

# BETRIEBSANLEITUNG INSTRUCTION MANUAL



**KEB COMBICOM**

**CANopen-ANSCHALTUNG  
CANopen-INTERFACE**



Seite D - 3 ..... D - 40

Das in dieser Betriebsanleitung verwendete Pictogramm entspricht folgender Bedeutung:



**Achtung,  
Unbedingt  
beachten**

In dieser Betriebsanleitung befindet sich auf Seite D - 38 ein Literaturverzeichnis, in dem Nachschlagewerke aufgeführt sind, die bestimmte Normen und Aussagen in dieser Anleitung erläutern. An den entsprechenden Textstellen, befinden sich mit eckigen Klammern [ ] gekennzeichnete Ziffern.



Page GB - 3 ..... GB - 40

The pictograph used in this manual means:



**Attention,  
observe at  
all costs**

On Page GB - 38 of this Manual you find a literature list which contains reference books. Certain standards and statements are explained in these reference books. At the corresponding text passage you find digits in square brackets [ ].

<b>1.</b>	<b>General</b> .....	<b>4</b>
<b>2.</b>	<b>Ordering Information</b> .....	<b>4</b>
<b>3.</b>	<b>F5-CAN-Operator</b> .....	<b>4</b>
<b>4.</b>	<b>Hardware Description</b> .....	<b>5</b>
4.1	Diagnostic interface .....	5
4.2	CAN-Interface.....	6
<b>5.</b>	<b>Change from F4-CAN to F5-CAN-Interface</b> .....	<b>7</b>
<b>6.</b>	<b>Fundamentals of CAN-Bus</b> .....	<b>8</b>
<b>7.</b>	<b>Functions</b> .....	<b>9</b>
7.1	Characteristic of the High-Speed-PDO .....	11
7.2	Characteristic of the Low-Speed-PDO .....	11
7.3	Automatic CAN-Bit rate detection .....	12
7.4	Process-Data Mapping .....	12
7.5	CANopen Bootup-Sequenz .....	13
7.6	Bootup-Message .....	15
7.7	Node-Guarding .....	15
<b>8.</b>	<b>Coding of Data in the 4 Types of CAN-Telegrams</b> .....	<b>16</b>
8.1	<b>SDO(rx)-Telegram</b> .....	<b>16</b>
8.1.1	Initiate Domain Download Request (Write Request of the Master) .....	17
8.1.2	Initiate Domain Upload Request (Read Request of the Master) .....	17
8.2	<b>SDO(tx)-Telegram</b> .....	<b>18</b>
8.2.1	Initiate Domain Download Response (Write Confirmation from Inverter) .....	18
8.2.2	Initiate Domain Upload Response (Read Confirmation from Inverter) .....	18
8.2.3	Abort Domain Transfer (Error answer from the Inverter) .....	18
8.3	<b>PDO1(rx)-Telegram</b> .....	<b>19</b>
8.4	<b>PDO1(tx)-Telegram</b> .....	<b>19</b>
8.5	<b>PDO2(rx)-Telegram</b> .....	<b>19</b>
8.6	<b>PDO2(tx)-Telegram</b> .....	<b>19</b>
<b>9.</b>	<b>Configuration Parameters</b> .....	<b>20</b>
<b>10.</b>	<b>Access to Operator-Parameters via Diagnostic Interface</b> .....	<b>32</b>
<b>11.</b>	<b>Changing the Transmission-Type of the PDO's</b> .....	<b>36</b>
11.1	Asynchronous manufacturer-specific (Value = 254d/FEh) or Asynchronous profile-specific (Value = 255d/FFh) .....	36
11.2	Synchronous acyclic (Value = 0) or Synchronous cyclic (Value = 1) .....	36
<b>12.</b>	<b>Annex</b> .....	<b>37</b>
12.1	<b>CAN-Bit-Timing</b> .....	<b>37</b>
12.1.1	Important Warning Information .....	38
12.2	<b>Literature Index</b> .....	<b>38</b>
12.3	<b>Table of Configuration-Parameters according to CANopen</b> .....	<b>39</b>
12.4	<b>Compact Summary of CAN-communication</b> .....	<b>40</b>

1. General

This manual as well as the hardware and software are developments of the Karl E. Brinkmann GmbH. Errors and omissions excepted! Karl E. Brinkmann GmbH prepared the documents, software and hardware to the best of their knowledge, however, no guarantee is given that the specifications will bring the user the efficiency aimed at. The Karl E. Brinkmann GmbH reserves the right to change the specifications without obligation. All rights reserved!

GB

2. Ordering Information

This Instruction Manual: .....	CC.F5.010-K001
F5-CAN-Operator: .....	00.F5.060-5000
Accessory for diagnostic interface:	
HSP5-cable between PC and adapter: .....	00.F5.0C0-0001
Adapter DSUB9 / Western: .....	00.F5.0C0-0002

3. F5-CAN-Interface

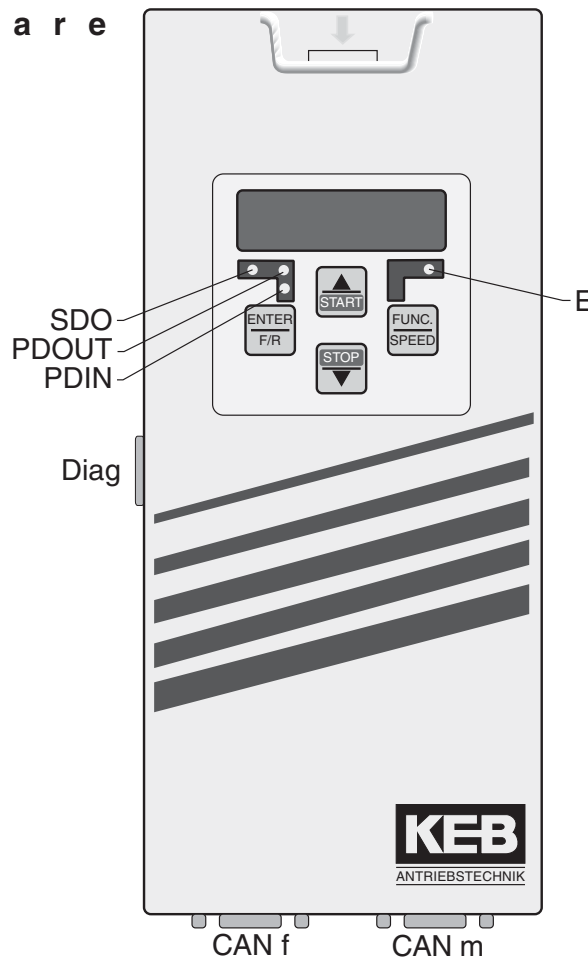
KEB-Antriebstechnik develops, produces and sells static frequency inverters worldwide in the industrial power range. Inverter type **F5** can be optionally equipped with a **CAN**-interface (**C**ontroller-**A**rea-**N**etwork). This is an intelligent interface which controls the access via CAN to the parameters in the inverter.

The F5-CAN-operator is integrated into the inverter housing by insertion and fits into every KEB-F5-frequency inverter. Parallel to the field bus operation the operation via the integrated display/keyboard as well as a further serial interface for diagnosis/parameterizing (KEB COMBIVIS) is possible.



To program the KEB F5 inverter with CAN you need the respective manual for the inverter control [1] in addition to this manual.

## 4. Hardware description



SDO: (green)	SDO-Communication active
PDOOUT: (green)	- PDOOUT-data are written to the FI control. - lights up during the automatic bit-rate detection on CAN.
PDIN: (green)	- PDIN-data are read from the FI control. - Lights up briefly after a Reset-command over CAN.
E (red):	
on ==>	Inverter ready for operation
blinking ==>	Inverter in error
off ==>	No supply voltage
Diag:	Diagnostic interface to the PC (see chapter 4.1)
CAN f:	CAN-interface (socket-connector)
CAN m:	CAN-interface (pin-connector)

### 4.1 Diagnostic interface



To prevent a destruction of the PC-interface, the diagnostic interface may only be connected to the PC over a special HSP5 cable with voltage adaption.

Over an adapter a HSP5 cable is connected to the diagnostic interface (see page 4 'Ordering Information'). By way of the PC-software KEB COMBIVIS 5 one has now normal access to all inverter parameters. The operator-internal parameters can also be read and adjusted or parameterized by means of download.

## 4.2 CAN-Interface

To CAN-Bus the operator provides one D-SUB-9 pole male connector and one D-SUB-9 pole female connector (in accordance with DIN41652 Part 1). Assignment of CAN-connector [2] according to:

Pin	Signal	Description
1	-	Reserved
2	<b>CAN_L</b>	<b>CAN-Bussignal dominant low</b>
3	CAN_GND	Not connected here
4	-	Reserved
5	(CAN_SHLD)	Not connected here
6	(GND)	Not connected here
7	<b>CAN_H</b>	<b>CAN-Bussignal dominant high</b>
8	-	Reserved
9	(CAN_V+)	Not connected here

**Transmission level of CAN:** In accordance with ISO/DIS 11898, ISO-High Speed.

**Transmission speed of CAN:** Adjustable via CAN (10, 20, 25, 50, 100, 125, 250, 500, 800, 1000 Kbit/s).

**Isolation:** Safe isolation in accordance with VDE0160.

**Bus connection:** 124 Ohm , must be done externally e.g. on the second D-SUB-connector (Pins 2 and 7).

GB

## 5. Change from F4-CAN to F5-CAN-Interface

Here the important changes of KEB-CAN-F5-Interface based on KEB-F4-CAN-Interface are summarized in a list to provide the user with a survey.

### **Added features of KEB-CAN-F5-Interface**

- Second CAN-connector for the continuation of the CAN-BUS or to plug-on a terminating resistor.
- Received extended CAN-telegrams (29-Bit-Identifier) are accepted and processed. As before only standard CAN-telegrams are transmitted (11-Bit-Identifier).
- Automatic CAN-bit rate detection disengageable (see page 12).
- Direct set addressing at inverter parameters via the SDO-Subindex: (see page 17).
- Programming and diagnosis via keyboard and display of the CAN-operator.
- Additional diagnosis and programming interface for KEB COMBIVIS (see page 5).

### **Omitted features of KEB-CAN-F5-Interface**

- No support of DRIVECOM-Profile parameters.
- No support of DSP402-Standards.
- No support of Life-Guarding.

### **Changes**

- Modified standard process data assignment (see page 26, 28).

### 6. Fundamentals of CAN-BUS

The **CAN(Controllor-Area-Network)**-BUS system and the terms often used for this system are described in this chapter.

CAN is a **Multi-Master-System**. This means every node has access to the BUS and can send telegrams. In order to prevent problems when two nodes simultaneously access the BUS, the CAN-BUS has an arbitration phase which determines who may continue to send his telegram. When there is a conflict in accessing BUS the user with the lowest telegram number (identifier) has priority. This user then can completely send his telegram without repeating the first part. All other nodes go into receiving status and stop sending their telegram. The available telegram numbers in the CAN version 2.0A are limited to 2032 identifiers (0...2031).

CAN-telegrams can have a maximum of 8 byte user data.

When speaking of the **logical CAN-Master**, the CAN-nodes is meant who manages the control of the entire CAN-System. Even though there are only Masters in CAN it usually is the case that one or more node have the control. In connection with this the KEB inverter is seen as the command receiver (logical slave).

## 7. Function

The CAN protocol is uniformly standardized up to layer 2. The complete processing of the protocol is done by the CAN-controller. The **CAN in Automation Association (CiA)** adopted a standard for the higher protocol level called **CAN Application Layer (CAL)**. Based on this standard the '**CAL-based Communication Profile**' (CiA,DS301) was published in September 1995. This standard is the basis for all **CANopen**-device-profiles. In the '**CAL-based Communication Profile**' certain functions of the CAL-Standard are selected. The communication profile defines the **Minimum Capability Device**. This is the minimum required functionality that a CANopen-node must provide. The KEB CAN-Interface realizes such a Minimum Capability Device.

An important function in every CAN network is the distribution of telegram numbers (identifiers). The numbers are limited to 2032 in CAN V2.0 A. A special process for this is defined in the CAL-standard that realizes this identifier distribution dynamically through its own protocol (DBT = Distributor). This relatively extensive procedure for allocating identifiers is not required and not integrated in the KEB-CAN-Interface. Thus a simple process is specified in the communication profile for agreement of the identifier allocation. This process is supported by the KEB-CAN-Interface and is as follows:

Every inverter has a CAN-address (the **Module\_Id**). This results from the parameter **inverter address**, which every KEB-inverter supports (see manual of the KEB-inverter used):

$$\text{Module\_Id} = \text{Inverter Address (SY.06)} + 1$$



After delivery all KEB-inverters have the inverter address = 1. If several KEB inverters should be networked via CAN, they must have different inverter addresses. This is done for example via the keyboard on the operator.

Every inverter is assigned 6 identifiers.

Via one identifier the logical CAN-BUS-Master requests the reading (upload) or writing (download) of any parameter in the KEB-inverter (**Request-Identifier**).

Another identifier is reserved for the respective answer from the inverter (**Response-Identifier**). This mechanism of request and response is called **confirmed service**. The CANopen communication profile calls this function **Service-Data-Object (SDO)**.

$$\begin{aligned} \text{SDO(rx)} = \text{Request-Identifier} &= 1536 + \text{Module\_Id} = 1537 + \text{inverter addr. (SY.06)} \\ \text{SDO(tx)} = \text{Response-Identifier} &= 1408 + \text{Module\_Id} = 1409 + \text{inverter addr. (SY.06)} \end{aligned}$$

*Example:* Inverter address = 30 ==> Read/Write request via Identifier = 1567  
 ==> Read/Write confirmation via Identifier = 1439

*Note:* In principle the function of the SDO is completely sufficient to control the KEB F5-frequency inverter via CAN. Each parameter value in the inverter can be changed or queried.

The 3rd identifier enters data in the inverter which is not addressed nor confirmed. This identifier is called **OUT1-Identifier**, because it goes from the Master to the Slave.

The inverter uses the 4th identifier to provide new data unaddressed and unconfirmed to the CAN-Master (**IN1-Identifier**).

These functions are characterized by the communication profile as **Process-Data-Object (PDO)**. Both identifiers are called PDO1(rx) and PDO1(tx).

$\begin{aligned} \text{PDO1(rx)} &= \text{Out-Identifier} = 512 + \text{Module\_Id} = 513 + \text{inverter address (SY.06)} \\ \text{PDO1(tx)} &= \text{IN-Identifier} = 384 + \text{Module\_Id} = 385 + \text{inverter address (SY.06)} \end{aligned}$
---

Starting with software version 1.3 the PDO-function is available twice in the KEB F5-CAN-interface. This so-called 2. PDO occupies identifier five to six:

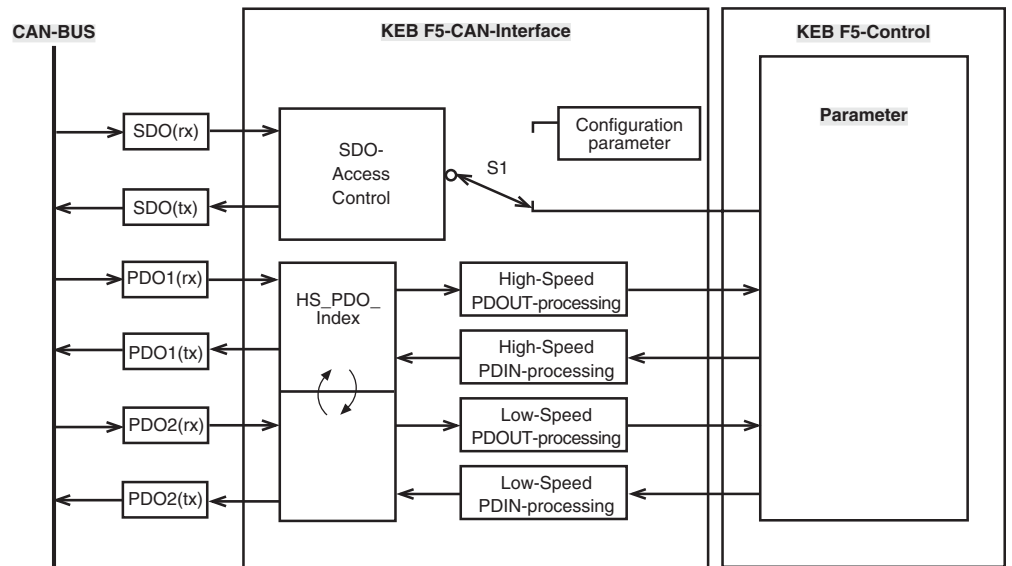
$\begin{aligned} \text{PDO2(rx)} &= \text{Out-Identifier} = 768 + \text{Module\_Id} = 769 + \text{inverter address (SY.06)} \\ \text{PDO2(tx)} &= \text{IN-Identifier} = 640 + \text{Module\_Id} = 641 + \text{inverter address (SY.06)} \end{aligned}$
---

The 5th identifier enters data in the inverter which is not addressed nor confirmed. This identifier is called **OUT2-Identifier**, because it goes from the Master to the Slave.

The inverter uses the 6th identifier to provide new data unaddressed and unconfirmed to the CAN-Master (**IN2-Identifier**).

The two PDO's are identical concerning the external management, but differ clearly in the form of internal processing. Only one of the two can be handled as High-Speed PDO like in the previous software version. Regarding the processing the added PDO is of equal status as the SDO-commands and is referred to as Low-Speed-PDO in the following. It can be adjusted which of the two PDO's is the High-Speed-PDO. On delivery the first PDO is 'High-Speed' and the second PDO is 'Low-Speed' and disabled. Therefore existing CAN-application have not to be modified.

The CAN-Interface controls the data flow from CAN (SDO(rx) and PDO(rx)) to the inverter control and also from the inverter to CAN-BUS (SDO(tx) and PDO(tx)):



The diagram above shows the function of CAN-Interface. The position of switch **S1** is determined solely by the parameter-address (16 Bit Index plus 8 Bit Subindex) in the CAN-SDO(rx)-telegram. In a certain index range the so-called configuration data of CAN-interface is found. These parameters determine the behaviour of the CAN-interface and thus are realized in it. Access to parameters in the index range 2000(hex) to 5EFF(hex) are transferred as read/write services to the inverter control.

### 7.1 Characteristic of the High-Speed-PDO

- The process data mapping is in the inverter control. The corresponding parameters are in the system parameter group (SY). As the coding of the PD-mapping of the inverter does not match with CANopen it is automatically converted by the CAN-operator accordingly.
- The presetting of new process output data by CAN is converted only on one special process data service for the inverter control.
- The minimum cycle time for new process output data is approx. 3 ms.
- The cyclic reading of process input data is executed by only one special process data read service.
- The minimum attainable cycle time for the reading of process input data is approx. 3ms.
- Not all parameters of the inverter control can be mapped on the High-Speed-PDO.

### 7.2 Characteristic of the Low-Speed-PDO

- The process data mapping is managed exclusively by the CAN-operator.
- The presetting of new process output data by CAN is converted to 'n' simple services (like SDO-commands) for the inverter control, at that 'n' corresponds to the number of mapped parameters in the PDO-mapping.
- The minimum cycle time for new process output data is approx. 'n' \* 5 ms.
- The cyclic reading of process input data is executed by 'n' simple read services, at that 'n' corresponds to the number of mapped parameters in the PDO-mapping.
- The minimum attainable cycle time for the reading of process input data is approx. 'n' \* 5 ms.
- All parameters of the inverter control can be mapped on the Low-Speed-PDO.

### 7.3 Automatic CAN-Bit rate detection

The KEB F5-CAN-Operator supports an automatic bit rate detection on CAN. It is enabled in the standard setting and it is activated by the value 255 of the Operator parameter CAN\_BAUD and CAN-BAUD2 (see below). It is disabled by adjusting a value uneven 255 in the above mentioned parameters.

In order for the automatic bit rate detection to be successful the following points must be observed:

- The KEB-F5-CAN-Operator recognizes only bit rates which are listed as values in the above mentioned parameters.
- The Bit-Timing of all CAN-nodes must also match (see chapter CAN-Bit-Timing).
- In order to recognize the CAN-bit rate the KEB-F5-CAN-Operator must receive several CAN-telegrams. Depending on the bit rate the KEB-CAN-node must receive between 3 up to 30 telegrams, before it participates in the CAN-communication. Note, that for that purpose the necessary processing depends on the actual Bus-setup. If only the logical master and the KEB-CAN-node are connected to the CAN-Bus, it is sufficient to request from the master the transmission of one telegram. Since the CAN-controller of the logical master repeats the telegram for as long as it receives an acknowledgement from the counter-station, the transmission will only be stopped if the KEB-CAN-node has found the bit rate. In case several nodes are connected to the CAN-Bus, which are already successfully communicating, then several telegrams must be ordered so that the added KEB-F5-CAN-Operator can find the bit rate.

In the phase of the automatic bit rate detection the time between two telegrams succeeding one another should not be larger than 1 s.

### 7.4 Process - Data Mapping

The type of determination of the destination for the data in the PDO(rx)-telegrams and the source for the data in the PDO(tx)-telegrams complies with the regulations of the CANopen-Communication Profile (see [12] in literature index). For every data direction a complex object (parameter) defines the PDO-mapping.

Another object determines the communication definition (PDO communication parameter) for each data direction. See parameter description from:

- |                                     |                                     |
|-------------------------------------|-------------------------------------|
| - <b>1st receive PDO Mapping</b>    | - <b>2nd receive PDO Mapping</b>    |
| - <b>1st transmit PDO Mapping</b>   | - <b>2nd transmit PDO Mapping</b>   |
| - <b>1st receive PDO Parameter</b>  | - <b>2nd receive PDO Parameter</b>  |
| - <b>1st transmit PDO Parameter</b> | - <b>2nd transmit PDO Parameter</b> |

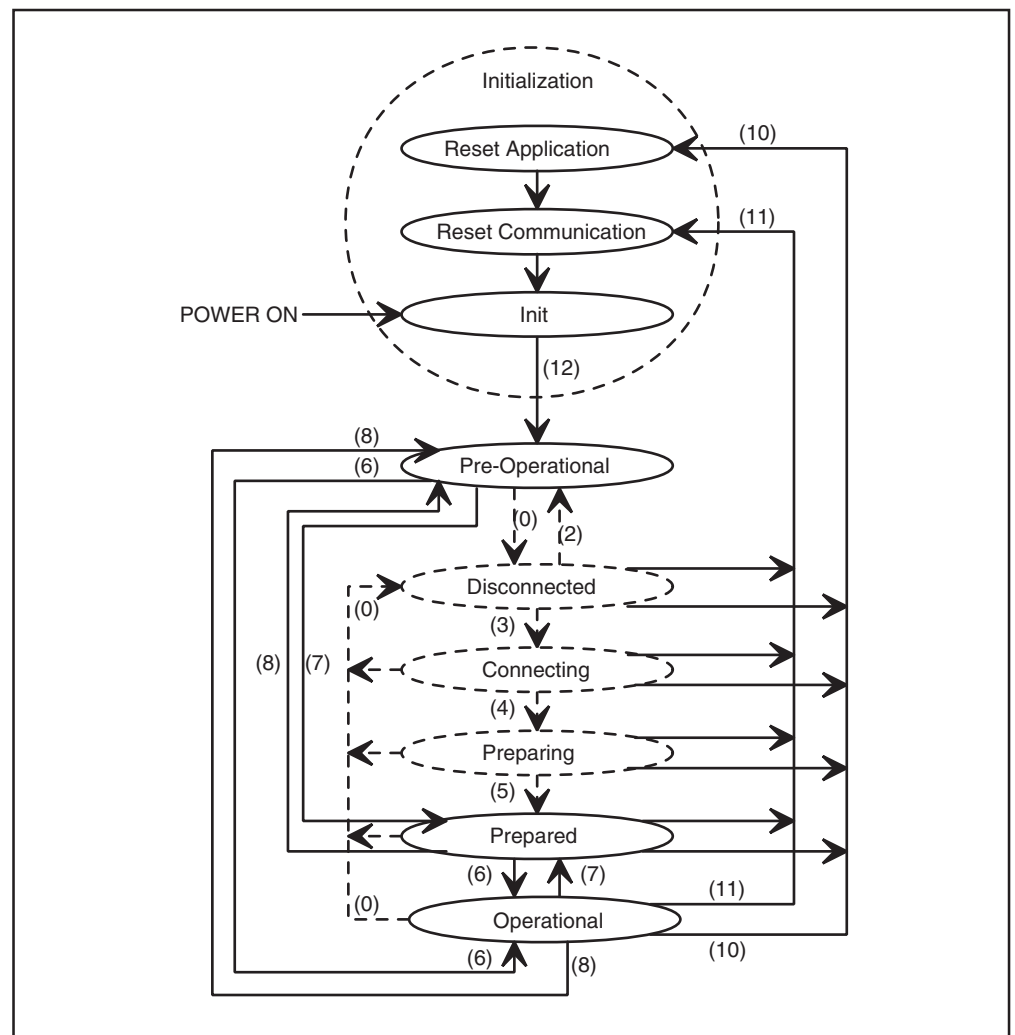
in this manual.

## 7.5 CANopen Bootup-Sequenz

The KEB-CAN-Interface automatically goes into **Pre-Operational** status after the initialization phase. In this status the communication is activated via the SDO(rx) and SDO(tx) with the services Domain Download (write parameter) and Domain Upload (read parameter). The process-data communication is inactive in this status. This is released by the NMT-Command Start\_Remote\_Node() (see below). After this command the KEB-CANopen-Slave switches its operation state to **Operational**. In this state the communication is completely activated. Certain CAN-nodes are addressed by the NMT-protocol by the **Node-Id**. This results as with the Module-Id from the value of the parameter inverter address:

Node-Id = Module-Id = Inverter Address (SY.06) + 1

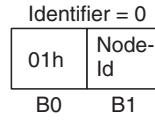
GB



The KEB-CANopen-Interface realizes the following transitions (characterized by a solid line):

6: **Start\_Remote\_Node()**

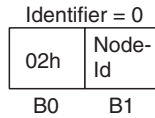
CAN-telegram:



with Node\_Id = 0 (all NMT-Slaves are addressed) or  
 with Node\_Id = Inverter-Address + 1 (only 1 inverter is addressed)

7: **Stop\_Remote\_Node()**

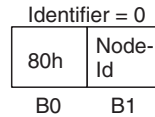
CAN-telegram:



with Node\_Id = 0 (all NMT-Slaves are addressed) or  
 with Node\_Id = Inverter-Address + 1 (only 1 inverter is addressed)

8: **Enter\_Pre-Operational\_State()**

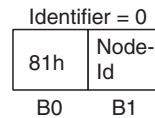
CAN-telegram:



with Node\_Id = 0 (all NMT-Slaves are addressed) or  
 with Node\_Id = Inverter-Address + 1 (only 1 inverter is addressed)

10: **Reset\_Node():** When this function is carried out a Software-Reset is done in the KEB-CAN-Interface.

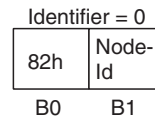
CAN-telegram:



with Node\_Id = 0 (all NMT-Slaves are addressed) or  
 with Node\_Id = Inverter-Address + 1 (only 1 inverter is addressed)

11: **Reset\_Communication():** Function as with Reset\_Node().

CAN-telegram:



with Node\_Id = 0 (all NMT-Slaves are addressed) or  
 with Node\_Id = Inverter-Address + 1 (only 1 inverter is addressed)

12: **Enter Pre-Operational automatically():** s.o.

GB

**7.6 Bootup-Message**

The KEB-F5-CAN-Operator gives a bootup-message, when after POWER ON the initialization period has finished. This is a telegram on identifier = 1793 + inverter address (SY.06), data length = 1, value = 0.

**7.7 Node-Guarding**

CAN (**CAN Application Layer**) defines a protocol through which a CAN-node can require the actual node-status of any node. This is part of the network management functionality (NMT) of a CAN-node and is called Node-Guarding. This is supported by the KEB-CAN-node. The node-guarding-request is done by sending a remote frame with the node-guarding-identifier. The response is returned as data-telegram with 1 byte of data on the same identifier. This byte contains the actual node-status of that node plus one bit (MSBit) which is altered from one to the next message. Each node has its own node-guarding-identifier. For minimum capability devices this identifier directly results from the node-id as follows:

Node-Guarding-Identifier = 1792 + Node-Id = 1793 + inverter addr. (SY.06)

Value of node-state	Meaning
1	DISCONNECTED
2	CONNECTING
3	PREPARING
4	PREPARED
5	OPERATIONAL
127d	PRE_OPERATIONAL

**8. Coding of Data in the 4 Types of CAN Telegrams**

Using this telegram the logical CAN-Master can inquire (read) or change (write) the value of one parameter. In the communication profile a write-service is called a **Domain Download** and a read-service a **Domain Upload**. The KEB-CAN-Interface only supports the expedited form of these services, so that only one telegram is exchanged for the service request and another for the service confirmation between logical CAN-Master and the KEB-CAN-Interface.

**8.1 SDO(rx)-Telegram**

The parameter is addressed by the unsigned 16-Bit-Index and 8-Bit-Subindex. The parameters of the inverter control lie in the index range 2000(hex) to 5EFF(hex). The CAN-Index results from the parameter address (see parameter description of the inverter control used) by addition with the Offset 2000(hex):

$$\text{CAN-Index} = \text{KEB-Parameter-Address} + 2000(\text{hex})$$

The subindex is used as an additional addressing for complex parameters of the operator. Likewise it can be used with parameters of the frequency control for the set addressing. For that applies:

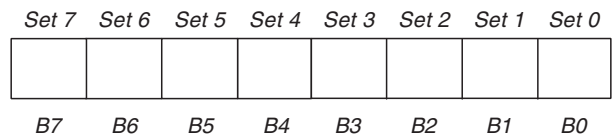
***Subindex = 0***

For set-programmable parameters the value of the parameter FR.09 determines the selected set.

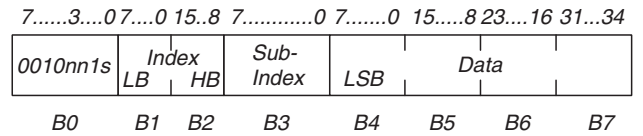
***Subindex uneven 0***

For set-programmable parameters the subindex determines the selected set. Here it must be noted, that the set is bit-coded. It is thus possible to simultaneously change the value of the parameter in several sets through writing. If on reading several sets are addressed simultaneously then the value of the parameter is only returned, if it is the same in all addressed sets. If the values are not equal an error is returned.

Subindex (if uneven 0):



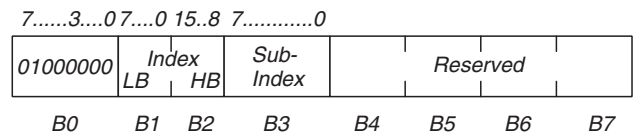
### 8.1.1 Initiate Domain Download Request (Write Request of the Master)



- nn:** only valid, when s=1: contains the number of bytes of the datafield which does not contain data.
- s:** When = 1, then contains nn the number of bytes in the data field which do not contain data. Otherwise no display of the data length in nn.
- Index:** 16-Bit (unsigned) addressing of the parameter (see above).
- Subindex:** 8-Bit (unsigned) sub-address for complex parameters and direct set addressing.
- Data:** data to be transmitted. The LSbyte is transmitted first.

GB

### 8.1.2 Initiate Domain Upload Request (Read Request of the Master)



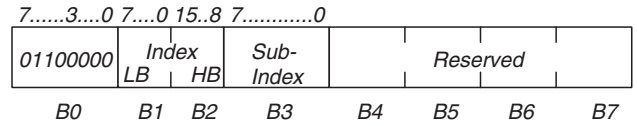
- Index:** 16-Bit (unsigned) addressing of the parameter (see above).
- Subindex:** 8-Bit (unsigned) sub-addressing for complex parameters and direct set addressing.

8.2 SDO(tx)-Telegram

8.2.1 Initiate Domain Download Response (Write Confirmation from Inverter)

This answer is sent from the KEB-CAN-Interface when the requested write service could be completed without an error.

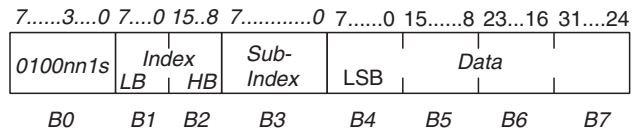
**Index:** see above  
**Subindex:** see above



8.2.2 Initiate Domain Upload Response (Read Confirmation from Inverter)

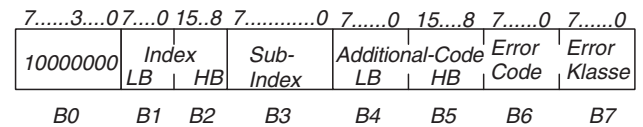
This answer is sent by the KEB-CAN-Interface when the requested read service could be completed without an error.

**nn:** see above  
**s:** see above  
**Index:** see above  
**Subindex:** see above  
**Data:** see above



8.2.3 Abort Domain Transfer (Error Answer from the Inverter)

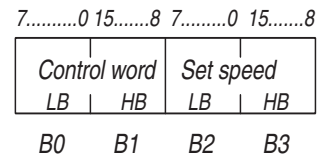
The KEB-CAN-Interface sends this answer when the requested read/write services cannot be completed. In this case an error description is supplied.



Error Class	Error Code	Additional Code	Significanca
6	1	0000h	Invalid access to one parameter, e. g. writing to a Read_Only-Parameter.
6	1	0010h	Invalid password.
6	1	0011h	Operation not possible.
6	4	0000h	The addressed parameter does not exist.
6	4	0041h	Invalid PD-assignment.
6	6	0000h	The internal communication between operator and FI-control is faulty.
6	7	0010h	Invalid data length.
6	9	0011h	Invalid subindex.
6	9	0012h	Invalid language identification.
6	9	0030h	The written value is outside the valid value range.
8	0	0022h	Inverter busy.

### 8.3 PDO1(rx)-Telegram

Using this telegram the logical CAN-Master transfers the new process-output data to the inverter. In the standard setting the KEB-CAN interface waits for a telegram with  $\geq 4$  byte data with the following contents:



The length and assignment of the PDO1(rx)-telegram can be changed by different configuration parameters. This change can only be done with the SDO(tx)-telegram (see above).

The following configuration parameters have an influence on the structure of the process-output data:

- 1st receive PDO Mapping
- 1st receive PDO Parameter

GB

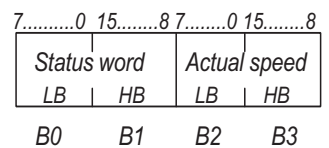
### 8.4 PDO1(tx)-Telegram

Using this telegram the KEB-CAN-Interface provides the (logical) CAN-Master with the process-input data.

The length, assignment and control of this telegram is influenced by the following configuration parameters:

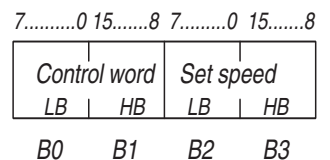
- 1st transmit PDO Mapping
- 1st transmit PDO Parameter

The standard setting causes the following telegram structure:



### 8.5 PDO2(rx)-Telegram

Using this telegram the logical CAN-Master transfers the new process-output data to the inverter. In the standard setting the KEB-CAN interface waits for a telegram with  $\geq 4$  byte data with the following contents:



The length and assignment of the PDO2(rx)-telegram can be changed by different configuration parameters. This change can only be done with the SDO(tx)-telegram (see above).

The following configuration parameters have an influence on the structure of the process-output data:

- 2nd receive PDO Mapping
- 2nd receive PDO Parameter

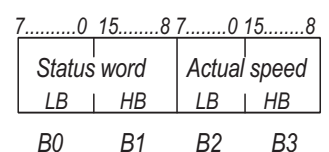
### 8.6 PDO2(tx)-Telegram

Using this telegram the KEB-CAN-Interface provides the (logical) CAN-Master with the process-input data.

The length, assignment and control of this telegram is influenced by the following configuration parameters:

- 2nd transmit PDO Mapping
- 2nd transmit PDO Parameter

The standard setting causes the following telegram structure:



## 9. Configuration Parameters

These parameters determine the configuration of the KEB-F5-CAN interface and thus are realized in the F5-CAN and not in the inverter control.

<b>CAN_Baud</b>	<b>Index:</b> 5FFFh <b>Object-Type:</b> Simple Variable (Var) <b>Subindex:</b> 0 <b>Data length:</b> 1 Byte <b>Significance:</b> Index for CAN-transmission speed <b>Coding:</b> <ul style="list-style-type: none"> <li>0 = 10 Kbit/s</li> <li>1 = 20 Kbit/s</li> <li>2 = 50 Kbit/s</li> <li>3 = 100 Kbit/s</li> <li>4 = 125 Kbit/s</li> <li>5 = 250 Kbit/s</li> <li>6 = 500 Kbit/s*</li> <li>7 = 1000 Kbit/s*</li> <li>8 = 800 Kbit/s*</li> <li>9 = 25 Kbit/s</li> <li>255 = Automatic Bit rate detection</li> </ul>
-----------------	--

**Standard Setting:** 255 = Automatic Bit rate detection

**Note:** A value changed is immediately active but not automatically stored non-volatily. The Bit-Timing adheres to the specifications of the Working Group Physical-Layer of CiA (see Literature index [2]). See annex for Bit-Timing. Which transmission speeds can be used depends on the line length, the sum of the deceleration times and the Bit-timing. Value 255 can be written only via CAN-BAUD2.

<b>SAVE_CAN_Baud</b>	<b>Index:</b> 5FFEh <b>Object-Type:</b> Simple Variable (Var) <b>Subindex:</b> 0 <b>Data length:</b> 1 Byte <b>Significance:</b> Used for non-volatile storage of adjusted CAN-transmission speed. <b>Coding Write:</b> FFh = non-volatile storage of CAN_Baud. 0 = no storage. <b>Coding Read:</b> FFh = Adjusted value corresponds to the stored value. 00h = Adjusted value is not equal to stored value.
----------------------	--

\* See chapter 12.1.1 'Important Warning Information'



**CAN\_Baud2**    **Index:**                    **5FECh**  
**Object-Type:**            **Simple Variable (Var)**  
**Subindex:**                **0**  
**Data length:**            **1 Byte**  
**Significance:**            Index for CAN-transmission speed alternatively to CAN\_Baud (see above).  
**Coding:**                    0 = 10 Kbit/s  
                                   1 = 20 Kbit/s  
                                   2 = 50 Kbit/s  
                                   3 = 100 Kbit/s  
                                   4 = 125 Kbit/s  
                                   5 = 250 Kbit/s  
                                   6 = 500 Kbit/s\*  
                                   7 = 1000 Kbit/s\*  
                                   8 = 800 Kbit/s\*  
                                   9 = 25 Kbit/s  
                                   255 = Automatic Bit rate detection  
**Standard Setting:** **255 = Automatic Bit rate detection**  
**Note:**                        Contrary to parameter CAN-BAUD a modified value is immediately stored non-volatile, but it becomes active only after a Reset Node-command or after the next switch on.

**WD\_Inhibit**    **Index:**                    **5FF9h**  
**Object-Type:**            **Simple Variable**  
**Subindex:**                **0**  
**Data length:**            **1 Byte**  
**Significance:**            Determines onto which events the Fieldbus-Watchdog is triggered. The Fieldbus-Watchdog is used to bring the frequency inverter into the error condition if no activities take place on CAN. The actual activation and programming of the Watchdog is adjusted in the FI-control. The parameters to be adjusted for that are found in the instruction manual of the FI-control.  
**Coding:**                    Bit-coded:  
                                   Bit 0 = 1  
                                   When starting a PDOOUT-telegram to the FI-control the Watchdog is reset. Note, that the occurrence of this event is also dependent on the setting of the parameter 1st Receive PDO Parameter.Tx\_type.  
                                   Bit 1 = 1  
                                   When starting the processing of a SDO-job, the Watchdog is reset.  
                                   Bit 2 = 1  
                                   If the node does not detect transfer problems on CAN, the Watchdog is reset.  
**Standard Setting:** **07h**  
                                   The Watchdog is reset if:  
                                   - process output data is written to the FI-control,  
                                   - a SDO-job is started and  
                                   - no transmission problems are found on CAN .  
**Note:**                        A changed value is immediately active and non-volatile stored.

\* See chapter 12.1.1 'Important Warning Information'

**PD\_Stored**    **Index:**            **5FE2h**  
**Object-Type:**    **Simple Variable**  
**Subindex:**        **0**  
**Data length:**    **1 Byte**  
**Significance:**    Determines whether the actual process data assignment is read from the EEPROM or whether it is operated with the standard PD-assignment.  
**Coding:**            FFh        ==> operates with the stored PD-assignment  
                          otherwise ==> operates with the standard PD-assignment.  
**Standard Setting:** **FFh**  
**Note:**                A changed value is immediately active and non-volatile stored.

**PDIN1\_Cycle\_Time**    **Index:**            **5FE6h**  
**Object-Type:**    **Simple Variable (Var)**  
**Subindex:**        **0**  
**Data length:**    **2 Byte**  
**Significance:**    Indicates the cycle time, in which the process input data of PDO1 in the status OPERATIONAL are read from the inverter control.  
**Coding:**            n \* 1ms.  
**Standard Setting:** **25 = 25ms**  
**Note:**                A changed value is immediately active and non-volatile stored.

**PDIN2\_Cycle\_Time**    **Index:**            **5FE7h**  
**Object-Type:**    **Simple Variable (Var)**  
**Subindex:**        **0**  
**Data length:**    **2 Byte**  
**Significance:**    Indicates the cycle time, in which the process output data of PDO2 in the status OPERATIONAL are read from the inverter control  
**Coding:**            n \* 1ms.  
**Standard Setting:** **100 = 100ms**  
**Note:**                A changed value is immediately active and non-volatile stored.

**HS\_PDO\_Index**        **Index:**            **5FE5h**  
**Object-Type:**    **Simple Variable (Var)**  
**Subindex:**        **0**  
**Data length:**    **1 Byte**  
**Significance:**    With this parameter it is specified which PDO should be the High-Speed-PDO.  
**Coding:**            0 : 1.PDO is High-Speed-PDO / 1 : 2.PDO is High-Speed-PDO  
**Note:**                A changed value is stored non-volatile, but it becomes active only after the next switch-on or Reset Node-command. Furthermore, a change in value causes all PDO's to be deactivated (Bit 31 of PDO parameter CobID = 1).

**GB**

**device type**  
(in accordance with CANopen (13))

**Index:** 1000h  
**Object-Type:** Simple Variable (Var)  
**Subindex:** 0  
**Data length:** 4 Byte  
**Significance:** Describes the unit type in accordance with the CANopen communication profile.  
**Coding:** Not specified at this point.  
**Standard Setting:** 0  
**Note:** This parameter is constant and can only be read.

**error register**  
(in accordance with CANopen (13))

**Index:** 1001h  
**Object-Type:** Simple Variable (Var)  
**Subindex:** 0  
**Data length:** 1 Byte  
**Significance:** Shows the error status of CANopen-devices.  
**Coding:** Bit0 : = 1 ==> error has occurred.  
**Standard Setting:** 0  
**Note:** This parameter can only be read. Every read out of this parameter returns the value to zero.

**Identify Object**

**Index:** 1018h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of entries in this object)  
**Data length:** 1 Byte  
**Significance:** Provides the number of entries in this object.  
**Coding:** 1  
**Standard Setting:** 2  
**Note:** The value of this parameter can only be read.  
**Subindex:** 1 (Vendor-ID)  
**Data length:** 4 Byte  
**Significance:** Manufacturer designation of CAN assigned in the automation user group.  
**Coding:** Bit31 . . . Bit24: Department  
 Bit23 . . . Bit0: Company  
**Standard Setting:** 00000014h  
**Note:** The value of this parameter can only be read.  
**Subindex:** 2 (Product code)  
**Data length:** 4 Byte  
**Significance:** Product significance  
**Coding:** 00000004h = Type F4  
 00000005h = Type F5  
**Standard Setting:** 00000005h  
**Note:** The value of this parameter can only be read.

<b>1st receive PDO Parameter</b>	<b>Index:</b>	<b>1400h</b>
	<b>Object-Type:</b>	<b>Structured Variable (Record)</b>
	<b>Subindex:</b>	<b>0 (Number of supported entries in the record)</b>
	<b>Data length:</b>	<b>1 Byte</b>
	<b>Significance:</b>	Provides the number of entries which can be addressed under this object.
	<b>Coding:</b>	1
	<b>Standard Setting:</b>	<b>2</b>
	<b>Note:</b>	The value of this parameter can only be read.
	<b>Subindex:</b>	<b>1 (COB-ID)</b>
	<b>Data length:</b>	<b>4 Byte</b>
	<b>Significance:</b>	Specifies, on which identifier the PDO(rx) is send for the transfer of the process-output data. Control information for this PDO is found in the upper Bits.
	<b>Coding:</b>	<p>Bit31(MSB) = 0 ==&gt; The processing of the process output data is activated.</p> <p>Bit31(MSB) = 1 ==&gt; The process-output data are not processed.</p> <p>Bit30 = 0 ==&gt; Remote Frame of the respective identifier is answered.</p> <p>Bit30 = 1 ==&gt; Remote Frame on the respective Identifier is not answered.</p> <p>Bit29 = 0 ==&gt; 11-Bit Identifier (CAN V2.0A)</p> <p>Bit29 = 1 ==&gt; 29-Bit Identifier (CAN V2.0B), here not adjustable. But 29-Bit identifier telegrams are received and processed.</p> <p>Bit28...Bit0: Identifier (Bit0 = LSB), here for Bit28 to Bit11 = fixed = 0.</p>
	<b>Standard Setting:</b>	<b>00000201h + Inverter-Address (SY.06)</b>
	<b>Note:</b>	A changed value is immediately active and non-volatile stored. At switch-on of the process data processing (Bit31 from "1" to "0") the setting of the parameter 1st Receive PDO Mapping (Index = 1600h) is transferred to the inverter control. If the FI-control does not accept the mapping, an error response is returned at this point and the process output data processing remains switched off. If the PD-mapping is accepted by the frequency inverter, it is automatically non-volatile stored and the process output data processing is enabled as desired.
	<b>Subindex:</b>	<b>2 (transmission type)</b>
	<b>Data length:</b>	<b>1 Byte</b>
	<b>Significance:</b>	Determines when and how this object is being sent on the CAN-Bus.
	<b>Coding:</b>	<p><u>0 (synchronous acyclic) or 1 (synchronous cyclic):</u> With receipt of a SYNC-command (identifier = 128d, data length = 0) the current process output data is transferred to the FI-control.</p> <p><u>254d (asynchronous, specified by manufacturer):</u> The process output data are completely transferred to the FI-control as soon as at least one byte has changed.</p> <p><u>255d (asynchronous, profile specific):</u> See asynchronous, specified by manufacturer.</p>
	<b>Note:</b>	A changed value is immediately active and non-volatile stored.

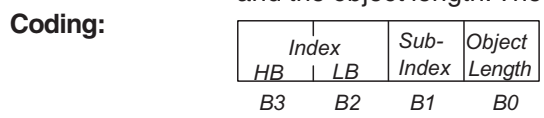
GB

<b>2nd receive PDO Parameter</b>	<p><b>Index:</b> 1401h  <b>Object-Type:</b> Structured Variable (Record)  <b>Subindex:</b> 0 (Number of supported entries in the record)  <b>Data length:</b> 1 Byte  <b>Significance:</b> Provides the number of entries which can be addressed under this object.  <b>Coding:</b> 1  <b>Standard Setting:</b> 2  <b>Note:</b> The value of this parameter can only be read.</p>
	<p><b>Subindex:</b> 1 (COB-ID)  <b>Data length:</b> 4 Byte  <b>Significance:</b> Specifies, on which identifier the PDO(rx) is send for the transfer of the process-output data. Control information for this PDO is found in the upper Bits.  <b>Coding:</b> Bit31 (MSB) = 0 ==&gt; The processing of the process output data is activated.                      Bit31 (MSB) = 1 ==&gt; The process-output data are not processed.                      Bit30 = 0 ==&gt; Remote Frame of the respective identifier is answered.                      Bit30 = 1 ==&gt; Remote Frame on the respective Identifier is not answered.                      Bit29 = 0 ==&gt; 11-Bit Identifier (CAN V2.0A)                      Bit29 = 1 ==&gt; 29-Bit Identifier (CAN V2.0B), here not adjustable. But 29-Bit identifier telegrams are received and processed.                      Bit28...Bit0: Identifier (Bit0 = LSB), here for Bit28 to Bit11 = fixed = 0.</p>
	<p><b>Standard Setting:</b> 80000301h + Inverter-Address (SY.06)  <b>Note:</b> A changed value is immediately active and non-volatile stored. When activating the process data processing (Bit 31 from "1" to "0") the adjustment of the 2nd receive PDO mapping is transferred to a corresponding inverter mapping. If this could successfully be completed, the mapping is automatically stored non-volatile.</p>
	<p><b>Subindex:</b> 2 (transmission type)  <b>Data length:</b> 1 Byte  <b>Significance:</b> Determines when and how this object is being sent on the CAN-Bus.  <b>Coding:</b> 0 (synchronous acyclic) or 1 (synchronous cyclic):                      With receipt of a SYNC-command (identifier = 128d, data length = 0) the current process output data is transferred to the FI-control.                      254d (asynchronous, specified by manufacturer):                      The process output data are completely transferred to the FI-control as soon as at least one byte has changed.                      255d (asynchronous, profile specific):                      See asynchronous, specified by manufacturer.</p>
	<p><b>Note:</b> A changed value is immediately active and non-volatile stored.</p>

## 1st receive PDO Mapping

**Index:** 1600h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of mapped objects in PDO)  
**Data length:** 1 Byte  
**Significance:** Provides the number of entries, which can be addressed in this object.  
**Coding:** 1 (maximum valid value range 1.....4).  
**Standard Setting:** 2  
**Note:** A writing of this parameter causes the automatic switch-off of the process output data processing (Bit31 of index 1400h, subindex = 1 is set to "1")

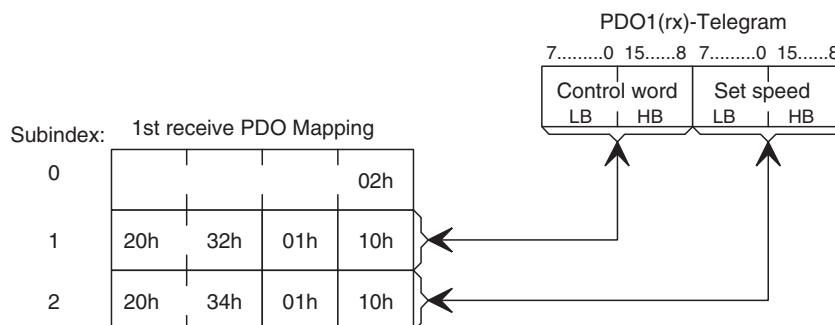
**Subindex:** 1 to maximum 4 (nth object to be mapped)  
**Data length:** 4 Byte  
**Significance:** Describes an object mapping. It consists of the index, subindex and the object length. The object length is specified in Bits.



**Standard-Setting:** See below.

**Note:** A writing of this parameter causes the automatic switch-off of the process output data processing (Bit31 of index 1400h, subindex=1 is set to "1").

The diagram below shows the relationship between the process-output data mapping and the respective PDO1(rx) telegram structure in the standard assignment.



*Example for coding of data on CAN-BUS:*

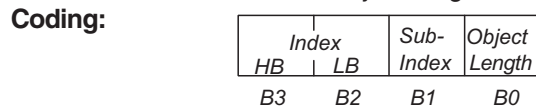
The first mapped parameter in the receive-PDO should be changed to the parameter with index = 2302h and subindex = 1. In this case the complete 8 data bytes of the Initiate Domain Download Request have to be set as follows:

23h	00h	16h	01h	10h	01h	02h	23h
B0	B1	B2	B3	B4	B5	B6	B7

**2nd receive PDO Mapping**

**Index:** 1601h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of mapped objects in PDO)  
**Data length:** 1 Byte  
**Significance:** Provides the number of entries, which can be addressed in this object.  
**Coding:** 1 (maximum valid value range 1.....4).  
**Standard Setting:** 2  
**Note:** A writing of this parameter causes the automatic switch-off of the process output data processing (Bit31 of index 1401h, subindex = 1 is set to "1")

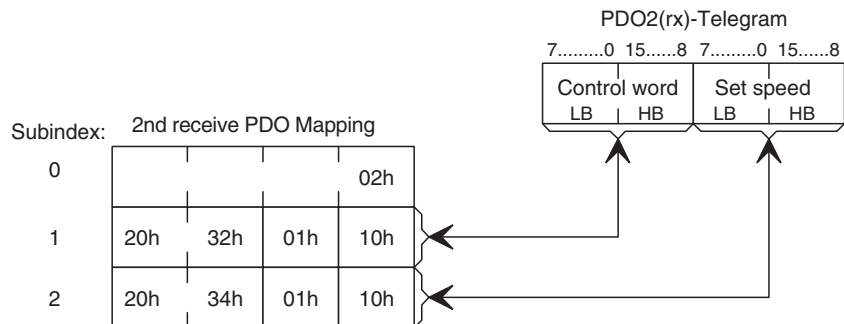
**Subindex:** 1 to maximum 4 (nth object to be mapped)  
**Data length:** 4 Byte  
**Significance:** Describes an object mapping. It consists of the index, subindex and the object length. The object length is specified in Bits.



**Standard-Setting:** See below.

**Note:** A writing of this parameter causes the automatic switch-off of the process output data processing (Bit31 of index 1401h, subindex=1 is set to "1").

The diagram below shows the relationship between the process-output data mapping and the respective PDO2(rx) telegram structure in the standard assignment.



<b>1st transmit PDO Parameter</b>	<b>Index:</b>	<b>1800h</b>
	<b>Object-Type:</b>	<b>Structured Variable (Record)</b>
	<b>Subindex:</b>	<b>0 (Number of supported entries in the record)</b>
	<b>Data length:</b>	1 Byte
	<b>Significance:</b>	Provides the number of entries in this object.
	<b>Coding:</b>	1
	<b>Standard Setting:</b>	<b>3</b>
	<b>Note:</b>	The value of this parameter can only be read.
	<b>Subindex:</b>	<b>1 (COB-ID)</b>
	<b>Data length:</b>	4 Byte
	<b>Significance:</b>	Specifies on which identifier the PDO(tx) is sent for the transfer of the process input data. Control information for this PDO is found in the upper Bits.
	<b>Coding:</b>	<p>Bit31(MSB) = 0 ==&gt; The processing of the process input data is active.</p> <p>Bit31(MSB) = 1 ==&gt; The process input data is not processed.</p> <p>Bit30 = 0 ==&gt; Remote Frame is answered on the respective identifier.</p> <p>Bit30 = 1 ==&gt; Remote Frame is not answered.</p> <p>Bit29 = 0 ==&gt; 11-Bit-Identifier (CAN V2.0A)</p> <p>Bit29 = 1 ==&gt; 29-Bit Identifier (CAN V2.0B), here not adjustable.</p> <p>Bit28...Bit0: Identifier (Bit0 = LSB), here for Bit28 to Bit11 = fixed = 0.</p>
	<b>Standard Setting:</b>	<b>0000181h + Inverter-Address (SY.06)</b>
	<b>Note:</b>	A changed value is immediately active and non-volatile stored. At switch-on of the process data processing (Bit31 from "1" nach "0") the setting of the parameter 1st transmit PDO Mapping (index 1A00h) is transferred to the inveter control. If the FI-control does not accept the mapping, an error response is returned at this point and the process output data processing remains switched off. If the PD-mapping is accepted by the frequency inverter, it is automatically non-volatile stored and the process output data processing is enabled as desired.
	<b>Subindex:</b>	<b>2 (transmission type)</b>
	<b>Data length:</b>	1 Byte
	<b>Significance:</b>	Determines when and how this object can be sent to the CAN-Bus.
	<b>Coding:</b>	<p><u>0 (synchronous acyclic) or 1 (synchronous cyclic):</u> With receipt of a SYNC-command (identifier = 128d, data length = 0) the current process input data are transmitted on CAN.</p> <p><u>254d (asynchronous, specified by manufacturer):</u> The process input data are transmitted on CAN as soon as at least one byte has changed.</p> <p><u>255d (asynchronous, profile specific):</u> See asynchronous, specified by manufacturer.</p>
	<b>Note:</b>	A changed value is immediately active and non-volatile stored.
	<b>Subindex:</b>	<b>3 (inhibit time)</b>
	<b>Data length:</b>	2 Byte
	<b>Significance:</b>	Indicates the minimum time interval between two CAN-telegrams on this identifier.
	<b>Coding:</b>	100 μs
	<b>Standard Setting:</b>	<b>150 (= 15 ms)</b>
	<b>Note:</b>	A changed value is immediately active and non-volatile stored. The internal resolution for the Inhibit-Time is 1ms. Thus the adjusted value has an inaccuracy of +/- 1ms.

GB

**2nd transmit PDO Parameter**

**Index:** 1801h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of supported entries in the record)  
**Data length:** 1 Byte  
**Significance:** Provides the number of entries in this object.  
**Coding:** 1  
**Standard Setting:** 3  
**Note:** The value of this parameter can only be read.  
**Subindex:** 1 (COB-ID)  
**Data length:** 4 Byte  
**Significance:** Specifies on which identifier the PDO(tx) is sent for the transfer of the process input data. Control information for this PDO is found in the upper Bits.  
**Coding:** Bit31 (MSB) = 0 ==> The processing of the process input data is active.  
 Bit31 (MSB) = 1 ==> The process input data is not processed.  
 Bit30 = 0 ==> Remote Frame is answered on the respective identifier.  
 Bit30 = 1 ==> Remote Frame is not answered.  
 Bit29 = 0 ==> 11-Bit-Identifier (CAN V2.0A)  
 Bit29 = 1 ==> 29-Bit Identifier (CAN V2.0B), here not adjustable.  
 Bit28...Bit0: Identifier (Bit0 = LSB), here for Bit28 to Bit11 = fixed = 0.  
**Standard Setting:** 80000281h + Inverter-Address (SY.06)  
**Note:** A changed value is immediately active and non-volatile stored. When activating the process data processing (Bit31 from "1" to "0") the adjustment of the 2nd transmit PDO mapping is transferred to a corresponding inverter mapping. If this could successfully be completed, the mapping is automatically stored non-volatile.  
**Subindex:** 2 (transmission type)  
**Data length:** 1 Byte  
**Significance:** Determines when and how this object can be sent to the CAN-Bus.  
**Coding:** 0 (synchronous acyclic) or 1 (synchronous cyclic):  
 With receipt of a SYNC-command (identifier = 128d, data length = 0) the current process input data are transmitted on CAN.  
 254d (asynchronous, specified by manufacturer):  
 The process input data are transmitted on CAN as soon as at least one byte has changed.  
 255d (asynchronous, profile specific):  
 See asynchronous, specified by manufacturer.  
**Note:** A changed value is immediately active and non-volatile stored.  
**Subindex:** 3 (inhibit time)  
**Data length:** 2 Byte  
**Significance:** Indicates the minimum time interval between two CAN-telegrams on this identifier.  
**Coding:** 100 μs  
**Standard Setting:** 1000 (= 100 ms)  
**Note:** A changed value is immediately active and non-volatile stored. The internal resolution for the Inhibit-Time is 1ms. Thus the adjusted value has an inaccuracy of +/- 1ms.

## 1st transmit PDO Mapping

**Index:** 1A00h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of mapped objects in PDO)  
**Data length:** 1 Byte  
**Significance:** Specifies the number of entries that can be addressed in this object.  
**Coding:** 1 (Maximum valid value range 1.....4).  
**Standard Setting:** 2  
**Note:** A writing of this parameter causes the automatic switch-off of the process input data processing (Bit31 from Index 1800h, Subindex = 1 is set to "1").

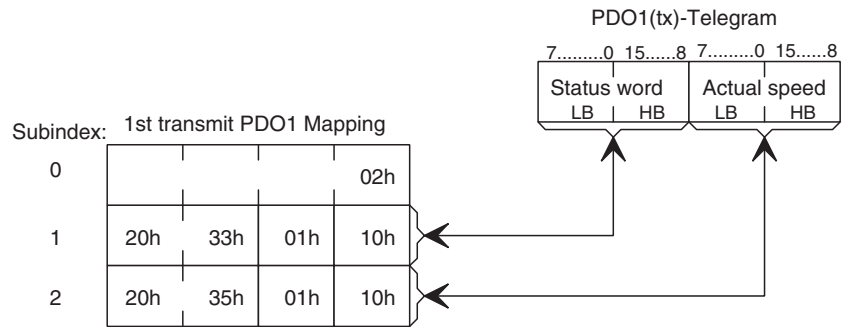
**Subindex:** 1 to maximum 4 (nth object to be mapped)  
**Data length:** 4 Byte  
**Significance:** Specifies the object mapping. It consists of the index, subindex and the object length. The object length is specified in Bits.



**Standard Setting:** See above

**Note:** A writing of this parameter causes the automatic switch-off of the process input data processing (Bit31 from Index 1800h, Subindex = 1 is set to "1").

The diagram below shows the relation between the process input data mapping and the respective PDO1(tx) telegram structure in the standard assignment.

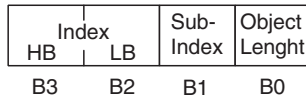


2nd transmit PDO Mapping

**Index:** 1A01h  
**Object-Type:** Structured Variable (Record)  
**Subindex:** 0 (Number of mapped objects in PDO)  
**Data length:** 1 Byte  
**Significance:** Specifies the number of entries that can be addressed in this object.  
**Coding:** 1 (Maximum valid value range 1.....4).  
**Standard Setting:** 2  
**Note:** A writing of this parameter causes the automatic switch-off of the process input data processing (Bit31 from Index 1801h, Subindex = 1 is set to "1").

**Subindex:** 1 to maximum 4 (nth object to be mapped)  
**Data length:** 4 Byte  
**Significance:** Specifies the object mapping. It consists of the index, subindex and the object length. The object length is specified in Bits.

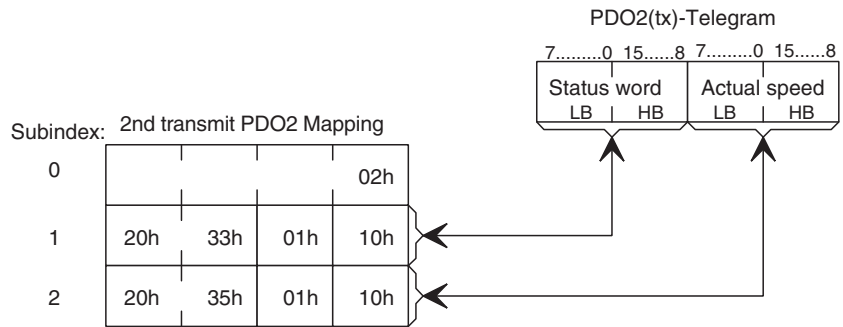
**Coding:**



**Standard Setting:** See above

**Note:** A writing of this parameter causes the automatic switch-off of the process input data processing (Bit31 from Index 1801h, Subindex = 1 is set to "1").

The diagram below shows the relation between the process input data mapping and the respective PDO2(tx) telegram structure in the standard assignment.



## 10. Access to Operator-Parameters via Diagnostic Interface

The parameters which the operator organizes are also called Operator-parameters. Some of these parameters are pure diagnostic parameters and partly of no interest to the user.

Other parameters are mirrored fieldbus parameters, which in normal operation are programmed over the fieldbus, but for the initial commissioning and for test purposes they can also be modified over the diagnostic interface.

At that it is important to know, that parameters exist which are coded differently on the fieldbus than on the diagnostic interface. The parameters for setting the process data assignment are an example for it.



We explicitly warn you not to change parameters parallel over the fieldbus as well as over the diagnostic interface. Unreproducible failure can be the consequence of such a procedure!!!

In the following the Operator-parameters are listed. All parameters visible in the KEB COMBIVIS but not specified in the following are only for KEB internal use and may not be changed by the user.

**Date Mmm DD YYYY Type:**  
(mit Mmm = month, DD = day,  
YYYY = year of the software  
date)

**Significance:**

Specifies the operator type and version. It concerns a 32-Bit-value, which consists of two 16-Bit-values. The high-word indicates the operator type, the Low-word indicates the version.

**Data length:**

4 Byte

**Coding:**

Operator type:

- 0001h: Standard-Operator
- 0002h: Interbus-Operator
- 0003h: Profibus-Operator
- 0004h: CANopen-Operator

Example: 00040001h means, it concerns a CANopen-Operator of the version 1.

**Note:**

This parameter is Read\_Only and not available via CAN.

**Parameter Count**

**Significance:**

Number of available Operator-parameters.

**Data length:**

1 Byte

**Coding:**

1

**Note:**

This parameter is Read\_Only and not available via CAN.

**Response Delay Time**

**Significance:**

Minimum response delay for inquiries over the diagnostic interface.

**Data length:**

1 Byte

**Coding:**

1 ms

**Note:**

This parameter is not available via CAN.

**Current Password**

**Significance:**

Currently adjusted Password level

**Data length:**

2 Byte

**Coding:**

See coding of the parameter password in the FI-control.

**Note:**

This parameter is Read\_Only and not available via CAN.

<b>HSP5_Max_InvBusy_Retries</b>	<b>Significance:</b>	Indicates how often a HSP5-service to the inverter control is repeated, if the inverter rejects the service with error 'Inverter busy'.
	<b>Data type:</b>	Unsigned8
	<b>Coding:</b>	1
<b>HSP5Tout Count</b>	<b>Significance:</b>	Counts the time overrun at the internal communication between the operator and the FI-control.
	<b>Data length:</b>	2 Byte
	<b>Coding:</b>	1
	<b>Note:</b>	This parameter is not available via CAN.
<b>Watchdog inhibit</b>	<b>Data length:</b>	1 Byte
	<b>Note:</b>	This parameter is identical to the CAN-parameter WD_Inhibit (see above).
<b>ActPDO_Index</b>	<b>Significance:</b>	With this parameter it is possible to toggle between the PDO's in the Combivis parameter view. The following parameters apply then in each case to the selected PDO.
	<b>Data type:</b>	Unsigned8
	<b>Coding:</b>	0 : 1.PDO 1 : 2.PDO
<b>PD_In_Para_CobID</b>	<b>Data length:</b>	4 Byte
	<b>Note:</b>	This parameter corresponds to the CAN-parameter „nth transmit PDO Parameter, Cob ID“ *
<b>PD_In_Para_TxType</b>	<b>Data length:</b>	1 Byte
	<b>Note:</b>	This parameter corresponds to the CAN-parameter „nth transmit PDO Parameter, TxType“ *
<b>PD_In_Para_Inhibit</b>	<b>Data length:</b>	2 Byte
	<b>Note:</b>	This parameter corresponds to the CAN-parameter „nth transmit PDO Parameter, Inhibit Time“ *
<b>PD_In_Cycle</b>	<b>Note:</b>	This parameter corresponds to the CAN-parameter PDIN_Cycle_Time *

\* Which PDO is addressed depends on the current value of the parameter ActPDO\_Index.

<b>Nr_PDIn_Objs</b>	<b>Data length:</b> 1 Byte <b>Note:</b>	This parameter corresponds to the least significant byte (LSB) of the CAN-parameter „nth transmit PDO Mapping, Nr Mapped Objects“ *
<b>PD_Inx Index (where x = 1...4)</b>	<b>Data length:</b> 2 Byte <b>Note:</b>	This parameter corresponds to the most significant word of the parameter nth transmit PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>PD_Inx Set (where x = 1 . . . 4)</b>	<b>Data length:</b> 1 Byte <b>Note:</b>	This parameter corresponds to the third-most significant byte of the parameter nth transmit PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>PD_Inx _BitDlen (where x=1...4)</b>	<b>Note:</b>	This parameter corresponds to the low byte of the parameter nth transmit PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>PD_Out_Para_CobID</b>	<b>Data length:</b> 4 Byte <b>Note:</b>	This parameter corresponds to the CAN-parameter „nth Receive PDO Parameter, Cob ID“ *
<b>PD_Out_Para_TxType</b>	<b>Data length:</b> 1 Byte <b>Note:</b>	This parameter corresponds to the CAN-parameter „nth Receive PDO Parameter, TxType“ *
<b>Nr_PDOut_Objs</b>	<b>Data length:</b> 1 Byte <b>Note:</b>	This parameter corresponds to the least significant byte (LSB) of the parameter nth Receive PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>PD_Outx Index (where x = 1...4)</b>	<b>Data length:</b> 2 Byte <b>Note:</b>	This parameter corresponds to the most significant word of the parameter nth Receive PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>PD_Outx Set (where x = 1 ... 4)</b>	<b>Data length:</b> 1 Byte <b>Note:</b>	This parameter corresponds to the third-most significant byte of the parameter nth Receive PDO Mapping, PDO Mapping for the nth application object to be mapped *

**GB**

<b>PD_Outx_BitDlen (where x = 1...4)</b>	<b>Note:</b>	This parameter corresponds to the least significant byte of the parameter nth Receive PDO Mapping, PDO Mapping for the nth application object to be mapped *
<b>ProcessData Inx (where x = 1...4)</b>	<b>Significance:</b> <b>Data length:</b> <b>Coding:</b> <b>Note:</b>	x. process input data word 2 Byte Depending on mapped parameter This parameter is Read_Only and corresponds to the x.word of the PDO (tx) - telegram on CAN.
<b>ProcessData Outx (where x=1...4)</b>	<b>Significance:</b> <b>Data length:</b> <b>Coding:</b> <b>Note:</b>	x. process output data word 2 Byte Depending on mapped parameter This parameter is Read_Only and corresponds to the x.word of the PDO (rx) - telegram on CAN.
<b>Take Stored PD-Map</b>	<b>Note:</b>	This parameter corresponds to the CAN parameter PD_Stored (Index = 5FE2h), (s. o.).
<b>Check PD Setting</b>	<b>Significance:</b> <b>Data length:</b> <b>Coding:</b> <b>Note:</b>	Indicates whether the PD-assignment change adjusted last was executed error-free. 1 Byte 0: Error occurred in the last PD-assignment change. 255d: Last PD-assignment change was executed error-free. This parameter is not available on CAN.
<b>CAN_Baud2</b>	<b>Data length:</b> <b>Note:</b>	1 Byte This parameter corresponds to the CAN-parameter CAN_Baud2 (see above).
<b>Act_CAN_Baud</b>	<b>Significance:</b> <b>Data length:</b> <b>Coding:</b> <b>Note:</b>	Shows the currently adjusted CAN-Bit rate. It differs only from the value of the parameter CAN_BAUD if the automatic bit rate detection is activated on CAN (CAN_Baud = 255d bzw. (-1)). 1 Byte see CAN_Baud without the value 255d or (-1). This parameter is Read_Only and not available on CAN.
<b>HS_PDO_Index</b>	<b>Note:</b>	This parameter corresponds to the configuration parameter HS_PDO_Index.

### 11. Changing the transmission-type of the PDOs

The transmission-type of parameter **1st/2nd receive PDO Parameter** and **1st/2nd transmit PDO Parameter** is changeable. Valid values are:

- Asynchronous manufacturer-specific (Value = 254 = Standard) and
- Asynchronous profile-specific (Value = 255)
- Synchronous acyclic (Value = 0)
- Synchronous cyclic (Value = 1).

This chapter describes the difference between the adjustments.

GB

#### 11.1 Asynchronous manufacturer-specific (Value = 254d/FEh) or Asynchronous profile-specific (Value = 255d/FFh)

If this value of the transmission-type is adjusted in parameter **1st/2nd receive PDO Parameter**, the process output data in OPERATIONAL state are transferred to the inverter control upon receiving a valid PDO(rx)-telegram, if at least one value has changed. A valid PDO(rx) telegram is a telegram on the corresponding identifier with a data length of  $\geq$  the data length which results from the PDO(rx)-Mapping. Standardly that means all telegrams on the OUT-identifier with a data length of  $\geq 4$  Byte are accepted.

In OPERATIONAL state the process input data are read cyclicly from the FI-control. If in parameter **1st/2nd transmit PDO Parameter** the value 254d or 255d is adjusted a PDO(tx)-telegram is sent on the CAN, if the process input data have changed.

#### 11.2 Synchronous acyclic (Value = 0) or synchronous cyclic (Value = 1)

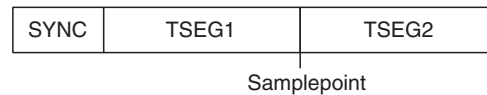
If in parameter **1st/2nd receive PDO Parameter** the transmission type is adjusted to one of these values, the process output data in OPERATIONAL status are transferred to the inverter control upon receiving a SYNC-telegram. Condition: A valid PDO(rx)-telegram was received first.

For parameter **1st/2nd transmit PDO Parameter** the value transmission-type = 0/1 means that in OPERATIONAL state a PDO(tx)-telegram is sent on the CAN immediately after the receipt of a SYNC-telegram .

## 12. Annex

### 12.1 CAN-Bit-Timing

The KEB-CAN interface adheres to the adjusted Bit-timings defined by the CIA standards (see [2] in index literature):  
The nominal Bit-timing is as follows:



The following is valid for all adjustable baudrates:

- $t_q$ : Base time unit. All segments of the Bit-timing result in a multiple of this time unit.
- SYNC: = 0 ==> Only the edges which go from recessive to dominant are used for synchronization.
- SJW: = 0 ==> Synchronization jump width =  $1 * t_q$
- TSEG2: = 1 ==>  $t_{SEG2} = 2 * t_q$

GB

Baudrate	Time-Quantum ( $t_q$ )	Sample-Point	TSEG1
10 Kbit/s	6.25 $\mu$ s	$14 * t_q = 87.5 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
20 Kbit/s	3.125 $\mu$ s	$14 * t_q = 43.75 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
25 Kbit/s	2,5 $\mu$ s	$14 * t_q = 35,0 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
50 Kbit/s	1.25 $\mu$ s	$14 * t_q = 17.5 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
100 Kbit/s	625 ns	$14 * t_q = 8.75 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
125 Kbit/s	500 ns	$14 * t_q = 7.0 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
250 Kbit/s	250 ns	$14 * t_q = 3.5 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
500 Kbit/s	125 ns	$14 * t_q = 1.75 \mu$ s	12 ==> $t_{SEG1} = 13 * t_q$
800 Kbit/s	125 ns	$8 * t_q = 1,0 \mu$ s	6 ==> $t_{SEG1} = 7 * t_q$
1000 Kbit/s	125 ns	$6 * t_q = 750$ ns	4 ==> $t_{SEG1} = 5 * t_q$

The transmission speeds marked grey in the table must be considered as very critical in relation to the possible line length.

### 12.1.1 Important Warning Information



The KEB-CAN-Module has a potential isolated CAN-Interface. The line length or the possible transmission speed is reduced by additional delay elements (optocoupler) in the signal line. The possible line length or transmission speed is dependent on the delay times of all users in the CAN-network. It is up to the user to assess which bitrate is possible for the line length. The necessary information for the KEB-CAN-Interface is listed in the following:

Sum of the input/output delay times

of the CAN-controller: ..... ≤ 36 ns.

Transmission-delay time of the CAN-driver: ..... ≤ 80 ns.

Receiving-delay time of the CAN-driver: ..... ≤ 70 ns.

Transmission-delay time of the used optocoupler: ..... ≤ 40 ns.

Receiving-delay time of the used optocoupler: ..... ≤ 40 ns.

**In any case the smallest CAN-transmission speed should be used which is required by the process.**

### 12.2 Literature Index

- [1]: Instruction Manual KEB COMBIVERT F5 with Application Manual.
- [2]: Document of Specifications of Working Group Physical-Layer the **CAN in Automation (CiA) User Group** : CiA/DS 102-1. Publisher: CiA International Users and Manufacturers Group Association, Am Weichselgarten 26, D-91058 Erlangen. Documents of Specification of Working Group Higher-Layer-Protocols of the CiA (Publisher see above).
- [3]: CiA/WG2/DS201 : CAN in the OSI Reference Model.
- [4]: CiA/WG2/DS202-1 : CMS Service Specification.
- [5]: CiA/WG2/DS202-2 : CMS Protocol Specification.
- [6]: CiA/WG2/DS202-3 : CMS Encoding Rules.
- [7]: CiA/WG2/DS203-1 : NMT Service Specification.
- [8]: CiA/WG2/DS203-2 : NMT Protocol Specification.
- [9]: CiA/WG2/DS204-1 : DBT Service Specification.
- [10]: CiA/WG2/DS204-2 : DBT Protocol Specification.
- [11]: CiA/WG2/DS207 : Application Layer Naming Conventions.
- [12]: CiA/DS301 V.4.01 : Application Layer and Communication Profile from 01.06.2000.

12.3 Table of configuration parameters according to CANopen

Index	Name	Object Type	Subindex	Data Length in Byte	Data Type	Acc.
5FFFh	CAN_Baud	VAR	0	1	UNSIGNED8	rw
5FFEh	SAVE_CAN_Baud	VAR	0	1	UNSIGNED8	rw
5FECh	CAN_Baud2	VAR	0	1	UNSIGNED8	rw
5FF9h	WD_Inhibit	VAR	0	1	UNSIGNED8	rw
5FE2h	PD_Stored	VAR	0	1	UNSIGNED8	rw
5FE5h	HS_PDO_Index	VAR	0	1	UNSIGNED8	rw
5FE6h	PDIN1_Cycle_Time	VAR	0	2	UNSIGNED16	rw
5FE7h	PDIN2_Cycle_Time	VAR	0	2	UNSIGNED16	rw
1000h	Device type	VAR	0	4	UNSIGNED32	ro
1001h	Error register	VAR	0	1	UNSIGNED8	ro
1018h	Identify Object	RECORD			IDENTIFY	
1018h	Number of supported entries	VAR	0	1	UNSIGNED8	ro
1018h	Vendor_ID	VAR	1	4	UNSIGNED32	ro
1018h	Product code	VAR	2	4	UNSIGNED32	ro
1400h	1st receive PDO Parameter	RECORD			PDOCommPar	
1400h	Number of supported entries	VAR	0	1	UNSIGNED8	ro
1400h	COB-ID	VAR	1	4	UNSIGNED32	rw
1400h	transmission type	VAR	2	1	UNSIGNED8	rw
1401h	2nd receive PDO Parameter	RECORD			PDOCommPar	
1401h	Number of supported entries	VAR	0	1	UNSIGNED8	ro
1401h	COB-ID	VAR	1	4	UNSIGNED32	rw
1401h	transmission type	VAR	2	1	UNSIGNED8	rw
1600h	1st receive PDO Mapping	RECORD			PDOMapping	
1600h	Number of mapped objects	VAR	0	1	UNSIGNED8	rw
1600h	nth object to be mapped	VAR	1 - max. 4	4	UNSIGNED32	rw
1601h	2nd receive PDO Mapping	RECORD			PDOMapping	
1601h	Number of mapped objects	VAR	0	1	UNSIGNED8	rw
1601h	nth object to be mapped	VAR	1 - max. 4	4	UNSIGNED32	rw
1800h	1st transmit PDO Parameter	RECORD			PDOCommPar	
1800h	Number of supported entries	VAR	0	1	UNSIGNED8	ro
1800h	COB-ID	VAR	1	4	UNSIGNED32	rw
1800h	transmission type	VAR	2	1	UNSIGNED8	rw
1800h	Inhibit time	VAR	3	2	UNSIGNED16	rw
1801h	2nd transmit PDO Parameter	RECORD			PDOCommPar	
1801h	Number of supported entries	VAR	0	1	UNSIGNED8	ro
1801h	COB-ID	VAR	1	4	UNSIGNED32	rw
1801h	transmission type	VAR	2	1	UNSIGNED8	rw
1801h	Inhibit time	VAR	3	2	UNSIGNED16	rw
1A00h	1st transmit PDO Mapping	RECORD			PDOMapping	
1A00h	Number of mapped objects	VAR	0	1	UNSIGNED8	rw
1A00h	nth object to be mapped	VAR	1 - max. 4	4	UNSIGNED32	rw
1A01h	2nd transmit PDO Mapping	RECORD			PDOMapping	
1A01h	Number of mapped objects	VAR	0	1	UNSIGNED8	rw
1A01h	nth object to be mapped	VAR	1 - max. 4	4	UNSIGNED32	rw

GB



## 12.4 Compact Summary of CAN-Communication

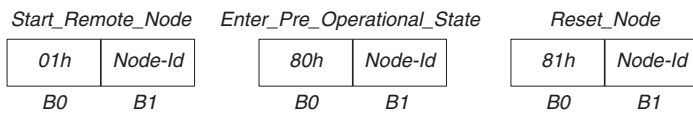
**Module-Id = Node-Id = inverter address (SY.06) + 1**

**Fixed identifier allocation:**

SDO(rx)-Identifier = 1537 + SY.06 : SDO-request to KEB-FI  
 SDO(tx)-Identifier = 1409 + SY.06 : SDO-confirmation from KEB-FI  
 PDO1(rx)-Identifier = 513 + SY.06 : process data to KEB-FI  
 PDO1(tx)-Identifier = 385 + SY.06 : process data from KEB-FI  
 PDO2(rx)-Identifier = 769 + SY.06 : process data to KEB-FI  
 PDO2(tx)-Identifier = 641 + SY.06 : process data from KEB-FI  
 Node-Guarding-Identifier = 1793 + SY.06

GB

**The most important NMT-commands (Telegrams) on identifier = 0:**



**The most important values of the Node-State.**

PRE\_ OPERATIONAL = 7Fh : communication active except the PDOs  
 OPERATIONAL = 05h : communication completely active

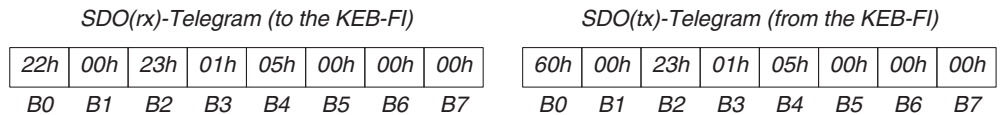
**Example for SDO-communication:**

Reading of parameter *Digital setpoint frequency setting* (op.03)  
 in Set 4 ==> Index = 2303h, Subindex = 10h



In this example the read value is = 1000 (03E8h)

Writing value = 5 to parameter *Setpoint source* (op.00)  
 in set 0 ==> Index = 2300h, Subindex = 01h

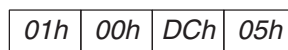


**Example for the setting of new process data with the PDO1(rx)-Telegram:**

Here the standard process data assignment is assumed.

The parameter *Control word* (SY.50) shall receive the value = 1 ,  
 the parameter *Setpoint speed* (SY.52) shall receive the value = 1500 (05DCh)

*PDO1(rx)-Telegram (to the KEB-FI)*







### **Karl E. Brinkmann GmbH**

Försterweg 36 - 38 • D - 32683 Barntrup  
Telefon 00 49 / 52 63 / 4 01 - 0 • Fax 00 49 / 52 63 / 4 01 - 1 16  
Internet: www.keb.de • E-mail: info@keb.de

### **KEB Antriebstechnik GmbH & Co. KG**

Wildbacher Str. 5 • D - 08289 Schneeberg  
Telefon 0049 / 37 72 / 67 - 0 • Telefax 0049 / 37 72 / 67 - 2 81  
E-mail: info@keb-combidrive.de

### **KEB Antriebstechnik Austria GmbH**

Ritzstraße 8 • A - 4614 Marchtrenk  
Tel.: 0043 / 7243 / 53586 - 0 • FAX: 0043 / 7243 / 53586 - 21  
Kostelni 32/1226 • CZ - 370 04 České Budejovice  
Tel.: 00420 / 38 / 731 92 23 • FAX: 00420 / 38 / 733 06 97  
E-mail: info@keb.at

### **KEB Antriebstechnik**

Herenveld 2 • B - 9500 Geraadsbergen  
Tel.: 0032 / 5443 / 7860 • FAX: 0032 / 5443 / 7898  
E-mail: koen.detaeye@keb.de

### **KEB China**

Xianxia Road 299 • CHN - 200051 Shanghai  
Tel.: 0086 / 21 / 62350922 • FAX: 0086 / 21 / 62350015  
Internet: www.keb-cn.com • E-mail: info@keb-cn.com

### **Société Française KEB**

Z.I. de la Croix St. Nicolas • 14, rue Gustave Eiffel  
F - 94510 LA QUEUE EN BRIE  
Tél.: 0033 / 1 / 49620101 • FAX: 0033 / 1 / 45767495  
E-mail: sfkeb.4@wanadoo.fr

### **KEB (UK) Ltd.**

6 Chieftain Business Park, Morris Close  
Park Farm, Wellingborough, GB - Northants, NN8 6 XF  
Tel.: 0044 / 1933 / 402220 • FAX: 0044 / 1933 / 400724  
Internet: www.keb-uk.co.uk • E-mail: info@keb-uk.co.uk

### **KEB Italia S.r.l.**

Via Newton, 2 • I - 20019 Settimo Milanese (Milano)  
Tel.: 0039 / 02 / 33500782 • FAX: 0039 / 02 / 33500790  
Internet: www.keb.it • E-mail: kebitalia@keb.it

### **KEB - YAMAKYU Ltd.**

15 - 16, 2 - Chome, Takanawa Minato-ku  
J - Tokyo 108 -0074  
Tel.: 0081 / 33 / 445-8515 • FAX: 0081 / 33 / 445-8215  
E-mail: kebjt001@d4.dion.ne.jp

### **KEB Portugal**

Lugar de Salgueiros - Pavilhao A, Mouquim  
P - 4760 V. N. de Famalicao  
Tel.: 00351 / 252 / 371 318 • FAX: 00351 / 252 / 371 320  
E-mail: keb.portugal@netc.pt

### **KEB Taiwan Ltd.**

1F, No.19-5, Shi Chou Rd., Tounan Town  
R.O.C. - Yin-Lin Hsian / Taiwan  
Tel.: 00886 / 5 / 5964242 • FAX: 00886 / 5 / 5964240  
E-mail: keb\_taiwan@mail.apol.com.tw

### **KEBCO Inc.**

1335 Mendota Heights Road  
USA - Mendota Heights, MN 55120  
Tel.: 001 / 651 / 4546162 • FAX: 001 / 651 / 4546198  
Internet: www.kebco.com • E-mail: info@kebco.com