Software Interface Description

Contents

Change index template CI 04.02

2008. 12. 17

Template valid as of

1 Introduction

1.1 Abbreviations and Definitions

Abbreviation, term	Description
Change index CI	Change index (Version of a module) e.g. as a result of modifications to components or layout (the predecessor version is usually no longer produced after a new CI has been released) (format: ##.##)
COB	Communication OBject In a CAN2.0A system, there are 2048 different COBs that consist of 0 to 8 data bytes.
COB-ID	Communication OBject IDentifier
LSS	Layer Setting Services – CANOpen protocol for configuring slave modules
NMT	Network Management – CANOpen protocol for monitoring CAN nodes
Node ID	Node number of CANOpen nodes
PDO	Process Data Object – CANOpen designation for event-driven and broadcast messages
Product code	R&S internal designation for all items which are manufactured by R&S or required for this and which can be ordered internally (format: ####.####)
RTR	Remote Transmission Request – A CAN message (without data section) used for requesting a PDO.
SDO	Service Data Object – CANOpen protocol for the point-to-point transfer of large quantities of data
Variant	A version of a hardware item, e.g. component variants or documentation in various languages (format: ## and added to the end of the product code, usually separated by a period symbol)
Vendor ID	Code that describes a specific manufacturer (CANOpen)

1.2 Revision List

Version	Date	Author	Comment
01.00	16.03.18	Hoenselaar	Created

1.3 Storage Location of This File

This file is located in the RTC component th__upc under \upc_doc\SDD_TX9-PA.

2 CAN Bus

2.1 CAN/CANOpen – Basics

With the CAN bus, it is necessary to make a distinction between CAN and CANOpen. CAN describes the bus as such, and CANOpen describes a higher-level protocol which is optimized for CAN.

The used baud rate is 125 kbits/s

The key points of CAN are:

- A CAN message essentially consists of the following:
  - an 11 bit (CAN2.0A) or 29 bit (CAN2.0B) ID = COB-ID (**C**ommunication **Ob**ject **Id**entifier),
  - length information,
  - 0 to 8 data bytes and
  - a CRC.

  CAN2.0A and B can always be used simultaneously on the bus; however, CANOpen uses only CAN2.0A.
- CAN is always multimaster-capable (i.e. every bus subscriber can send messages at any time of its own accord).
- CAN is possible on all physical media that enable a dominant and a recessive level (dominant takes precedence over recessive). Example: CAN via optical fiber: As soon as a subscriber introduces light into the optical fiber (dominant), it is bright in the optical fiber, irrespective of how many other subscribers introduce light or not.
- Every CAN bus subscriber also checks the current level on the bus during the transmission phase and verifies whether the transmit bit and receive bit match.
- There is an arbitration phase at the beginning of a message which primarily uses the ID. Every subscriber that started to send a message coincidentally at the same point in time, applies dominant and recessive levels one after the other on the bus. A module that applies a recessive level and sees a dominant level ends transmission and switches over to receive mode. This means: the ID contains a priority (the message with the most dominant levels wins) and, at the end of arbitration, there is only allowed to be one node that is in transmit mode.
- Owing to various fault mechanisms, a message is received either by all connected subscribers or by no subscribers.

Structure of a CAN message:

SOF (StartOfFrame)	Arbitration (ID)	Control field	Data field	CRC	ACK	EOF (EndOfFrame)
1 bit	12 or 32 bits	6 bits	0 to 8 bytes	16 bits	2 bits	7 bits

In addition, so-called stuffing bits can occur which increase the length of the message and are used to synchronize the bus subscribers.

Mechanisms of CANOpen:

- Node numbers (**node IDs**): Every module is assigned a unique node ID. The node ID is added to basic COB-IDs in order to permit unambiguous assignment of messages to modules.
  Example: The heartbeat has the basic ID 0x700. The node with node ID 5 sends a heartbeat on COB-ID 0x705.
- **LSS** (Layer Setting Services): Mechanism used to transfer a node ID to a node that has not yet been assigned a node ID, or to change the transmission rate. The product code, serial number, variant, change index of the module and vendor ID are used to ensure that there is no ambiguousness.
  - Vendor ID: The vendor ID is set to 0x04 for all modules
  - Product ID: Product code ofPMU905 the module (**with PMU905 via ACB: 2109.0959**)
  - Revision number: Change index of the module
  - Serial number: Serial number + (variant * 1000000) of the module
- **Note: The CAN serial number is a number used for unambiguous identification of a module. The described format is the same regardless of how the serial number and variant of the module are stored/processed internally.**

- **PDO** (Process Data Object): Normal transmit and receive messages of a module. It consists of the COB-ID and 0 to 8 data bytes and can be transmitted when certain events occur, and can also be sent universally as a broadcast. PDOs are always unconfirmed. A COB-ID is required for every PDO. Example: Status messages.
- **SDO** (Service Data Object): Information that rarely occurs or requires more than 8 data bytes can be transmitted using SDOs. For this purpose, an SDO channel (a related pair of COB-IDs) is opened between a master and a slave (1:1 connection). The master uses a COB-ID to request a specific data item which the master wants to write or read, and the slave replies accordingly. The alternating transmission is repeated until all data of the SDO object has been sent (the channel is then closed and can be opened again for the next transmission). With each SDO, 16 million different objects of unlimited length can be addressed on each module.
  Example: Type plate or module software update.
  In the case of the SDO transfer, a distinction is generally made between:
  - Segmented transfer: Every CAN message, which can consist of max. 8 data bytes, is confirmed by the receiving station.
  - Block transfer: Only the end of a large block of CAN messages is confirmed by the receiving station.
  - Expedited SDO A shortened protocol is provided for messages with 0 to 4 bytes.
- **NMT** (Network Management): This includes messages such as ResetNode, ResetCommunication or Heartbeat. A master can use this mechanism to perform basic control and monitoring tasks.

2.2 Allocation of COB-IDs and Node IDs:

The allocation of the COB-IDs is based heavily on the standard for CANOpen. One difference is, for example, that PDO pairs 3 and 4 from the standard are used as SDO channels 2 and 3. This allows the data of the modules to be accessed by several masters at the same time without the masters having to know anything about the others. Additionally, PDOs are defined at the address 0x100+NodeID and 0x680+NodeID. All PDOs can be freely used as transmit or receive PDOs.

The structure is such that up to 127 nodes (node IDs 1 to 127) are possible on one physical CAN bus. This is a theoretical value that is intended to make the assignment of node numbers and modules easier. In practical terms, the used CAN transceiver modules reach the limits of their driver performance here; a maximum number of approx. 100 modules on one CAN bus should therefore not be exceeded. If possible, modules always leave the production facility unconfigured and do not have any node numbers assigned.

The node IDs are allocated as follows:

Min.	Max.	Number of reserved node IDs	Max. number simultaneously on one CAN bus	Module
1	29	29	24	Amplifier (PMU905)
30	49	20		
50	59	10		
60	69	10		
70	99			
100	104	5		
105	110	6		
111	120			
121	121	1		
122	122	1		
123	123	1		
124	127	4		

On the one hand, fixed allocation of the node IDs makes debugging easier and, on the other hand, helps maintain an overview if there are several master modules in the system (see also 3.1.3).

Communication object	COB-ID(s) **hex**	Slave nodes
NMT node control	000	Receive only
Sync	080	Receive only
Emergency	080 + NodeID	Transmit
TimeStamp	100	Receive only
PDO	100 + NodeID	4th transmit PDO
PDO	180 + NodeID 200 + NodeID 280 + NodeID 300 + NodeID	1st transmit PDO 1st receive PDO 2nd transmit PDO 2nd receive PDO
SDO 3	380 + NodeID 400 + NodeID	3rd transmit 3rd receive
SDO 2	480 + NodeID 500 + NodeID	2nd transmit 2nd receive
SDO 1	580 + NodeID 600 + NodeID	1st transmit 1st receive
PDO	680 + NodeID	3rd transmit PDO
NMT node monitoring (node guarding/heartbeat)	700 + NodeID	Transmit
PDO	7C0 – 7CF	Transmit (reserved for service mode for amplifier 1)
PDO	7D0 – 7DF	Transmit (reserved for service mode for amplifier 2)
LSS	7E4 7E5	Transmit Receive

Table 1 – COB-ID allocation

	COP-ID	Richtung	Verstärker
PDO	100 + NodeID	Transmit	NoBasicConfig
	180 + NodeID	Transmit	Baugruppen Status (von BG an den CAN-Master)
	200	Receive	BasicConfiguration (gesendet von CAN-Master)*
	300	Receive	Reset
SDO 1	580 + NodeID	Transmit	verwendet von Exciter A
	600 + NodeID	Receive	

Table 2 – Use of PDO and SDO on PMU905

2.3 Multimaster Operation under CANOpen

Genuine multimaster operation (several CANOpen masters negotiating a mastership with each other via the CAN bus) is not supported by the system. However, multiple SDO channels which are independent of each other are used so that several master modules can be present in the system at the same time (Drive9-SP A/B and Ctrl). The channel to be used by the master depends on the function of the master (SP-A/SP-B/Control).

SDO1 is used by SP-A,
SDO2 is used by SP-B,
SDO3 is used by control.

The channels are each implemented in such a way that they can be used independently of each other (even if, for example, data ranges such as the electronic type plate are retrieved from the EEProm).

2.4 Persistent Node Numbers (Node IDs)

The modules keep their node IDs in a fail-safe memory and, following a reboot, register themselves on the bus using this node number and corresponding heartbeat messages. If a module receives its own heartbeat from the CAN bus (e.g. because components have been replaced), it resets its node ID in order to avoid conflicts, and then responds to the LSS query as to whether unconfigured nodes are present in the system, whereupon they are located and configured again by the system.

In the EEPROM and RAM on the modules is a memory location at which the node number is stored. Both values can differ, e.g. in cases where "Reset Communication" has been sent to a component. Only valid node IDs (1 to 127) are stored in the EEPROM so that, following a reset, a module always contains a valid node ID as the initial value. The node ID in the EEPROM can only be set to an otherwise invalid value (e.g. 0xFF = unconfigured) by means of an SDO.

Example: A module currently has the node ID 5 (stored in EEPROM and RAM) and receives the message "ResetCommunication". This causes communication to be initialized and the node ID in the RAM to be set to 0xFF; however, the memory location in the EEPROM remains at 5. If a power failure occurs, the module restarts with the node ID from the EEPROM (5). If a new (valid) node ID is assigned, this is entered both in the EEPROM and in the RAM. In production, the node ID in the EEPROM should be set to 0xFF following the test.

2.5 LSS (Layer Setting Services)

LSS is used in the system for node number assignment. The two messages 0x7E5 (from the master to the slave) and 0x7E4 (from the slave to the master) are used for LSS. A master scans the system for unconfigured nodes at an interval of ≤ 5 s.
(**Note:** This message is simultaneously interpreted by the slaves as a heartbeat from the LSS master.)

3 Execution Scenarios

3.1 System Switch-On

3.1.1 Initialization of Amplifiers

Since after a restart it is essential that amplifiers undergo basic configuration (ON/OFF, peak/AV, reference, DC voltage, etc.) before they are allowed to deliver power, it is necessary for the amplifiers to be able to receive this configuration irrespective of the assignment of node numbers. The following occurs so that this can be performed quickly:

- An amplifier sends (as soon as it is receive-ready on the CAN bus) a broadcast message (without data) in order to signal that it is not yet configured. It repeats this until it receives a corresponding message (see section 5.6).
- After receiving this message, the master that is connected to the amplifier places the corresponding broadcast message with the basic setup on the bus (see section 0).

3.1.2 Finding CAN Modules

CAN modules in the system can either be in the "unconfigured" state or already have a persistent node number and be in the "operational" state.

- Unconfigured modules can be found by means of LSS and set to the "operational" state.
- Configured modules in the "operational" state register themselves on the bus by means of a heartbeat (every 2000 ms). A master can then clearly identify the module by reading out the type plate and integrate this module in the system.

Note: A module that remains in the intermediate state between "unconfigured" and "operational" (e.g. in the "configured" or "preoperational" state) for longer than 30 s, resets itself to the "unconfigured" state and can then be found again by means of LSS.

3.1.3 Conflicts of Modules with Identical Node Number

It is possible, particularly when starting up a transmitter or when replacing components such as amplifiers or sample ports, that (owing to the persistent storage) node IDs are present multiple times in the system; this must be avoided at all costs as otherwise conflicts and problems will occur on the CAN bus.

In order to resolve this situation, each module monitors whether its own heartbeat is being sent from another module on the CAN bus and, if this is the case, it triggers the following actions:

The node resets its own state to "unconfigured", thereby also resetting the node ID. It then sends the message "ResetCommunication NodeID xy" in order to force all other nodes which have this node ID and which are connected to the bus, to be set to the "unconfigured" state. The LSS routine will now again find the nodes that were already found in the system, and assign the known node IDs. The newly added module is assigned a new node ID.

3.2 Connection of Amplifier to Power Supply

After the amplifier is connected to the power supply, it is able to communicate via the CAN bus.

The amplifier can have a CAN node ID or be in an unconfigured state. In the case of new amplifiers, the unconfigured state is the delivery state. If the amplifier has already been put into operation once before and a node ID was assigned by a CAN master, then the amplifier has stored this node ID permanently and therefore has a CAN node ID.

The amplifier behaves in the following way:

CAN node ID present:
The amplifier sends a heartbeat under cyclic (0x700+NodeID) and its status PDO under (0x180 + NodeID). If the amplifier detects its own CAN node ID (output by another subscriber) on the CAN bus, the amplifier automatically resets to the unconfigured state and sends the CAN command "Reset Communication" on the CAN bus to the other subscriber with the same node ID in order to also set this subscriber to the unconfigured state.

Unconfigured state (CAN node ID = 0xff):
The amplifier does not send any CAN messages because it does not have a CAN node ID. The amplifier responds to the CAN query "are there any unconfigured subscribers?". By means of the CAN LSS commands, the amplifier can then be assigned a CAN node ID by the master.

The amplifier continuously monitors the CAN bus with respect to LSS commands. LSS commands that are absent over a long period of time are interpreted by the amplifier as a missing master and are indicated by means of an LED (see 5.3 Status LED).

3.3 Setting / Changing CAN Node ID

Each amplifier has a unique identifier (vendor ID, product code, revision no., serial no.).
This identifier can be read out via the LSS protocol.

Examples:

//are there any unconfigured subscribers?
0x7E5 4C 00 00 00 00 00 00 00
// send vendor id, reply received on 0x7E4
0x7E5 5A 00 00 00 00 00 00 00
// address one specific amplifier
0x7E5 41 xx xx ..
0x7E5 42 xx xx .. etc.
// set new node ID to xx
0x7E5 11 xx 00 00 …
// store configuration
0x7E5 17 xx 00 00 …

// switch to operational mode
0x7E5 04 00 …

3.4 Reset Communication / "unconfigured" Mode

Each CAN module, i.e. also the amplifiers, checks the CAN node IDs present on the bus. Sending the "Reset Communication" command to the respective CAN module sets this module to the "unconfigured" state.
If a module reads its own node ID on the bus (output by another module), this module sets itself and the other module on the bus to the "unconfigured" state. The other module is reset by means of autonomous sending of the CAN command "Reset Communication". New CAN node IDs must be assigned by the master.

Important:
Resetting to the "unconfigured" mode after the CAN command "Rest Communication" has been received is a R&S implementation and not compliant with CANOpen.

Examples:

//send the subscriber with CAN node ID 0x64 Reset Communication?
0x000 82 64 00 00 00 00 00 00
//are there any unconfigured subscribers? 0x50 must appear on 0x7E4.
0x7E5 4C 00 00 00 00 00 00 00

3.5 LSS

Valid serial number range

The valid range for the LSS search is from 0 to 99999999.

Valid range for HW revision

The valid range for the LSS search is from 0 to 9999.

3.6 FW Update

Not described here

4 Type Declarations

4.1 Percentage Values

Values where the ratio is more informative than the absolute value are transferred as a percentage value. Percentage values can be represented as signed 8 bit values in % or as signed 16 bit values in 1/100 %.

	Number range	Value range	Comment
8 bit (signed)	–100 to +100	–100 % to +100 %	< –100 or > +100 is invalid
16 bit (signed)	–10000 to +10000	–100.00 – +100.00 %	> +10000 and < –10000 is invalid

4.2 Voltages

Voltages consist of 16 bit values and are transferred in mV. The top 2 bits stand for the exponent (10^x). It is therefore possible to represent voltages from 0.000 V to 16 kV.

15…exponent …14	13 …… bits ……0	Voltage
0	0 to 16383	0 to 16383 mV
1	1639 to 16383	16.39 to 163.83 V
2	1639 to 16383	163.9 to 1638.3 V
3	1639 to 16381	1639 V to 16381 V
3	16382	≥ 16382 V
3	16383 (0x3FFF)	Value invalid

4.3 Currents

Currents consist of 16 bit values and are transferred in mA. The top 2 bits stand for the exponent (10^x). It is therefore possible to represent currents from 0 mA to 16 kA.

15…exponent …14	13 …… bits ……0	Current
0	0 to 16383	0 to 16383 mA
1	1639 to 16383	16.39 to 163.83 A
2	1639 to 16383	163.9 to 1638.3 A
3	1639 to 16381	1639 A to 16381 A
3	16382	≥ 16382 A
3	16383 (0x3FFF)	Value invalid

4.4 Temperatures

4.4.1 8 Bit

Temperatures with low accuracy are transferred as signed 8 bit values in 1 K.

Number range	Value range
–126 to +126	–126 °C to +126 °C
–127	≤ –127 °C
+127	≥ +127 °C
–128	Value invalid

4.4.2 16 Bit

Temperatures with high accuracy are transferred as signed 16 bit values in 0.1 K.

Number range	Value range
–32766 to +32766	–3276.6 °C to +3276.6 °C

–32767 (0x8001)	≤ –3276.7 °C
+32767 (0x7FFF)	≥ +3276.7 °C
–32768 (0x8000)	Value invalid

4.5 Power

The **power values** consist of 16 bit values. The top 2 bits stand for the exponent:

$$0 \triangleq 10^0;\ 1 \triangleq 10^2;\ 2 \triangleq 10^4;\ 3 \triangleq 10^6$$

It is therefore possible to represent powers from 0.000 W to 16 MW.

15…exponent …14	13 …… bits ……0	Power
0	0 to 16383	0 to 16383 mW
1	164 to 16383	16.4 to 1638.3 W
2	164 to 16383	1640 to 163830 W
3	164 to 16381	164000 W to 16381000 W
3	16382	≥ 16381000 W
3	16383 (0x3FFF)	Value invalid

4.6 Power (dBm)

Power values in dBm are transferred as signed 16 bit values in 0.01 dBm.

It is therefore possible to represent powers from –327.66 dBm to +327.66 dBm.

Number range	Value range
–32766 to +32766	–327.66 dBm to +327.66 dBm
–32767 (0x8001)	≤ –327.67 dBm
+32767 (0x7FFF)	≥ +327.67 dBm
–32768 (0x8000)	Value invalid

4.7 dB

Ratios in dBm are transferred as signed 16 bit values in 0.01 dBm.

It is therefore possible to represent ratios from –327.66 dBm to +327.66 dBm.

Number range	Value range
–32766 to +32766	–327.66 dB to +327.66 dB
–32767 (0x8001)	≤ –327.67 dB
+32767 (0x7FFF)	≥ +327.67 dB
–32768 (0x8000)	Value invalid

5 Messages from Modules

5.1 Timestamp Object

The timestamp object defined in CANOpen is used to distribute a uniform, system-wide time: The timestamp contains the UTC time of the transmitter.
The structure of the CANOpen timestamp object corresponds to the data type TIME_OF_DAY (6 bytes) and is:

```
STRUCT OF
  UNSIGNED28 ms,
  VOID4 reserved,
  UNSIGNED16 days
TIME_OF_DAY
```

The days are counted from January 1, 1984.
The resolution is ms.

5.2 Identify

Each module must be able to identify itself by means of suitable mechanisms in order to allow a user to perform and check sorting operations. Normally this is done by means of rhythmic flashing of the "status" LED (→5.3). The module can use an SDO to pass a time between 0 (immediately off) and 59 seconds during which the module activates Identify. The value is not stored permanently and is always initialized with 0 after a reset or restart. If a new time is passed during the runtime, this new time starts immediately.

SDO

Index	Subindex	read/write	Bytes	Content
0x2300	0x01	w	1	Time [s] for which Identify is to remain activated. 1 byte 0 = off 1-59 = 1 s to 59 s >59 = undefined (ignore)

5.3 Status LED

Modules that have a status LED use it to indicate the following CAN states:

1. "unconfigured" The module does not have a node number, master is present
   – rapid flashing (approx. 10 Hz) –
2. "no master" The master is, however, not visible to the module (regardless of whether the node number is valid)
   – LED is on for 750 ms and off for 250 ms –
3. "everything OK" Master is visible and node number is present
   – LED is on –
4. "identify" The module is identifying itself (due to a command from the master)
   – LED is on for 125 ms and off for 375 ms –

Modules that have a two-colored status LED and can therefore display the colors red, green and yellow additionally indicate the sum state using the different colors (SumWarning = yellow (green and red simultaneously) and SumFault = red only). The overall state can therefore be determined from the color and the CAN state from the flashing sequence. The green LED means no warning and no fault.

5.4 Heartbeat

As provided for in the CANOpen standard, the heartbeat is implemented using the COB-ID 0x700+NodeID. The control unit sets an appropriate value (2000 ms) (which is stored in a fail-safe memory) and then waits for a corresponding heartbeat every 2000 ms. The heartbeat message allows the control unit to do two things:

1. if the heartbeat is absent, to recognize that the module has been removed and
2. if the system is restarted (the node IDs are persistent and the modules do not respond to the LSS query "are there any unconfigured modules?"), to recognize that modules with the node number xy

(COB-ID 0x700 = node ID) are present in the system and can determine the exact type by querying the type plate.

For its part, a slave module expects messages from a control module.
The master modules also send a heartbeat on the COB-ID 0x700+NodeID. The masters can use this to monitor each other via CAN. (Note: Owing to the filter settings, this message is not received by the slaves.)

Detection of "No Connect" in slave

If a slave does not receive an LSS message after max. 10 s, it automatically switches over to the "No master" state until an LSS message is received again.

5.5 Reset Faults

A "Reset Faults" of a module is executed via a PDO. The PDO must always be sent by the master. It resets the faults.
One bit per module type (e.g. all GD900 units) can be used to reset one specific module type.

Byte 0 and byte 1 of the PDO are defined as follows:

Byte 0							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 t.b.d.	0 t.b.d.	ParIO12_9	ParIO80_32	RCB	RFSWITCH	PMU905	GD900

Byte 1							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 t.b.d.	0 t.b.d.	0 t.b.d.	0 t.b.d.	0 t.b.d.	0 t.b.d.	0 t.b.d.	0 t.b.d.

PDO

COB-ID	receive/transmit	Content
0x300, without addition of node ID	r	Byte 0 to 1: Reset (16 bit) Byte 2 to 7: Reserved Length: 8 bytes

5.6 Broadcast Message, "noBasicConfiguration"

If an amplifier has not yet received the global configuration (see the broadcast message "Global Configuration"), the amplifier sends a PDO without data (conflict avoidance in the case of multiple amplifiers). The message is sent by the amplifier by means of a PDO with fixed ID and length 0 and has the following structure.

Cycle time = Inhibit time = 500 ms
Length of PDO = 0 bytes

PDO broadcast on address 0x280							
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
omitted							

5.7 Basic Configuration

The basic configuration is used to disclose the settings of the transmitter system globally. This is done using a PDO.

According to CAN SDD, it is possible to use a broadcast message to command all of the amplifiers in the system in a single command. The message is sent using a PDO.

It is also possible to send the basic configuration to each amplifier separately using an SDO (see object directory entry 0x2690). This mechanism allows each amplifier to receive its own configuration independently of the other amplifiers within the transmitter.

PDO

COB-ID	receive/transmit	Content
0x200, without addition of node ID	r	See above. Length: 8 bytes

Structure:

Basic configuration						
Byte 0	Byte 1/2		Byte 3/4		Byte 5	Byte 6	Byte 7
Bit 0: PowerOn 0 = PowerOff, 1 = PowerOn **Bit 1: PEAK/AV** 1 = Peak, 0 = AV **Bit 2 to 3: IDQ-SELECT** 00b: IDQ0 01b: IDQ1 10b: IDQ2 11b: IDQ3 **Bit 4: ActivateBlocking** 0 = inactive, 1 = active **Bit 5 to 7: reserved (0)** With PMU905: ON 0x01, OFF 0x00	**VREF_PWR** in mVolt Range (0 to 5000)		**VDC_CTRL** in 0.01 % Range (4800 to 10000) (4800 = 48 % to 10000 = 100 %)		**Bit 0 to 4**: FREQ_ID **Bit 5 to 7**: reserved (0) With PMU905: 0x00	Reserved (0)	Reserved (0)
Alignment:	Bit 7 to bit 0	Bit 15 to bit 8	Bit 7 to bit 0	Bit 15 to bit 8			

5.8 Status Message – Status PDO

The status PDO is always sent with 8 bytes. If a status value is changed, the PDO is sent autonomously and directly by the amplifier.
Inhibit time: 500 ms.
Cycle time: 10000 ms.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Global component (CAN SDD)	Amplifier state						
Bit 7 to bit 0	Bit 7 to bit 0	Bit 7 to bit 0	Bit 7 to bit 0	Bit 7 to bit 0	Bit 15 to bit 8	Bit 23 to bit 16	Bit 31 to bit 24

Byte	Bit	
0	0	Reserved (0)
	1	Reserved (0)
	2	State of sorting line 1
	3	State of sorting line 2
	4	Reserved (0)
	5	Sw UpdateRunning
	6	SW/HW mismatch
	7	Reserved (0)
1	0	Reserved (0)
	1	Reserved (0)
	2	Reserved (0)
	3	Reserved (0)
	4	Reserved (0)
	5	Reserved (0)
	6	Reserved (0)
	7	Reserved (0)
2	0	Reserved (0)
	1	Reserved (0)
	2	Reserved (0)
	3	Reserved (0)
	4	Reserved (0)
	5	Reserved (0)
	6	Reserved (0)
	7	Reserved (0)
3	0	FREQ_FAIL
	1	REDUCED_PWR
	2	PSU_RES
	3	FAN_FAIL
	4	Reserved (0)
	5	SUPPLY_FAIL_1
	6	SUPPLY_FAIL_2
	7	PWR_CALIBRATED

Byte	Bit	
4	0	DET_INHIB_MON
	1	AMPLIFIER_ON
	2	RF_IN_FAIL
	3	MUTE
	4	REFLECTION
	5	RF_POWER_FAIL
	6	REFLECTION_INT
	7	TEMP_FAIL
5	0	TEMP_FAIL_INT
	1	PA_FAIL
	2	DRV_FAIL
	3	TRANSISTOR_FAIL
	4	PEAK_AV
	5	RF_INHIB
	6	AC_OK_1
	7	AC_OK_2
6	0	Reserved (0)
	1	FAULT
	2	DC_OK_1
	3	DC_OK_2
	4	Reserved (0)
	5	DC_FAIL
	6	SUPPLY_FAIL
	7	REGULATION_FAIL
7	0	INIT_FAIL
	1	INIT_BUSY
	2	BIAS_FAIL_BIT0
	3	BIAS_FAIL_BIT1
	4	BIAS_ADJ
	5	SD_MON
	6	Reserved (0)
	7	Reserved (0)

6 CAN Object Directory of Amplifiers

The object directory provides the user with an interface which can be used to write / read values to and from the amplifier. SDOs are used to access the object directory.

6.1 Type Plates

Type plate of amplifier				
Idx	Sub	R/W	Bytes	Description
0x2001	0	R	1	Number of subindexes
0x2001	1	RW	sizeof	PADM IDENT 2 (size: 44 bytes)

Structure of PADM_IDENT_2 block:

	Offset	**Size in bytes**
Product code	12	4
Variant	16	1
Product index	18	2
Serial number	20	4
Product date	24	2
Name	30	14

6.2 Coupling Attenuation of Front Sample Point

Coupling attenuation of front sample point (RF_MON)				
Idx	Sub	R/W	Bytes	Description
0x2623	0	R	1	Number of subindexes
0x2623	1	RW	2	Coupling attenuation in dB
0x2623	2	RW	2	Reserved (0)
0x2623	3	RW	2	Reserved (0)
0x2623	4	RW	2	Reserved (0)

6.3 Trigger Commands to Amplifier

CAN control commands in trigger mode

Magic numbers (4 bytes, ASCII):

ASCII:	Meaning:
B1A$	BIAS_ADJUST (auto bias)
!B1A	BIAS_STOP
ZeR0	BIAS_ZERO

Idx	Sub	R/W	Bytes	Description
0x2630	0	W	4	Magic number of command

6.4 Basic Configuration SDO

Basic configuration

Idx	Sub	R/W	Bytes	Description
0x2690	0	R	1	Number of subindexes
0x2690	1	W	7	Basic configuration (see 5.7 Basic Configuration)

The SDO consists of 7 bytes of data (PDO 8 bytes). The eighth byte of the PDO is currently unused and is not transferred to the SDO. As a result, the SDO can be transferred in 2 instead of 3 segments. If necessary, the eighth data byte can be added to the SDO at a later stage.

6.5 Analog Measured Values

Analog measured values				
Idx	Sub	R/W	Bytes	Description
0x2640	0	R	56	Measured values: 32 x UInt16 according to CAN-SDD

Breakdown of measured value array:

	PMU905	
Array item	**Designation**	**Value range**
0	PWR_A	0..7500 mV
1	PWR_B	0..7500 mV
2	PWR_OUT	0..7500 mV
3	REFL_OUT	0..7500 mV
4	V_REG	0..12500 mV
5	V_TEMP	0..150 °C
6	I_DRV	0..33,3 A
7	I_PRE	0..5000 mA
8	I_1A	0..33,3 A
9	I_2A	0..33,3 A
10	reserved	-
11	V_REFL_SAVE	0..7500 mV
12	reserved	-
13	VPLUS_MON	0..57,5 V
14	V_I_DC	0..223,492 A
15	reserved	-
16	I_1B	0..33,3 A
17	I_2B	0..33,3 A
18	reserved	-
19	V_12V_MON	0..15000 mV
20	VREF_PWR_OPV	0..5000 mV
21	V_AUX_IN	0..36,75 V
22	V_5V_ACB	0..7500 mV
23	V_3V5	0..5000 mV
24	AIR_INLET	-30..120 °C
25	AIR_OUTLET	-30..120 °C
26	reserved	-
27	reserved	-