank

```
3A20000000022EC1020A3B0202020202020A510202020202020A880202020202020AA70202E2
...
3A20FFE000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF21
```

```
3A20AHAL00XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXCS
AH = Address high
AL = Address low
XX = Data to program
```

Acknowledge message (flash completed)

```
023059303603XX    FLASH COMPLETED
023159303603XX    FLASH COMPLETED
023159303603XX    FLASH COMPLETED
```

Checksum verification message

```
3AFE554433221100000000000003000000000XXXX0000000000000000000000000000000000CC

XXXX = checksum calculated by host
CC = message checksum
```

Checksum answer message

```
023059304503XX    CHECKSUM OK
023159304503XX    CHECKSUM OK
023159304503XX    CHECKSUM OK
OR
023059304403XX    CHECKSUM FAILED
023159304403XX    CHECKSUM FAILED
023159304403XX    CHECKSUM FAILED
```

Document created by Dilemma 2011