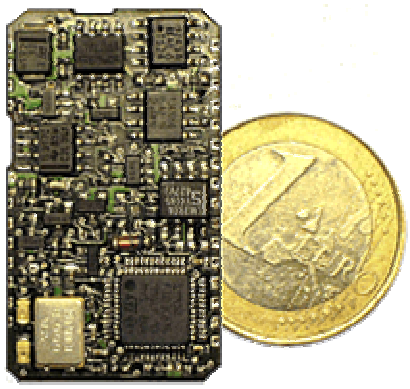
DXM1

DIAMEX®

OBD2 WITH SERIAL INTERFACE MODULE (Version 1.0)

Protocols: ISO9141-2, ISO14230-2 (KWP2000) ISO15765-CAN, J-1850 PWM, J-1850 VPWM
Automatic protocol detection, or pre-selection of the protocol
5 baud init Slow and Fast for KWP2000 Init
4 CAN protocols (11/29 Bit, 250/500 kbaud)
Automatic Wakeup at ISO/KWP2000
Short Fixed K-Line output
Pass-through capability
No external components associated with the OBD2 interface is needed.
Multi-frame-capable (multiple answers ECU's)
Serial interface with 9600 to 250,000 baud, compatible to MAX3232 and FT232R
Bios update boot loader possible
Super fast ARM Cortex-controller with 72 MHz clock frequency
AT command, it is compatible with many existing programs
Intelligent Protocol Scan function with storage of the last protocol
2 LEDs for Connect and dataflow
Changeable identification text for personalization of the chip
3.3 volt power supply for Controller
12 volt power supply from OBD socket
3.3-volt voltage regulator on-board (for supply of 12 volts)
Module size 35 x 20mm
Made in Germany



Description

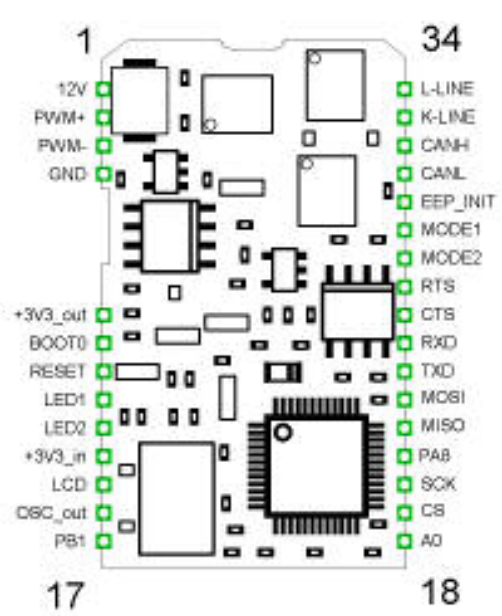
The new innovation for industrial and domestic on-board diagnostics - small, powerful, flexible and surprisingly low value. With the DXM you can quickly create own OBD2 applications, software as well as hardware. The module includes the complete hardware for OBD2 vehicle diagnostics, including the corresponding protocols and the associated communication firmware for the convenient connection via PC, notebook or co-processor. For the connection to the complex analysis, the standard interface TxRx isavail able.
To build a functioning OBD2 interface only the following components are required

- OBD2 connection (via 9-pin SUB-D connector)
- Shuttle board for placement of electronic components
- Serial Converters MAX3232 plus five 100n capacitors
- Protection diode + electrolyte capacitor 47µ, 35V

DXM is the result of several years OBD2 interface development. The result is an ultra-compact, easily integrated SMD-processor module that is easy to program and configure. Equipped with the latest 32-bit processor technology and the resulting performance, firmware, it serves as a base for various applications in industrial and private applications. Fully integrated hardware for immediate connection to the car diagnostic interface opens up completely new perspectives. Engineering and development services as well as costs can be saved. Valuable development time, particularly the resources and material-intensive adaptation to different applications, tolerances and standard deviations, can be elegantly circumvented. The result is a cheap and fast product implemented on a high technical level - an undemanding, powerful OBD2 heart.

***DXM** - solid, innovative, future-proof and inexpensive.*

DXM1 Pinout



PIN	Description
1	+12 volts for the vehicle to supply the OBD2 interfaces
2	PWM output + / VPWM to the OBD2 port PIN 2
3	output PWM to the OBD2 connector pin 10
4	Ground connection
5	Not available
6	Not available
7	Not available
8	Not available
9	3.3 volts output from the 12 volt vehicle power is won. This pin can be connected with PIN15 to supply the module with 3.3 V voltage. Maximum load capacity approx 100mA
10	boot 0, is only required for the programming of the module. Must remain open for normal operation.
11	RESET of the module, if this pin is laid on GND. Internal pullup resistor.
12	LED1, low active
13	LED2, low active
14	+3.3 Volt power supply for the micro controller on the DXM1
15	LCD output (for future extensions)
16	8 MHz clock output for connecting external micro controllers
17	PB1 (free port of the micro controller for future extensions)
18	A0, output (for future extensions)
19	CS, chip- select für die SPI-Schnittstelle (für spätere Erweiterungen)
20	SCK, SPI interface (for future extensions)
21	PA8 (free port of the microcontroller for future extensions)
22	MISO, SPI interface (for future extensions)
23	MOSI, SPI interface (for future extensions)
24	TXD, Serial Interface Output
25	RXD Serial interface input
26	CTS, Serial Interface, Clear to send, input (must not be connected)
27	RTS, Serial Interface, Request to send, starting (must not be connected)
28	MODE2 will be needed for future expansions, internal pullup
29	MODE1 will be needed for future expansions, internal pullup
30	Initializes the EEPROM on the module after RESET, internal pullup
31	L outputCAN OBD2 connector to pin 14
32	H output CAN OBD2 connector for PIN 6
33	K-LINE output to OBD2 connector PIN 7
34	L-LINE output to OBD2 connector Pin 15

Communication with the controller

The communication with the DXM1 module is done via the serial interface. The data is in delivered status fixed to 9600 baud, 8 data bits, 1 stop bit (9600, 8N1) is set. This speed is sufficient for the relatively small number of transferred data. You can optionally set the baud rate via AT command or via the boot loader up to 250,000 baud converted. The handshake lines RTS (Request To Send) and CTS (Clear To Send) may be used for the flow control, but they can also remain unconnected if it is guaranteed that the input or output buffer does not overflow.

After reset of the DXM-1-module the following message should be sent via the serial interface:

```
DIAMEX DXM1 v1.0

>
```

Note: This message varies depending on the version, and may additionally be changed via AT command.

Which message even appears, indicates that the communication of the interface works. The sign ">" means that the interface is ready to receive commands. The interface now distinguishes two different command groups:

1. Internal commands for configuration and initialization of the interface controller. All these commands begin with the characters "AT", this was applied by the control commands for modems, and means "Attention, Attention."
2. Data which are provided to the OBD2 bus for the control of the vehicle concerned.

All of these commands are transmitted as hexadecimal numbers; therefore it may only be entered ASCII characters 0-9 and AF as pairs.

All entered commands has to be finished with a end of line sign (Carriage Return, Dez. 13, HEX $0D)
Spaces or tabs are automatically filtered out, small and capital letters will not be distinguished. In the following examples, each entry has to be completed with the line ending characters, they will not be specified extra.

Examples:

```
at dp
```
Is internally changed to ATDP
```
A  T  Z
```
Is internally changed to ATZ
```
01 1c
```
Is internally changed to 011C

If the line ending characters CR is absent, the command will automatically be cancelled after 5 seconds and a "?" is used as error. Commands that the controller do not understand or incorrect entries in hex values, will also reported with a "?" as error message.

OBD2 commands must always start with even number of hex characters:

Example:
```
0100
```
or
```
01 00
```

An odd number of characters generated an error report.

Some AT commands require as additional parameters one or more hex characters. For the exact number, please refer to the command list. An incorrect number of parameters will also report an error message.

The AT command of DXM1 module

>ATZ

This command causes an immediate termination of all current functions and a reboot of the controller. All parameters are stored in the ground state and the self test (by means of flashing LED's) is performed. Finally, the identification text will be reported.

Issue:
```
Diamex DXM1 v1.0

>
```

Note: The text depends on the BIOS version and can be changed by AT commands.

>ATWS

This command causes an immediate termination of all current functions and a reboot of the controller. All parameters are stored in the ground state is set. Finally, the identification text will be reported. This command has the same effect as ATZ, but is much faster, since the self test is skipped.

Issue:
```
Diamex DXM1 v1.0

>
```
Note: The text depends on the BIOS version and can be changed by AT commands.

>ATI

Here only the identification text is reported, without performing a warm start.
Ongoing functions like the automatic wakeup for ISO or KWP2000 will not be canceled.

Issue:
```
Diamex DXM1 v1.0

>
```

>ATD

All parameters are stored in the ground state as after a cold or warm start.

Issue:
```
OK

>
```

>ATE0
>ATE1

This command switches the serial echo on (1) or off (0). All data which are received through the serial port will be sent back to the PC when the echo is switched on.

Issue:
```
OK

>
```

>ATL0
>ATL1

With his command the transmission of the linefeed character can be switched on (1) or off (0) at the end of the line. All responses sent to the PC are usually concluded with a Carriage Return (13 Dec., $ 0D Hex). The Linefeed sign (10 Dec, $0A Hex) is reported after every Carriage Return with switched on Linefeed. Especially when using a terminal program and the command istransmitted by hand it is quite useful to leave the linefeed switched on, because otherwise all the answers in a single line are displayed and new answers overwrite the old one.

Issue:
OK

>

>ATM0
>ATM1
Turns the memory function of the final protocol on or off.

If the same OBD2 protocol is often used, it may be useful to save this as presetting in the EEPROM. These turns on the memory function with ATM1, then sets a connection with the vehicle control unit, so that a protocol is activated, finally switch off the Memory Function with ATM0 again. The actual protocol is used as standard immediately.

If the Memory Function is switched on, also the parameters ATEX, ATLx, and SBX ATHx ca be saved. After saving, this function should be switched off again with ATM0 to prevent uncontrolled writing of the EEPROM.

The memory function can also be disabled by ATZ and ATWS.

>ATH0
>ATH1
With this command you can set if the OBD2 header answers and the check sum byte shall also be displayed.

Issue:
With ATH0:
>0100
41 00 E8 19 30 12

With ATH1:
>0100
48 6B 10 41 00 E8 19 30 12 47
48, 68, 10 are the 3 header bytes
47 is the check sum byte

Issue:
OK

>
As there do not exist any header and check sum bytes, the CAN-ID will be communicated instead. More information concerning this matter please refer to the CAN-protocol description later in this document.

>ATBD

The abbreviation of "**B**uffer **D**ump" causes the display of the OBD2 receiver storage. Valid data is only available only if an OBD2 command was executed before.

Example:
```
>0100
41 00 E8 19 30 12

>ATBD
0A 48 6B 10 41 00 E8 19 30 12 47 00 00
```

The first byte indicates the number of valid characters in memory. In this case it is 10 characters (Hex 0A). The last byte in this case is invalid and can include any value.

>ATB

Display of the OBD2 reception buffer. This command differs from the previous ATBD, that the complete contents of the receive buffer is displayed, instead of only the first maximum of 12 characters. This is especially necessary for multi-frame and multi-ECU answers.

For example (DTC query with 4 error codes):
> 03
43 01 15 02 30 03 50
43 04 60 00 00 00 00

```
>ATB
16 02
C7 F1 10 43 01 15 02 30 03 50 A6
C7 F1 10 43 04 50 00 00 00 00 6F
```

In the first response line is the total number of bytes in the buffer is specified (16) and the number of response frames (02). Among these also the contents of the frame buffer with header and check sum bytes.

>ATSR xx
Input of the RX ECU OBD2 filter address for OBD2 responses.
If in a "Functional Request several ECU answer, can be filtered the requested information by setting the RX-filter.

Example (KWP2000 with switched on headers):
> 0100
C6 F1 10 41 00 B8 7B B0 10 FB
C6 F1 18 41 00 08 28 00 00 40

Here 2 different ECU's answer. If only the answers of the 2nd ECU (18) should be filtered out, can this be done by entering the command ATSR18.

In KWP2000 is always the 3rd Byte (Target Address) address used as a filter. At ISO9141, PWM and VPW the 2nd or 3 Byte is used depending of the requested type. The 3rd byte is used by "Physical Request" and the 2nd +1 byte is used by "Functional Request".

Example (ISO9141 Functional Request):
Request Header 68 6A F1
Answers 48 6B 10

The filter address can only be switched off by entering ATZ, ATD ATWS or off.

This parameter has no function in the CAN protocol.

>ATN

Display of the current protocol used as Hex value. F0 .. F9

Example:

```
>ATN
F2

>
```

F0: no active protocol
F1: PWM protocol
F2: VPWM protocol
F3: ISO9141 protocol
F4: KWP2000 protocol (5 baud init)
Q5: KWP2000 protocol (Fast Init)
F6: CAN protocol 11Bit ID, 500kBaud
Q7: CAN protocol 29Bit ID, 500kBaud
Q8: CAN protocol 11Bit ID, 250kBaud
Q9: CAN protocol 29Bit ID, 250kBaud

An output of F0 means that no protocol iscurrently used (f or example, ATZ).

>ATDP

Display of the currently used protocol in plaintext.

Example:

```
>ATDP
ISO 9141-2

>
```

Here is the list of all possible issues:

NOT CONNECTED
SAE J1850 / PWM
SAE J1850 / VPWM
ISO 9141-2
ISO 14230-4, KWP2000 (5 Baud Init)
ISO 14230-4, KWP2000 (Fast Init)
ISO 15765-4, CAN (11/500)
ISO 15765-4, CAN (29/500)
ISO 15765-4, CAN (11/250)
ISO 15765-4, CAN (29/250)

>ATK

Displays the current keywords on ISO9141 and KWP2000.

Example:

```
>ATK
8F E9

>
```

If there is no connection to the vehicle, or a protocol is used that does not support keywords, the output is invalid.

>ATKW0

>ATKW1

For a Slow, Init keywords are distinguished between protocols 3 (ISO9141) and 4 (KWP2000). This automatic detection can be disabled with ATKW0. This is just for experimental purposes. For a secure detection of the protocol the detection must be switched on.

Issue:
OK

>

Default: ATKW1

>ATP[A]x
Manual setting of the current protocol or the automatic protocol search.

You can specify a protocol presided. No other protocol will be searched, but canceled with “UNABLE TO CONNECT” if the controller does not respond.

ATP1 PWM
ATP2 VPWM
ATP3 ISO9141-2
ATP4 KWP2000 5 Baud Init
ATP5 KWP2000 Fast Init
ATP6 CAN 11/500
ATP7 CAN 29/500
ATP8 CAN 11/250
ATP9 CAN 29/250

If instead ATPAx (x = log number) is entered, the controller automatically searches all the other protocols, if the current is not found.

With ATP0 or ATPA0 (identical), the automatic search is activated. There is no protocol default and all protocols will be searched after a reboot until the right one has been found.

Caution!
If a fixed protocol without auto-search is default, no other protocol will be searched, if the electronic control unit does not response on defaulted issues. In this case, please activate search function with ATP0 or ATPAx (x = current protocol).

If the memory function with ATM1 is switched on, the change of setting in the EEPROM will be saved immediately and reused for restart.
By entering ATP without additional parameter will the actual mode be displayed. In this case, no change will be done.

More information about this in the section entitled "OBD2 protocols”.

The factory default is ATP0. It supports all protocols searched.

>ATV

An overview of all the modifiable parameters of the DXM1 are displayed.

Example:

```
>ATV
E1 L1 H0 M0 KW1 SW19 SR00 SB00 (9600)
  HEADER: 00 00 00
  WAKEUP:
CA1 CC1
CAN-BAUD: 00000000
CAN-TXID: 00000000
CAN-RXID: 00000000
CAN-MASK: 00000000
CAN-FLOW: 00000000
FLOWDATA: 00 00 00 00
```

E1 = echo one (1), from (0)
L1 = linefeed one (1), from (0)
H0 = header (1), from (0)
M0 = Memory one (1), from (0)
KW1 = protocol detection by keyword
one (1), from (0)
SW19 = ISO / KWP wakeup 0x19 * 100ms
SR00 response = 00 bytes = auto
SB00 Serial baud rate = 00 = 9600
HEADER = ISO / KWP / PWM / VPWM header bytes
Wakeup = ISO / KWP wakeup command
CA1 = CAN auto-format (1), from (0)
CC1 = CAN flow control one (1), from (0)
CAN BAUD = baud rate bytes CAN controller
CAN-CAN TXID = Transmit ID
CAN-CAN = RXID reception ID
CAN-CAN-MASK = mask receive
CAN-CAN FLOW = Flow Control ID
FLOW DATA = CAN-message-flow data

>ATSB x

Set Baud rate of serial interface. In the factory default or after re initialization of the EEPROM memory is the baud rate set on 9600. A baud rate is changed only after the ATZ is enabled. The new fixed rate can be stored in the EEPROM when the store is released earlier with ATM1.

Example:

```
>ATSB4
115200
OK
>ATZ
```

Here the baud rate is set to 115200. Please make sure that, after the ATZ baud rate on the serial interface, controlled by the computer / controller is also changed, because otherwise no more commands are possible. If the selected baud rate stored in the EEPROM is unknown, try to consider all possible baud rates or EEPROM memory reinitialize. After that 9600 baud are set again. If the changed baud rate is not saved in the EEPROM, please reset or disconnects the power supply again to reactivate the baud rate saved in the EEPROM.

List of possible baud rates:

```
ATSB0 = 9600 Baud (Standardwert)
ATSB1 = 19200 Baud
ATSB2 = 38400 Baud
ATSB3 = 57600 Baud
ATSB4 = 115200 Baud
ATSB5 = 125000 Baud
ATSB6 = 250000 Baud
```

>ATSH xx yy zz

Manual setting of the header bytes for ISO9141, KWP2000, PWM and VPWM protocols.

There are 3 Hex values expected:
xx = priority/type-byte
yy = Target-address
zz = Source-address„

The exact structure of the 3 bytes, please refer to the SAE protocol specifications.

Regarding KWP2000, the length at the 1st byte (xx) is adjusted automatically.

If the protocol with ATPx is set, the header bytes can be changed before the 1 connection built up. If the auto-search function isenabled, the standard values are always used for the connection build up. Here the header bytes can only be used after changing the Connect.

Defaults:

ISO9141-2:	`68 6A F1`
KWP2000:	`Cx 33 F1` (x = length)
PWM:	`61 6A F1`
VPWM:	`68 6A F1`

The header bytes can be reset by entering ATZ, ATD ATWS.

This parameter has no function in the CAN protocol.

>ATWM xx yy zz aa [bb] [cc]
Manually setting of the wakeup message bytes for ISO9141 and KWP2000.

There are 4-6 Hex values expected:
xx = priority / type byte (with length indication in KWP)
yy = Target Address
zz = Source Address
aa, bb, cc = Command Bytes 1-3

In the 1st byte the KWP2000 must include the correct length.

Example (ISO9141-2):
```
>ATWM 68 6A F1 03
OK
```

Example (KWP2000):
```
>ATWM C2 33 F1 01 04
OK
```

Defaults:
ISO9141-2: `68 6A F1 01 00`
KWP2000: `C1 33 F1 3E`

The wakeup message bytes can be reset to the standard values by entering ATZ, ATD ATWS.

This parameter has no function in the PWM, and CAN VPWM protocol.

>**ATSW xx**
With this command, the length of time between automatic wakeup commands in an existing ISO9141 and KWP2000 can be set.

ISO9141 and KWP2000 vehicle controllers expect regular data traffic to the connected OBD2 interface. Should this e missed for longer time (according to SAE standard 5000ms), the connection will be separated and has to be build up again by a re-init.

This command expects the length of time as a 2-digit HEX code. The time between wakeup commands is derived from the hex value * 100 milliseconds.

Example:
```
>ATSW1E
OK

>
```

Here is the wakeup time set $ 1E (30) * 100ms = 3.0 seconds.
The standard value is set with ATZ, ATD or ATWS on $ 19 (25), which is equivalent to 2.5 seconds. If a value of 0 is entered, the automated wakeup is disabled and the connection will be separated after 5 seconds if no OBD2 commands are transmitted.

>**ATCA0**
>**ATCA1**
With this command, the automatic formatting of the transmitted and received CAN data can be switched on or off.

A switch off of the automatic formatting is only useful if the DXM Controller in CAN buses shall be operated, which are not OBD2 compatible.

Is the formatting switched off, has all required bytes (max. 8) of the CAN data packet to be entered with switched on formatting the length of max. 7 bytes will be automatically added.

Example:

With auto-format:
```
>0100
```
is converted to:
```
0201000000000000
```

In OBD2 mode 8 bytes are always sent in data packages. The following command is equivalent to the above mentioned example, but in a switched off auto-format:
```
>0201000000000000
```

If instead, only
```
>020100
```
is entered, only one packet is sent with 3 bytes, which is not OBD2 compliant.

The setting of the auto-formatting also affects the output of the responses received on the CAN bus.

Example:

With auto-format, all 8 bytes of the CAN-frame were sent. There will not be any evaluation of the received data:

```
06 41 00 B8 7B B0 10 00
```

Is the auto-format switched on, the received OBD2 data package will be evaluated. In this case the 1st byte shows that 6 valid bytes follow. Thus, the following output:

```
41 00 B8 7B B0 10
```

In multi-frame responses is the output of the data package in switched on format fitted to the ELM controller.

For example, auto-format.
Query the error codes with 4 errors:
```
>03
00A
0: 43 04 01 15 02 30
1: 03 50 04 60 00 00 00
```

The 1st Line indicates that the response consist of 00A (hex) = 10 valid bytes. Each additional line begins with a counter followed by a colon, which indicated order of the received data. It is counted from 0 to F (hex) and then starts again at 0 if there are more packets to be transmitted.
Switched off formatting is the answer as follows:

```
10 0A 43 04 01 15 02 30
21 03 50 04 60 00 00 00
```

Here the PCI-length byte of the PC software has to be evaluated.
For more info on OBD2 Data please refer to the SAE J1979 - specifications.

standard setting: On (ATCA1).

>**ATCC0**
>**ATCC1**

In OBD2 systems, multi-frame answers have to be sent flow control messages from the tester, which indicate the controller, that following packages have to be accepted. That happens trough the DXM controller automatically.

If the controller in the CAN systems, which are not OBD2 compliant, has to be applied, it may make sense, to switch off these flow control messages.

Example:
```
>ATCC0
OK

>
```

standard setting: On (ATCC1).

>**ATCD xx**

Flow control data packages, which hast o be sent in multi frame responses, contain beside the status byte (FS), a block size byte (BS) and also a byte for the duration (ST), which have to be added between the following response packages. This way the controller has time to process these data's. With this command the duration can be changed.

Example:
```
>ATCD 7F
OK

>
```
Set the length of time to the maximum value of 127ms.
This command is only for experimental purposes and may be available for testing simulators. In general, the setting is not changed.
Default: 0A (10 ms)

>ATCI xxx
>ATCI xxxxxxxx

The CAN-ID which has to be sent, is set with this command. This can be useful when the DXM controller is not set in OBD2 complaint CAN systems.

Defaults:

11 Bit ID	7DF
29 Bit ID	18 DB 33 F1

Example:

```
>ATCI 7E0
OK

>
>ATCI 18 DA 10 F1
OK

>
```

Please make sure that by the number of characters is determined whether the 11Bit or 29Bit ID has to be changed. For 11Bit ID must always 3 hex characters be entered and for 29Bit ID must always 8 hex characters be entered.

By entering ATWS, ATD or ATZ the ID is set back to the standard values.

>ATCF xxx
>ATCF xxxxxxxx

Set CAN RX-filter. If too much data is transferred on the CAN-bus, it may happen that the controller's internal buffer overflows if the data is not transferred in time to the PC. In this case, the data received should be filtered from the desired. The filter is used together with the RX-mask used with the command which can be changed with the ATCM command.

Standard values:

11 Bit Filter	7E8
29 Bit Filter	18 DA F1 00

Examples:

```
>ATCF 7E0
OK

>
>ATCI 18 DA F1 10
OK

>
```

Please make sure that by the number of characters is determined whether the 11Bit or 29Bit filter should be changed. For 11Bit filter must be entered always 3 hex characters for 29Bit filter must always be entered 8 hex characters.

By entering ATWS, ATD or ATZ the ID is set back to the standard values.

>ATCM xxx
>ATCM xxxxxxxx

Set CAN RX-mask. In conjunction with the RX filter (ATCF) the mask can be used, to filter out individual or a group of data.

standard values:

11 Bit mask	7F8
29 Bit mask	1F FF FF 00

Examples:

```
>ATCM 7F0
OK

>

>ATCM 1F FF 00 00
OK

>
```

In conjunction with the filter the first 1-bit in the mask indicates whether the incoming message has to be compared with the filters. If the mask bit is 0, this bit will be accepted as "ok".

Please make sure that it is determined by the entered number of characters whether the 11Bit or 29Bit mask should be changed. For 11Bit mask 3 hex characters has to be entered and for the 29 bit mask 8 hex signs has to be entered.

By entering ATWS, ATD or ATZ the ID is set back to the standard values.

>AT!00
Output of the serial number of the DXM controller. All DXM controllers have a unique serial number which is not modifiable. You can enquiries with this command.

Example:
```
>AT!00
X123456789

>
```

>AT!01
Output of the controller type and the BIOS version number. Since the identification string is changeable, there is no clear identification of the controller type and the BIOS version on the ATI command possible. For this reason, this command has been inserted, whose output is unchangeable and always displays the right controller type.

Example:
```
>AT!01
DXM1-10

>
```

The above mentioned example shows the output of a DXM-1 controller with BIOS version 1.0

>AT!10
Measurement of 12-volt vehicle power.

Example:
```
>AT!10
12.5V

>
```

Further commands, especially for the amendment of the identification document and the internal test of the DXM interface are not be listed here. Interested parties can request this information by e-mail, by indicating the use for the purpose of the developer of the DXM controller by e-mail.

Direct input of a OBD2 command

To enter an OBD2 command, only for this command necessary data's in hex code are committed. If previously no connection has been set up with the OBD2 bus in the vehicle, will this action be done by entering the command uniquely. The different protocols are inquired depending of the protocol settings (see section "OBD2 protocols")

Once the vehicle controller replies with a protocol on a inquiry, further test are cancelled and the recognized protocol is used for all further inquiries. The time for the completion of the first command consequently lasts max. 2.5 seconds, due to the long response time at Slow Init. VPWM, PWM and CAN be detected within 1 second.

Example:
Output at ISO9141 and KWP2000 Connect.

```
>0100
SEARCHING...
41 00 E8 19 30 12

>

>0100
41 00 E8 19 30 12
```

Example:
Output at VPWM, PWM or CAN Connect.

```
>0100
41 00 E8 19 30 12


>
```

Overview of the AT commands DXM1

General commands:

ATD reset all values
ATE0 echo off
ATE1 echo on (standard)
ATI identification Text
ATL0 line feed off (standard)
ATL1 line feed on
ATV display variable parameters
ATWS warm start
ATZ reset
AT! 00 serial numbers
AT! 01 Controller ID
AT! 10 voltage measurement of vehicle
ATSB x serial baud rate setting

OBD2 commands:

ATBD Buffer Dump
ATB display of the receiving buffer
ATDP display of the current protocol in plaintext
Ath0 header off (standard)
ATH1 header on
ATM0 protocol memory off(standard)
ATM1 protocol memory on
ATN display of the current protocol number
ATP display the current protocol configuration
ATP x preset protocol x
ATP Ax preset protocol x with auto research function
ATP0 always use protocol auto search
ATSH xx yy zz define header
ATSR xx set RX filter address

ISO/KWP2000- specific commands:

ATSW xx set wake-up interval times
ATWM xx yy zz aa [bb] [cc] set wake-ip message
ATK displays the current keywords
ATKW1 automatic detection of the protocol on (standard)
ATKW0 automatic detection of the protocol off

CAN-specific commands:

ATCA0 CAN auto format off
ATCA1 CAN auto format on (standard)
ATCC0 CAN flow control off
ATCC1 CAN flow control on (standard)
ATCD xx set CAN flow delay
ATCF xxx RX CAN message filter 11 Bit
ATCF xxxxxxxx CAN message filter RX 29 Bit
ATCI xxx CAN message identifier TX 11 Bit
ATCI TX xxxxxxxx CAN message identifier 29 bits
ATCM XXX CAN RX message mask 11 Bit
ATCM xxxxxxxx CAN RX message mask 29 Bit

OBD2 protocols

The DXM1 module supports all currently existing OBD2 protocols.

Modus	Protokoll
1	J-1850 PWM (41.6kBaud)
2	J-1850 VPWM (10.4kBaud)
3	ISO 9141-2 (5 Baud Init)
4	ISO 14230-4 KWP2000 (5 Baud Init)
5	ISO 14230-4 KWP2000 (Fast Init)
6	ISO 15765-4 CAN 11Bit-ID, 500 kBaud
7	ISO 15765-4 CAN 29Bit-ID, 500 kBaud
8	ISO 15765-4 CAN 11Bit-ID, 250 kBaud
9	ISO 15765-4 CAN 29Bit-ID, 250 kBaud

In the factory setting the DXM1 module is configured that all protocols are browsed automatically until a fitting (ATP0) is found. If no matching protocol is found, the following text is displayed:

```
UNABLE TO CONNECT
```

The protocols are tested in the following order:

1	J-1850 PWM
2	J-1850 VPWM
3	5: KWP2000 (Fast Init)
4	3+4: ISO/KWP2000 (5 Baud Init)
5	6: CAN 11/500
6	7: CAN 29/500
7	8: CAN 11/250
8	9: CAN 29/250

Once a matching protocol is found, the search is aborted and the found protocol is used or all further protocols. The current protocol can be displayed with the command ATDO.

```
>ATDP
ISO9141-4

>
```

At the same time, the found protocol is used as search start, when the connection was previously closed with ATWS or ATZ. This can be shown with the command ATP.

```
>ATP
AUTO 3 = ISO9141-4

>
```

If the memory function protocol is activated (ATM1), the current protocol is saved in the EEPROM of the module and will be even achieved after separation of the power supply.

After reconnecting to the power supply the contents of the EEPROM is read out and the saved protocol is used at once for the new search. If the saved record is not found, the protocol are tested in the mentioned order, but the previously negative tested protocol will be skipped.

If the search should start with the requested protocol, can this be done also manual:

```
>ATPA5
AUTO 5 = ISO 14230-4 KWP2000 Fast Init

>
```

Here, the search starts with the Protocol 5. If the protocol is not found, the protocols are tested in the order of the search list.

If the module DXM1 should always use one certain protocol, can this be determined by the following command:

```
>ATP5
5 = ISO 14230-4 KWP2000 Fast init

>
```

Here, only the Protocol 5 is used. If the protocol is not found, the search is aborted and UNABLE TO CONNECT or another error is displayed. If the memory function is switched on, the protocol will be saved and also achieved and obtained before and after the disconnection of the power supply. Therefore Attention! If no connection can be established, initially check with ATP which protocol is preset.

ERROR REPORT

Many of the following error reports will only appear during the first connection build up and only appear if a fixed protocol with ATPx is preset. In the auto search mode the error reports are disabled in order to continue the test with the next protocol.

?

Syntax error. Is reported, if the AT command does not exist or if parameter commands are missing or has been entered incorrect.

BUS BUSY

Occurs at PWM and VPWM, if a command could not be sent withing the prescribed time.

CAN ERROR

Errors in the CAN protocol. Because the CAN protocol is relatively secure against transmission errors on the bus, this report will only appear, if the connection to the CAB bus has been disconnected.

NO DATA

Is reported, if an OBD2 command has not been answered withing the prescribed time, for example if it is not supported in the controller.

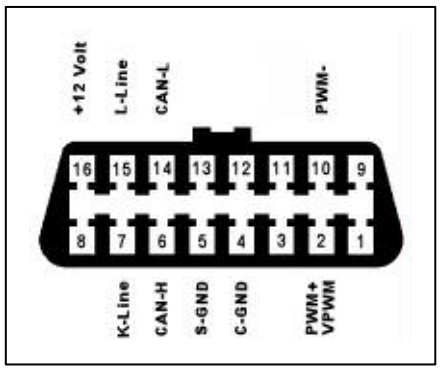
UNABLE TO CONNECT

Appears when no connection to a control unit could be built.

ERROR (xx)

An unspecified error occurred.

The OBD2 connector



All new manufactured vehicles must have such a 16-pin connector according to OBD2 standard feature. The jack must be in a 1 meter radius from the driver's set. Although the design is standardized, the various vehicle manufacturers use different transmission protocols.

Standardized, however, is which protocol belongs to which pin of the jack. In most cars, only the contacts of the 16-pin socket are present, which also lead signals. Often are additional (non-standard) contacts on pins in place and also connected with the controller unit, which are used for the internal diagnostics for the vehicle manufacturer. Since there are no official documentation for this contacts, it should these be left blank.

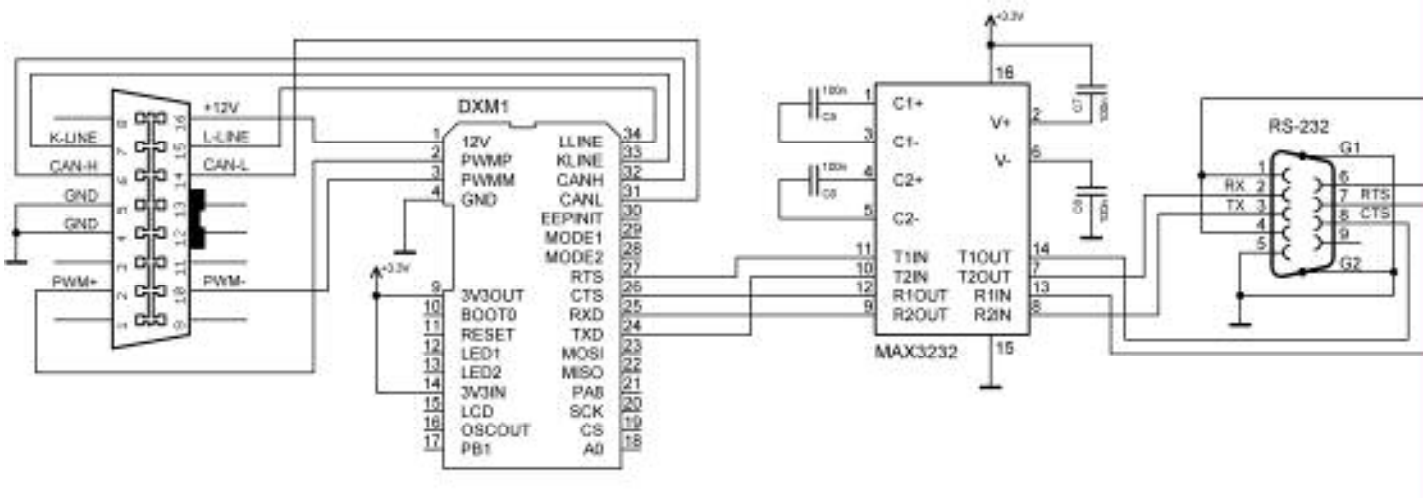
The following contacts are always present:

PIN 4	vehicle chassis ground (pin 4 + 5 are usually connected)
PIN 5	Signal Ground
PIN 16	Power + 12 Volt

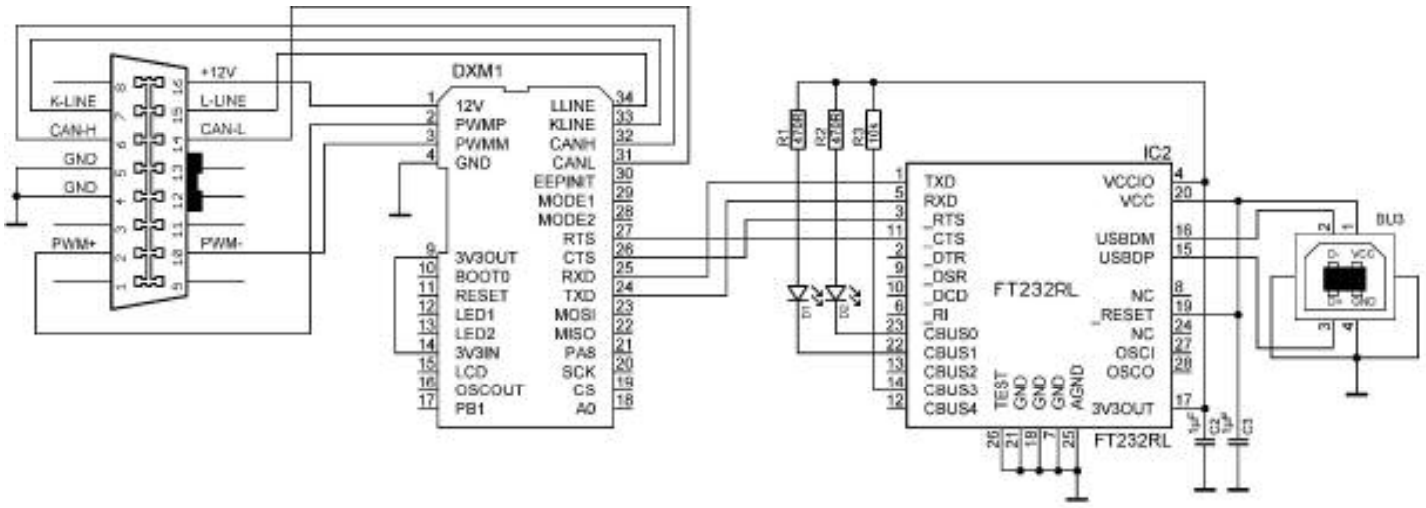
The following contacts exist depending on the used protocol and engaged:

PIN 7	ISO9141 / ISO14230 K-Line	(input and output)
PIN 15	ISO9141 / ISO14230 L-Line	(input)
PIN 2	J1850, PWM+	(differential signal bi-directional)
PIN 10	J1850, PWM-	(differential signal bi-directional)
PIN 2	J1850, VPWM	(Bi-directional)
PIN 6	ISO15765, CAN-H	(differential signal bi-directional)
PIN 14	ISO15765, CAN-L	(differential signal bi-directional)

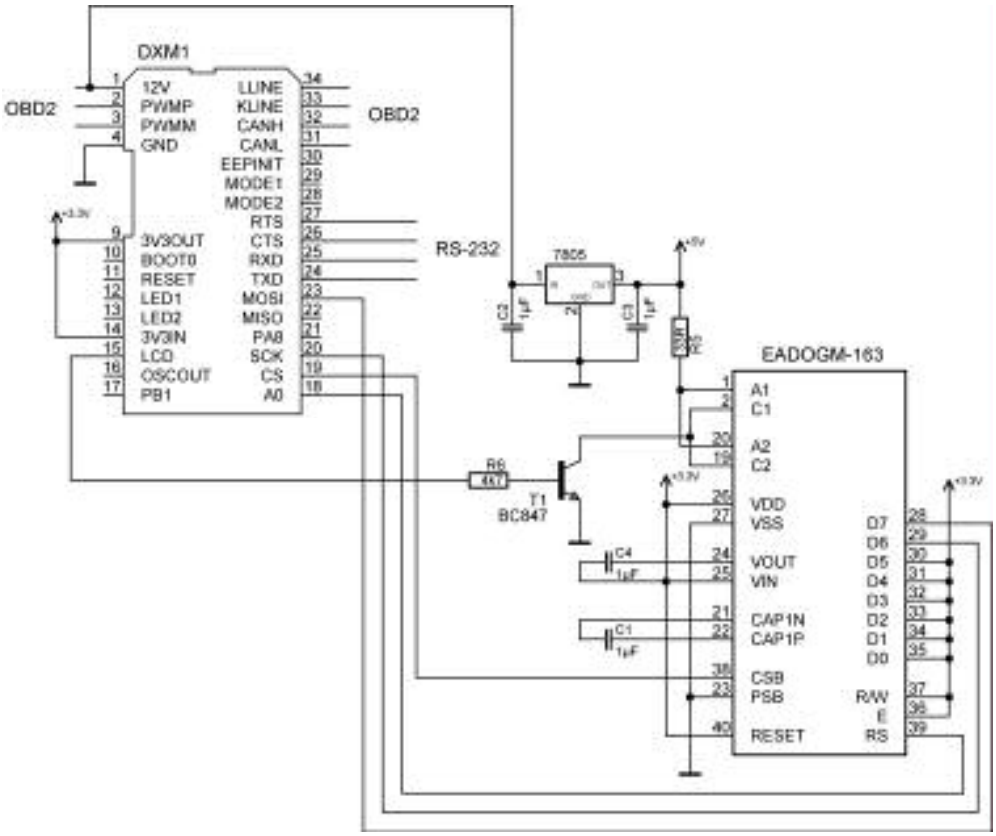
Schematic: DXM1 with MAX3232-Serial Converters



Schematic: DXM1 with FT232R USB converter



Schematic: LC-display on DXM1



Connections of a LCD display (3 x 16 characters) to the SPI port of the DXM1.

Over Transistor T1 the background light is switched on. The power supply of the display can be done from the 3.3 volt supply of the DXM1. But for the backlight a separate voltage regulator is required because the current demand is too high for the 3.3 volt regulator on the DXM1.

Hinweise

Changes on the Bios and instructions of the controller reserved.

DIAMEX ® is a registered trademark.

All brand names, trademarks and registered trademarks are the property of their rightful owners and are used here only for description.

Liability
The manufacturer assumes no liability for damage caused by the application of DIAMEX interface and the diagnostic software.

Contact, Forum, software, updates

http://www.dxm.obd-diag.net

info@diamex.de