Intro

Tuning typically consists of three tools. An editor to modify a binary file, A flashing tool to write these changes to an ECU and a data logging solution.

BIN Background

Obtaining a stock binary (tune) file for your car can be a little tricky or seem difficult at first. You can't read the ECUs natively, so it's not as simple as plugging your laptop into the car and reading it out. However the "Flashdaten" flash data files are available on the european VAG servers, so these files aren't too difficult to get ahold of if you know where to look.

We have tried to consolidate our efforts in developing less definition files etc. (more on that later). But generally for USDM spec cars, you will likely want to run/flash these files:

2015-2018 GTI / A3: 5G0906259L_0002
2015-2017 Golf R / S3: 8V0906259K_0003
2018 Golf R / S3: 8V0906259P_0001
2019+ GTI / GLI / A3: 5G0906259Q_0003
2019+ Golf R / S3: 8V0906259Q_0002

The 2015-2018 models use the Simos 18.1 or 18.6 ECU and 2019+ use Simos 18.10. The 15-18 Simos18.1/18.6 files use the "SC8S50" file structure while Simos18.10 use the "SCGA05" file structure. There are a vast number of other box codes and file structures beyond this, but most of our efforts have consolidated on these box codes/structures.

Once you have your desired frf file, you need to convert it into a .bin so you can read/edit the file in your editor program of choice. VW_Flash has a built-in function to do this. Follow the link below to download a stable "release" of the software. File -> Extract FRF.

This will spit out a 4MB .bin file and 5 other smaller .bin files. The smaller files will be labeled with what block of the memory they represent (CBOOT, ASW1, ASW2, ASW3, CAL). The 4MB .bin is a single file with all of these blocks. A very short explanation of these is that CBOOT is the bootloader, ASW1-2-3 is the application software (or Operating System) and CAL is the calibration.

Ultimately to "tune" the car, you will be editing the Calibration. You can edit the 4MB "full bin" or the smaller "calibration-only" file to get the same result– the flash tools discussed here support both formats, but your XDF/definition file needs to match (more on this next). I recommend sticking with the 4MB full bin for everything. If you're coming from Eurodyne, a full bin is synonymous with a loader file, while the cal only would be the maestro file. However with binary files, you can edit them both the same, I don't believe it is advantageous to split them up– it just complicates things down the road.

XDF/XML Background

A XDF or XML file is simply a definition file that maps out all of the tables, axis', values in the calibration, into a human-readable format in the editor software. There are some useful scripts out there to automate much of this definition file creation process, but that is beyond the scope of this "getting started" document.

Ultimately, you need a definition file to match your box code / file structure. For example if you are running 5G0906259L_0002 or 8V0906259K_0003, you need the SC8S50 definition file. XDF format is used by TunerPro, while XML is used by ecuEdit (and other software).

BIN Editor Options

There are currently two main programs being used in this effort: TunerPro and ecuEdit.

TunerPro is the "standard" software, but recently a few have been trying out other options. TunerPro is free and is still in development and getting features added (although slow). On the other hand, is it quite basic and lacking a lot of features that one might expect after spending a lot of time with commercial solutions like HPtuners, etc.

ecuEdit costs around $110 USD and has a lot of advanced editor features and a built-in logging application. The software is also hardware-locked to a single device, which may be a big issue for some.

Tunerpro is still the most popular option at this point in time.

Flashing/Datalogging Tool Options

The original free and open source flashing solution is VW_Flash. This program is (generally) for PC use, has an easy to use exe GUI in stable "releases". Supports Simos18.1/6/10 and DQ250 MQB flashing. Supported dongles include the Tactrix OpenPort and the Macchina A0 in either wired USB or BLE (bluetooth). Supports high-speed datalogging for ECU only. Additionally, you can run this in CLI and/or Linux/SocketCAN as originally intended (see github for more details).

More recently, SimosTools Android app was released which supports ECU flashing and also DQ250 flashing. Macchina A0 (BLE) is the only supported dongle. Supports standard PID and high-speed datalogging for ECU, and DSG logging.

DQ381 flashing is currently in development for VW_Flash (it works, but still has some bugs) and will eventually be rolled out with Simos Tools as well.

There are several possible combinations of hardware/software that will work, but the Macchina A0 is the most flexible– it works with both applications, and between the two you can flash ECU and DSG, and datalog everything from your android device over bluetooth.

Unfortunately at the time of writing, the Macchina A0 is sold out everywhere with no ETA on restock. However the device is open-sourced based and you can make your own similar esp32-based device for relatively cheap.

Datalogging Background

Mode 22 datalogging is using built-in diagnostic functions of the ECU or TCU controller. It works well, but is limited to parameters that VW deemed necessary for logging purposes.

A better logging option is a custom high-speed RAM logger. This option allows even faster datalogging and can literally log any measurement in the controller.

Mode 22 is plenty sufficient for diagnostic purposes, but RAM logging is highly desirable for tune development.

Links

Hardware

- Macchina A0: https://www.macchina.cc/catalog/a0-boards/a0-under-dash
- Tactrix OpenPort: https://www.tactrix.com/

Software

- VW_Flash: https://github.com/bri3d/VW_Flash
- SimosTools: https://play.google.com/store/apps/details?id=com.app.simostools
- TunerPro: https://www.tunerpro.net
- ecuEdit: https://www.epifansoft.com/ecuEdit.html