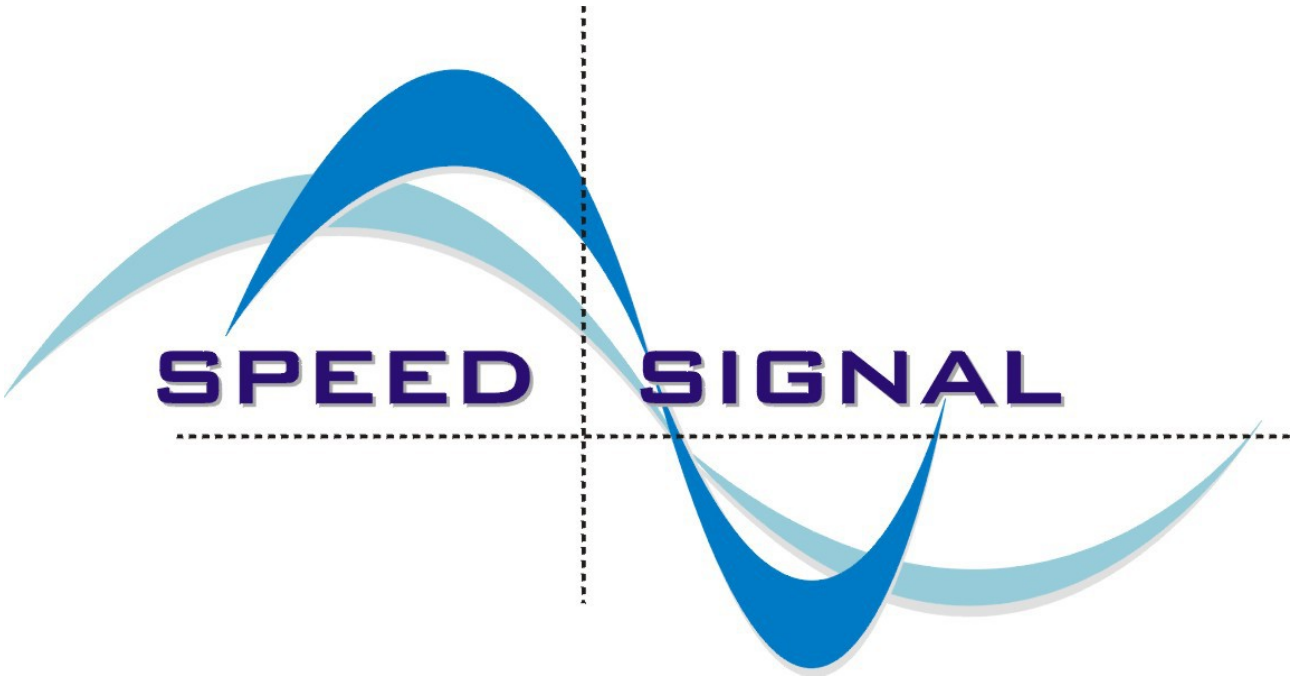

speedsignal GmbH

Software Design Document

Version R14

Date: 24.06.2015

CAN2COM Protocol

speedsignal GmbH
Carl-von-Ossietzky-Str. 3
83043 Bad Aibling

E-Mail: info@speedsignal.de
Tel: +49 8061 49518-0
Fax: +49 8061 49518-10

1 Changelog

File: X:\projekte\2009\2009_029_X2Y\Dokumentation\RS232Protokoll_R14 english.odt

date 24.06.15 11:38

version	*author*	*date*	*state*	*description*
0.01	Härtling	14.10.05	initial	initial version
0.02	Härtling	04.01.06	-	corrections after beta prototypes
0.03	Härtling	21.04.06	-	ID 0x17 implemented, for BMW E60
0.04	Günther	07.02.07	-	definition of steering wheel buttons
0.05	Dörrer	29.03.07	-	-
0.06	Dörrer	16.04.07	-	crc added, cfg message, date added
0.07	Dörrer	24.04.07	-	modes of operation
0.08	Dörrer	11.06.07		steering wheel & gearshift data modified
0.09	Härtling/ Kukalaj	05.12.13		added commands 0x04, 0x20 .. 0x25
0.10	Kukalaj	24.03.14		Example request mileage was not correct
0.11	Härtling	06.05.14		Added command 0x05, 0x25 changed response to set config string in case of success (up to now in implementation no response was sent at all)
0.12	Härtling	30.07.14		added command 0x06 get available data
R13	Stacheter	15.12.14		Revision of the whole document, added 0x26 and 0x27
R14	Stacheter	24.06.15		Corrected handling of more than one available services Added contact information Corrected command to adapter example

2 Directory

3 Definition

3.1 Protocol specification

The protocol defines the transmission of the vehicle data to an external device. It standardizes data from different vehicles.

3.1.1 Interface

The data exchange happens with serial interface (RS232) with 9600 baud, 8 data bits, no parity, 1 stop bit (9600, 8n1).

3.1.2 Blockframe

The dataframe starts with STX (0x02; start of text) and ends with ETX (0x03; end of text). In between the message ID, the databytes and the checksum (CRC; XOR over ID and data bytes) are transmitted.

STX	ID	DATA	CRC	ETX

In the following chapter, we focus on the ID and the data bytes. The ID and checksum have each 1 Byte. The data length varies and is thus n Bytes.

A dataframe with ID = 0x01 and n = 3 Bytes of data information is shown as an example:

[STX]	0x01	0xFF 0xFF 0xFF	0xFE	[ETX]

Every nibble of the ID, CRC and data Bytes has to be sent as an ASCII character! For example, if 0x0A is sent as databyte, 0x30 (ASCII for '0') and 0x41 (ASCII for 'A') is transmitted. So each Byte of ID, data and CRC is transmitted in two Bytes.

On the physical layer, the message will be transmitted as:

0x02	0x30 0x31	0x46 0x46 0x46 0x46 0x46 0x46	0x46 0x45	0x03
[STX]	[0][1]	[F][F] [F][F] [F][F]	[F][E]	[ETX]

A detailed example is given in Chapter 5.

3.1.3 Configuration

The C2C-Interface needs CAN activity to be awake. However, it doesn't send any data until certain data is requested (Ch. 4.2.1) or the configuration mask is set (Ch. 4.2.9)!

Mode	***Description***
request data manually (Ch. 4.2.1)	Sending the request commando results in receiving the data or a corresponding error message (4.3.26), if no data is available. It is also possible to request certain data at any time while the autosend mode is activated.
auto send (Ch. 4.2.9)	To activate the auto send mode, the configuration mask for the desired IDs has to be set. The corresponding data is sent periodically (as given for the corresponding ID) or when the state has changed.

3.1.4 Calculation of the Checksum (CRC)

The CRC is calculated as XOR over the ID byte and every data bytes.

It has to be calculated before the frame is transferred to ASCII.

Example:

ID: Byte[0] = 0x17;
Data: Byte[0] = 0x09; Byte[1] = 0x30; Byte[2] = 0x14

The checksum results to:

CRC = 0x17 ^ 0x09 ^ 0x30 ^ 0x14 = 0x3A

4 Command specification

4.1 Identifier Overview

4.1.1 Interface Command Identifier Overview

Messages to the adapter

ID (Hex)	*Description*
0x01	request data
0x02	set I/O
0x03	read I/O
0x04	command to adapter
0x05	Send string to adapter
0x06	get available data
0x0A	Get Version Info
0x0E	config I/O
0x0F	config datamask

4.1.2 Vehicle Data Identifier Overview

Messages coming from the adapter

ID (Hex)	*Description*
0x10	ignition
0x11	gearshift
0x12	illumination
0x13	VIN (vehicle identification number)
0x14	speed
0x15	rpm
0x16	steering wheel buttons
0x17	time
0x18	fuel level
0x19	mileage in km
0x1A	date
0x1B	illumination 2

ID (Hex)	*Description*
0x1C	warning illumination
0x1D	warning vehicle
0x1E	warning message
0x1F	service interval
0x20	adapter status
0x21	Battery voltage
0x22	External Temperature
0x23	Enginewater Temperature
0x24	Fuel Level 2
0x25	Send String
0x26	Window Status
0x27	Locking System Status
0xFE	Test Message
0xFF	Error Message

4.2 Commands to the Interface

4.2.1 ID 0x01 - Request data

Description:

Request particular data from the vehicle.

Respond from the CAN-Interface: see above (4.2.x)

If an requested data isn't availabile, the error message 0xFF (4.3.26) with the requested ID as databyte is sended as respond.

Data: 2 Bytes

byte	*value*	*description*
0	0x01	command: request data
1	0xXX	id from the data you request

Example:

description	*sendframe*
request: illumination	[STX][0x01][0x12][0x13][ETX]
respond: if data is available, upper beam headlights	[STX][0x10][0x03][0x13][ETX]
respond: if data isn't available	[STX][0xFF][0x10][0xEF][ETX]

On the RS232 line you see the frame [STX][0x01][0x12][0x13][ETX] as follows: [STX]011213[ETX]

4.2.2 ID 0x02 – Set I/O

!!! not implemented yet !!!

Descripiton:

set I/O ports, ON = 12V (0xFF), OFF = 0V (0x00)

Data:

5 Bytes

byte#	*value*	*description*
0	0x02	STX
1	0x02	command: set I/O
2	port	select I/O port (0x00 ... 0x03)
3	state	state: 0x00 = OFF (0V), 0xFF = ON (12V)

byte#	value	description
4	crc	checksum = byte#1 ^ byte#2 ^ byte#3
5	0x03	ETX

Example:

description	sendframe
set I/O #2 to 12V	[STX][0x02][0x01][0xFF][crc][ETX]

4.2.3 ID 0x03 – Read I/O

!!! not implemented yet !!!

Descripiton:

read I/O ports

databyte from respond: 0x00 low level (0V), 0xFF high level (12V)

Data:

1 Byte

Request:

byte#	value	description
0	0x02	STX
1	0x03	command: read I/O
2	port	select I/O port (0x00 ... 0x03)
3	crc	checksum = byte#1 ^ byte#2
4	0x03	ETX

Sending:

- on request

4.2.4 ID 0x04 – Commands to Adapter

Descripiton:

send a command to the adapter

Data:

1 Byte

value	meaning	description
0x00	enter shutdown	send adapter into shutdown to save current. Only possible when can is not active. Adapter awakes with can bus activity
0x01	unlock central locking	unlock the central locking system of the vehicle
0x02	lock central locking	lock the central locking system of the vehicle
0x03 .. 0xFF		reserved

response: [STX] 0x04 0x00 [CRC] [ETX] (ok)

response: [STX] 0x04 0x01 [CRC] [ETX] (not ok)

4.2.5 ID 0x05 – Send String to Adapter

!!! not implemented yet !!!

Descripiton:

send a string to the adapter

Data:

size	meaning	description
2	String type	Type of string 0: string represents the version of the sending device 1: tbd.
n	string	String of given type, null - terminated

Example: send „1.2.3\0“ to adapter as version string:

0x02 0x30 0x35 0x00 0x00 0x00 0x00 0x30 0x31 0x2E 0x30 0x32 0x2E 0x30 0x33 0x30 0x30 0x?? 0x?? 0x03

STX „05“ „0“ „1.2.3“ CRC ETX

STX	CMD	StrType	String	CRC	ETX
0x02	0x30 0x35	0x00 0x00 0x00 0x00	0x30 0x31 0x2E 0x30 0x32 0x2E 0x30 0x33 0x30 0x30	0x?? 0x??	0x03

note:

All the payload is always transmitted as pure ascii chars, where the high and low nibble of each byte of the payload (= command, data and crc) is converted to a separate ascii char in the range of „0“ to „9“ and „a“ to „f“ resp. „A“ to „F“.

Sending:

- on Event

4.2.6 ID 0x06 – Get available Data

!!! not implemented yet !!!

Description:

ask the adapter for data which are really available from the vehicle

Data:

no data

Response:

4 Data Bytes with the same Layout like command „0x0F – config sendmask“. Each Bit which is set to 1 means, that the according information is available in the current setup.

Hint:

The adapter stores the information, if a certain information was available on the CAN Bus on the last runtime. If it was available, the accordingly bit will be set in the response. It may happen, that the actual transmitted value for a certain information is a old stored value from the last time the adapter was active. The value will be updated as soon a new information was available from the vehicle.

4.2.7 ID 0x0E - Config I/O

!!! not implemented yet !!!

Description:

config I/O ports to input or output
BITn = 0, input (default)
BITn = 1, output

Data:

1 Byte

byte#	***value***	***description***
0	mask	e.g. 0x0A, port#3 and #1 are set to outputs

byte 0 - BITn							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
-	-	-	-	I/O #3	I/O #2	I/O #1	I/O #0

4.2.8 ID 0x0A – Version Info

Description:

When this ID is send, the interface will send an ASCII string containing the revision of the protocol and the implemented software version.

Example:

R13 SWV1 means R13 of the protocol and V1 of the implemented Software

4.2.9 ID 0x0F - Config Sendmask

Description:

If the bit is set to one, the corresponding data from the vehicle will be sent periodically or when the state changes.

BITn = 1, enable sending

BITn = 0, disable sending (default)

To configurate the sendmask successfully you've to send all 28 databytes. Unused Bits/Bytes configure with zero.

As answer you'll receive the same message with edit data bytes, if a bit is set to zero the requested data isn't available, but will be send as soon as its available. If config message and received answer are eqal all data are avialable.

Data:

28 Bytes

byte#	***description***
n	maskbytes

Byte 0: Bit 0 .. Bit 7							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
TIME	LKF	RPM	SPEED	FIN	ILLUMINATION	GEARSHIFT	IGNITION

Byte 1 – Bit 8 .. Bit 15							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
SERVICE INTERVAL 0x1F	WARNING MESSAGE 0x1E	WARNING VEHICLE 0x1D	WARNING ILLUMINATION 0x1C	ILLUMINATION2 0x1B	DATE	MILEAGE	FUELLEVEL

Byte 2 - Bit 16 .. Bit 23							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
Locking System Status 0x27	Window Status 0x26	send string 0x25	Fuel Tank Level 2 0x24	Engine Coolant Water Temp. 0x23	Evironmental Temprature 0x22	Battery Voltage 0x21	ADAPTER STATUS 0x20

! reserved ! Byte 3 - Bit 24 .. Bit 31							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
-	-	-	-	-	-	-	-

Respond:

description	***Send format***
error, false config message	[STX][0xFF][0x0F][0xF0][ETX]
configuration successfully	[STX][0x0F][0x0F][ETX]

4.3 Vehicle Data Messages

4.3.1 ID 0x10 - Ignition

Description:

state of the ignition / position of the key

Data: 1 Byte

Value Byte 0 (0x..)	***Description***
00	no key in lock
01	key in lock
02	1st position
03	2nd position
04	start engine

Sending:

- on request
- when state changes

4.3.2 ID 0x11 - Gearshift

Description:

Information about the current gear

Data: 1 Byte

Value Byte 0	***Description***
0x00	P *)
0x01	reverse gear
0x02	N *)
0x03	D *)
0x04	S **)
0x05	tiptronik activated **)
0xFF	no gear available

*) only for vehicles with automatic gearbox
**) if available at this vehicle

Sending:

- on request
- when state changes

4.3.3 ID 0x12 - Illumination

Description:

Information about activated headlights

Daten: 1 Byte

Value byte0	*description*
0x00	lights off
0x01	parking light
0x02	driving lights
0x03	upper beam headlights

Sending:

- on request
- when state changes

4.3.4 ID 0x13 - VIN

Description:

Vehicle identification number. It is possible that the VIN is not

Data: n Byte (17)

value byte#	*description*
0 .. n	VIN in ASCII

Sending:

- on request
- every 5000ms

4.3.5 ID 0x14 - Speed

Description:

Vehicle speed in km/h

Data: 2 Byte

byte#	description
0	high byte
1	low byte

speed [km/h] = Byte0Byte1 / 128

Sending:

- on request
- when state changes, max. every 100ms

4.3.6 ID 0x15 - RPM

Description:

rounds per minute of the engine

Data: 2 Byte

byte#	description
0	high byte
1	low byte

RPM = Byte0Byte1 / 4

Sending:

- on request
- when state changes, max. every 100ms

4.3.7 ID 0x16 - Steering Wheel Buttons

Description:

Indicates pushed steering wheel buttons. If no key is pushed, all bits are zero.
Reserved bits are always initialized with zero.

Data: 4 Bytes

byte#	description
4	data bytes

byte 0 - BITn							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***

byte 0 - BITn							
SKIP DOWN	SKIP UP	SKIP	DOWN	UP	VOLUME -	VOLUME +	VOLUME

byte 1 - BITn							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
MENU BACKWARD	MENU FORWARD	MODE	MUTE	SOURCE	END CALL	ANSWER CALL	PTT

! reserved ! Byte 2 - 3 BITn							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***
-	-	-	-	-	-	-	-

Sending:

when state changes, max. every 100ms

4.3.8 ID 0x17 - Time

Description:

time of the vehicle´s clock

Data: 3 Bytes

byte#	***Range (Hex)***	***Range (Dez)***	***Description***
0	0x00 .. 0x17	0 .. 23	hour
1	0x00 .. 0x3B	0 .. 59	minute
2	0x00 .. 0x3B	0 .. 59	second

Sending:

- on request
- when state changes, every 1000ms

4.3.9 ID 0x18 - Fuel Tank Level (deprecated, use Fuel Tank Level 2 instead!)

Description:

Fuel level in litre

Data: 1 Byte

byte#	Range	Description
0	0x00 .. 0xFF	fuel level in litre

Sending:

- on request
- when state changes

4.3.10 ID 0x19 - Mileage

Description:

milage in kilometre

Data: 4 Bytes

byte#	description
0	high Byte
1	
2	
3	low Byte

Mileage [km] = Byte0Byte1Byte2Byte3 / 100

Sending:

- on request
- every 5000ms

4.3.11 ID 0x1A - Date

Description:

Current date as shown in the vehicle display. The decimal values are transmitted as hexadecimal payload.

Data: 3 Bytes

byte#	Range (Hex)	Range (Dec)	description
0	0x01 .. 1F	1 .. 31	day
1	0x01 .. 0x0C	1 .. 12	month
2	0x00 - 0xFF	-	first byte of year data
3	0x00 - 0xFF	-	last byte of year data

e.g. 20nd November 2014 you'll receive the (hex) payload: 14 0B 07DE

Sending:

- on request
- when state changes, every 5000ms

4.3.12 ID 0x1B – Illumination 2

Description:

additional information about the status of the vehicle lamps

Data: 8 Bytes

For each information we have 2 Bits: 00: Lamp off, 01: Lamp on, 10: reserved, 11: signal not available

Byte 0: High Byte, Byte 7: low Byte

byte 0 - BIT 56..63							
[63]	***[62]***	***[61]***	***[60]***	***[59]***	***[58]***	***[57]***	***[56]***

...

byte 7- BIT 0..7							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***

Bits	***Signal***		***Bits***	***Signal***
0, 1	parking light		32, 33	Tagfahrlicht
2, 3	low beam front		34, 35	reserved
4, 5	low beam rear		36, 37	reserved
6, 7	high beam		38, 39	reserved
8, 9	fog light (front)		40, 41	reserved
10, 11	fog light (rear)		42, 43	reserved
12, 13	flasher left		44, 45	reserved
14, 15	flasher right		46, 47	reserved
16, 17	hazard flasher		48, 49	reserved
18, 19	reverse light		50, 51	reserved
20, 21	brake light		52, 53	reserved
22, 23	parking light left		54, 55	reserved
24, 25	parking light right		56, 57	reserved
26, 27	trailer flasher left		58, 59	reserved
28, 29	trailer flasher right		60, 61	reserved
30, 31	trailer flasher		62, 63	reserved

Sending:

- on request
- when state changes

4.3.13 ID 0x1C – Illumination Errors

Description:

Indicates defect lamps of the vehicle (if the information is available on CAN)

Data: 8 Bytes

For each information we have 2 Bits: 00: Warning not active, 01: Warning active, 10: reserved, 11: signal not available

Byte 0: High Byte, Byte 7: low Byte

byte 0 - BIT 56..63							
[63]	***[62]***	***[61]***	***[60]***	***[59]***	***[58]***	***[57]***	***[56]***

...

byte 7- BIT 0..7							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***

Bits	***Signal***		***Bits***	***Signal***
0, 1	parking front left		32, 33	brake right
2, 3	parking front right		34, 35	brake middle
4, 5	low beam left		36, 37	brake trailer
6, 7	low beam right		38, 39	fog front left
8, 9	high beam left		40, 41	fog front right
10, 11	high beam right		42, 43	fog rear
12, 13	driving light rear left		44, 45	reverse left
14, 15	driving light rear right		46, 47	reverse right
16, 17	flasher front left		48, 49	license plate
18, 19	flasher front left car wing / mirror		50, 51	Additional high beam left
20, 21	flasher front right car wing / mirror		52, 53	Additional high beam right
22, 23	flasher rear left		54, 55	flasher front right
24, 25	flasher rear right		56, 57	reserved
26, 27	flasher trailer		58, 59	reserved
28, 29	hazard flasher		60, 61	reserved

Bits	Signal		Bits	Signal
30, 31	brake left		62, 63	reserved

Sending:

- on request
- when state changes

4.3.14 ID 0x1D – Vehicle Warnings

Description:

Warning lamps and other warnings of the vehicle

Data: 8 Bytes

For each information we have 2 Bits: 00: Warning not active, 01: Warning active, 10: reserved, 11: signal not available

Byte 0: High Byte, Byte 7: low Byte

byte 0 - BIT 56..63							
[63]	***[62]***	***[61]***	***[60]***	***[59]***	***[58]***	***[57]***	***[56]***

...

byte 7- BIT 0..7							
[7]	***[6]***	***[5]***	***[4]***	***[3]***	***[2]***	***[1]***	***[0]***

Bits	Signal		Bits	Signal
0, 1	Oil pressure		32, 33	parking brake
2, 3	Oil level low		34, 35	fuel low
4, 5	Engine error		36, 37	tire pressure low front left
6, 7	wiper water low		38, 39	tire pressure low front right
8, 9	brake		40, 41	tire pressure low rear left
10, 11	electric generator		42, 43	tire pressure low rear right
12, 13	door open driver (left)		44, 45	tire pressure general
14, 15	door open codriver (right)		46, 47	light failure
16, 17	door open rear driver side		48, 49	ASR/ESP
18, 19	door open rear codriver side		50, 51	Seat belt driver
20, 21	door open hood		52, 53	Seat belt codriver
22, 23	door open trunk		54, 55	Glow Plug

Bits	Signal		Bits	Signal
24, 25	fuel tank cap		56, 57	Enginewater Overtemperature
26, 27	Airbag driver		58, 59	Rear Window open
28, 29	Airbag codriver		60, 61	reserved
30, 31	Airbag general		62, 63	reserved

Sending:

- on request
- when state changes

4.3.15 ID 0x1E – Warning message (String)

!!! not implemented yet !!!

Description:

Error messages as shown in the cockpit

Data: x Bytes

Byte#	Value	Description
0	length	length of string without trailing NULL
1 .. length	ascii	message to the user
length + 1	0	termination of string (NULL)

4.3.16 ID 0x1F – Service Interval

Description:

service interval of the vehicle to the next service

Data: 12 Bytes

Byte#	Value	Description	
		value	type
	type of service	0	General inspection
		1	Emission inspection
		2	Oil Service
		3	Brake Fluid
		4	Brake pads Front
		5	Brake pads Rear
0...1		6...0xff	reserved

<table>
<tr><th>Byte#</th><th>Value</th><th>Description</th></tr>
<tr><td>2..5</td><td>Distance till service</td><td>0x00..0xfffffffe distance till service
0xffffffff invalid, value not provided</td></tr>
<tr><td>6..9</td><td>Timestamp to Service OR
Time till service,
depends on time unit of Byte#11</td><td>
<table>
<tr><th>byte#</th><th>intervaltype</th><th></th></tr>
<tr><td>0</td><td>1 .. 31</td><td>Day, (0xff invalid)</td></tr>
<tr><td>1</td><td>1 .. 12</td><td>Month (0xff invalid)</td></tr>
<tr><td>2</td><td>0 .. 99</td><td>first digits of year, e.g. 20</td></tr>
<tr><td>3</td><td>0 .. 99</td><td>last digits of year, e.g. 07</td></tr>
</table>
OR

0x00..0xfffffffe time till service
0xffffffff invalid, value not provided
</td></tr>
<tr><td>10</td><td>Sign (till / over)</td><td>
<table>
<tr><th>value</th><th>type</th></tr>
<tr><td>0</td><td>till service (+)</td></tr>
<tr><td>1</td><td>over service (-)</td></tr>
</table>
</td></tr>
<tr><td>11</td><td>Bit 4...7 unit distance

Bit 0..3 unit time</td><td>Unit distance: 0 – kilometers
1 – miles
2...0xE – reserved
0xF – invalid

Unit time: 0 – time till service in days
1 – timestamp to service
2...0xE – reserved
0xF – invalid</td></tr>
<tr><td></td><td></td><td></td></tr>
</table>

Sending:

- on request: when requested, all available services are sent with 100ms delay
- when state changes

4.3.17 ID 0x20 – Adapter Status

Description:

Provides information about the current adapter status

Data: 4 Bytes

Byte	Bit	Signal / Information	description
0	24..31		reserved
1	16..23		reserved
2	8..15		reserved
3	1..7		reserved
3	0	CAN Bus Activity	if set to 1, can bus is active, else can bus is fallen asleep or not connected

Sending:

- on request
- when state changes

4.3.18 ID 0x21 – Battery Voltage

Description:

Battery voltage in Volts (0.1V steps)

Data: 4 Bytes

Byte	description
0	High byte
1	Low byte

Voltage [V] = Byte0Byte1/ 10;

Sending:

- on request
- when state changes

4.3.19 ID 0x22 – External Temperature

Description:

external temperature in °C

Data: 1 Bytes

Byte	value	description
0	-39.5°C … 87.5°C	External Temperature in °C

Temperature [°C] = (Byte0 / 2) – 40;

Sending:

- on request
- when state changes

4.3.20 ID 0x23 – Enginewater Temperature

Description:

Enginewater temperatur in °C

Data: 4 Bytes

Byte	*Range*	*Description*
0	-40°C … 215°C	Enginewater Temperature in °C

Temperature [°C] = Byte0 - 40

Sending:

- on request
- when state changes

4.3.21 ID 0x24 – Fuel Tank Level 2

Description:

tank level and unit

Data: 3 Byte

<table>
<tr><th>Byte#</th><th>Value</th><th>Description</th></tr>
<tr><td>0 ... 1</td><td>0x0000 .. 0xFFFF</td><td>tank level</td></tr>
<tr><td>2</td><td>unit</td><td>Unit value
<table>
<tr><th>type</th><th>interpretation of data</th></tr>
<tr><td>0</td><td>Tank level in liter</td></tr>
<tr><td>1</td><td>Tank level in gallons</td></tr>
<tr><td>2</td><td>Tank level in percent (%)</td></tr>
<tr><td>3..0xFF</td><td>reserved</td></tr>
</table></td></tr>
</table>

Sending:

- on request
- when state changes

4.3.22 ID 0x25 – Send String out from Adapter

!!! not implemented yet !!!

Same as command ID 0x05 (4.2.5), but to the other direction.

4.3.23 ID 0x26 – Window Status

Description:

Provides information about the status of each window of the vehicle (if available on CAN).

The first 8 Bytes are for the window position, while we have 1 Byte for each window:

Byte 0-Byte7	***Description***
0x00..0x64	Window Position in Percent
0xFF	invalid Data

Additionally we have 4 Bits for the status of each window:

(Byte 8-11) 4 Bits	***Description***
0000	window closed and not moving
0001	window open and not moving
001x	window moving up/closing
010x	window moving down/opening
1111	no information available

Data: 12 Bytes

Byte 0: High Byte, Byte 11: low Byte

Byte#	***Description***
0	Reserved
1	Top position (only convertible)
2	Headliner position
3	Sunroof position
4	Rear Passenger window position
5	Rear driver window position

Byte#	Description
6	Passenger window position
7	Driver window position

Byte 8 - BIT 24..31							
[31]	[30]	[29]	[28]	[27]	[26]	[25]	[24]

...

Byte 11 - BIT 0..7							
[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

Bits	Signal		Bits	Signal
0 .. 3	Driver window		16 .. 19	Sunroof
4 .. 7	Passenger window		20 .. 23	Headliner
8 .. 11	Rear driver window		24 .. 27	Top (only convertible)
12 .. 15	Rear passenger window		28 .. 31	Reserved

Sending:

- on request
- when state changes

4.3.24 ID 0x27 – Locking System Status

Description:

Provides information about the status of the locking system

For each window we have 4 Bits:

2 Bits	Description
00	Door locked
01	Door unlocked
10	Reserved
11	Invalid

Data: 3 Bytes

Byte 0: High Byte, Byte 2: low Byte

byte 0 - BIT 16..23							
[23]	[22]	[21]	[20]	[19]	[18]	[17]	[16]

...

byte 2- BIT 0..7							
[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

Bits	Signal		Bits	Signal
0, 1	All Doors (vehicle completely locked/not completely locked)		12, 13	Reserved
2, 3	Driver door locked/unlocked		14, 15	Reserved
4, 5	Passenger door locked/unlocked		16, 17	Reserved
6, 7	Rear driver door locked/unlocked		18, 19	Reserved
8, 9	Rear passenger door locked/unlocked		20, 21	Reserved
10, 11	Hatchdoor locked/unlocked		22, 23	Reserved

Sending:

- on request
- when state changes

4.3.25 ID 0xFE – Test Message

!!! not implemented yet !!!

Description:

Will send a defined test message if requested

Data: 4 Bytes

Byte#	Value	Description
0	0x35	-
1	0xA7	-
2	0x21	-
3	0x5E	-

Sending:

on request

After being requested, you will receive: 2735A7215E[crc]

4.3.26 ID 0xFF – Error Messages

Description:

Will send an error message with the corresponding ID, if certain data was requested, but is not available.

Data: 1 Bytes

Byte#	*Value*	*Description*
0	0x35	ID of the requested data

Sending:

- In case of Error, if requested Data is unavailable

5 Receiving Data - Example

Blockframe for communication between the CAN-Interface and Uart like in 3.1.2 defined.

In this example the time is requested. Therefore, the command for requesting (0x01, see 4.2.1) has to be sent, followed by the ID for the time (0x17, see 4.3.8) and the checksum (0x16) as calculated in 3.1.4.

So the transmitted message is: 0x01 0x17 0x16

Since the nibbles have to be transmitted in ASCII, the message on the physical layer is received as:

0x02	0x30	0x31	0x31	0x37	0x31	0x36	0x03
[STX]	0	1	1	7	1	6	[ETX]

The answer on the physical layer is in this example:

0x02	0x31	0x37	0x30	0x39	0x33	0x30	0x31	0x34	0x33	0x41	0x03
[STX]	1	7	0	9	3	0	1	4	3	A	[ETX]

So we got

0x17 0x09 0x30 0x14 0x3A

as answer, with 0x17 as ID, 0x09 0x30 0x14 as databytes and 0x3A as the checksum.

The time is given as the databytes, transformed to decimal values:

09:48:20 (hh:mm:ss)