HUD ECU Hacker

Latest Version: 5.3.1 (20.sep.2024)

Features

- HUD ECU Hacker can **scan** ECUs from Delphi, Bosch, Honda, Deni, Liteon, Motion, Yeson, Lifan, Rongmao, HiSun and more.
- HUD ECU Hacker can **clear** the fault codes (DTC).
- HUD ECU Hacker can **flash** and **tune** the Delphi MT05.
- HUD ECU Hacker is **charityware** (click the link for an explanation).
- HUD ECU Hacker can connect to the ECU over **K-Line** and **CAN bus**.
- HUD ECU Hacker implements the **protocols** CAN Raw, ISO 15765, ISO 9141, ISO 14230, KW1281 and Honda.
- HUD ECU Hacker works with **cheap adapters**: J2534, VAG KKL, VAG K+CAN, ELM327, OBDLink and UsbCAN
- HUD ECU Hacker allows to send commands to the ECU manually (**command injection**).
- HUD ECU Hacker contains a **CAN bus analyzer**, CAN bus debugger, CAN bus logger and CAN Terminal with command injection.
- HUD ECU Hacker can write **logfiles** and convert them into **graphs**.
- HUD ECU Hacker has a RS232 Terminal
- HUD ECU Hacker can **emulate** any ECU.
- HUD ECU Hacker can **sniff** the commands and responses to the ECU on any CAN bus or K-Line connection.
- HUD ECU Hacker can load / extract / convert proprietary tuning files: BIN, PAC, S19, HEX, CAL, PTP,...
- HUD ECU Hacker can be automated with **macros** in the C# language.
- HUD ECU Hacker can be **adapted** to new ECU models by adding XML files.
- HUD ECU Hacker can be **translated** to other languages by adding XML files.
- HUD ECU Hacker is **high quality** software from a software developer with 40 years of programming and reverse engineering experience.
- HUD ECU Hacker **Download**

Supported ECU's

HUD ECU Hacker can scan all ECU's from all motorbikes, ATV's, cars and trucks if they support the OBD2 protocol.
OBD2 also permits to see and clear the fault codes (DTC = Diagnostic Trouble Codes) that the ECU is reporting.
OBD2 is supported by all newer vehicles, but gives few details for the service technician
because it's only purpose is to control the compliance of exhaust emission laws.

Motorbikes + ATV's + Cars + Buses + Trucks

ECU Model	Connection	OBD2 Parameters	Fault Codes	Documentation
Autodetect OBD2	K-Line or CAN bus (14 protocols)	194	2435	**Wikipedia OBD2**

Additionally HUD ECU Hacker has implemented the **vendor specific** (proprietary) parameters of the following ECUs.
These give more details than the OBD2 parameters, but they are a secret of each vendor and not published anywhere.
Normally you have to buy an expensive scantool or software from the vendor to see these parameters.

And what concerns **flashing**: No vendor allows an ordinary vehicle owner (like YOU) to upload calibrations to the ECU.
Flashing is done in the factory and by some workshops which have a special license and pay a lot of money for it.
To avoid that YOU can flash your ECU, they protect the access to the memory by a security key, require a checksum, and more.

Motorbikes + ATV's + Honda Watercraft + Kohler Lawn Mowers

ECU Model	Connection	Speed (baud)	Scanning	Flashing	Tuning	ECU Connector Pins / Diagram	More Details
Delphi MT05 and MT05.2	K-Line ISO 14230	10.400	115 params	YES	YES	**Delphi MT05 & MC21 Manual.pdf**	
Kohler Engines MT05	K-Line ISO 14230	10.400	136 params	YES	YES	**Delphi MT05 & MC21 Manual.pdf**	
Briggs & Stratton MT05	K-Line ISO 14230	10.400	136 params	YES	YES	**Delphi MT05 & MC21 Manual.pdf**	
Rongmao MT05	K-Line ISO 14230	10.400	115 params	NO	NO	**Delphi MT05 & MC21 Manual.pdf**	See **Appendix 1**
Rongmao RM08	K-Line ISO 14230	10.400	115 params	NO	NO	**Delphi MT05 & MC21 Manual.pdf**	See **Appendix 1**
HiSun MT05	CAN ISO 15765	500.000	180 params	NO	NO	**Delphi MT05 & MC21 Manual.pdf**	See **Appendix 3**
Chinese Fake MT05	ISO 14230 or ISO 15765	10.400 or 500.000	OBD2 only	NO	NO	**Delphi MT05 & MC21 Manual.pdf**	See **Appendix 4**
Delphi MT05.3	CAN ISO 15765	500.000	180 params	In future	NO	**Delphi MT05 & MC21 Manual.pdf**	See **Appendix 2**
Bosch MSE 3.0	K-Line ISO 14230	10.400	41 params	NO	NO	**Bosch MSE 3.0 Manual.pdf**	
Bosch MSE 6.0 K-Line	K-Line ISO 14230	10.400	70 params	NO	NO	**Bosch MSE 6.0 Manual.pdf**	
Bosch MSE 6.0 CAN	CAN ISO 15765	500.000	88 params	NO	NO	**Bosch MSE 6.0 Manual.pdf**	
Bosch MSE 8.0	CAN ISO 15765	500.000	76 params	NO	NO	**Bosch MSE 8.0 Manual.pdf**	
Deni E1700	CAN Raw proprietary	250.000	55 params	NO	NO	**Deni EXX00 Manual.pdf**	
Honda K-Line	K-Line proprietary	10.400	77 params	In future	NO	**Honda Manual.pdf**	
Lifan EFI 9 Euro 4	K-Line ISO 14230	10.400	40 params	NO	NO	**Lifan EFI 9 Euro 4 Manual.pdf**	
Liteon MC21	K-Line ISO 14230	9.600	80 params	NO	NO	**Delphi MT05 & MC21 Manual.pdf**	
Motion SE08	K-Line ISO 14230	10.400	180 params	NO	NO	**Motion SE08 & DE08 Manual.pdf**	
Motion DE08	K-Line ISO 14230	10.400	180 params	NO	NO	**Motion SE08 & DE08 Manual.pdf**	
Yeson 28S-06 and 28S-16	K-Line ISO 9141	10.400	51 params	NO	NO	**Yeson 28S Manual.pdf**	

Cars + Vans

ECU Model	Connection	Speed (baud)	Scanning	Flashing	Tuning	ECU Connector Pins / Diagram	More Details
Bosch MSA 15 (Diesel)	K-Line KW 1281	9.600	48 params	NO	NO		
Bosch PSG 5 (Injection Pump)	K-Line KW 1281	9.600	10 params	In next version	NO	**Bosch PSG 5 Manual.pdf**	

Not Implemented

The vendor specific parameters of the following ECU's are not yet implemented but a manual is available.

ECU Model	Connection	Speed (baud)	Scanning	Flashing	Tuning	ECU Connector Pins / Diagram	More Details
Deni E0900	K-Line + CAN Raw		OBD2 only	NO	NO	**Deni EXX00 Manual.pdf**	
Deni E1900	K-Line		OBD2 only	NO	NO	**Deni EXX00 Manual.pdf**	
Harley Dyna 2016	CAN Bus	500.000	OBD2 only	NO	NO	**Harley Dyna 2016 Manual.pdf**	See **Appendix 5**
Rojo GY6-125	ISO 15765	500.000	Proprietary	NO	NO	**Rojo GY6 Manual.pdf**	
FAI	J1939	250.000	Proprietary	NO	NO	**Fai Manual.pdf**	

More ECU models will be added in the future. You can also add your own ECU.
HUD ECU Hacker can be **extended by the user** by adding XML files.
By defining the commands, parameters and formulas it can be adapted to other ECU's. See **below**.

OBD2 Scanner

I have a Regal Raptor 350 motorbike (still sold in 2023) which always ran perfectly...

Show Full Size

...until one day the **EFI** light turned on, which indicates a fault. (EFI = Electronic Fuel Injection)
On other motorbikes it is named **MIL** (Malfunction Indicator Lamp) or **CEL** (Check Engine Light) or **FI** (Fault Indicator).

km/h
0 6 1 2
180
EFI
N

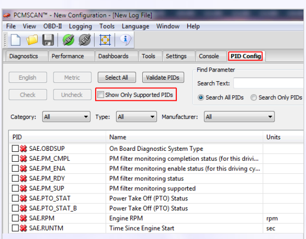
Show Full Size

Although the engine was running without noticeable problem, something was wrong.

I read in internet that all modern cars and motorbikes have an **OBD2** plug (OBD = On Board Diagnostics).
The **ECU** (Engine Control Unit) of the vehicle informs about the cause of the fault by returning a **DTC** (Diagnostic Trouble Code).

I searched for **OBD2** software which shows me this error code.
I found that nearly all software is **paid software** and not working without buying a license.
Or even worse: Some companies sell software together with hardware which acts as a **dongle**.

I tested for example **PCMScan** from Palmer and found that it is not able to read one single parameter of my motorbike:


PCMSCAN™ - New Configuration - [New Log File]
File View OBD-II Logging Tools Language Window Help
Diagnostics Performance Dashboards Tools Settings Console PID Config
English Metric Select All Validate PIDs
Check Uncheck Show Only Supported PIDs
Find Parameter
Search Text:
Search All PIDs Search Only PIDs

☐ Show Only Supported PIDs

● Search All PIDs ○ Search Only PIDs

Category: All Type: All Manufacturer: All

PID	Name	Units
☐ SAE.OBDSUP	On Board Diagnostic System Type	
☐ SAE.PM_CMPL	PM filter monitoring completion status (for this drivi...	
☐ SAE.PM_ENA	PM filter monitoring enable status (for this driving cy...	
☐ SAE.PM_RDY	PM filter monitoring status	
☐ SAE.PM_SUP	PM filter monitoring supported	
☐ SAE.PTO_STAT	Power Take Off (PTO) Status	
☐ SAE.PTO_STAT_B	Power Take Off (PTO) Status	
☐ SAE.RPM	Engine RPM	rpm
☐ SAE.RUNTM	Time Since Engine Start	sec

Show Full Size

The software told me that it has connected, but all parameters were marked with a **red cross** inidicating that the ECU does not support this parameter.
Not even such a basic parameter like 'Engine RPM' was displayed, nor did I see any fault code.
I analyzed the data traffic and found that the ECU answered all commands with **0x7F**, which is an error code.
What a luck that I did not purchase a license for this software, which is completely useless for me!

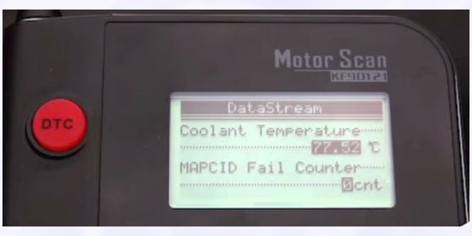
And no other OBD2 software that I tested was not able to communicate with my motorbike.
The Regal Raptor 350 uses a Delphi MT05 ECU.

> **The older MT05 ECU is not OBD 2 compliant.**
> **The newer Delphi MT05.2 implements a basic OBD2 support.**
> **But OBD2 was designed to check that a vehicle complies the emission laws.**
> **OBD2 data has a limited usefulness for the service technician.**
> **HUD ECU Hacker can display the OBD2 data from any vehicle,**
> **but much more useful is the vendor specific scan data.**
> **HUD ECU Hacker displays 90 detailed scan parameters with vendor-specific information**
> **of the Delphi MT05 which you will not find in any "universal" OBD2 scantool.**

Scantool

To scan the MT05 you would normally have to buy an expensive scantool like this.
On Youtube there is a **video** showing how to use it.

Show Full Size



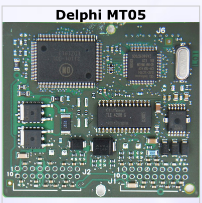
It comes in a suitecase which is bigger than a notebook. Price is approx $200 - $300 US.
It is **very primitive**: It has only 5 buttons and the LCD display displays **only 2 parameters** at once.
You have a much better display of all 90 parameters at once by using a notebook with **HUD ECU Hacker**.

ECU's from Delphi Electronics

Delphi MT05

Show Full Size

Delphi MT05



Show Full Size

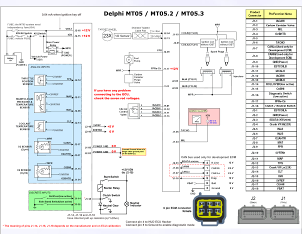
The Delphi **MT05** controls 1 or 2 cylinders. It is used mainly in motorbikes, ATV's and UTV's.
The Liteon **MC21** is used in Quads from Hercules and Cectec. It has a Motorla MC9S12D64 processor.

The **Delphi MT05 & MC21 Manual** (PDF) shows how these ECU's are connected.

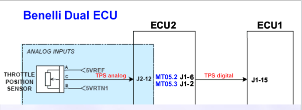
The older Liteon **MC21** connects with 9600 baud.
The MC21 diagram can be found in the Delphi Manual. You will not find a pin for the MAP sensor.
The MC21 has the MAP Sensor built into the ECU. The ECU is connected with a hose to the manifold.

Page 76 shows the 6 pin **diagnostic plug** (ECM connector)
The Delphi **MT05** and **MT05.2** use the ISO 14230 protocol over K-Line at 10400 baud and Fast Init.
The Delphi **MT05.3** uses CAN bus while K-Line is optional and it's functionality is crippled.

I added several missing details to the following MT05 diagram:


Delphi MT05 / MT05.2 / MT05.3
ANALOG INPUTS
DISCRETE INPUTS
If you have any problem connecting to the ECU, check the seven red voltages.
6 pin ECM connector female
J2
J1

Show Full Size



Show Full Size

Some Benelli motorbikes (QJ600, BN600, TNT600) have two MT05 ECU's, each controlling 2 cylinders in a 4 cylinder engine.
One ECU takes the analog TPS signal and forwards it as a digital signal to the second ECU.

The following table shows the pins of the MT05.3 which may be configured by the manufacturer in the calibrations.
Some pins may also be configured for testing purposes.

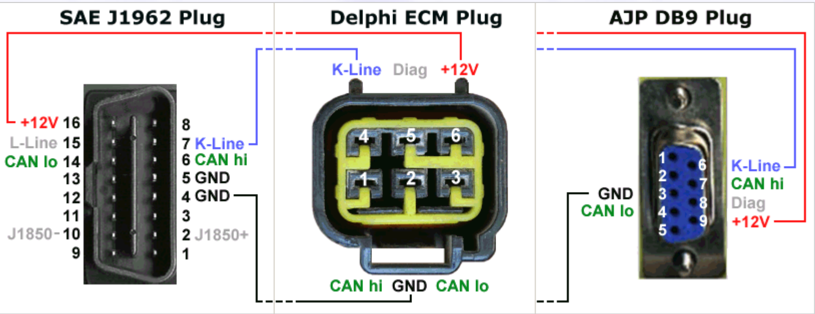
Pin	Type	Usage Options
J1-2	Output	ECP valve or Start motor disable relay or AC cooling fan relay or Head light relay or Second air injection valve
J1-3	Output	MIL Lamp or Misfire generation status of cylinder 1
J1-4	Output	O2 heater B or AC cooling fan relay or Head light relay or Second air injection valve
J1-5	Input	O2 Sensor B or Analog input 1
J1-6	Output	Tacho (RPM) or Toggle signal when MAP is read or Output of TPS duty cycle signal to second ECU (4 cylinder engines)
J1-14	Input	Rollover Sensor Input or Power adding switch input
J1-15	Input	O2 Sensor B or Analog input 2 or Input for TPS duty cycle signal from first ECU (4 cylinder engines)
J1-16	Input	Reset EEprom by short to ground or Idle RPM adjust or Differential lock
J1-18	Input	Clutch/Neutral switch or Reverse gear switch
J2-3	Output	K-Line communication or Head light relay or Second air injection valve
J2-6	Output	Injector B or Second air injection valve or Cooling fan relay
J2-12	Input	TPS sensor input or MAP sensor only system

Connecting to the MT05

HUD ECU Hacker supports the following adapters to connect with the ECU, which are described in the following chapters:

1. K-Line (VAG KKL) adapter (only K-Line)
2. VAG K+CAN adapter (K-Line + CAN bus)
3. J2534 (e.g. Tactrix Openport) adapter (K-Line + CAN bus)
4. ELM327 / OBDLink (USB, Bluetooth, WIFI) adapter (K-Line + CAN bus)
5. ZLG (Polaris) UsbCAN adapter (only CAN bus)

All these adapters have a standardized plug with 16 pins: The **J1962** plug.
ATTENTION: These adapters are never connected directly to the plugs at the ECU.
Motorbikes and ATV's have a separate diagnostic plug which is normally **under the seat**.
Most motorbikes and ATV's with the Delphi MT05 use the original ECM plug from Delphi (a black / yellow plastic plug with 6 pins).
AJP uses it's proprietary DB9 plug.



You only have to connect **3 wires** between the J1962 plug of the adapter and the motorbike: **Ground**, **+12V** and **K-Line**.
There are also 2 pins for **CAN** bus. But the firmware does not use them. They are only for the Delphi developers.

Normally pin 5 (**Diag**) is connected to the MT05 pin J1-16 (diagnostic switch) which enables **Diagnostic Mode** when switched to ground.
You can prove this easily in the dashboard. The blue ball must appear when you connect pin 5 (Diag) and pin 2 (Ground):

Diagnostic Mode

The meaning of 'Diagnostic Mode' depends on scalar 'J1-16 Input Usage' in the calibrations:

- 'J1-16 Input Usage' = 0 means that the pin is ignored (disabled).
- 'J1-16 Input Usage' = 1 resets the EEPROM at next key-off (see **Self-Learning**).
- 'J1-16 Input Usage' = 3 applies an alternate fuel cut-off if also scalar 'Alternate Overspeed Fuel Cutoff' is enabled.

Some **Benelli** motorbikes put a dummy plug with a jumper between 4 and 5 onto the ECM plug.
The LCD display in the dashboard is connected to pin 5 and sends commands over K-Line to obtain the coolant temperature from the ECU.

Which adapter to buy?

	K-Line / VAG KKL Adapter	J2534 Adapter	ELM327 / OBDLink Adapter
Connection	Only K-Line	K-Line + CAN bus	K-Line + CAN bus
Advantage	As there is no intelligence in the adapter even the counterfeit are working perfectly. You can also make your own **cheap** DIY adapter with 2 transitors.	**Professional** adapters for scanning and flashing. They are technically the best choice.	**Hobbyist** adapters. Bluetooth and WIFI versions available. There is no advantage over J2534 adapters.
Disadvantage	No support for CAN bus. Cannot measure battery voltage. Timing depends on computer speed.	The genuine are expensive but chinese clones exist which do work.	Misdesigned (see **below**) ECU Emulator and CAN Raw protocol do not work. All ELM327 adapters in internet are fake, except genuine OBDLink, ScanTool.
Summary	**OK** (limited use)	**recommended**	**deprecated**
Price Genuine	$5 USD	$180 ... $500 USD	OBDLink: $40 USD
Price Counterfeit	$5 USD **Do NOT buy Ross-Tech**	Chinese Clone: $20 USD is OK **Do NOT buy Mini-VCI fake**	$10 USD **Do NOT buy cheap fake!**

Here you see the order of most recommended to least recommended adapters:

1. **Genuine Tactrix J2534** **(100% recommended)**
2. **Chinese Tactrix clone** **(do not buy Mini-VCI fake!)**
3. **K-Line (VAG KKL)** **(not if your ECU uses CAN bus)**
4. **VAG K+CAN** **(only two CAN baudrates)**
5. **Genuine Scantool OBDLink USB**
6. **Genuine Scantool OBDLink Bluetooth**
7. **BMW K+DCAN** **(only K-Line)**
8. **Chinese fake ELM327 USB**
9. **Chinese fake ELM327 Bluetooth**
10. **Chinese fake ELM327 Wifi** **(most stupid design of all)**

HUD ECU Hacker Function		K-Line + BMW Adapters	J2534 Adapter	OBDLink Adapter	Fake Elm327 Adapter	UsbCAN Adapter	VAG K+CAN Adapter
Parameter Scanning	K-Line ISO 9141 K-Line ISO 14230	✓	✓	✓	✓	✗	✓
	CAN ISO 15765	✗	✓	✓	✓	✓	✓⚠
	CAN Raw	✗	✓	✗	✗	✓	✓⚠
Data Slewing	K-Line ISO 9141 K-Line ISO 14230	✓	✓	✓	✗	✗	✓
	CAN ISO 15765	✗	✓	✓	✗	✓	✓⚠
	CAN Raw	✗	✓	✗	✗	✓	✓⚠
Sniffing	K-Line ISO 9141 K-Line ISO 14230	✓	✓	✓	✗	✗	✓
	CAN ISO 15765	✗	✓	✓	✗	✓	✓⚠
	CAN J1939	✗	✓	✓	✗	✓	✓⚠
	CAN Raw	✗	✓	✓	✗	✓	✓⚠
Flashing MT05 / MT05.2	K-Line ISO 14230	✓	✓	✓	✗	✗	✓
ECU Emulator	K-Line Fast Init	✓	✓	✗	✗	✗	✓
	K-Line 5-Baud Init	✓	✗	✗	✗	✗	✓
	CAN ISO 15765	✗	✓	✗	✗	✓	✓⚠
	CAN Raw	✗	✓	✗	✗	✓	✓⚠
Proprietary Protocols	K-Line Honda	✓	✓	✓	✗	✗	✓
	K-Line KW 1281	✓	✗	✗	✗	✗	✓
Can measure battery voltage		✗	✓	✓	✓	✗	✗

✓⚠ VAG K+CAN adapters support only CAN buses with 100 kBaud and 500 kBaud.

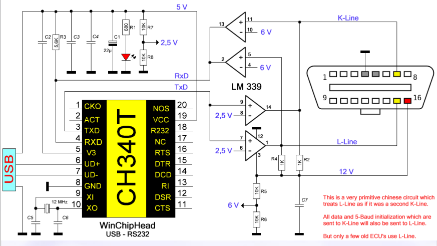
Option 1: K-Line Adapter

The **cheapest** option to connect to an ECU with K-Line protocol is using a VAG KKL adapter (approx $5 USD).
Additionally K-Line adapter are faster than J2534 adapters and much faster than ELM327 adapters.

IMPORTANT:
The ISO 14230 fast initialization requires a very precise timing.
K-Line adapters do not have a microprocessor, so the timing depends on the computer.
Windows is not a real-time operating system and delays are normal.
But K-Line adapters work very well on most computers. Only a few users have reported connection problems.
In case of a connection error close all software on your computer which consumes much CPU and try to connect multiple times.
See chapter **Trouble Shooting**.

VAG KKL Adapter

VAG KKL Adapter


5 V
2,5 V
RxD
TxD
K-Line
6 V
LM 339
L-Line
12 V
USB
CKO
ACT
TXD
RXD
V3
UD+
UD-
GND
XI
XO
CH340T
NOS
VCC
R232
NC
RTS
DTR
DCD
RI
DSR
CTS
12 MHz
WinChipHead
USB - RS232
This is a very primitive chinese circuit which treats L-Line as if it was a second K-Line.
All data and 5-Baud initialization which are sent to K-Line will also be sent to L-Line.
But only a few old ECU's use L-Line.

Show Full Size

The +12V on the diagnostic plug are always present, even when the ignition key is off.
If you don't need the adapter anymore don't let it connected for hours because it permanently draws current from the battery.

In the HUD ECU Hacker **toolbar** (button **Install USB Driver**) you can install the drivers for all types of K-Line adapters.
If you live in Iran you can request a free K-Line adapter. Contact **EfixMotor** in China.

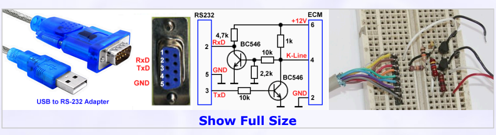
For Electronic Experts Only

You can also build your own K-Line adapter with a cheap (approx $1 USD) USB to RS232 adapter and 2 transistors.
For a quick testing this can be built on a breadboard.

In internet you may find circuits that use an opto-coupler. This is complete nonsense!
There is no need to use any opto-coupler.

IMPORTANT: Build your own adapter ONLY if you have experience with electronics!
I receive emails from beginners who fail to build their own circuit. I will not give you support for this.
Buy a complete adapter if you do not understand how a transistor works.
Simply forget it if you don't even have an **oscilloscope**.

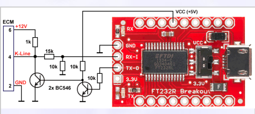
Option 1:
If you already have an USB to RS232 adapter you can use it.



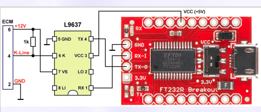
Show Full Size

The signal must be **inverted** from RS232 to K-Line and back.
The RS232 lines TxD and RxD are low when idle, while K-Line is high when idle.

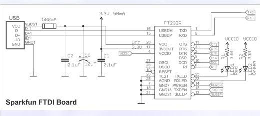
Option 2:
You can also use a **Sparkfun breakout board** with the high quality FTDI chip.
In this case the Rx and Tx signals must **not** be inverted.
The Tx and Rx pins of the FTDI chip are high when idle.
This FTDI board has two LED's which are flashing when data is transferred on Rx and Tx.
If you use another board where the Rx LED is stupidly connected directly to the RX pin of the board you must remove this LED.



Show Full Size



Show Full Size

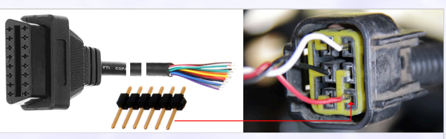


Show Full Size

K-Line is half duplex, so only the computer or the ECU can send data alternately, but not at the same time.
All data sent from the computer via TxD is then received as **echo** on RxD.
The computer will always first receive the echo of it's own command and then, after a pause, the response from the ECU.
This allows to easily detect connection problems. If **no echo** is received, there is always a **hardware problem**.
HUD ECU Hacker always verifies the echo but it does not show the echo in the Trace pane, except the echo is corrupt.

1a) Build your own ECM Cable (for Delphi MT05)

You can buy a J1962 Female Connector and solder the 3 wires to 3 **pin headers** which perfectly fit into the ECM plug.



Show Full Size

Or you buy the 6 Pin Furukawa FW090 Male Connector **FW-C-6M-B** at Cycleterminal: **here** or at Taobao: **here** or **here** or **here**.
Or you **search on Google** for more companies selling this plug on eBay, Alibaba or AliExpress.

Then you can either use a crimping plier or solder the wires to the contacts and build your cable as shown in this **video**.

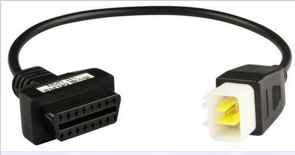
1b) Buy a complete ECM Cable (for Delphi MT05)

If you have time you can also buy a complete cable in China. Shipping may take between 1 and 3 months.

Cable USB to ECM from **Taobao**

Show Full Size

Cable J1962 to ECM from **AliExpress**



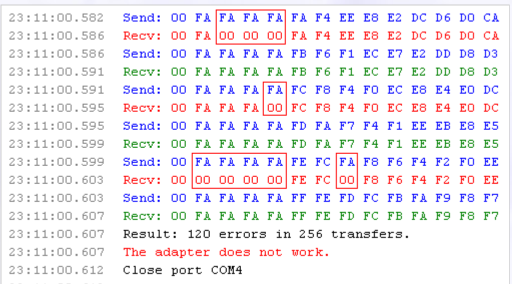
Show Full Size

The red cable from Taobao has a very limited support of baudrates.
In the window "Configure Adapter" you must switch to **Fast Init Mode 2** otherwise you get a timeout when connecting.

K-Line Adapter Echo Test

I discovered severe problems with cheap chinese USB to RS232 adpaters containing the widely used **CH340 chip**.
UPDATE: After many years WinChipHead finally fixed the bugs in the CH340 chip and the driver.
Verify in the Trace pane that you have driver version 3.8 or higher and a chip with firmware version 2.63 or higher.

I implented the **Echo Test** into HUD ECU Hacker.
It sends data to K-Line, verifies the echo and measures the speed.
Here you see the test result of a faulty CH340 adapter (green = correct response, red = wrong response):



```
23:11:00.582  Send: 00 FA FA FA FA FA F4 EE E8 E2 DC D6 D0 CA
23:11:00.586  Recv: 00 FA 00 00 00 FA F4 EE E8 E2 DC D6 D0 CA
23:11:00.586  Send: 00 FA FA FA FA FB F6 F1 EC E7 E2 DD D8 D3
23:11:00.591  Recv: 00 FA FA FA FA FB F6 F1 EC E7 E2 DD D8 D3
23:11:00.591  Send: 00 FA FA FA FA FC F8 F4 F0 EC E8 E4 E0 DC
23:11:00.595  Recv: 00 FA FA FA 00 FC F8 F4 F0 EC E8 E4 E0 DC
23:11:00.595  Send: 00 FA FA FA FA FD FA F7 F4 F1 EE EB E8 E5
23:11:00.599  Recv: 00 FA FA FA FA FD FA F7 F4 F1 EE EB E8 E5
23:11:00.599  Send: 00 FA FA FA FA FE FC FA F8 F6 F4 F2 F0 EE
23:11:00.603  Recv: 00 00 00 00 00 FE FC 00 F8 F6 F4 F2 F0 EE
23:11:00.603  Send: 00 FA FA FA FA FF FE FD FC FB FA F9 F8 F7
23:11:00.607  Recv: 00 FA FA FA FA FF FE FD FC FB FA F9 F8 F7
23:11:00.607  Result: 120 errors in 256 transfers.
23:11:00.607  The adapter does not work.
23:11:00.612  Close port COM4
23:11:00.612  ==========================================
```

You can execute the echo test after connecting the K-Line / VAG adapter to the motorbike.
The echo test will fail if you connect only the adapter over USB to the computer. The +12V are required.
The +12V at the ECM plug are connected directly to the battery and are not affected by the ignition key.

Option 2: VAG K+CAN Adapter

VAG K+CAN

Show Full Size

This cheap USB adapter has a FTDI chip and works perfectly as **K-Line** adapter.
Additionally it has a PIC18F25K80 microprocessor that can communicate with **CAN bus**.
But how to activate CAN mode and what commands to send to the adapter is completely undocumented.
There is only one software that can use this adapter over CAN bus: VAG-K+CAN Commander.
The Chinese sell a clone of the original VAG adapter with a cracked version of the VAG software on a CD.
I had to do a complex reverse engineering to find out how to configure this adapter for CAN bus.

ATTENTION:
The original adapter was developed by VAG for Volkswagen and Audi cars.
These use 500 kBaud for the Drive train CAN bus and 100 kBaud for the Convenience and Infotainment CAN bus.
While this adapter works on K-Line with any baudrate, for CAN bus only **100 and 500 kBaud** are implemented.
If you need CAN bus, it is recommended to buy a J2534 adapter instead which supports a wide range of baudrates.
However, many CAN bus ECU's use 500 kBaud, so this adapter may be useful for the majority of CAN bus ECU's.

Option 3: BMW K+DCAN Adapter (INPA)

VAG K+CAN

Show Full Size

This cheap USB adapter has a FTDI chip and works perfectly as **K-Line** adapter.
Additionally it has an ATMEGA162 microprocessor that can communicate with the **DCAN bus** from BMW.
The adapter has a switch that connects pin 7 (K-Line) with pin 8 which is normally unused. Only older BMW cars use pin 8.
The Chinese sell a clone of the original BMW adapter with a cracked version of the BMW software INPA on a CD.

ATTENTION:
The original adapter was developed by BMW specifically for their cars.
The CAN bus functionality is **only** useful for BMW: It supports only 100 kBaud and 500 kBaud.
It listens only on the 11 bit CAN ID's 130 and 600 ... 6FF and supports only ISO 15765 protocol with extended addressing.
All this configuration is hard coded in the firmware of the adapter and cannot be changed.
Additionally the adapter has a higher power consumption than other adapters. It becomes warm.
Do not buy this adapter. If you already have one, it is only useful for K-Line. The position of the switch is irrelevant.

Option 4: J2534 Adapter (recommended)

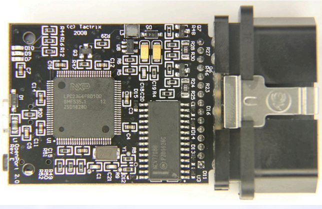
J2534 adapters are the recommended choice. They support K-Line and CAN bus.
J2534 (PassThru) is an international standard for communication with ECU's.
If you are interested in the details read the **API Documentation** (PDF) for programmers.
The genuine J2534 adapters (for example **Tactrix OpenPort** or **Drewtech Mongoose**) are very expensive ($180 ... $500 USD).
Chinese clones are significantly cheaper. See below.

ATTENTION: Do not buy **XHorse Mini-VCI** adapters. They are Chinese fake garbage. They do NOT work with HUD ECU Hacker!

Genuine Tactrix

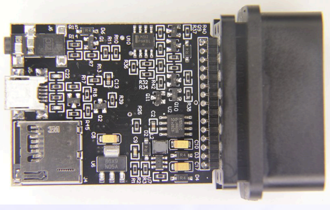
Show Full Size

Genuine Adapter Top



Show Full Size

Genuine Adapter Bottom



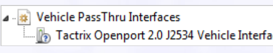
Show Full Size

Do NOT buy!

!! FAKE !!
MINI-VCI
J2534
Standard OBDII
Communication Interface

Show Full Size

When you plug in the adapter for the first time Windows installs a default driver and assigns a COM port.
ATTENTION: This is the wrong driver and the COM port will never work.
In the HUD ECU Hacker **toolbar** (button **Install USB Driver**) you can install the original Tactrix driver.
After installing the correct driver the COM port will disappear and a J2534 device will show up in Control Panel:

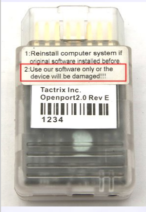

Vehicle PassThru Interfaces
Tactrix Openport 2.0 J2534 Vehicle Interface

J2534 K-Line Echo Test

If you get a timout error when connecting to the ECU over K-Line you can execute the echo test.
HUD ECU Hacker will send some data packets to K-Line and verify that the echo is received correctly.
If this test fails you have either a defective adapter or K-Line has a shortcut.
Make sure that K-Line has +12 Volt while the adapter is connected.

Chinese J2534 Adapter Clone

Tactrix Clone


1:Reinstall computer system if original software installed before
2:Use our software only or the device will be damaged!!!
Tactrix Inc.
Openport2.0 Rev E
1234

Show Full Size

Here you see a Chinese clone of a Tactrix adapter that is significantly cheaper than the original.
These clones are an identical copy of the original hardware and they work perfectly.
You can buy a cheap J2534 clone from **EfixMotor** in China for $20 US or search on **Alibaba**.
If you see "1234" below the barcode you have a Chinese counterfeit adapter.
The genuine Tactrix shows the unique serial number below the barcode which are 8 letters.

The only problem is that the label says:
"Reinstall computer system if original software installed before."
"Use our software only or the device will be damaged!!!"

Nothing will be damaged when you use this adapter with HUD ECU Hacker.
Use the toolbar button **Install USB Driver** in HUD ECU Hacker and install the original Tactrix driver.

This label says that you must **NOT** use the EcuFlash software from Tactrix to upload a new firmware to the adapter.
The Chinese use their own firmware. But there is no need to update the firmware in the adapter.
By the way: The latest firmware is from 2016.

Option 5: ELM327 Adapter (deprecated)

ELM327 / OBDLink / Scantool adapters support K-Line and CAN bus.
There are 3 types of ELM327 adapters:

1. **Chinese ELM327 clones**:
   The internet is full of fake ELM327 adapters. You find them on eBay, Amazon, AliExpress, etc.
   All these adapters are **garbage**. The Chinese did not even implement half of the command set.
   You send a command to the adapter, it answers with 'OK' but it does not execute the command.
   These adapters are **fraud**. All adapters for less than $40 USD are **fake**!
   DO NOT BUY THIS CRAP!
   If you already have one you can use it to scan the parameters, but ECU Emulator and Data Slewing and Flash Up/Download will not work.

2. **Genuine ELM327 adapters**:
   Genuine ELM327 adapters have the ELM327 chip inside. Download **Datasheet** (PDF)

   ELM Electronics sells only the **ELM327 chip** ($21 CAD), but they do not offer an own adapter.
   Genuine adapters are difficult to find because very few companies offer them: **WGsoft** (105€), **Warenhuis** (109€).
   But even if you have a genuine adapter, Flash Upload will not work because they do not support to set a long timeout.

3. **Genuine OBDLink adapters**:
   Genuine OBDLink adapters have a STN11XX chip inside. Download **Datasheet** (PDF)

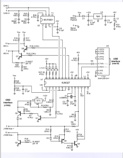
   They are sold by **Scantool** and **OBDLink** ($40 USD).
   They implement the same AT commands as genuine ELM327 adapters and have additional ST commands.
   If you want to use an ELM327 adapter you should **ONLY** buy it from Scantool or OBDLink.
   My adapter was sold with a very old firmware. Don't forget to update the firmware.
   Even in a genuine OBDLink adapter I found a severe bugs.
   Even with the genuine OBDLink adapters the ECU Emulator will not work.

4. **Other adapters**:
   There are more adapters which support the extended ST commands, like
   vLinker MC WiFi, vLinker MC+, vLinker FS BT. I did not test them.

ELM327 adapters are a **misdesign**. They are also significantly **slower** than the other adapters.
They have too many commands which makes programming complicated.
Instead of leaving the intelligence in the controlling software (as J2534 adapters do) all the intelligence is in the chip.
The chip must be configured with hundreds of commands.
Instead of transmitting binary data directly (as J2534 adapters do), they use ASCII strings, which is simply a bad design.
ISO 15765 data is passed by the adapter with missing CAN ID or must be parsed in the controlling software (STUPID design!)
Instead of using internally an USB capable processor (as J2534 adapters do) they convert USB first to RS232 which is slower and the COM port must be configured with the correct baudrate, while J2534 adapters neither need COM ports nor baudrates.
If you have very fast CAN bus traffic the ELM327 loses packets because the internal serial connection is too slow.
Another issue is that the ELM327 can store configuration in non-volatile memory resulting in not predetermined behaviour.
None of the ELM327 adapters (not even the genuine) supports the CAN Raw protocol.
The internal buffer in the adapter is too small for high speed CAN bus traffic. You will get BUFFER FULL errors.
So, if you already have a ELM327 or OBDLink adapter study the Trace pane to see how many errors you get.
But if you don't have an adapter yet, **do not buy it**. See summary **above**.

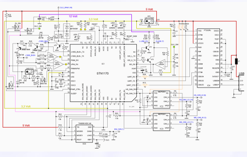
Diagrams of a genuine adapters:

EML327



Show Full Size

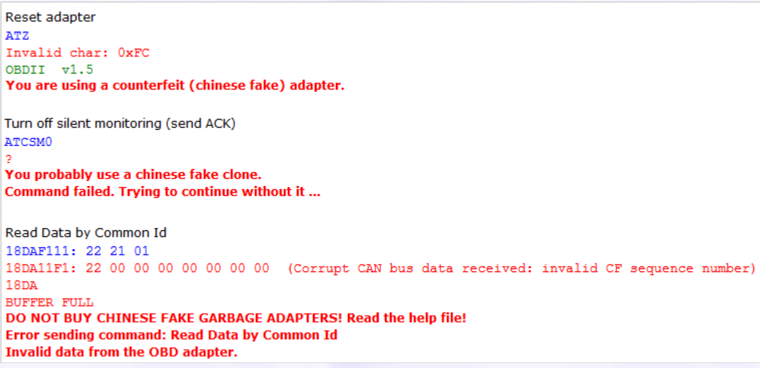
OBDLink



Show Full Size

Counterfeit ELM327 Adapters

Do **NOT** buy any of the cheap Chinese ELM327 adapters which are sold on **Amazon** or **eBay**!
They are all fake and work only partially. Many ELM327 commands are buggy or even not implemented at all.
If you see one of the following errors in the HUD ECU Hacker Trace pane, the Chinese have betrayed you:



```
Reset adapter
ATZ
Invalid char: 0xFC
OBDII  v1.5
You are using a counterfeit (chinese fake) adapter.

Turn off silent monitoring (send ACK)
ATCSM0
?
You probably use a chinese fake clone.
Command failed. Trying to continue without it ...

Read Data by Common Id
18DAF111: 22 21 01
18DA11F1: 22 00 00 00 00 00 00 00  (Corrupt CAN bus data received: invalid CF sequence number)
18DA
BUFFER FULL
DO NOT BUY CHINESE FAKE GARBAGE ADAPTERS! Read the help file!
Error sending command: Read Data by Common Id
Invalid data from the OBD adapter.
```

The buffer of the fake adapters is so tiny that it cannot even receive a 128 byte ECU response!
So you can only scan the OBD2 commands which give few information, but not the detailed vendor specific commands.

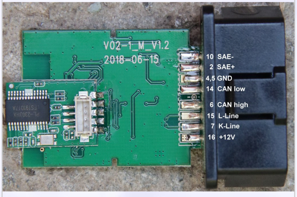
You will see none of the above errors if you use a genuine OBDLink adapter.

Counterfeit ELM327 USB Adapters

Chinese Clone

Show Full Size

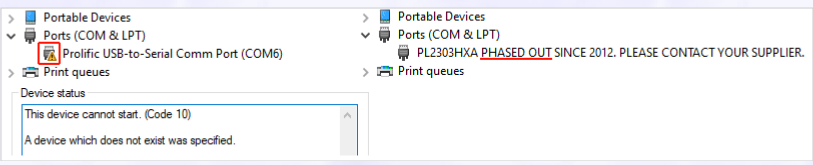
Chinese Clone



Show Full Size

Some chinese USB adapters use a counterfeit PL2303 chip (right photo) which converts from USB to RS232.
On Windows XP and Windows 7 this works perfectly.
But on **Windows 8 and 10** the latest drivers from Prolific detect the counterfeit chip and refuse to work.
The driver returns the undocumented **Error 433**: "A device which does not exist was specified."

If you connect the adapter and Windows 10 does not find an installed driver it downloads the latest **version 3.8** from Windows Update and you will see a yellow exclamation mark or a 'PHASED OUT' error:



This device cannot start. (Code 10)

A device which does not exist was specified.

PL2303HXA PHASED OUT SINCE 2012. PLEASE CONTACT YOUR SUPPLIER.

Do not use the drivers from a CD or from Windows Update.
I have implemented the driver installation into the toolbar at the top of HUD ECU Hacker.
Install the Prolific driver version 3.3 from 2008 which also works on Windows 8 and 10.

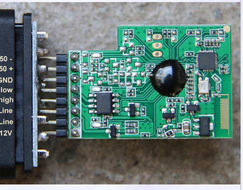
Counterfeit ELM327 Bluetooth Adapters

Chinese Clone

Vgate™ Scan
Power
OBD
PC
OBD
Scan
ADVANCED OBD SCAN TOOL

Show Full Size

Chinese Clone

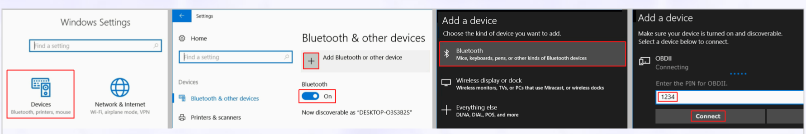


Show Full Size

On the right photo you see that there are several SMD parts missing (4 transistors and 16 passive components).
This means that the J1850 bus will not work. Only CAN and K-Line are implemented.
If a vendor sells this as a universal ELM327 adapter, this is a fraud.
However, the J1850 bus was used by older GM and Ford vehicles, but is not used in modern cars anymore.

Installing a Bluetooth Adapter

Follow these steps on Windows 10 to add a bluetooth adapter: (on Windows 7 it is similar)


Windows Settings
Find a setting
Devices
Bluetooth, printers, mouse
Network & Internet
Wi-Fi, airplane mode, VPN
Settings
Home
Find a setting
Devices
Bluetooth & other devices
Printers & scanners
Bluetooth & other devices
Add Bluetooth or other device
Bluetooth
On
Add a device
Bluetooth
Mice, keyboards, pens, or other kinds of Bluetooth devices
Wireless display or dock
Wireless monitors, TVs, or PCs that use Miracast, or wireless docks
Everything else
DLNA, DIAL, POS, and more
Add a device
OBDII
Connecting
Connect

Now discoverable as "DESKTOP-O3S3B2S"

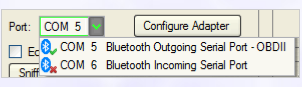
Choose the kind of device you want to add.

Make sure your device is turned on and discoverable.
Select a device below to connect.

Enter the PIN for OBDII.
1234

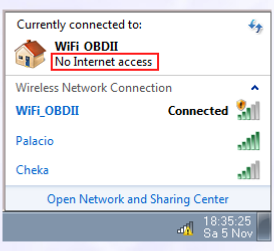
Show Full Size

If it was successful you see 2 COM ports in the combobox 'Port:' in HUD ECU Hacker:


Port:
COM 5
Configure Adapter
COM 5 Bluetooth Outgoing Serial Port - OBDII
COM 6 Bluetooth Incoming Serial Port

Select the Outgoing serial port which is connected to the bluetooth device "OBDII".
The Incoming serial port is useless.

Counterfeit ELM327 WIFI Adapters



These adapters are the most stupid design because they represent a WIFI access point.
Mostly they respond on the fix IP address **192.168.0.10** and port **35000** without WIFI password.
The problem is that you must disconnect your notebook from your router to connect it to the adapter.
So after connecting to the OBD adapter you lose internet access.
The signal strength and maximum distance between computer and adapter are worse than with Bluetooth adapters.
The adapter uses the most used Wifi Channel 11. So conficts with your or your neighbour's router are probable.
There is absolutely no reason why should buy these adapters.

Option 6: UsbCAN Adapter (deprecated)

CAN0H
CAN0L
CAN0R+
CAN0R-
CAN1H
CAN1L
CAN1R+
CAN1R-
CAN1
CAN0
STATUS
USBCAN
USB
51EDA Technology
Professional CAN-bus tools

Show Full Size

If you already have a Chinese ZLG (Polaris) UsbCAN adapter, you can use it with HUD ECU Hacker.
However, do not buy one because it supports only CAN bus and the driver is worst Chinese 'quality' with many bugs.
Most of the pins are fake. Only the uppermost (CAN0H and CAN0L) are connected internally.
The LED's are extremely stupid: Red LED blinking means OK. Green LED illuminated means CAN bus error.
To install the driver click the toolbar button **Install USB Driver** in HUD ECU Hacker.

KW 1281 Protocol

KW 1281 is a proprietary Bosch/Volkswagen protocol and the oldest protocol implemented in HUD ECU Hacker.
It was used in old ECU's from Bosch (Made in Germany) in cars from Volkswagen and Opel.
KW 1281 uses K-Line with 5-Baud Init and 9600 baud data transfer.

These are the transferred bytes that you see in the Trace pane when reading an analog ECU input (sensor):

```
Command  (from PC):  04 61 ( 08 02 ) 03
Response (from ECU): 05 62 ( FB 55 01 ) 03
```

But invisibly in the background the double amount of bytes is transferred.
Blue: Bytes sent by the tester (HUD ECU Hacker)
Green: Bytes sent by the ECU

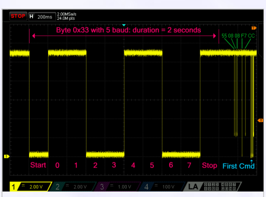
Command				Response			
Header 1	04		Packet Length	Header 1	05		Packet Length
		FB	Inverted: FB = 04 XOR FF			FA	Inverted: FA = 05 XOR FF
Header 2	61		Counter	Header 2	62		Counter + 1
		9E	Inverted: 9E = 61 XOR FF			9D	Inverted: 9D = 62 XOR FF
Payload 1	**08**		Command Read ADC Channel	Payload 1	**FB**		Response Read ADC Channel
		F7	Inverted: F7 = 08 XOR FF			04	Inverted: 04 = FB XOR FF
Payload 2	**02**		Channel no. 2	Payload 2	**55**		ADC Value High Byte
		FD	Inverted: FD = 02 XOR FF			AA	Inverted: AA = 55 XOR FF
ETX	03		End of Transfer	Payload 3	**01**		ADC Value Low Byte
						FE	Inverted: FE = 01 XOR FF
				ETX	03		End of Transfer

As you see the KW 1281 protocol is very stupidly designed:
Each and every received byte is sent back inverted which produces 100% overhead.
Addionally K-Line adapters have a very bad performance when single bytes are transferred one by one over USB.
The result of this inefficent transfer is that KW 1281 is much **slower** than modern ISO protocols.

IMPORTANT: The Bosch ECU's are extremely primitive. They don't have a mechanism to handle transfer errors.
If anything goes wrong during the transfer the ECU sends 4 garbage bytes and then stops communication.
Timing is very critical. If an expected byte needs longer than 40 ms the ECU blocks itself and does not respond anymore.
To avoid timing problems do not run processes on your computer that consume much CPU.

ISO 9141 Protocol

The ISO 9141 protocol is the oldest of the standardized OBD protocols. It is quite primitive.
The data transfer on **K-Line** (Pin 7) is like RS-232, sending the least significant bit first, but the voltage is inverted.



Show Full Size

ISO 9141 uses the extremely slow **5 Baud Init** to wake up the ECU. It takes 2 seconds to transmit the address byte (0x33).
Some ECU's require this 5 Baud Init to be sent additionally on **L-Line** (Pin 15).

5-Baud Initialization			
Sender	**Data**	**Baudrate**	**Meaning**
Tester	0x33	5 Baud	Address
ECU	0x55	10400 Baud	Synchronization
ECU	0x08 (0x94)	10400 Baud	Keyword 1
ECU	0x08 (0x94)	10400 Baud	Keyword 2
Tester	0xF7 (0x6B)	10400 Baud	Keyword 2 (inverted: F7 = 08 XOR FF)
ECU	0xCC	10400 Baud	Address (inverted: CC = 33 XOR FF)
Tester	Packet	10400 Baud	First Command

Next you see the first OBD2 command 01 00 (Request supported PID's) and the response from the ECU.
The command is embedded into a packet which starts with a header and ends with a checksum.

```
Command  (from PC):  68 6A F1 ( 01 00 ) C4
Response (from ECU): 48 6B 11 ( 41 00 BE 36 B0 03 ) A9
```

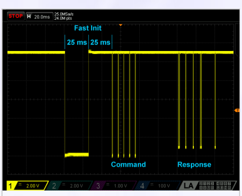
OBD2 Command '01 00'			Response (1...7 data bytes)		
Header 1	68	Fix value	Header 1	48	Fix value
Header 2	6A	Fix value	Header 2	6B	Fix value
Header 3	F1	Source address (Tester)	Header 3	11	Source address (ECU)
Payload 1	**01**	OBD2 Service 1	Payload 1	**41**	Service confirmation = **01** + 40
Payload 2	**00**	PID 0	Payload 2	**00**	PID confirmation
Checksum	C4	68+6A+F1+01+00 = C4	Payload 3	**BE**	8 Bits encoding supported PID's
			Payload 4	**36**	8 Bits encoding supported PID's
			Payload 5	**B0**	8 Bits encoding supported PID's
			Payload 6	**03**	8 Bits encoding supported PID's
			Checksum	A9	48+6B+11+41+00+BE+36+B0+03 = A9

ISO 9141 does not define error codes.
The ECU simply does not respond when it does not understand a command.

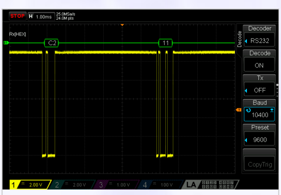
ISO 14230 Protocol

The Keyword 2000 protocol (KWP 2000) is defined in **ISO 14230**. It offers much more functionality than ISO 9141.
ISO 14230 normally uses **Fast Init** to wake up the ECU.
Fast Init means that the K-Line goes low for exactly 25 ms and then high for 25 ms. After that the communication starts with 10400 baud.
However, there are a few ECU's which combine the ISO 14230 protocol with 5-Baud Init.

On the left you see the fast init, followed by the command 'Start Communication' and the response from the ECU.
On the right you see long pauses between the bytes which are needed if the ECU is old and very slow.



Show Full Size



Show Full Size

In detail the command 'Start Communication' (Service = 81) looks like this:

```
Command  (from PC):  81 11 F1 ( 81 ) 04
Response (from ECU): 83 F1 11 ( C1 EF 8F ) C4
```

The Delphi MT05 uses the address 11. The application on the PC (the tester) uses the address F1.
The MT05 responds with 2 **key bytes** EF and 8F which define how the ECU wants the commands to be formatted.
They define how to transmit the packet length and if the source/target addresses are to be sent.
You see the meaning of the key bytes in the Trace pane in magenta when connecting.

Command 'Start Communication'			Response Short (1...63 data bytes)		
Header 1	81	80 + length of data (1 byte)	Header 1	83	80 + length of data (3 bytes)
Header 2	11	Destination address (ECU)	Header 2	F1	Destination address (tester)
Header 3	F1	Source address (tester)	Header 3	11	Source address (ECU)
Payload 1	**81**	Service 'Start Communication'	Payload 1	**C1**	Service confirmation = **81** + 40
Checksum	04	81+11+F1+81 = 04	Payload 2	**EF**	Key byte 1 (bit flags)
			Payload 3	**8F**	Key byte 2 (always 0x8F)
			Checksum	C4	83+F1+11+C1+EF+8F = C4

The first header byte is called **format byte**.
It may contain the packet length and it's bits define if addresses are sent and the type of addresses (physical, functional).

To simplify reading the binary data HUD ECU Hacker displays the data bytes in parenthesis in the Trace pane:
81 11 F1 (81) 04
83 F1 11 (C1 EF 8F) C4
The other bytes are not really interesting as they are generated automatically.

The following table shows a long response (102 data bytes) which contains an additional length byte (header 4).

Command 'Read Data'			Response Long (64...255 data bytes)		
Header 1	82	80 + length of data (2 byte)	Header 1	80	Extra length byte follows
Header 2	11	Destination address (ECU)	Header 2	F1	Destination address (tester)
Header 3	F1	Source address (tester)	Header 3	11	Source address (ECU)
Payload 1	**21**	Service 'Read Data'	Header 4	66	Length of data (102 byte)
Payload 2	**01**	Subfunction 1	Payload 1	61	Service confirmation = **21** + 40
Checksum	A6	82+11+F1+21+01 = A6	Payload 2	01	Subfunction confirmation = **01**
			Payload 3	...	Parameter raw data byte 1
			Payload 102	...	Parameter raw data byte 100
			Checksum	...	80+F1+11+66+61+01+...

HUD ECU Hacker tries first to connect with the physical address defined in the parameter XML file (Default = 0x11).
If this fails it waits 5 seconds and tries again with the functional address 0x33.
A **physical address** (format byte contains 0x80) means that one specific device on a bus is addressed.
A **functional address** (format byte contains 0xC0) is like a broadcast address to a group of devices.
There are functional addresses for Steering Controllers, ABS Systems, Air Condition, Audio, Lightning, etc.
The functional address 0x33 is used to address "Engine Controllers" (ECUs).
As normally only one ECU is connected this can be used when the physical ECU address is unknown.
As you see here the ECU responds to the functional address 0x33 with it's physical address 0x11:
C1 33 F1 (81) 66
83 F1 11 (C1 EF 8F) C4
The Delphi MT05 does not need functional addressing, but there are strange ECU's which do not respond to their own physical address!

Errors

ISO 14230 defines several error codes which the ECU can return.
If the ECU does not understand a command it sends 7F (failure) followed by the service and the error code.

Keep-Alive

If the ECU does not receive commands it switches to sleep mode after 5 seconds.
While HUD ECU Hacker is polling data this will never happen because polling takes place 3 to 5 times per second.
Only if you switch to manually enter commands (in the Trace pane), polling stops and HUD ECU Hacker sends a Keep-Alive every 3 seconds.

```
Command  (from PC):  81 11 F1 ( 3E ) C1
Response (from ECU): 81 F1 11 ( 7E ) 01
```

Honda Protocol

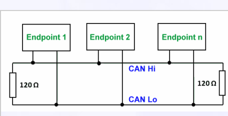
Honda uses an undocumented proprietary protocol over K-Line at 10400 baud.
A pulse of 70 ms similar to the Fast Init of ISO14230 is sent to wake up the ECU before communication starts.

```
Command  (from PC):  72 05 ( 00 F0 ) 99
Response (from ECU): 02 04 ( 00 ) FA
```

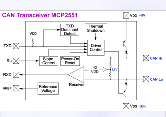
Command 'Initialize'			Response		
Header 1	72	Command	Header 1	02	Command AND 0x0F
Header 2	05	Byte count in this packet	Header 2	04	Byte count in this packet
Payload 1	00	Parameter byte 1	Payload 1	00	Response byte 1
Payload 2	F0	Parameter byte 2	Checksum	FA	02+04+00+FA = 00
Checksum	99	72+05+00+F0+99 = 00			

CAN Bus

Newer vehicles are equipped with **CAN bus** which uses 2 wires (CAN Hi and CAN Lo) and runs mostly at 250 or 500 kBaud.
Many controllers and sensors may be connected to the same bus. Each endpoint has at least one unique ID.
The neutral voltage on CAN bus is 2.5 Volt. Each endpoint has a transmitter which pulls CAN Hi up and CAN Lo down.
The receiver measures the difference between CAN Hi and CAN Lo which makes CAN robust against electromagnetic interference.



Show Full Size

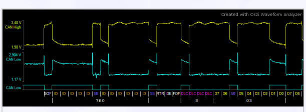


Show Full Size

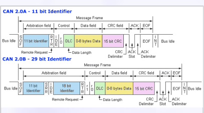
Additionally an intelligent arbitration system in each endpoint detects if 2 endpoints try to send data at the same time.
In case of such a collision it is clearly defined which enpoint has priority.
The endpoint which loses arbitration must stop transmission and try to send the packet again later.

Below you see a raw CAN frame which contains the identifier (11 bit or 29 bit) and max 8 data bytes.
Data transfer is very robust because a 15 bit CRC assures that each frame is received without error.

If you have to analyze problems with CAN bus and you have an oscilloscope that can connect to a computer over USB,
you can use my open source software **Oszi Waveform Analyzer** which converts your oscilloscope into a logic analyzer.
After A/D conversion of the analog inputs the CAN bus decoder shows the bits of the CAN packets, as you see here:



Show Full Size



Show Full Size

The CAN bus Identifier (ID) is similar to the **ECU address** in the ISO 14230 protocol.

Some ECU's permanently **broadcast** data to the CAN bus.
This is mostly engine speed, coolant temperature and MIL lamp status which the dashboard displays to the driver.
In motorbikes with ABS system you may also see the ABS ECU broadcasting the front and rear wheel speed.

As you see in the diagram above there is an **ACK Slot** in the CAN frame.
The idea is that the receiver of a packet sets the ACK bit to **ack**nowledge that it has received the packet.
If the packet was not acknowledged the sender may send it again.
If you see the same CAN packet beeing sent over and over again within one millisecond the cause may be that nobody has acknowledged it.
But broadcast messages are normally sent in fix intervals and need not necessarily be acknowledged.
ELM327, OBDLink and UsbCAN adapters allow to chose if you want to silently monitor the CAN bus without interfering by not setting the ACK bit.
However J2534 and VAG K+CAN adapters always acknowledge all packets on the bus. You cannot turn this off.

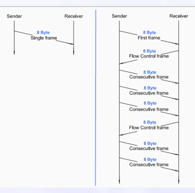
CAN Raw Protocol

If your ECU uses the **CAN Raw** protocol you will see completely undocumented and proprietary packets which differ with each vendor.
Mostly these ECU's have one address (CAN ID) on which they receive commands and multiple addresses on which they send responses.
Here you see the CAN Raw traffic of the Deni 1700 ECU which broadcasts data on multiple CAN ID's:

```
104: 5D C4 C4 82 00 7F 34 03
105: 00 00 20 00 00 98 4B 76
106: 00 00 00 00 00 00 00 B4
107: 90 80 E8 03 10 27 01 29
108: 00 F0 64 00 00 00 64 00
109: 7F 02 00 00 7F 03 00 00
110: 00 00 00 00 00 00 00 00
111: 64 00 64 00 00 00 00 00
```

ISO 15765 Protocol

The **ISO 15765** protocol has been designed to overcome the limit of 8 data bytes in CAN bus frames.
ISO 15765 can send messages up to 4095 data bytes in multiple CAN Raw frames.
It occupies the first data byte of each CAN frame as a control byte.
The first byte may define that the frame is a **SF** (Single Frame) which transmits only 7 data bytes.
Or a **FF** (First Frame) followed by multiple **CF** (Consecutive Frames) and **FC** (Flow Control Frames) transport a larger payload.
You can configure HUD ECU Hacker to show with the ISO 15765 data additionally the CAN Raw frames in the Trace pane:



Show Full Size

```
-------- Command -----------
(Tx) 7E0: 22 F1 95
(Tx) 7E0: 03 22 F1 95 00 00 00 00 (SF: Len= 3)

-------- Response ----------
(Rx) 7E8: 10 11 62 F1 95 53 45 30 (FF: Len= 17)
(Tx) 7E0: 30 00 00 00 00 00 00 00 (FC: Continue)
(Rx) 7E8: 21 35 50 33 53 31 56 31 (CF: Seq= 1)
(Rx) 7E8: 22 31 38 30 37 00 00 00 (CF: Seq= 2)
(Rx) 7E8: 62 F1 95 53 45 30 35 50 33 53 31 56 31 31 38 30 37
```

Above in blue you see a 3 byte ISO 15765 command sent to the Delphi MT05.3
The next line in gray shows this command encoded as a Single Frame in a CAN Raw packet.
The ECU responds with the First Frame.
Then HUD ECU Hacker sends a Flow Control Frame.
The ECU sends two Consecutive Frames.
And finally in green you see the ISO 15765 decoded ECU response.

OBD2 compliant ECU's with 11 bit ID mostly reveive commands on address **7E0** and send responses on **7E8**.
If your vehicle has more than one ECU, the second ECU may use the pair 7E1 / 7E9.
OBD2 compliant ECU's may use the range **7E0 ... 7E7** for receiving and **7E8 ... 7EF** for responding.
The scantool can send a command to the **broadcast** address 7DF where all connected ECU's must respond with their own address.
ECU's with 29 bit ID use 18DB33F1 for broadcast and 18DAF1xx for commands and 18DAxxF1 for responses.
ECU's with ISO 15765 protocol may additionally broadcast **autonomous** CAN Raw packets on other addresses.
Mostly these data packets contain MIL status, engine speed and temperature which are displayed by the **dashboard**.

J1939

J1939 is used in trucks, buses, military, marine and agricultural vehicles where many devices are connected to CAN bus.
Each device sends permamently it's data in certain intervals (e.g. every 100 ms).
For example the engine ECU sends engine RPM, the ABS ECU sends wheel speed and the torque converter sends the input shaft speed.
J1939 uses 29 bit CAN ID's which encode the source and destination of a packet, the priority and a PGN (Parameter Group Number).
The PGN defines the meaning of the data in the packet. J1939 defines more than 2000 standard PGN's. Some examples:

```
D100 : Air Suspension Control 6
DC00 : Anti-theft Status
F00B : Electronic Steering Control
F017 : Engine Knock Level #1
F032 : Linear Displacement Sensor
F07B : Power Converter 2 Limits Active Power
FC92 : Air Fuel Ratio
FD71 : Brake actuator stroke status
FD7C : Diesel Particulate Filter Control 1
FDD3 : Turbocharger Information 6
FE4E : Door Control 1
FE71 : Laser Tracer Position
FEAF : Fuel Consumption (Gaseous)
FEE1 : Retarder Configuration
FEF4 : Tire Condition Message 1
```

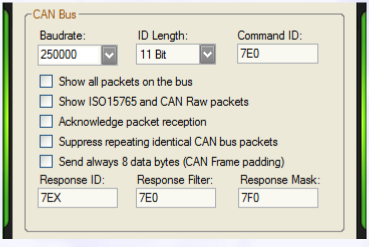
CAN Bus Filter

If you sniff the CAN bus traffic in a car or a truck you get a tremendous amount of data.
CAN bus traffic comes from ECU, ABS, air bags, seat belts, multimedia, door sensors, window- and mirror control, and more.
A filter and a mask will be needed to exclude all traffic on the CAN bus except with the ECU.
But even if you only connect one ECU which **autonomously** sends packets you may want to exlude them from the sniffed data.

The filtering also tells the adapter which packets to acknowledge (set the bit in the **ACK** slot of the CAN frame).
If HUD ECU Hacker runs in sniff mode it will never modify the ACK bit.
Otherwise, when communicating with the ECU or in Emulator mode the received packets must be ACKnowledged.
If a CAN packet is not ACK'ed by the receiver the sender assumes that it was not received and sends it again.
If a CAN packet is not ACK'ed mutiple times the sender generates an error and stops the communication.

For CAN Raw protocol you must enter `RespFilter` and `RespMask` in the parameter XML file.
For ISO 15765 these are not needed. You enter a fix `RespID` instead.

For Sniffing you can use the following window to define the filter and mask:



Baudrate: 250000
ID Length: 11 Bit
Command ID: 7E0

☐ Show all packets on the bus
☐ Show ISO15765 and CAN Raw packets
☐ Acknowledge packet reception
☐ Suppress repeating identical CAN bus packets
☐ Send always 8 data bytes (CAN Frame padding)

Response ID: 7EX
Response Filter: 7E0
Response Mask: 7F0

All checkboxes have a tooltip which explains the purpose.

In the field 'Response ID' enter an ID on which the ECU responds and the filter and mask will be calculated automatically.
Example 1.)
If you know the ECU ID enter it into the field 'Response ID'.
If you enter 7E8, the filter and mask will be calculated as 7E8 and 7FF.
Example 2.)
Otherwise use the letter 'X' to specify that a digit does not matter (e.g. 7EX). A digit corresponds to 4 bits.
If you enter 7EX, the filter and mask will be calculated as 7E0 and 7F0.
Example 3.)
If the precision of 1 digit = 4 bits = 16 matches is not enough you must enter filter and mask manually.

	Example 1	Example 2	Example 3
Field 'Response ID'	7E8 = 111 1110 1000	7EX = 111 1110 XXXX	empty
Field 'Response **Filter**'	7E8 = 111 1110 1000	7E0 = 111 1110 0000	7E8 = 111 1110 1000
Field 'Response **Mask**'	7FF = 111 1111 1111	7F0 = 111 1111 0000	7FC = 111 1111 1100
Received ID's that match the filter and mask	7E8 = 111 1110 1000 only 1 ID matches	7E0 = 111 1110 0000 7E1 = 111 1110 0001 7E2 = 111 1110 0010 7E3 = 111 1110 0011 7E4 = 111 1110 0100 7E5 = 111 1110 0101 7E6 = 111 1110 0110 7E7 = 111 1110 0111 7E8 = 111 1110 1000 7E9 = 111 1110 1001 7EA = 111 1110 1010 7EB = 111 1110 1011 7EC = 111 1110 1100 7ED = 111 1110 1101 7EE = 111 1110 1110 7EF = 111 1110 1111 16 ID's match	7E8 = 111 1110 1000 7E9 = 111 1110 1001 7EA = 111 1110 1010 7EB = 111 1110 1011 4 ID's match

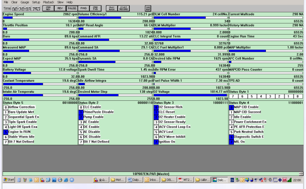
Each bit in the mask which is one defines that the same bit in the filter must match the same bit in the ID of the received CAN frame.
Each bit in the mask which is zero defines that the same bit does not matter, neither in the filter nor in the ID of the received CAN frame.

If you enter ResponseID = XXX or Mask = 000 the filter is turned off and all CAN ID's will pass through.

PCHUD & Diag Tool

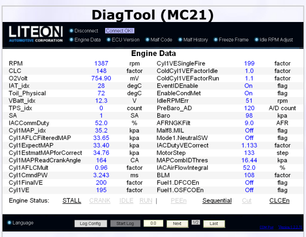
The Delphi manuals for MT05 and for MT20 explain a software **PCHUD**.
PCHUD ('Heads Up Display' for PC) is a very old program from **Delco** Electronics written in 1993 for Windows 3.
The manual for the MC21 explains a software **Diag Tool** from **LITEON** written in 2009.
Previously these were the <u>only</u> programs that could communicate with these ECU's.

PCHUD (MT05)



Show Full Size

DiagTool (MC21)



Engine Data

RPM	1387	rpm	Cyl1VESingleFire	199	factor
CLC	148	factor	ColdCyl1VEFactorIdle	1.0	factor
O2Volt	754.90	mV	ColdCyl1VEFactorRun	1.1	factor
IAT_idx	28	degC	EventIDEnable	On	flag
Toil_Physical	72	degC	EnableCondMet	On	flag
VBatt_idx	12.3	V	IdleRPMErr	51	rpm
TPS_idx	0	count	PreBaro_AD	120	A/D count
SA	1	SA	Baro	98	kpa
IACCommDuty	52.0	%	AFRNGKFilt	9.0	AFR
Cyl1MAP_idx	35.2	kpa	Malf8.MIL	Off	flag
Cyl1AFLCFilteredMAP	33.65	kpa	Mode1.NeutralSW	Off	flag
Cyl1ExpectMAP	33.40	kpa	IACDutyVECorrect	1.133	factor
Cyl1EstmatMAPforCorrect	34.76	kpa	MotorStep	133	step
Cyl1MAPReadCrankAngle	164	CA	MAPCombIDThres	16.44	kpa
Cyl1AFLCMult	0.96	factor	IACAirFlowIntegral	52.0	%
Cyl1CmndPW	3.243	ms	BLM	108	factor
Cyl1FinalVE	200	factor	Fuel1.DFCOEn	Off	flag
Cyl1VE	195	factor	Fuel1.OSFCOEn	Off	flag

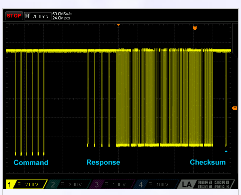
Engine Status: STALL CRANK IDLE RUN | PEEn Sequential Cut CLCEn

Show Full Size

Today it is practically impossible to find this software in internet.
I found lots of dead links and a fake PCHUD download on a chinese website which was a trojan.
But in the forum **China Riders** I found a thread from the (ex)user **'katflap'** talking about PCHUD.
Only thanks to 'katfalp' I could still in the year 2020 download and analyze this software.

The ancient 16 bit program PCHUD does not run on 64 bit Windows because Microsoft has removed the support for 16 bit applications on 64 bit platforms.
Running it on a 32 bit Windows in the 16 bit emulator (NTVDM.exe) I notice that it permanently occupies **100%** of one CPU core.

While PCHUD is displaying the data from the ECU it sends every 200 ms the same command (21 01) which the ECU responds with a data block of 100 bytes.
This 'parameter polling' looks like this:



Show Full Size

It was a lot of work to analyze which meaning has each of the 100 bytes in the response
and to find the formulas which convert the raw values into temperature, voltage and pressure.

HUD ECU Hacker

The ancient **PCHUD** from Delco is obsolete because

- it supports only one ECU model (MT05)
- it does not run on 64 bit Windows
- it occupies permanently 100% of a CPU core
- it cannot be connected over an ELM327 or J2534 adapter
- it cannot clear DTC fault codes (the menu is permanently grayed out)
- it can only display 36 parameters at the same time
- it shows the gauge for negative values wrongly
- it is clumsy to use and uses undocumented PAR, HUD, SLW, LGC, LGG, SCR, CFG and PLY files

The **Diag Tool** from LITEON is obsolete because

- it supports only one ECU model (MC21)
- it is a very sloppy software with ugly bugs
- it cannot be connected over an ELM327 or J2534 adapter
- it shows definitely wrong data for some parameters
- it can only display 36 parameters at the same time
- in the english version many translations (from Chinese) are missing

The new **HUD ECU Hacker** from ElmüSoft

- runs on Windows XP, 7, 8, 10 and 11 (not on Linux)
- runs on 32 bit and 64 bit Windows
- supports K-Line, VAG K+CAN, J2534, ELM327 (USB, Bluetooth, Wifi) and UsbCAN adapters
- can install the Windows drivers for all supported adapters
- supports multiple ECU models over K-Line and CAN bus
- automatically detects protocol, baudrate, bus init when using autodetect mode
- shows fault codes (DTC) with a text explanation
- can clear fault codes (if supported by the ECU model)
- shows all ECU parameters at once in a user-configurable dashboard (90 params for the MT05)
- shows detailed tooltips for all parameters and their meaning
- can be adapted to any KW 1281 / ISO 9141 / ISO 14230 / ISO 15765 / J1939 / CAN Raw ECU by editing 3 XML files
- the user can enter formulas to convert raw data into temperature, voltage or pressure
- can capture the parameter data from the ECU in a logfile
- can export a logfile to a CSV file
- can create graphs from a logfile
- shows the entire communication with the adapter in the Trace pane
- allows you to manually enter commands and send them to the ECU for testing
- has a built-in CAN bus Debugger / CAN bus Analyzer / CAN bus Logger
- can emulate any KW 1281 / ISO 14230 / ISO 9141 / ISO 15765 / J1939 / CAN Raw ECU
- has a built-in formula finder in the emulator
- can be automated with macros in the terminal and emulator
- can sniff the data traffic on the bus (for example from a scan tool or from a software to the ECU)
- can sniff ISO protocols, CAN Raw, Honda Keihin and KW 1281
- can extract vendor specific PAC files for MT05, MT05.2, MT05.3, SE08, MSE6.0, MSE8.0, Athena, MC10, MC21
- can decode hexadecimal S19, CAL, HEX, CUT, PTP, EFT files
- is optimized in each line of it's code for the highest possible speed
- has multi-language support. You can translate the user interface and scan parameters into any language.

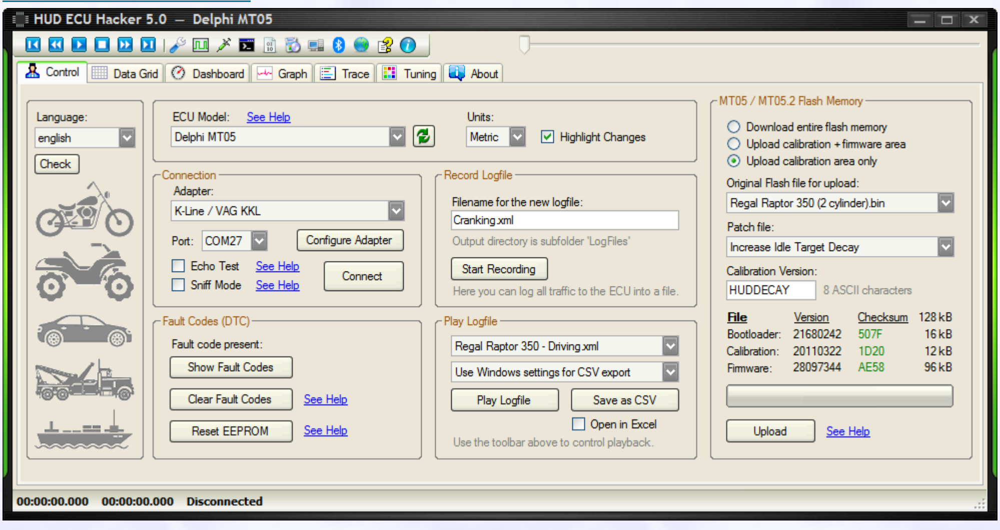
<u>MT05 specific:</u>

- has full tuning support (editing calibration maps, tables, scalars), also with 3D Editor
- can download the flash memory from the Delphi MT05
- automatically detects the addresses and types of maps, tables, scalars and DTC codes in the flash memory
- has a built-in Hex viewer which shows the binary flash memory
- can program the flash memory with the calibration tables and ECU firmware into the MT05
- the user can create Patch files which contain the changes to be applied to a flash file before uploading

**HUD ECU Hacker allows you to scan several ECU's and tune the MT05 without asking for a license fee.
And HUD ECU Hacker will never show you any advertising.
HUD ECU Hacker is the result of 4 years of full-time programming!
There is absolutely no documentation about the internals of the MT05.
Every detail you see in HUD ECU Hacker has required a very time consuming reverse engineering.
However, scanning and MT05 tuning is charityware: the author does not earn money with it.
But if this program has helped you saving money by not needing expensive commercial software
or an expensive scan tool you are asked to give a donation to a non-profit organization of your choice.
Like for example Shanti Bavan, a project which gives education for free to the poorest of the poor in India.
There is an excellent documentary about this very special residential school on Netflix: Daugthers of Destiny**

Apart from that HUD ECU Hacker has been designed to be **community software**.
Every user can adapt the program to his needs.
When you have **adapted** the XML parameter file for another ECU, you are asked to send it to me for publishing it.

HUD ECU Hacker - Control


HUD ECU Hacker 5.0 — Delphi MT05

Language: english

ECU Model: Delphi MT05

Units: Metric

☑ Highlight Changes

Connection

Adapter: K-Line / VAG KKL

Port: COM27

☐ Echo Test

☐ Sniff Mode

Record Logfile

Filename for the new logfile: Cranking.xml

Output directory is subfolder 'LogFiles'

Here you can log all traffic to the ECU into a file.

Fault Codes (DTC)

Fault code present:

Play Logfile

Regal Raptor 350 - Driving.xml

Use Windows settings for CSV export

☐ Open in Excel

Use the toolbar above to control playback.

MT05 / MT05.2 Flash Memory

○ Download entire flash memory
○ Upload calibration + firmware area
● Upload calibration area only

Original Flash file for upload: Regal Raptor 350 (2 cylinder).bin

Patch file: Increase Idle Target Decay

Calibration Version: HUDDECAY 8 ASCII characters

File	Version	Checksum	128 kB
Bootloader:	21680242	507F	16 kB
Calibration:	20110322	1D20	12 kB
Firmware:	28097344	AE58	96 kB

00:00:00.000 00:00:00.000 Disconnected

This is the main window of HDU ECU Hacker.

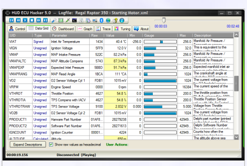
Clearing Fault Codes (DTC)

The button "Clear Fault Codes" clears the historic fault code(s) from the non-volatile ECU memory.
But if a fault is still present it will not be cleared: You click the button and nothing happens.
This button will clear only historic fault codes which are not present anymore.
The MIL / EFI lamp is only on if there is a current fault present. It is off if there are only historic DTC's.
Some current faults disapear immediately when the fault has been fixed (e.g. Oxygen sensor cable disconnected).
Other current faults will disapear alone after driving some minutes.
Some historic fault codes are erased automatically after driving 30 times without further faults.

The ECU can report multiple Current DTC's at once and it can store multiple Historic DTC's.
If there are multiple DTC's present, they are displayed alternating once a second in the Dashboard.
You see all faults at once with their status when you click the button **Show Fault Codes**.

If you get the fault codes **P0171** or **P0172** please read the chapter **Self-Learning**.

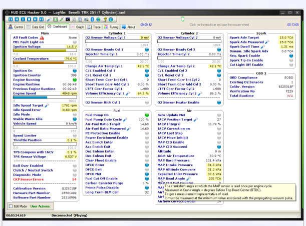
HUD ECU Hacker - Data Grid

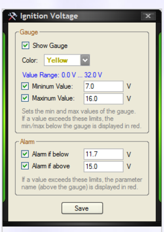

Show Full Size

This screenshot shows the playback of the logfile **Regal Raptor 350 - Starting Motor.xml**

For each parameter you see the raw value and it's meaning and the minimum and maximum values.
A gauge displays the value graphically. If the value can also be negative, the gauge starts in the middle.
Values that have changed since the previous sample have a yellow background. You can turn off this highlighting.

HUD ECU Hacker - Dashboard


Show Full Size


Ignition Voltage

Gauge

☑ Show Gauge

Color: Yellow

Value Range: 0.0 V ... 32.0 V

☑ Minimum Value: 7.0 V

☑ Maximum Value: 16.0 V

Sets the min and max values of the gauge.
If a value exceeds these limits, the min/max below the gauge is displayed in red.

Alarm

☑ Alarm if below 11.7 V

☑ Alarm if above 15.0 V

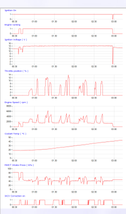
If a value exceeds these limits, the parameter name (above the gauge) is displayed in red.

Show Full Size

On the left screenshot you see a **tooltip** which appears when you hover the mouse over a parameter.
Some parameters show a **wrench** icon. 🔧 You can click on it and modify these values in the ECU. See **Data Slewing**.

The dashboard can be configured 100% by the user after checking the checkbox **Edit Mode** below.
You can create, edit and delete groups and assign parameters to them.
You can move around the groups, change the order of parameters and drag and drop them to another group.
In the dialog on the right you can configure a value parameter.
The ignition voltage has a minimum of 0 Volt and a maximum of 32 Volt.
You can restrict the range of the **gauge** to something more useful like 7 V to 16 V.
When you set an **alarm** the parameter will be displayed in red if the value exceeds the given limits.

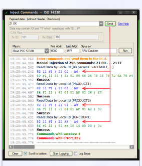
HUD ECU Hacker - Graph


Show Full Size

Show Full Size

These images are graphs created from the logfile **Regal Raptor 350 - Driving.xml**
You can chose the parameters that you want to include.

HUD ECU Hacker - Manual Command Injection


Show Full Size

HUD ECU Hacker allows to send commands manually to the ECU and study the response.
For the purpose of hacking you can also enter **XX**, **YY**, which will be replaced with all values from 00 to FF.
In the example above entering '21 XX' has sent 256 commands from '21 00' to '21 FF' to the ECU.
Here the ECU (a Delphi MT05) has only answered 4 of the 256 commands, for the others it has returned an error.
Entering '22 XX YY' will send 65536 commands from '22 00 00' to '22 FF FF' to the ECU.

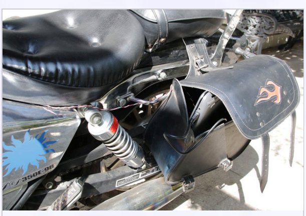
For the **CAN Raw** protocol you must additionally specify the Tx CAN ID for sending the command and the Rx ID for receiving the response.
Some ECU's send multiple responses on multiple CAN ID's to one command. In this case enter all Rx ID's separated by commas.

Macros

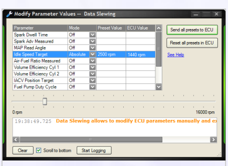
The Injector window supports macros that send commands to the ECU and process complex responses.
For more details see **Macros**

Recording Logfiles

Below in the Trace pane with the button **Start Logging** you can create logfiles after connecting to the ECU.
You can also record a log file while you are driving.
Connect the cables and put a notebook into a saddlebag or backpack.


Show Full Size

HUD ECU Hacker - Data Slewing


Modify Parameter Values -- Data Slewing

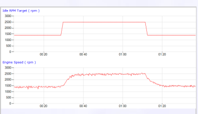
Parameter	Mode	Preset Value	ECU Value
Spark Dwell Time	Off		
Spark Adv Measured	Off		
MAP Read Angle	Off		
Idle Speed Target	Absolute	2500 rpm	1440 rpm
Air-Fuel Ratio Measured	Off		
Volume Efficiency Cyl 1	Off		
Volume Efficiency Cyl 2	Off		
IACV Position Target	Off		
Fuel Pump Duty Cycle	Off		

Show Full Size

The MT05 allows to manually modify some of the parameter values which have been measured or calculated.
The purpose of data slewing is to analyze an engine which is not running correctly.
You can set absolute (fix) preset values or you can add a delta (± offset) to the current ECU values.
First set all the preset values that you want to change in the list then click 'Send all presets to ECU'.
These changes have effect on the running motor.

Idle Speed

When the motor is runing idle and you set **Idle RPM Target** to 2500 rpm you will hear how it slowly becomes faster.


Show Full Size

This graph shows the logfile **Regal Raptor 350 - Data Slewing.xml** where the engine was running idle with 1400 rpm.
At 00:00:29.806 I have set the slew parameter **Idle RPM Target** to 2500 rpm. The ECU slowly adapted the idle speed.
At 00:01:12.480 I have clicked the button **Reset all presets in ECU**.

NOTE: On a Benelli TRK251 (1 cylinder) you can set the idle speed target but the engine speed is not adjusted correctly.

Fuel Pump

When the engine is off and you set **Fuel Pump Duty Cycle** to 15% you will hear the fuel pump running quietly.

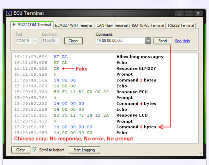
IACV

You can control the **Idle Air Control Valve** with the slew parameter **IACV Target Step**.
If you have problems with the idle speed read appendix **IACV Calibration**.

The modified slew values are not stored in the non-volatile memory of the ECU.
However this feature is for experts only. Wrong values can produce knocking or stall the motor.
I saw that the ECU does not go to sleep mode after changing some of the values.
Do not forget to click 'Reset all presets in ECU' when you are finished with your testing.

ATTENTION:
Data Slewing does **not** work with my chinese **ELM327** adapters. But J2534 and K-Line adapters do work.
The **ELM327 Datasheet** says (page 31) that the ELM327 limits the bytes that can be sent to the maximum for OBD2.
Therefore HUD ECU Hacker sends the command **ATAL** which allows longer commands.
My chinese adapter answers ATAL with 'OK', but it still refuses to send more than 4 data bytes.
You will see a timeout error in HUD ECU Hacker.

HUD ECU Hacker - ELM327 Terminal

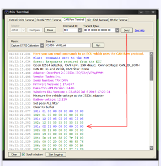

Show Full Size

As there are so many problems with chinese ELM327 fake clones I implemented the ELM327 Terminal.
Here you can test your adapter by sending commands and studying the responses.
The screenshot shows that my ELM327 clone sends commands only up to 4 data bytes.
If I send **5 data bytes** or more (like the Slewing commands) there is no response, no error and no prompt.
I verified on the oscilloscope that the adapter indeed does **not send anything**.
The command ATAL is simply ignored although it was answered with a fake 'OK'.
It is a **fraud** to sell this crap.
By the way: It is completely irrelevant if a chinese adapter claims to be version 1.5 or 2.1. They are all crap.
And I saw people complaining in internet about ELM327 adapters which have even less functionality than mine.

CAN Bus Analyzer

HUD ECU Hacker can also be used as CAN Bus Analyzer / CAN Bus Debugger / CAN Bus Logger / CAN Bus Terminal.
In **Sniff Mode** you can see the entire traffic on the CAN Bus (not only to the ECU) in real time.
You can set **filters** to show only the packets which you are interested in.

And the **CAN Raw / ISO 15765 Terminal** allows to send commands manually to the CAN bus:


Show Full Size

Normally you must buy expensive proprietary adapters for expensive CAN bus analyzer software.
HUD ECU Hacker allows to use a cheap chinese J2534 clone.

Macros

The CAN bus Terminals (CAN Raw and ISO 15765) support macros that communicate with the ECU.
For more details see **Macros**

HUD ECU Hacker - RS232 Terminal

In the RS232 Terminal you can manually send TX data to a **COM port** and see the received RX data.
You can see all RS232 events like CTS, DSR, CD, Ring, Break and set/reset DTR and RTS.

Macros

Macros are implemented for very advanced users who have knowledge of programming.
HUD ECU Hacker allows to write macro scripts in the C# language to automate complex tasks.
They will be compiled into native assembler code, so execution is lightning fast.

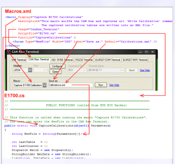
Macro scripts are supported in:

1. the **Sniff Terminal** window
2. the **CAN Raw Terminal** window
3. the **ISO 15765 Terminal** window
4. the **RS232 Terminal** window
5. the **Manual Injection** window
6. the **ECU Emulator** window

In case of a macro for the Emulator you simply declare it in the corresponding Emulator XML file:
`<Xml Version="1" EcuModel="Delphi MT05.2" MacroFile="Flashing.cs" >`

For all other types of macros you must declare it in the file **Macros.xml** in a subfolder under 'Macros'.
A macro script must contain at least one public static function that is called when you run the macro.
Each macro function may take any amount of parameters in which user values are passed to the script.
For each script parameter a control (ComboBox, TextBox, CheckBox,...) will be created where the user must enter a value.

The following example shows how a CAN bus communication can be automated with a macro script.
In the XML file you declare the display name, description and the parameters that the user must enter.



Show Full Size

There are different types of macros which support different functions that you can call in your code:

Macro Function	Sniff Terminal	CAN Raw Terminal	ISO 15765 Terminal	RS232 Terminal	Injection Window	ECU Emulator
`void SendPacket(Packet TxPack)` Sends a CAN Raw packet.		YES			Only for CAN Raw	
`Packet ReceivePacket(int Timeout, bool Throw)` Receives a CAN Raw packet.		YES			Only for CAN Raw	
`byte[] SendCommand(int Timeout, params byte[] Payload)` Sends a command to the ECU and receives the response.			YES		All except CAN Raw	
`byte[] SendBusInit(byte[] InitCommand)` Sends a K-Line bus init and then the InitCommand.					Only K-Line	
`void Disconnect()` Disconnects (stop scanning) when the macro has finished.					YES	
`void OpenCanRaw(int Baudrate, bool b29bit, int CmdID, int Filter, int Mask, bool Tx8Bytes)` Opens a new CAN Raw connection.		YES				
`void IdlePolling(int MinInterval)` Polls the parameters in the Parameter XML file. (Refresh Dashboard)					YES	
`string GetParameter(string UID)` Gets the value of a scan parameter as displayed in the dashboard.					YES	
`void RefreshParams(params string[] UIDs)` Refreshes ReadOnce scan parameter values in the dashboard.					YES	
`void WritePort(params byte[] Data)` Writes bytes to the RS232 port.				YES		
`byte[] ReadPort(int ByteCount, int Timeout, bool Throw)` Reads bytes from the RS232 port.				YES		
`void ClearPortRx()` Clears the receive buffer of the RS232 port.				YES		
`Packet[] OnPacketReceived(Packet RxPack)` Generates the ECU reponse(s) for an ECU command that is not defined in the Emulator XML file.						YES
`Packet[] GetEmuXmlResponse(string CmdID)` Returns the response(s) for CmdID from the Emulator XML file.						YES
`void OnSniffData(Packet RxPack)` Is called when a packet has been sniffed.	YES					
`void ChangeKLineBaudrate(int Baudrate)` Changes the baudrate for K-Line.	YES				YES	YES
`bool UserAborted()` Returns true when the user wants to abort the maro.	YES	YES	YES	YES	YES	
`void PrintTrace(string Text)` Prints a text to the Trace pane.	YES	YES	YES	YES	YES	YES
`void Sleep(int Interval)` Pauses the marco with a precision of 1 millisecond. If you don't need this precision use `Thread.Sleep()` instead.		YES	YES	YES	YES	
`DialogResult MessageBox(String Message, MessageBoxButtons Buttons, MessageBoxIcon Icon)` Shows a messagebox that blocks the macro until a button is clicked.		YES	YES	YES	YES	
Macro Function	**Sniff Terminal**	**CAN Raw Terminal**	**ISO 15765 Terminal**	**RS232 Terminal**	**Injection Window**	**ECU Emulator**

Apart from these functions HUD ECU Hacker extends the functionality for `byte[]` arrays.

- `string Hex = ByteArray.ToHex(int First, int Count)`
  Converts some or all bytes from ByteArray into a hex string "5C A4 CC 9F".
  The parameters First and Count are optional.
- `string Text = ByteArray.ToAscii(int First, int Count)`
  Converts some or all bytes from ByteArray into an ASCII text. Invalid bytes are displayed as '?'.
  The parameters First and Count are optional.
- `byte[] New = ByteArray.Extract(int First, int Count)`
  Extracts some bytes from ByteArray into a new array.
  The parameter Count is optional.
- `byte[] New = ByteArray.Append(byte[] Data, int First, int Count)`
  Appends some or all bytes from the array Data to the bytes in ByteArray.
  The parameters First and Count are optional.
- `byte[] New = ByteArray.ReplaceAt(int Pos, byte[] Replace)`
  Replaces the bytes at position Pos in ByteArray with the bytes in Replace.
- `bool ByteArray.Matches(params byte[] Data)`
  Returns true if all bytes in both arrays are identical.
- `bool ByteArray.StartsWith(params byte[] Data)`
  Returns true if the array starts with the bytes in Data.
- `int ByteArray.Find(int Start, params byte[] Pattern)`
  Returns the position in the array where Pattern was found or -1 if no match.
- `int ByteArray.DiffBytes(params byte[] Data)`
  Returns the count of bytes that are different in both arrays. Zero means they are identical.
- `void ByteArray.Fill(byte Value)`
  Fills the entire array with the given byte value.

```
Example:
byte[] MyTest = new byte[] { 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77 };
string StrHex = MyTest.ToHex(2, 3); // here StrHex is "33 44 55"
```

Please study the existing macro scripts which are full of explaining comments:

- **Macros\Honda Sniff\HondaSniff.cs**
  This script for the Sniff Terminal switches the baudrate when a firmware is uploaded to a Honda ECU.
- **Macros\Deni E1700\E1700.cs**
  This script for the CAN Raw Terminal captures the CAN bus traffic of a DENI software while uploading calibration tables over a UsbCAN adpater to the E1700 ECU and writes the calibration tables into an XML file.
- **ECU\MT05\Emulator\Flashing.cs**
  This script for the Emulator automates the upload and download of flash data for the Delphi MT05 / MT05.2.

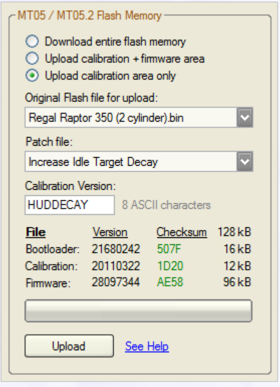
Download / Upload Flash Memory (MT05 / MT05.2 only)

The heart of the the Delphi MT05 is a 16 bit Infineon processor.
The flash memory in the processor is divided into 4 areas:

1. The **Bootloader** is required to start up the ECU.
   It will never be overwritten when flashing. This is a protected area.
2. The **Configuration Data** will always change when you turn off the ignition key.
   The ECU stores non-volatile data here when you turn the ignition key off, like:
   fault codes, ignition counter, statistics, fuel learning (**BLM**), airflow learning and throttle learning.
   HUD ECU Hacker does not write into this area, but you can erase the content of this erea. See **Reset EEPROM**.
3. The **Calibration Tables** are used to calculate the optimal operation of the motor depending on
   factors like speed, engine load and temperature, etc. They control fuel injection, spark timing, etc.
4. The **Firmware** area contains the executable program code.
   You should normally not overwrite this area except you know exactly what you are doing.

Processor	Delphi MT05		Delphi MT05.2	
Model	**SAK-XC164CM-16F40F**		**SAK-XC164CS-32F40BB**	
Flash Memory	128 kB		256 kB	
RAM	8 kB		12 kB	
Clock	32 MHz		32 MHz	
Flash Memory	**Delphi MT05**		**Delphi MT05.2**	
Bootloader	000000 - 003FFF	16 kB	000000 - 003FFF	16 kB
Configuration Data	004000 - 004FFF	4 kB	004000 - 004FFF	4 kB
Calibration Tables	005000 - 007FFF	12 kB	005000 - 00AFFF	24 kB
Firmware	008000 - 01FFFF	96 kB	00B000 - 03FFFF	212 kB

HUD ECU Hacker can download the flash memory into a file (flash download).
HUD ECU Hacker can also program the flash memory from a file (flash upload).
In the main window you see the versions and the checksums of the flash memory areas.
Green means the checksum is correct. **Red** means it is wrong and will be fixed when uploading.


MT05 / MT05.2 Flash Memory

○ Download entire flash memory
○ Upload calibration + firmware area
● Upload calibration area only

Original Flash file for upload: Regal Raptor 350 (2 cylinder).bin

Patch file: Increase Idle Target Decay

Calibration Version: HUDDECAY 8 ASCII characters

File	Version	Checksum	128 kB
Bootloader:	21680242	507F	16 kB
Calibration:	20110322	1D20	12 kB
Firmware:	28097344	AE58	96 kB

ATTENTION:
If you use a **K-Line** adapter execute the **Echo Test** to assure that it works correctly.
If you use an **ELM327** adapter it must be a genuine OBDLink adapter.

Before flashing for the first time store your original flash file in a secure place!
If flashing of only the calibration tables goes wrong your ECU may still communicate over K-Line.
But if flashing the firmware area goes wrong your ECU will probably be bricked.
See Rescue a bricked MT05.

Tuning (MT05 / MT05.2 only)

Tuning with Commercial Software

Tuning means to modify the calibration tables to get more power, cleaner emissions, or better fuel efficiency.
For tuning you normally have to purchase 2 expensive programs:

1.)
One program which only downloads and uploads the flash memory:
BitBox (250€, english) or **CombiLoader** (base: 21500 rouble + MT05: 8000 rouble) Also from **EcuTools**

2.)
Another program for editing the calibration tables:
BitEdit (100€, english) or **ChipTuningPro** (base: 2400 rouble + MT05: 10000 rouble) Also from **EcuTools**

Additionally you have to buy an USB dongle (30€) which protects their software from piracy.
BitEdit does not support all MT05 versions. Click **here** for a list of supported calibration versions.

While BitEdit shows 36 tables for the MT05 (in english), ChipTuningPro shows more than 200 tables.
BitEdit is compared with ChipTuningPro like a toy. It has bugs and shows some data and axis wrongly.

3.)
Other tuning software like **ECM Titanium** is even more expensive (> $1000 USD).

4.)
If you have a MT05 from Kohler or from Briggs & Stratton they sell you diagnostic software that is very restricted.
You are forced to buy their proprietary adapter (> $300 USD) which acts like a dongle.
Each time you start the software, it connects with their server and checks if you have a valid license.
The license is only valid for one year.

As you see clearly: All the tuning and even ECU scanning is a very **profitable business**. All these companies want your money.

IMPORTANT: A time consuming analysis must be done to get the correct meaning of scalars, tables, maps and their axes.
The companies which sell expensive tuning software do not invest the required time to do this tremendous work.
Doing a real analysis of each and every firmware version of each and every ECU model would result in a price that nobody is willing to pay.
So they enter much of the data by **guessing** and by copying it from other firmware versions, which results in a lot of wrong information.

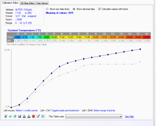
Tuning MT05 with HUD ECU Hacker

Download the flash memory of your ECU into a BIN file, go to the tab "Tuning", select your file and HUD ECU Hacker will auto-detect all calibrations.

With the charityware HUD ECU Hacker you save a lot of money by not having to buy commercial software.
Flash download, checksum correction and flash upload of the MT05 can also be done with HUD ECU Hacker.
Please do not forget to give a donation for using HUD ECU Hacker.

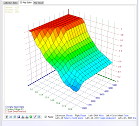
An issue with the MT05 is that each ECU **firmware version** stores the calibration tables at another address in the flash memory.
This means that there is no way to know how many tables exist and where each table starts and where it ends, what the binary data
in the table means, what are the values, meaning and units of the axes and which formula converts the raw table values into display values.
But HUD ECU Hacker analyzes the ECU assembler code and **finds automatically** 200 calibration tables and 500 scalar values.
It even finds the values and meaning of the **axes** of nearly all tables and maps by auto detection.
This works with all firmware versions and takes less than one second. The result is 100% reliable.

Calibration Editor



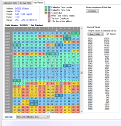
Show Full Size

3D Editor



Show Full Size

Hex Viewer



Show Full Size

Whenever you load a new flash memory BIN file it will be **automatically analyzed** and the result is written into a file.
This file has a name like "Firmware_5D06BA79.definitions" and contains the maps, tables, scalars and axes that were found.
The hex number in the filename "5D06BA79" is a CRC (similar to a checksum) of the firmware area in the flash memory.
Each firmware version will generate it's own definition file because each firmware version stores the calibrations at different addresses.
HUD ECU Hacker also detects the count of rows and columns and if the table data is 8 bit or 16 bit and if the data is signed or unsigned.

The **auto-detection** analyzes the assembler code in the ECU which gives reliable results independent of the firmware version.
The assembler code in the ECU is eternally long (printed on paper it would be one kilometer).
However HUD ECU Hacker does this analysis in less than a second. Here you see a snippet of the huge Delphi code:

```
c1768e f2 fc f8 f7      mov        r12,[0xF7F8]
c17692 f2 fd fc f7      mov        r13,[0xF7FC]
c17696 d7 40 01 03      extp       #0x301, #1
c1769a f2 fe 22 29      mov        r14,[0x2922]
c1769e da c1 f8 14      calls      FUN_c114f8
c176a2 f0 c4            mov        r12,r4
c176a4 f6 fc f8 f7      mov        [0xF7F8],r12
c176a8 f2 fd fc f7      mov        r13,[0xF7FC]
c176ac 42 fd f8 f7      cmp        r13,[0xF7F8]
c176b0 fd 08            jmpr       cc_ULE,LAB_c176c2
c176b2 22 fd f8 f7      sub        r13,[0xF7F8]
c176b6 f6 fd f6 f7      mov        [0xF7F6],r13
c176ba e1 12            movb       RL1,#0x1
c176bc f7 f2 e6 f7      movb       [0xF7E6],RL1
c176c0 0d 08            jmpr       cc_UC,LAB_c176d2
```

As you see, assembler code is extremely cryptic. I spend several weeks in understanding what the ECU does internally.
I hope that you honor this tremendous work and be so honest to respect the **charityware** policy of HUD ECU Hacker.

Hex Viewer

The **Hex Viewer** shows how scalars, tables and maps are lined up in the calibration area of the flash memory.
Reverse Lookup tables appear with **bold** text.
Axis values are calculated by a formula. The parameters for the formula are not stored in the calibration area.

You will see mostly two **white areas** of approx 30 bytes at the beginning of the calibration area.
All white areas contain tables which are never used by the firmware or they are filled with zeroes.
These tables are **orphans**. They contain data, that may be used in other firmware versions.

There are other tables which don't have a header, so the length and type of data cannot be auto-detected.
They appear **purple** in the Hex Viewer.

The Hex Viewer can also **compare two BIN files** and show the differences.

Delphi Calibration Data

Please read this chapter in the help file.

Completing Auto-Detection Results

Please read this chapter in the help file.

Editing Calibrations

Please read this chapter in the help file.

Working with Patches

Please read this chapter in the help file.

Adapting to other ECU's

HUD ECU Hacker can be adapted to any ECU which uses the KW 1281 / ISO 9141 / ISO 14230 / CAN Raw / ISO 15765 protocol.
This is a process in 5 steps.
You need the ECU and a scantool or software **from the vendor** which understands the **vendor specific** ECU data.
The traffic between ECU and scantool must be captured and reverse engineered. Doing this is **not illegal**.

IMPORTANT: A **Universal OBD2 Scantool** which only shows OBD2 data is useless. HUD ECU Hacker can already display OBD2 data.
The OBD2 standard has been designed to verify that a vehicle complies the emission laws.
OBD2 gives very limited information because the manufacturers implement only few commands, just the minimum to fulfil the law.
OBD2 may only show you Vehicle Speed, Engine Speed, Coolant Temperature, O2 Sensor and Throttle Position, and that's it.
Generally ECU's can report much more details to the service technician, but in a proprietary and secret data format of the manufacturer.
Only an expensive scantool or software from the ECU vendor may give you this information, but **not** a "universal" OBD2 scantool for all vehicles.

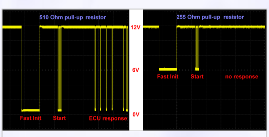
For example for the Delphi MT05 the vendor specific command **30** allows data slewing.
And the vendor specific command **21** returns details like crankshaft errors, stepper motor position, block learning (BLM) and much more.
These details (90 scan parameters) can **not** be obtained with a "universal" OBD2 scantool or OBD2 computer software.
The response of command 21 01 is a proprietary and undocumented data packet from Delphi.
I obtained the meaning of this 100 byte packet by analyzing the ancient PCHUD software.
You can do the same for any other ECU if you have a scantool or software which shows these details.

Step 1. Sniff Data

When you enable the checkbox 'Sniff Mode' you can capture the traffic between the ECU and a scantool / OBD software.

Connecting an additional sniff adapter over a splitter cable to the K-Line will mostly not work.
The reason is that each adapter has a pull-up resistor (mostly 510 Ω) between K-Line and +12V.
When you connect 2 adapters the parallel pull-up resistor becomes 255 Ω.
Adapters and ECU have a **current limitation** to protect them from shortcuts.
Most adapters don't provide enough current (50 mA) to pull K-Line to ground over 255 Ω.
Depending on the pull-up resistor and the current limitation you will not capture anything or the scantool stops working.
Here you see the result of connecting two adapters at the same time. The voltage does not reach 0V anymore.

Show Full Size

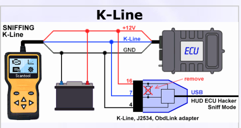


Show Full Size

It may also happen that you can sniff data as long as the motor is off.
But when the motor runs the battery voltage rises to 15V and now you don't capture data anymore or get crippled data.
The higher the battery voltage the more current is required to pull K-Line to ground.
However, there are also adapters with an internal pull up resistor of 1 kΩ. They may function unchanged.

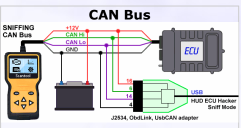
The only bullet-proof solution is to modify an adapter and **remove** the SMD pull-up resistor between pin 7 and 16.
You can either convert an adapter into a sniff adapter by removing this resistor completely
or you can insert a **switch** into the adapter which allows to chose between normal mode and sniff mode.

K-Line



Show Full Size

CAN Bus



Show Full Size

No modification in the adapter is required for CAN bus sniffing.
The J2534 and OBDLink adapters require 4 pins to be connected.
If you use the **UsbCAN** adapter only 2 pins must be connected: CAN0H and CAN0L.

Store the sniffed data into a **logfile** by clicking the button 'Start Logging' in the Trace pane.
Navigate through all menus of the scantool to capture all commands.
IMPORTANT: If the logfile has many "Invalid Data" you have the wrong baudrate or the wrong protocol.

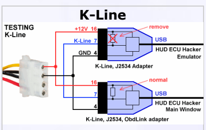
Step 2. Test the ECU Emulator

Connect HUD ECU Hacker to it's own Emulator to learn how to use it.
You can use a battery or the 12 Volt from a cheap computer power supply (yellow wire).

ATTENTION:
ELM327 / OBDLink adapters do not have the functionality required for the emulator.
J2534 adapters will not work with any protocol that uses the 5-baud initialization.
Use a **K-Line** adapter for K-Line Emulation.

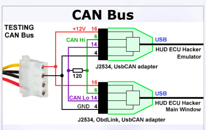
For CAN Bus an additional 100 Ω or 120 Ω resistor between CAN Hi and CAN Lo is **indispensable** because adapters do not have it built-in.

K-Line



Show Full Size

CAN Bus



Show Full Size

To test K-Line select ECU model "Delphi MT05.2" in the emulator and in the main window.
To test CAN bus select ECU model "Autodetect ...." in the emulator and with button 'Configure' switch to ISO 15765.
Then in the Emulator window click "Open", then in the main window click "Connect".
Now you should see the data coming from the emulated ECU in the dashboard.
Change the values of command 21 01 or 22 21 01 in the emulator and **study** their effect on the display in the dashboard.

Step 3. Simulate the ECU

Please read this chapter in the help file.

Step 4. Enter the XML Commands and Parameters

Please read this chapter in the help file.

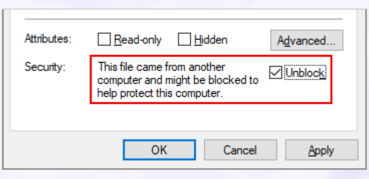
Step 5. Enter the XML Names and Descriptions

Please read this chapter in the help file.

Download and Installation

	Program	Version	Date	Size
Download	**HUD ECU Hacker**	5.3.1	20.sep.2024	21 MB

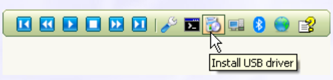
Windows may block the installer in the downloaded ZIP file because it has no digital certificate.
Please right click the ZIP file, select 'Properties' and check 'Unblock'.



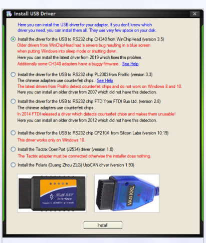
Attributes: ☐ Read-only ☐ Hidden Advanced...
Security: This file came from another computer and might be blocked to help protect this computer. ☑ Unblock

You need the **.NET framework 4.0** or higher. On Windows 10 and 11 this is already installed.

Drivers



In the **toolbar** at the top you can then install the **drivers**.
Click the toolbar button **Device Manager** to see if there are yellow exclamation marks which indicate a missing driver.
The toolbar has a **tooltip** for each button which appears when you hover the mouse over it.



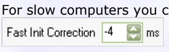
Show Full Size

Trouble Shooting

If your ECU is not listed under "ECU Model" connect with "Autodetect OBD2" which tries multiple protocols, init modes and ECU addresses.
All newer ECUs will respond to OBD2 commands.

Errors when connecting to the ECU:

- The **ignition key** must be on.
- The **kill switch** must be in the position where it allows the motor to run.
- Some motorbikes (Benelli) require the **side stand** to be up otherwise the ECU will not respond.
- It is **not** necessary to start the motor to establish a connection.
- Check that you have connected the **three** wires correctly as shown in the **connector diagram**.
- If you have a MT05 ECU verify that the **seven** voltages are correct that are marked red in the **MT05 diagram**.
- The voltage at the K-Line wire MUST be **+12 Volt** while the adapter is connected to the ECU.
  Some adapters do not enable the pull up resistor when they are in power safe mode.
  Measure the voltage while clicking the "Connect" button in HUD ECU Hacker which will activate the adapter.
- If you use a K-Line or J2534 adapter ecxecute the **Echo Test** to check the adapter.
- There are 2 types of timeout errors which indicate different errors:
  - **Timeout waiting for echo** means always that you have a hardware problem or the wrong COM port.
  - **Timeout waiting for response** (or received garbage characters) with **ELM327** adapter may mean that the baudrate is wrong.
    You can change the baudrate in the window "Configure Adapter". Normally ELM327 adapters use 38400 baud or 115200 baud.
  - **Timeout waiting for response** with **K-Line / VAG** adapter may happen rarely.
    The reason is that the ISO 14230 protocol is very time critical. It demands 50 ±1 ms for the fast init.
    But Windows as a multitasking OS is not very precise and the interval seen on an oscilloscope may vary from 45 ms up to 70 ms.
    If the interval between fast init and the command 'Start Communication' exceeds the limits the ECU does not respond.
    K-Line adapters are the only adapters where timing depends on the computer. J2534 and ELM327 adapters create a precise timing.
    If you get this type of timeout error, try the following:
    1. Click 'Connect' **several times** until it works. It may work 8 of 10 times.
    2. Some adapters (e.g. some SiliconLabs chips) do not support the way how HUD ECU normally generates the fast init pulse.
       Try switching **Fast Init Mode 1** / **Fast Init Mode 2** in the window "Configure Adapter".
    3. For slow computers you can enter in the same window a K-Line **timing correction** which is added to the 50 ms interval:

       Fast Init Correction -4 ms

       ATTENTION: If you enter an invalid value here you may screw up the fast initialization forever.
       If changing this value did not solve your problem, **reset the correction to zero** otherwise you may never be able to connect.
       To verify the timing you need a digital oscilloscope, otherwise it is pure try and error.
- **BUSINIT: ERROR** from an ELM327 adapter means that the adapter did not receive a valid response from the ECU.
- You can also change the configuration in "Autodetect OBD2.xml". Some older ECU's use 9600 baud.
- If you have tried everything and the ECU still does not respond, test your adapter: Connect HUD ECU Hacker to it's own emulator.
  Therefore you need a second adapter. See **Emulator**

If you have any problem you can send me a **Trace logfile** with the error message.
Do **not** send me screenshots from the Trace pane or even screen photos!
You find my **email** below at the end of the page.

Appendix

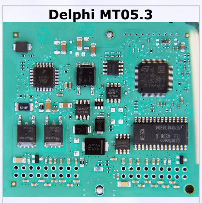
Delphi MT05.3

The new Delphi ECU is the MT05.3 which is Euro 5 compliant and uses CAN bus.
This ECU is a complely new development, using a modern 32 bit processor: SPC 572L.
The consequence is that all my endless work for flashing the MT05 / MT05.2 is completely useless now for the MT05.3
May be some day in the future I will add support for flashing the MT05.3.
But this is the hard work of another entire year.

HiSun MT05

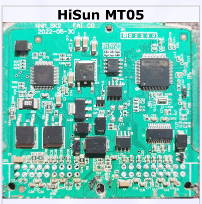
HiSun makes a proprietary ECU with another 32 bit processor: FS32K144.
But it supports the same commands for scanning as the Delphi MT05.3 over CAN bus.

Delphi MT05.3



Show Full Size

HiSun MT05



Show Full Size

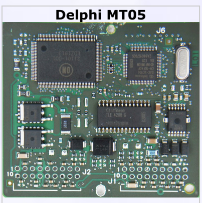
Rongmao MT05 and RM08

Rongmao uses the same case and plugs as the Delphi MT05.
It also sends the same **scan parameters**, so it can be scanned with HUD ECU Hacker.
But inside is a different board with different chips. (see photo below)
The PCB boards of the Rongmao MT05 and RM08 are identical.
Rongmao has milled away the processor label, so the processor model is hidden.
But I know that the Rongmao processor is an Infineon SAK-XC2365B-40F80LR with 320 kB flash memory.
However, flashing is not possible because Rongmao uses their own commands and even a proprietary protocol for flashing.

Harley Davidson MT05

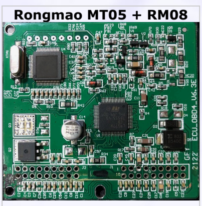
Also Harley Davidson uses an ECU which looks from the outside like the Delphi MT05.
But inside is another processor (MC9S12XEP) and obviously another firmware with different calibations.
This ECU cannot be flashed with HUD ECU Hacker because it is completely different.
It only supports CAN bus and even the ECU pins are connected differently. See **Manual**

Delphi MT05



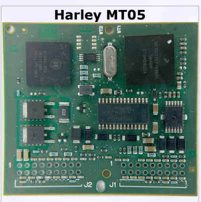
Show Full Size

Rongmao MT05 + RM08



Show Full Size

Harley MT05



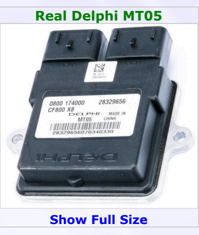
Show Full Size

Chinese Fake MT05

Chinese fake clones of the MT05 or MT05.2 have appeared in the market which are garbage.
They are full of bugs and only the very basic OBD2 commands are implemented.
These ECU's are so extremely buggy that they are not even able to send a correctly formatted DTC response!
Detailed scan data is not available, Data Slewing and flashing are not possible.
There are even fake MT05 which respond only on CAN bus instead of K-Line. (The real MT05 / MT05.2 does not respond on CAN bus)

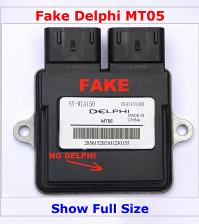
An ECU is only a Delphi **MT05** if it has the SAK-XC164CM-16 **processor** from Infineon inside.
An ECU is only a Delphi **MT05.2** if it has the SAK-XC164CS-32 processor.
An ECU with any other processor that says "Delphi MT05" on the label is a fake.

Real Delphi MT05



Show Full Size

Fake Delphi MT05



Show Full Size

On the real ECU's you see that "DELPHI" is engraved in the plastic of the cover. The fake does not have this.
HUD ECU Hacker will detect when you have connected a fake ECU and show an error message.

Chinese MT05 Clones

ATTENTION: There are also Chinese MT05 **clones**. These are not fake. They are an exact copy of the original Delphi ECU.
A clone has the **same processor** as the orignal, so it can even be flashed with HUD ECU Hacker.

Overheating Risk

A big problem of all combustion motors is overheating. If cooling fails, the motor will be damaged.
A water cooled motor will reach 80 degree Celsius, max 95 degree.
If the motor is air cooled the temperature may reach 140 degree.

1. The first damaged part will be the cylinder head gasket. As a result coolant will enter into the cylinders and vaporize.
   You will lose coolant through the exaust pipe. A vicious circle accelerating the damage.
   Replacing the gasket is expensive because the motor must be opened.
2. If you drive longer with an overheated motor the cylinders will be ruined and you need a new motor.

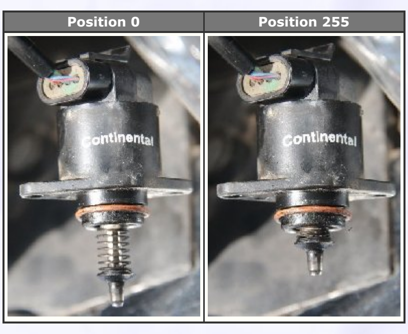
The cause of overheating is a failure in the cooling system. This may be due to a defective ventilator or lack of coolant.
Some motorbikes (like my Regal Raptor) neither have a temperature display nor an overheating lamp.
This is a severe problem because the owner has no chance to check the temperature of the motor.
ATTENTION: The Delphi MT05 is so stupidly programmed that it does **NOT** protect the motor from overheating!
Although the ECU has a temperature sensor and knows the exact temperature, it will not turn the motor off when it becomes too hot.
The MIL/EFI light may turn on when it is already too late (Error P0117 at approx 200 degree) or it never turns on.
Check regularly if your motorbike has sufficient coolant! Use HUD ECU Hacker to check the current temperature.
If coolant disappears within a few days or weeks without a leak your motor is probably already damaged.

IACV Calibration

The IACV (Idle Air Control Valve) is like a bypass for the throttle valve.
It allows a small amount of air to enter into the engine while the throttle is closed.
If the IACV does not work correctly you may have the following problems:

- The engine cannot be started
- The engine goes off alone while running idle (especially when it is cold)
- The idle speed is irregular
- The ECU may generate fault code **P0505**.

The IACV has a stepper motor which moves a pintle precisely to a position between 0 and 255.


Position 0 **Position 255**

To maintain the stepper motor in the desired position a current must flow permanently which generates a magnetic field.
Therefore the stepper motor becomes warm although it does not move.

- Position 0: The valve is fully closed. Air can only enter through the throttle into the intake manifold.
- Position 168: The valve is in the **parking** position for the next cranking. (Defined in the calibration **IAC Park Position**).
- Position 200: The maximum position that the ECU will use. (Defined in the calibration **IAC Position Max**).
- Position 255: The valve is fully open.

You can use the **Data Slewing** window to test the IACV while the engine runs.
If you enter a **delta** value of +30 steps, more air enters and you notice that the idle speed increases.
If you enter a **delta** value of -30 steps, less air enters, the idle speed decreases and the engine may stall.

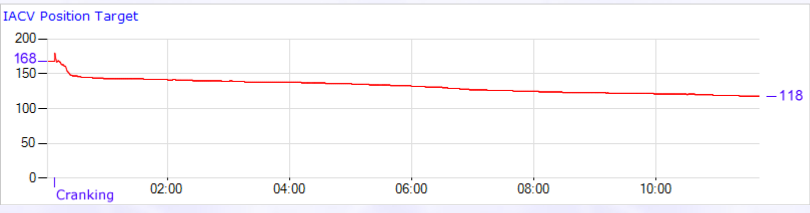
Aprox 5 seconds after turning the ignition key off the ECU parks the IACV and stores the pintle position in flash memory.
The IACV has no sensor which reports the current mechanical position to the ECU.
The ECU simply trusts that the position stored in the flash memory is the same as the real mechanical position.
But the range which the stepper motor can move is wider than the programmable range from 0 to 255.
So if the ECU loses synchronisation with the mechanical position you will have one of the problems listed above.

I found the following way to calibrate the IACV (while the engine is off) with the older **MT05** ECU.

1. Take the IACV out so you can see the pintle position.
2. Connect HUD ECU Hacker.
3. In the **Data Slewing** window move the IACV to the **absolute** position 255.
4. Now disconnect the battery so the ECU cannot store this position in the flash memory.
5. Reconnect the battery. You should hear the fuel pump running. Now the ECU assumes the IACV in parking position.
6. Repeat steps 2 to 5 until the pintle does not move anymore. The spring must be completely compressed.
7. Now you have the pintle in the **real** mechanical position 255.
8. In the Data Slewing window click **Reset all presets in ECU** which moves the pintle to the parking position.
9. Turn off the ignition key. Now the ECU stores the correct parking position in flash memory.
10. Mount the IACV back into it's place.
11. When driving the next time the airflow self-learnig will adapt to the new conditions.

The newer **MT05.2** already has the **EEPROM Reset**, which also adjusts the IACV, but in a different way.
The ICAV must be mounted in the throttle body while executing the EEPROM Reset.
The ECU will move the stepper motor until the pintle is mechanically blocked when the IACV is fully closed.
This will be the new position 0.

Play the logfile **Regal Raptor 350 - IACV Idle Warmup.xml** and create a graph with the preset **IACV and Idle**.
Here you see how the MT05 slowly adjusts the IACV during a 12 minute idle warm-up from 18 °C to 80 °C:



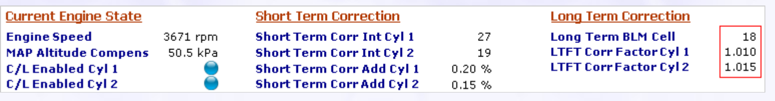
Self Learning

The ECU adapts to changing load, atmospheric pressure and fuel quality to keep the emissions at a minimum while running in closed loop.
It also compensates a worn out fuel pump or a dirty air filter.
Based on the O2 sensor feedback, the short term adaption increases (> 0) or decreases (< 0) the amount of fuel to get the optimal air/fuel mix.
If the short term adaption (Integrator) deviates too much, the long term adaption will be adjusted.
The long term fuel adjustment is stored in a table which contains Block Learn Multipliers (BLM).
Multipliers have values between 0.0 and 2.0 where values > 1.0 mean more fuel and values < 1.0 mean less fuel.
The Delphi MT05 uses a table with 36 cells (16 bit) for each cylinder which is stored in the flash memory.

MAP	TPS	< 1900	< 2800	< 3750	< 4500	< 5800	< 7200	< 9000	> 9000 rpm
< 30 kPa	< 4 %	Cell 0	Cell 1	Cell 2	Cell 3	Cell 4	Cell 5	Cell 6	Cell 7
< 46 kPa	< 10 %	Cell 8	Cell 9	Cell 10	Cell 11	Cell 12	Cell 13	Cell 14	Cell 15
< 62 kPa	< 19 %	Cell 16	Cell 17	Cell 18	Cell 19	Cell 20	Cell 21	Cell 22	Cell 23
> 62 kPa	> 19 %	Cell 24	Cell 25	Cell 26	Cell 27	Cell 28	Cell 29	Cell 30	Cell 31

Rolling Idle Cells			
Cell 32	Cell 33	Cell 34	Cell 35

The X and Y axis values come from the lookup tables 'BLM MAP Boundary', 'BLM TPS Boundary', and 'BLM RPM Boundary'.
The Y axis may be based on MAP compensated pressure or throttle position. This is defined by scalar 'BLM Load Option'.
The rolling idle cells are used when the engine is idling.
While the engine is running you see in the dashboard which cell the ECU is currently using and what is the value of this cell:



Current Engine State		Short Term Correction		Long Term Correction	
Engine Speed	3671 rpm	Short Term Corr Int Cyl 1	27	Long Term BLM Cell	18
MAP Altitude Compens	50.5 kPa	Short Term Corr Int Cyl 2	19	LTFT Corr Factor Cyl 1	1.010
C/L Enabled Cyl 1	●	Short Term Corr Add Cyl 1	0.20 %	LTFT Corr Factor Cyl 2	1.015
C/L Enabled Cyl 2	●	Short Term Corr Add Cyl 2	0.15 %		

This capture is from logfile 'Regal Raptor 350 - Driving.xml' at 01:16:246
A long term correction factor of 1.015 means adding 1.5% more fuel.

The ECU also learns automatically which voltage of the throttle sensor corresponds to 0% throttle position (TPS auto-zero).

Reset EEPROM (NVRAM)

If some BLM cells reach the **minimum or maximum adaption limit**, the fault codes **P0171** or **P0172** will be generated.
These errors mean that there is a defect (IACV, injector, fuel pump, air filter) which the ECU cannot correct anymore.
If you get these errors you must reset the self learning data in the **configuration area** with the button **Reset EEPROM** which will:

1. Erase the fuel self-learning data (BLM Table). All correction factors will be reset to 1.0 (no correction).
2. Erase the airflow self-learning data (Airflow Table). All correction factors will be reset to 1.0 (no correction).
3. Erase the throttle self-learning data (Auto-Zero). The throttle zero will be set to the default in the scalar 'TPS Raw Intercept'.
4. Erase all statistics. This will reset all time counters (except total runtime), max temperature, max batt voltage, max speed,...
5. Erase all historic DTC fault codes
6. Reset the IACV pintle position

ATTENTION: If you do not repair the underlying hardware defect the errors P0171 / P0172 will come back soon.

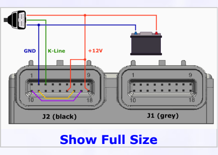
You need the EEPROM reset also after replacing the **throttle sensor**.
After erasing the configuration data the ECU will write fresh data the next time you turn the ignition key off.

Some older MT05 firmware versions do not implement the command which HUD ECU Hacker uses when you click the button **Reset EEPROM**.
In this case you may have luck trying one of the following two options:

1. Turn the ignition key off while pin 5 of the **ECM plug** is connected to ground (pin 2), then wait 10 seconds.
   This will only work if the scalar 'J1-16 Input Usage' is 1.
2. Or turn the ignition key off, wait 10 seconds, turn the key 5 times on/off within 5 seconds, then (while off) wait another 10 seconds.
   This will only work if the scalar 'J1-16 Input Usage' is not 1.

Rescue a bricked MT05

If you have uploaded a wrong flash file or interrupted the upload you may have 'bricked' your ECU.
A 'bricked' ECU will neither allow to start the motor nor will it respond on K-Line.
'Bricked' means that your ECU is now as useful as a brick. Congratulations!
In this case you have to switch the ECU into 'bootloader mode' and then you can connect again.
Unplug the plugs J1 and J2 and connect the ECU only to the battery and the K-Line or J2534 or ELM327 adapter:



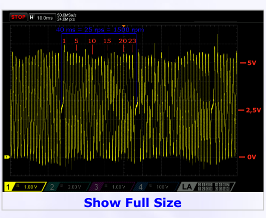
Show Full Size

You need one jumper between pins 10 and 17 and another jumper between pins 11 and 16.
This switches the ECU into 'bootloader mode' and allows to upload a valid flash file.
Additionally you connect +12V to pins 15 and 18, Ground to pin 2 and K-Line to pin 3.

The first 16 kB of the flash memory contain the bootloader.
This memory area will never be overwritten when you upload a flash file.
This assures that the bootloader stays always intact even when flashing goes wrong.

Crankshaft Position Sensor

The crankshaft position sensor reports the exact position of the crankshaft to the ECU.
The ECU needs this to calculate the moment of spark generation and of measuring the Intake Air Pressure sensor.
On the crankshaft there is a flywheel with teeth every 15 degree. Each tooth induces a pulse in a fixed pick up coil.
There are 24 positions on the 360 degree rotation. One of them is missing, so there are 23 pulses per rotation.
The gap from the missing tooth indicates the position near BDC (Bottom Dead Center) of cylinder 1.
Example: The motor runs with 1500 rpm. This is 1500 / 60 = 25 rotations per second = 40 ms per rotation.
This oscilloscope capture measured at ECU pin J2-04 shows the 25 * 23 = 575 pulses/second.

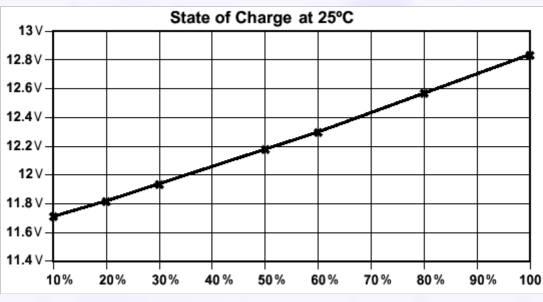


Show Full Size

The faster the motor runs the higher becomes the voltage.
The logfile **Benelli TRK 251 (1 Cylinder).xml** shows several **CKP Sensor Errors** which are increasing with the time.
But they are still not enough to turn the MIL/EFI indicator light on.

Battery

Lead-Acid batteries allow to easily detect their charge status by simply measuring the voltage while the ignition key is off.
At 12.8 Volt it is completely full.



Very detailed information about batteries can be found at the **Battery University**.
The lifetime of a motorbike battery is approx one year when used frequently.
When the voltage of the fully charged battery drops below 9 Volt while cranking the battery should be replaced.

If the alternator / generator works correctly the voltage should be between 13.5 and 14.5 Volt while the engine is running.
If the regulator is defective and the voltage rises to more than 15 Volt the battery will be damaged.

Software from ElmüSoft

Desktop Organizer & Event Calendar
My program PTBSync (in english + deutsch) has an event calendar / scheduler with lots of features, calculates the holidays of 70 countries, can display multiple gadgets on the desktop (current weather, birthdays, moon phase, menstruation calendar, world clocks,...) pins notes on the desktop, replaces the primitive Windows trayclock and adds a detailed tooltip, synchronizes with an atomic clock, and much more.

ECU Hacker & Tuner
connects to ECU's (Engine Control Units) from motorbikes, cars and trucks, shows parameters, clears DTC's, has tuning for a Delphi ECU, has an ECU emulator, can analyze the CAN bus using cheap adapters, supports many protocols, has a sniffer and manual data injection for K-Line and CAN Bus, allows you to write your own macro scripts that communicate over K-Line and CAN Bus.

Oscilloscope Waveform Logicanalyzer
displays oscilloscope waveforms and has A/D converters, noise suppression, precise time measurement, logic analyzer with decoders for UART, SPI, I²C, CAN bus (also CAN FD) and chip-specific decoders. Can split half-duplex communication. Transfers signals over USB (SCPI) from the oscilloscope to the computer and remote controls the oscilloscope. It is open-source and you can extend it with new features.

Subtitle Tuner
calculates the display duration of the subtitles based on the count of characters in the text. This avoids that badly created subtitles disappear before you could read them. Can remove deaf texts, lyrics, italic/bold. Can resynchronize a subtile to a movie with different framerate or change the subtitle offset.

Secure Doorlock with RFID Cards
is a project with an RFID antenna and a Teensy processor that opens a door with Mifare Desfire cards. The RFID cards of up to 64 users can be stored in the flash memory. It is extremely secure because it uses strong encryption. A big battery assures that the door can be opened during a power failure.

Remote Control your Computer with Infrared
An infrared receiver and a Teensy processor allow to remote control the music player on your computer with any old remote control that you have from a TV, amplifier, DVD player, SAT tuner or whatever. The Teensy simulates a keyboard and sends keystrokes to the music player.

3D Render and Editor Control in C#
A .NET control that can be easily implemented into System.Windows.Forms applications to display 3D graphs. The user can move and rotate the graph with the mouse and edit single points. Many display modes are available: 3D surface, 3D shapes, 3D lines, 3D objects, also animated 3D objects.

USB HID Touchscreen
A Teensy processor emulates keyboard, mouse and touchscreen at the same time to remote control a computer for automation purposes.

You can write me in english, oder auf deutsch, o en español.

elmue@gmx.de