CAN open configuration FAQ No.0004

Part	Version	Revision	Date	Status
en	6.2.3.0	001	2019-01-01	Released

Content

Introduction 2
Basic CAN information 2
C6 Interface 2
C6 Econ / Perform Runtime Configuration 2
C6 CAN Master 3
Adding Devices 3
Install eds file 4
CANopen Manager configuration 5
C6 CANopen Slave configuration 6
PDO (Process Data Object) configuration 7
SDO (Service Data Object) configuration 8
Additional C6 Compact configuration 10
C6 CAN Slave 12
Adding Devices 12
Sample Code 14
PLC_PRG 14
GVL UserODList 15
GVL UserVars 19
SDO Parameters for C6 Slave 20
Disclaimer 25

KEB Automation KG
Südstraße 38
32683 Barntrup, Germany
E-mail: info@keb.de
Tel. +49 5263 401-0
Fax +49 5263 401-116
www.keb.de

District Court Lemgo HRA 5649
DUNS-No. 314108728
VAT-No. DE309087075
Bank Details: Sparkasse Paderborn-Detmold
IBAN DE 19 4765 0130 0000 0060 07
BIC WELADE3L

General Partner:
Vittorio Tavella
KEB Verwaltungs-GmbH, Barntrup
District Court: Lemgo HRB 8965
Directors: Curt Bauer CMO,
Ralf Lutter COO, Vittorio Tavella CFO,
Wolfgang Wiele CTO

Introduction

This HowTo describes the basic setup to use KEB C6 controls as a CAN master or can slave. The basic setup will include some basic information on the CAN bus, PDO and SDO configuration, synchronisation, heartbeat and node guarding.

Basic CAN information

66This chapter provides some basic information about the CANopen fieldbus system.

C6 Interface

To use the CAN fieldbus you must have some special hardware installed. When using a C6 make sure there is a CAN extension card installed.
The extension for C6 Econ / Perform provides two D-sub connectors of type **DE-9M**.
The extension for C6 Compact provides one D-sub connector of type **DE-9F**.

The pin mapping is CANopen compatible. The first and last device on the fieldbus must use a 120Ω resistor for bus termination.

Connector pin	Description
2	CAN_L Bus line
7	CAN_H Bus line
5	CAN cable shield
Case	CAN cable shield

C6 Econ / Perform Runtime Configuration

When you have a C6 Econ or Perform you have to change the configuration file of the runtime plc. Go to the installation folder of the runtime (should be *C:\Program Files\3S CoDeSys\CoDeSys Control RTE3*) and rename the file “**CoDeSysControl.cfg**” to "**CoDeSysControl_original.cfg**" (or something similar). Then rename the "**Econ-SM-Can-CoDeSysControl.cfg**" to “**CoDeSysControl.cfg**” and restart the Runtime. Now you should be able to use the CAN interface properly.

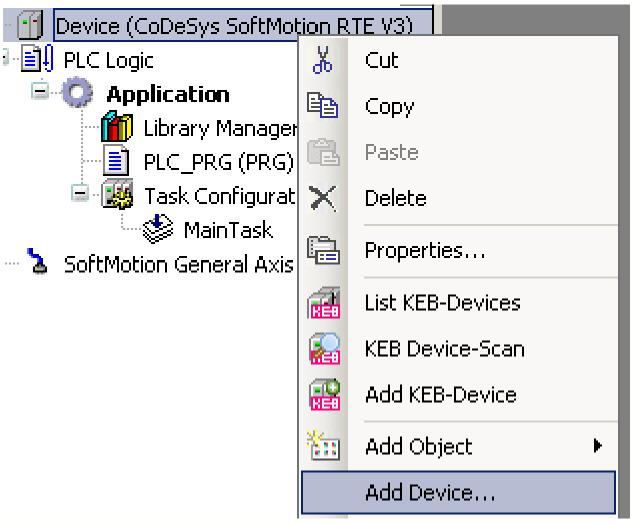
C6 CAN Master

This chapter will describe how you configure your C6 as CAN master device. There are some differences between the C6 compact and the C6 Econ/Perform.
There are some sample projects installed with Combivis studio 6.

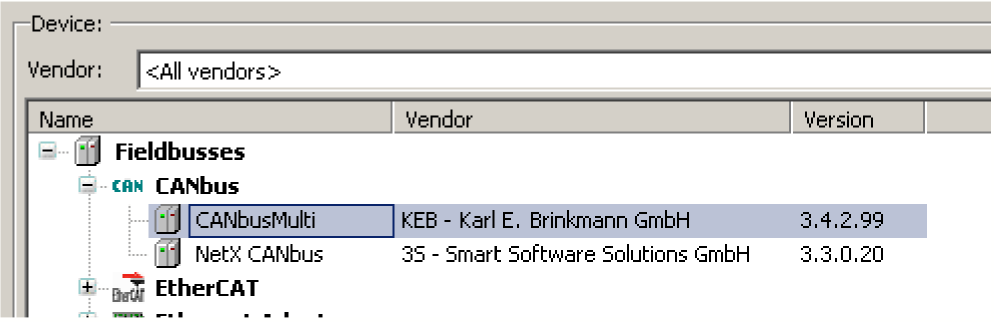
This chapter describes the setup of a C6 Econ as CAN master with one CAN slave (also C6), starting with a default project.

Adding Devices

First right click on the PLC device and add a new device.

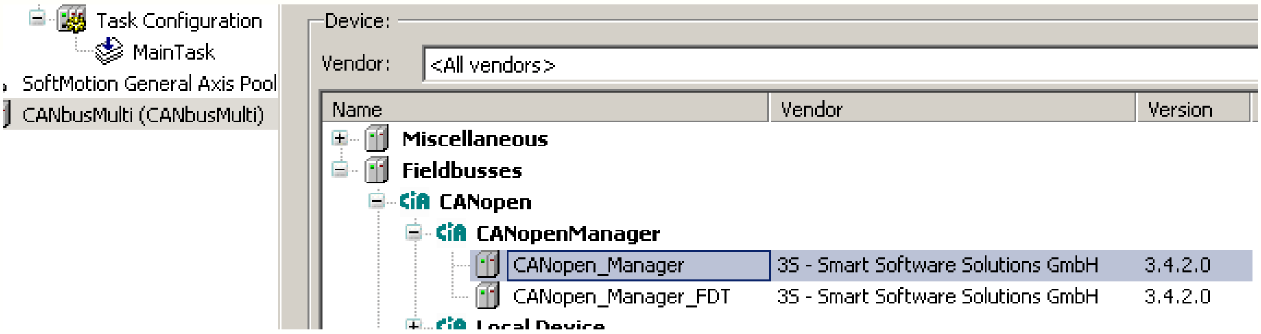


Then select the CANbusMulti Device from your device repository and add it to your project.



Name	Vendor	Version
Fieldbusses		
CANbus		
CANbusMulti	KEB - Karl E. Brinkmann GmbH	3.4.2.99
NetX CANbus	3S - Smart Software Solutions GmbH	3.3.0.20
EtherCAT		

Now we need to add a CANopen_Manager object to the CANbusMulti device. You can simply do that with left clicking on the recently added CANbus device if the add device dialog is still open.



Task Configuration
MainTask
SoftMotion General Axis Pool
CANbusMulti (CANbusMulti)

Device:

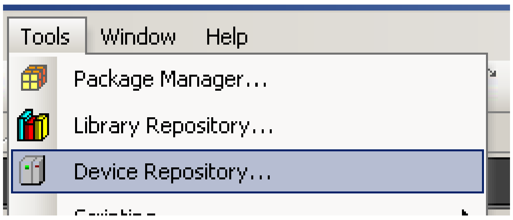
Vendor: <All vendors>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
CANopen		
CANopenManager		
CANopen_Manager	3S - Smart Software Solutions GmbH	3.4.2.0
CANopen_Manager_FDT	3S - Smart Software Solutions GmbH	3.4.2.0
Local Device		

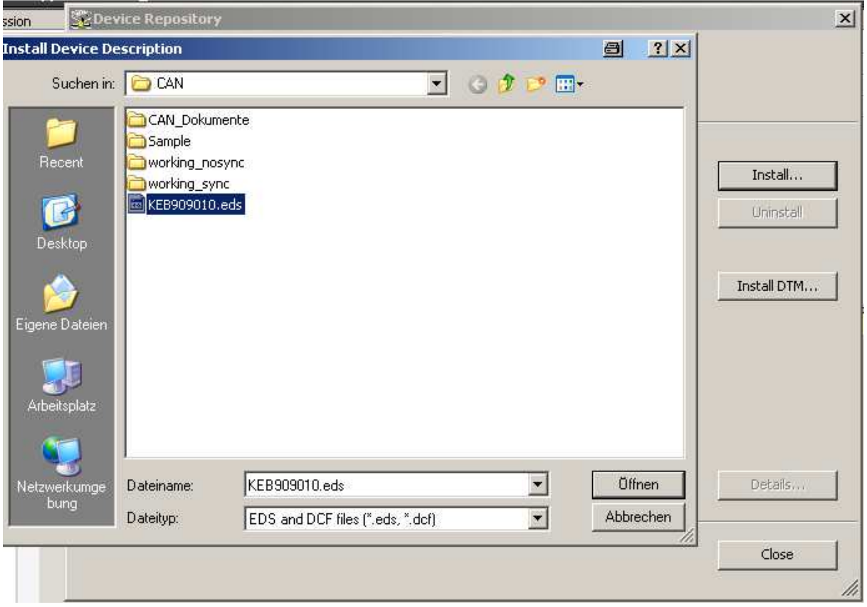
Now we need to add a slave to the Manager device. When you are using a C6 as slave device you need to install the provided .eds description file into your device repository.

Install eds file

To install an eds file you have to open the Device repository (In Combivis studio click Tools and the Device Repository)



Choose install and change the file type to EDS and DCF. Then select the KEB909010.eds.



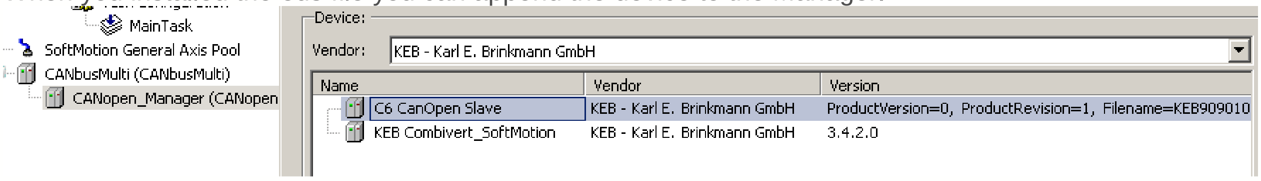
Suchen in: CAN

CAN_Dokumente
Sample
working_nosync
working_sync
KEB909010.eds

Dateiname: KEB909010.eds

Dateityp: EDS and DCF files (*.eds, *.dcf)

When you installed the eds file you can append the device to the Manager.



MainTask
SoftMotion General Axis Pool
CANbusMulti (CANbusMulti)
CANopen_Manager (CANopen

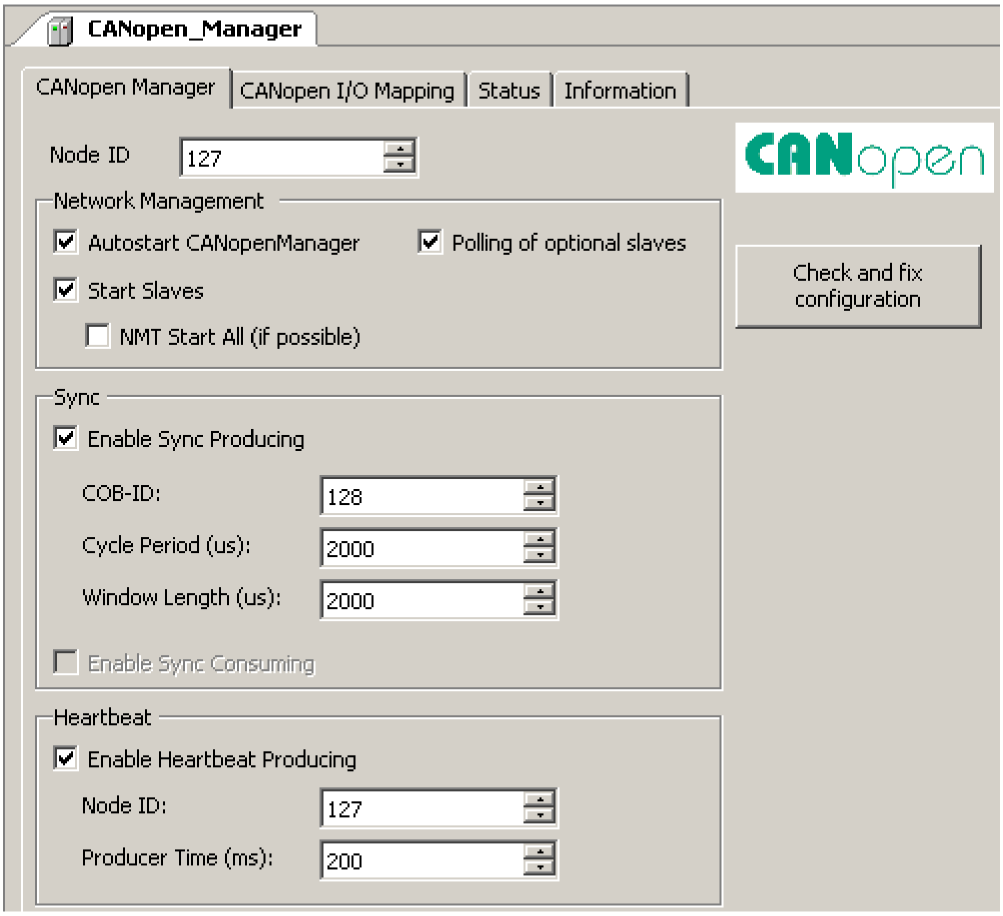
Device:

Vendor: KEB - Karl E. Brinkmann GmbH

Name	Vendor	Version
C6 CanOpen Slave	KEB - Karl E. Brinkmann GmbH	ProductVersion=0, ProductRevision=1, Filename=KEB909010
KEB Combivert_SoftMotion	KEB - Karl E. Brinkmann GmbH	3.4.2.0

CANopen Manager configuration

Now open the CANopen_Manager to edit the configuration.



CANopen_Manager

Node ID: 127

Network Management

☑ Autostart CANopenManager

☑ Polling of optional slaves

☑ Start Slaves

□ NMT Start All (if possible)

Sync

☑ Enable Sync Producing

COB-ID: 128

Cycle Period (us): 2000

Window Length (us): 2000

□ Enable Sync Consuming

Heartbeat

☑ Enable Heartbeat Producing

Node ID: 127

Producer Time (ms): 200

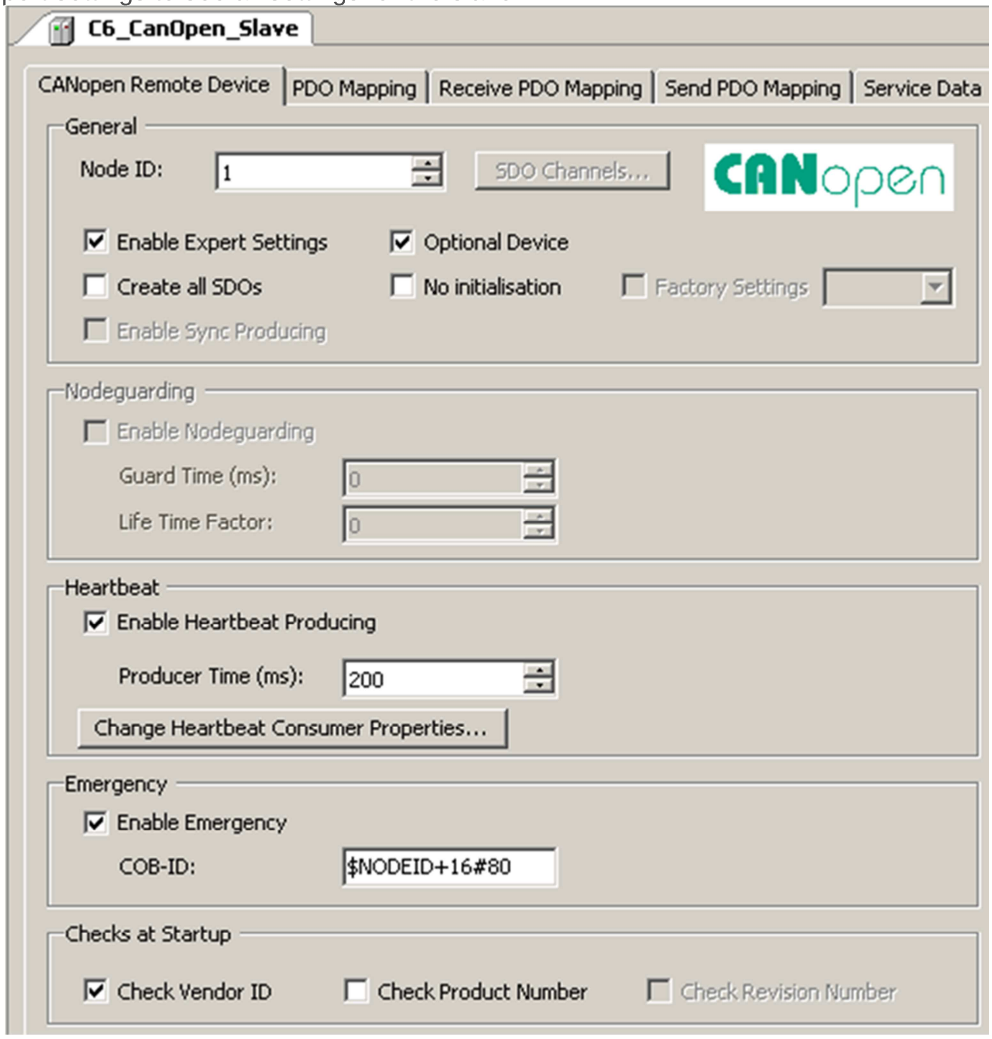
The CANopen_Manager is the Master device and controls some basic CAN features.

When you want to use synchronous communication you have to enable sync producing.
Hint: Make sure the PLC cycle time is the same as the cycle period you set here in the manager.

To check for slave fails you can enable heartbeat producing or NodeGuarding. The heartbeat is sent to all slaves and slaves send an answer to it. With this feature the master can detect if any slave failed and on the other hand the slaves can also check if there is some failure with the CAN master. The producer time is the time interval in milliseconds for the heartbeat.

C6 CANopen Slave configuration

Now you need to configure the slaves that are appended to the master.
Enable expert settings to see all settings for the slave.



C6_CanOpen_Slave

General

Node ID: 1

☑ Enable Expert Settings ☑ Optional Device

☐ Create all SDOs ☐ No initialisation ☐ Factory Settings

☐ Enable Sync Producing

Nodeguarding

☐ Enable Nodeguarding

Guard Time (ms): 0

Life Time Factor: 0

Heartbeat

☑ Enable Heartbeat Producing

Producer Time (ms): 200

Change Heartbeat Consumer Properties...

Emergency

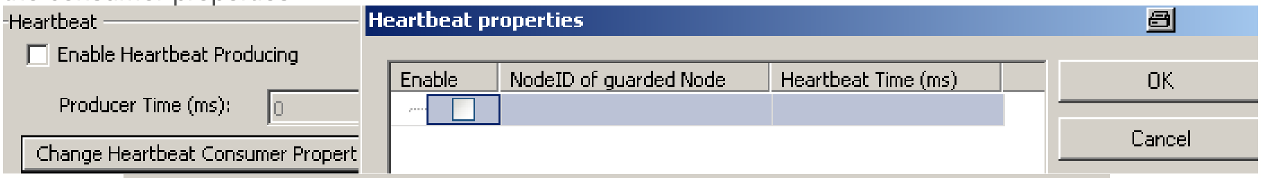
☑ Enable Emergency

COB-ID: $NODEID+16#80

Checks at Startup

☑ Check Vendor ID ☐ Check Product Number ☐ Check Revision Number

You have two possibilities for bus monitoring. One method is the heartbeat producing (must be enabled in master and slave with identic producer time). The second method is the so called Nodeguarding. To enable the checkboxes for Nodeguarding you must uncheck “Enable Heartbeat Producing” and disable the consumer properties.



Heartbeat

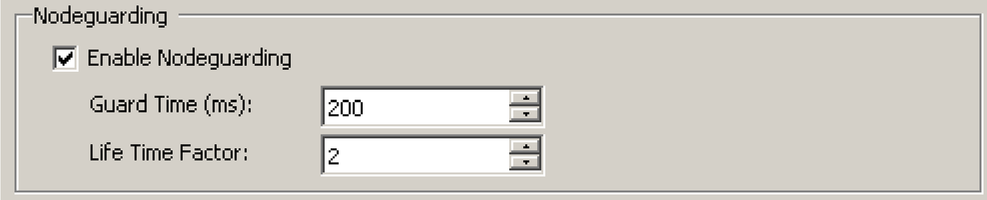
☐ Enable Heartbeat Producing

Producer Time (ms): 0

Change Heartbeat Consumer Propert

Heartbeat properties

Enable	NodeID of guarded Node	Heartbeat Time (ms)
☐		



Nodeguarding

☑ Enable Nodeguarding

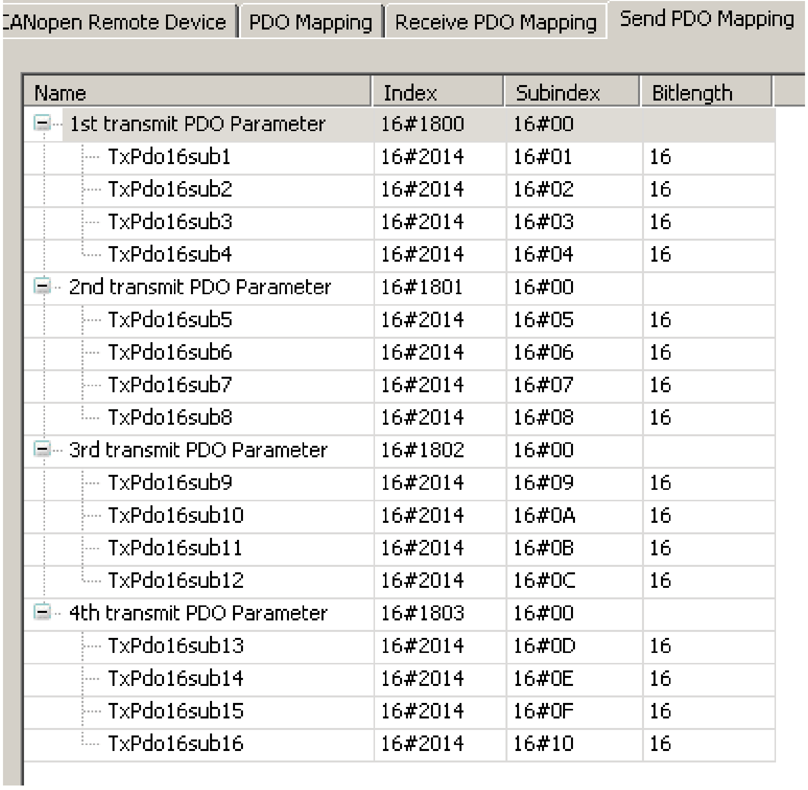
Guard Time (ms): 200

Life Time Factor: 2

You can only use one of this two methods. When using heartbeat make sure all heartbeat times are the same. When NodeGuarding is active the master can detect when a slave fails. But the slave itself will not detect that the CAN master is missing. To fix this you must enable lifeguarding on the slave (default active in KEB C6 Slave).

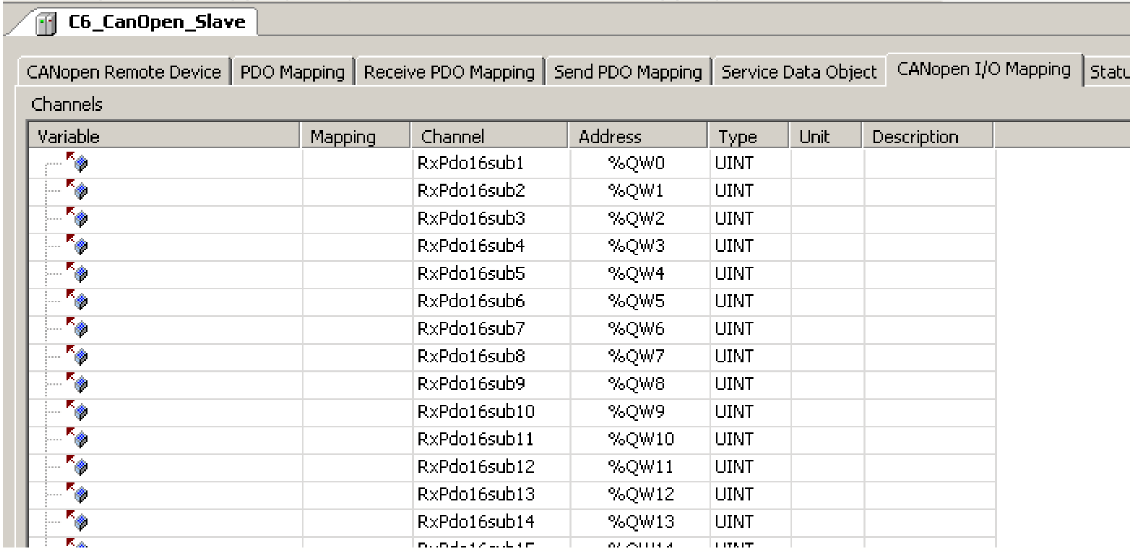
PDO (Process Data Object) configuration

Now you can have a look on your PDO data. There are 4 PDO groups for receiving and 4 groups for transmitting. Advanced settings like synchronous operation are described in SDO configuration chapter.



Name	Index	Subindex	Bitlength
1st transmit PDO Parameter	16#1800	16#00	
TxPdo16sub1	16#2014	16#01	16
TxPdo16sub2	16#2014	16#02	16
TxPdo16sub3	16#2014	16#03	16
TxPdo16sub4	16#2014	16#04	16
2nd transmit PDO Parameter	16#1801	16#00	
TxPdo16sub5	16#2014	16#05	16
TxPdo16sub6	16#2014	16#06	16
TxPdo16sub7	16#2014	16#07	16
TxPdo16sub8	16#2014	16#08	16
3rd transmit PDO Parameter	16#1802	16#00	
TxPdo16sub9	16#2014	16#09	16
TxPdo16sub10	16#2014	16#0A	16
TxPdo16sub11	16#2014	16#0B	16
TxPdo16sub12	16#2014	16#0C	16
4th transmit PDO Parameter	16#1803	16#00	
TxPdo16sub13	16#2014	16#0D	16
TxPdo16sub14	16#2014	16#0E	16
TxPdo16sub15	16#2014	16#0F	16
TxPdo16sub16	16#2014	16#10	16

In the CANopen I/O mapping tab you can add a variable mapping to the PDO data.



Variable	Mapping	Channel	Address	Type	Unit	Description
		RxPdo16sub1	%QW0	UINT		
		RxPdo16sub2	%QW1	UINT		
		RxPdo16sub3	%QW2	UINT		
		RxPdo16sub4	%QW3	UINT		
		RxPdo16sub5	%QW4	UINT		
		RxPdo16sub6	%QW5	UINT		
		RxPdo16sub7	%QW6	UINT		
		RxPdo16sub8	%QW7	UINT		
		RxPdo16sub9	%QW8	UINT		
		RxPdo16sub10	%QW9	UINT		
		RxPdo16sub11	%QW10	UINT		
		RxPdo16sub12	%QW11	UINT		
		RxPdo16sub13	%QW12	UINT		
		RxPdo16sub14	%QW13	UINT		
		RxPdo16sub15	%QW14	UINT		

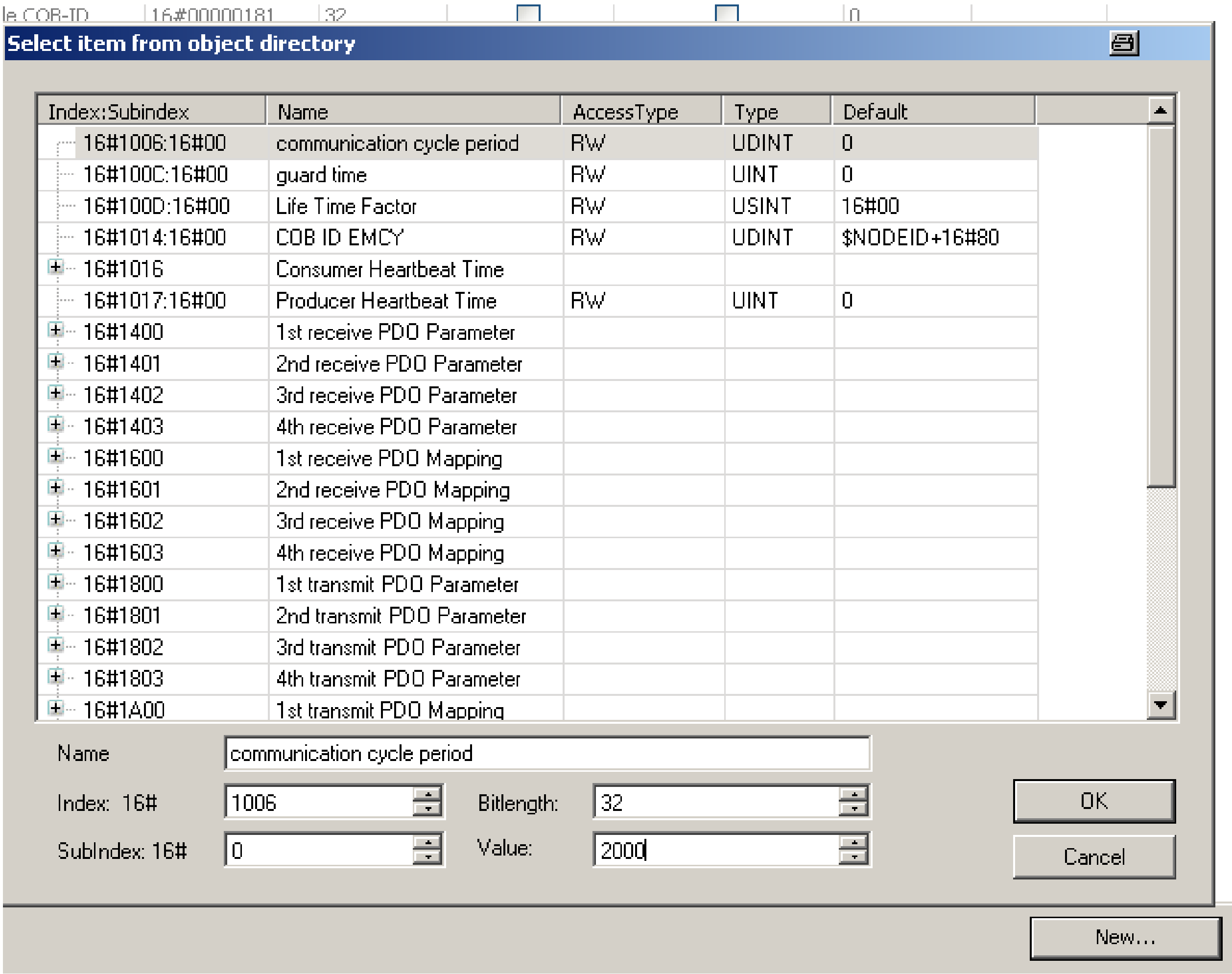
Hint: Receive and Transmit are chosen from slave view!

SDO (Service Data Object) configuration

A standard SDO is provided by the eds file you installed earlier. Now you can work with these basic settings but when you want to use cyclic or synchronous data you have to add some extra settings in the SDO tab in the configuration editor. The grey data is the part of the standard SDO and you can not edit or delete it. But you can add the same parameters at the end again to overwrite the data and use your own settings.

Synchronous PDO

To enable synchronous communication you must add the cycle time to your SDO.
Choose New… in the SDO tab and select the communication cycle period entry and change the value to your cycle time.



le COB-ID 16#00000181 32 0

Index:Subindex	Name	AccessType	Type	Default
16#1006:16#00	communication cycle period	RW	UDINT	0
16#100C:16#00	guard time	RW	UINT	0
16#100D:16#00	Life Time Factor	RW	USINT	16#00
16#1014:16#00	COB ID EMCY	RW	UDINT	$NODEID+16#80
16#1016	Consumer Heartbeat Time			
16#1017:16#00	Producer Heartbeat Time	RW	UINT	0
16#1400	1st receive PDO Parameter			
16#1401	2nd receive PDO Parameter			
16#1402	3rd receive PDO Parameter			
16#1403	4th receive PDO Parameter			
16#1600	1st receive PDO Mapping			
16#1601	2nd receive PDO Mapping			
16#1602	3rd receive PDO Mapping			
16#1603	4th receive PDO Mapping			
16#1800	1st transmit PDO Parameter			
16#1801	2nd transmit PDO Parameter			
16#1802	3rd transmit PDO Parameter			
16#1803	4th transmit PDO Parameter			
16#1A00	1st transmit PDO Mapping			

Name: communication cycle period

Index: 16#: 1006 Bitlength: 32

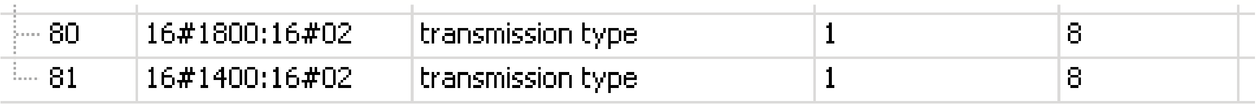
SubIndex: 16#: 0 Value: 2000

When you have done this you can choose different settings for transmission for each group of PDO data.
The default setting for the PDO is transmission type 254 which means asynchronous transmission.

The table shows the possible transmission types:

Value (decimal)	Type
0	Acyclic synchronous
1 – 240	Cyclic synchronous. The number means the number of cycles between each transmission (so 1 to 240 cycles are possible)
241 – 251	reserved
252	Synchronous RTR (Remote Transmison Request)
253	Asynchronous RTR (Remote Transmison Request)
254 – 255	Asynchronous transmission (**default**)

Now you can change the transmission types of the different PDO groups.
When you want to use the first RX and first TX group for synchronous communication then you would need to add this data to the SDO:



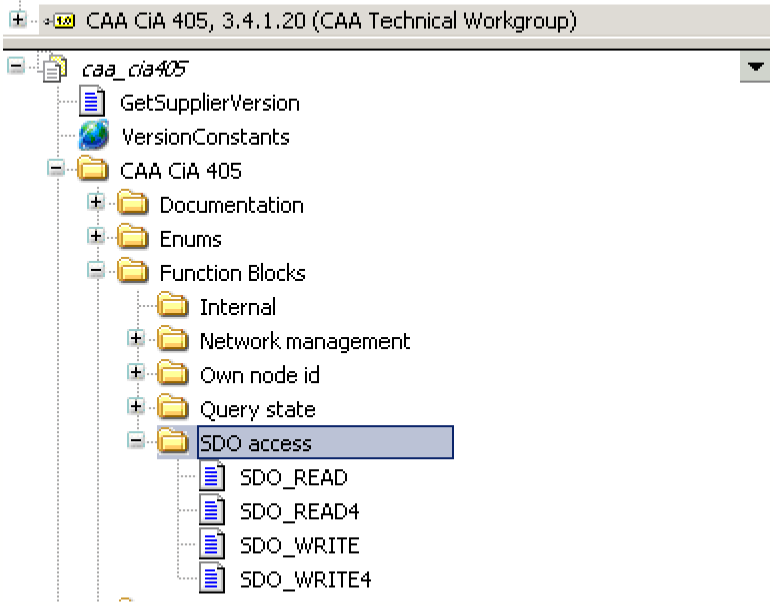
80	16#1800:16#02	transmission type	1	8
81	16#1400:16#02	transmission type	1	8

This is a basic setup to enable synchronous PDO data.

Additional SDO Notes

The SDO you can configure in the graphical interface is just transmitted one time when the slave connects to the master.
When you want to send a SDO while your application is running you have to use a CAA function block. To add this function block you have to add the CiA 405 library to your project. The function blocks SDO_READ and SDO_WRITE can be used to exchange new SDO data with a slave.

A list of available SDO parameters can be found at the end of this document.

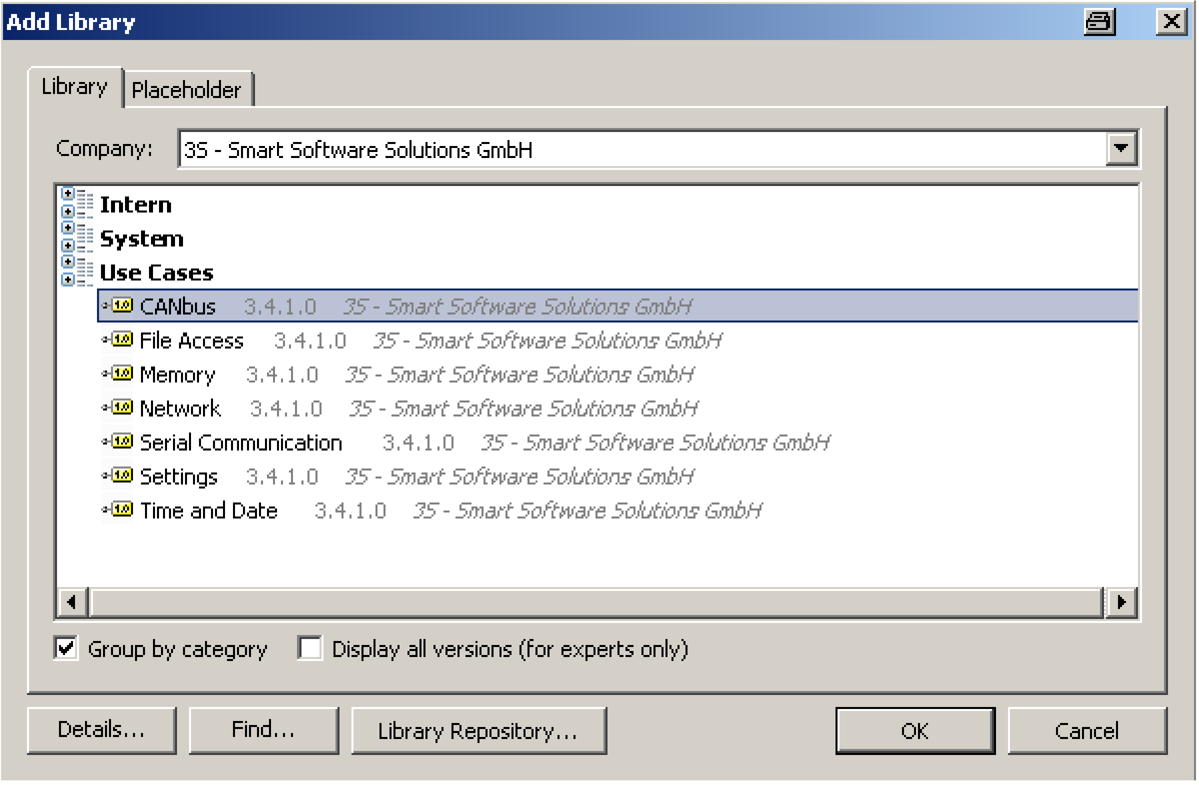


When you are done with your setup, compile the application and login to your device.

Additional C6 Compact configuration

When you want to use a C6 Compact as master you have to do some extra configuration in your project to get it working **when you use a C6 as Slave.**

First we need an additional library to gain access to some specific CAN master functions.



Add Library

Library | Placeholder

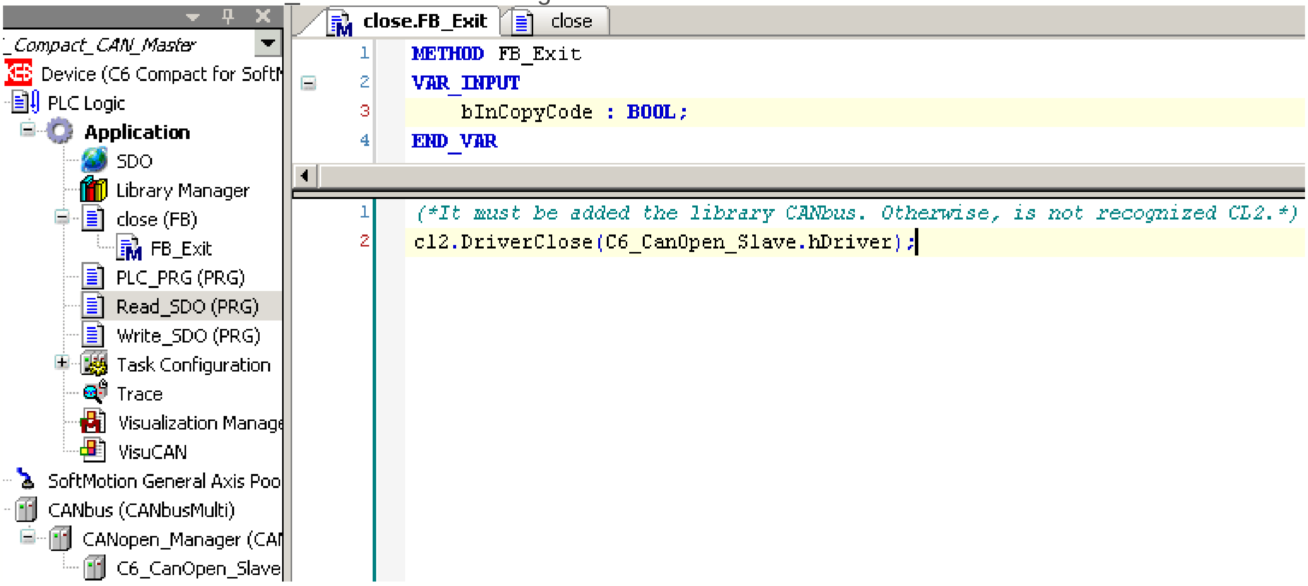
Company: 3S - Smart Software Solutions GmbH

Intern
System
Use Cases

CANbus	3.4.1.0	3S - Smart Software Solutions GmbH
File Access	3.4.1.0	3S - Smart Software Solutions GmbH
Memory	3.4.1.0	3S - Smart Software Solutions GmbH
Network	3.4.1.0	3S - Smart Software Solutions GmbH
Serial Communication	3.4.1.0	3S - Smart Software Solutions GmbH
Settings	3.4.1.0	3S - Smart Software Solutions GmbH
Time and Date	3.4.1.0	3S - Smart Software Solutions GmbH

☑ Group by category ☐ Display all versions (for experts only)

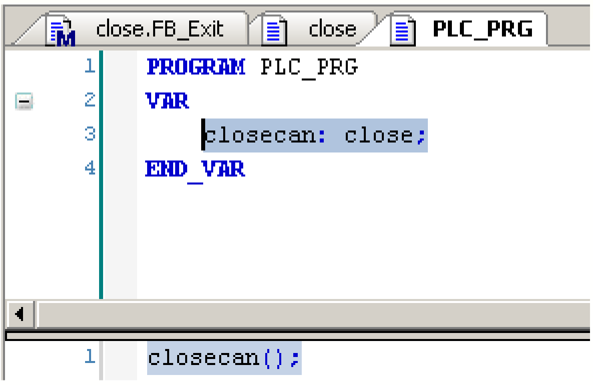
Now you must create a new function block in your project. Do not insert any source code in it. Just add a method with name FB_Exit and the following code in it:



```
METHOD FB_Exit
VAR_INPUT
    bInCopyCode : BOOL;
END_VAR
```

```
(*It must be added the library CANbus. Otherwise, is not recognized CL2.*)
cl2.DriverClose(C6_CanOpen_Slave.hDriver);
```

Then call the function block in your PLC_PRG:



```
PROGRAM PLC_PRG
VAR
	closecan: close;
END_VAR
```

```
closecan();
```

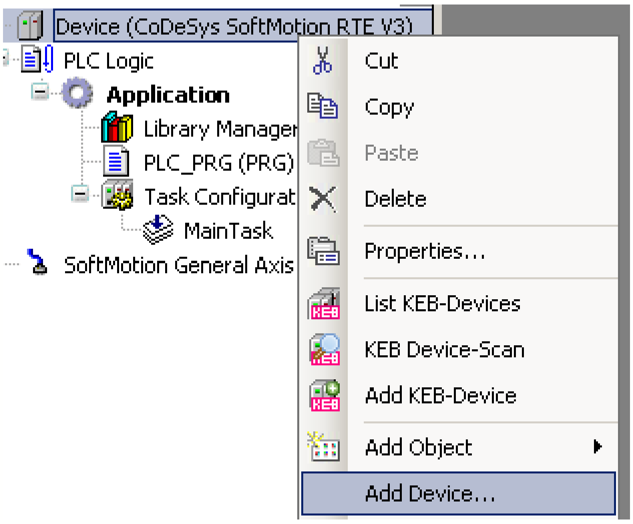
Now you can use the C6 Compact as CAN master. The rest of the setup is the same as with the C6 Econ/Perform.

C6 CAN Slave

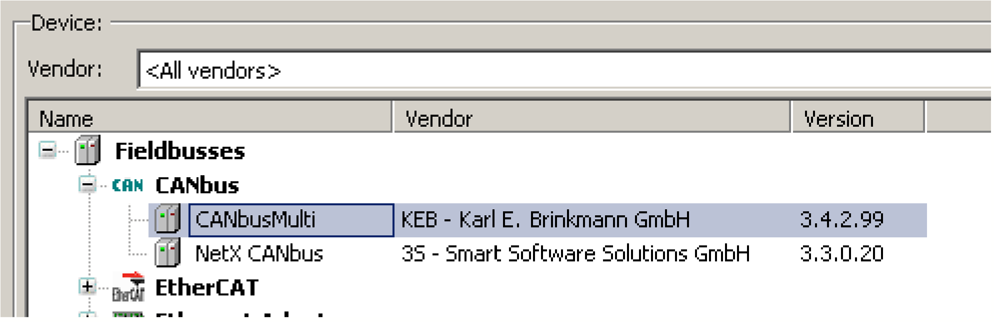
This chapter explains how to setup a C6 Compact as a CAN slave, starting with a default project.
Hint: The C6 Econ/Perform cannot be used as CAN Slave.

Adding Devices

First right click on the PLC device and add a new device.



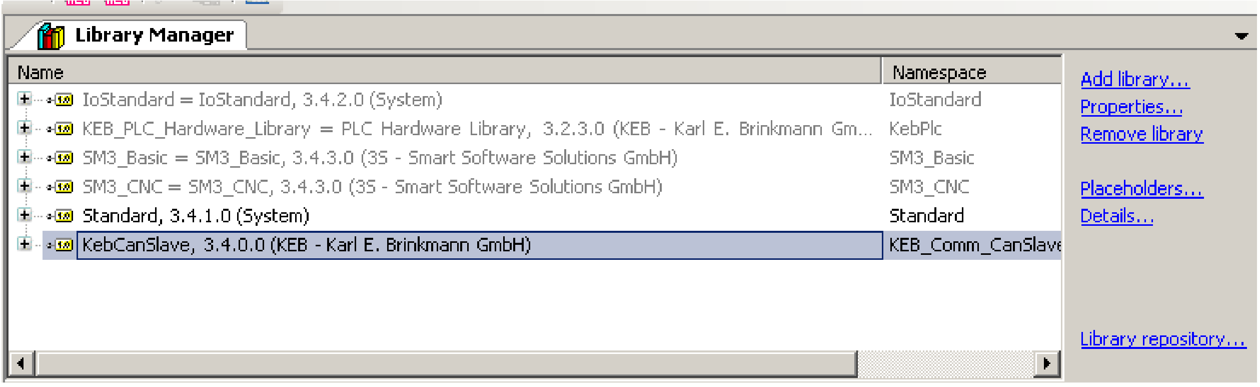
Then select the CANbusMulti Device from your device repository and add it to your project.



Vendor: <All vendors>

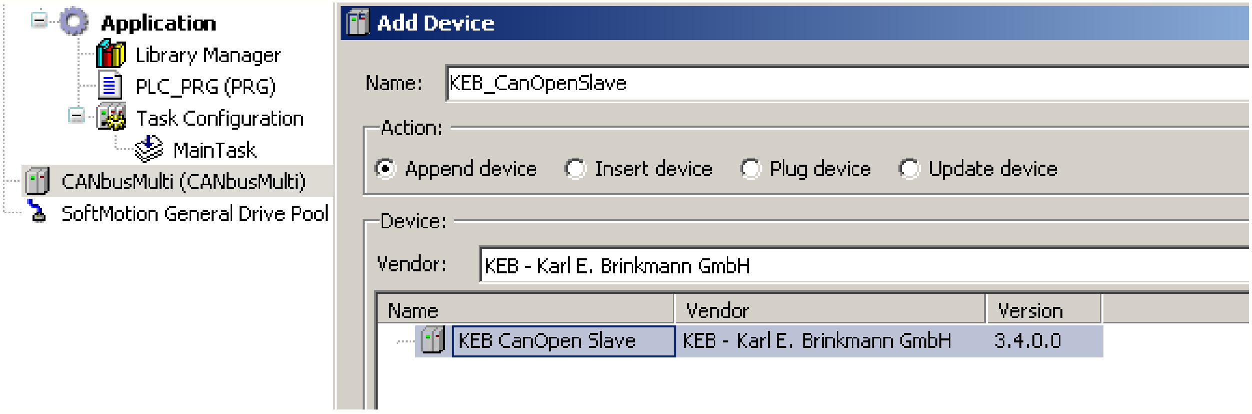
Name	Vendor	Version
Fieldbusses		
CANbus		
CANbusMulti	KEB - Karl E. Brinkmann GmbH	3.4.2.99
NetX CANbus	3S - Smart Software Solutions GmbH	3.3.0.20
EtherCAT		

Now we need to add the KEBCanSlave library.



Name	Namespace
IoStandard = IoStandard, 3.4.2.0 (System)	IoStandard
KEB_PLC_Hardware_Library = PLC Hardware Library, 3.2.3.0 (KEB - Karl E. Brinkmann Gm...	KebPlc
SM3_Basic = SM3_Basic, 3.4.3.0 (3S - Smart Software Solutions GmbH)	SM3_Basic
SM3_CNC = SM3_CNC, 3.4.3.0 (3S - Smart Software Solutions GmbH)	SM3_CNC
Standard, 3.4.1.0 (System)	Standard
KebCanSlave, 3.4.0.0 (KEB - Karl E. Brinkmann GmbH)	KEB_Comm_CanSlave

Once the library is added we can add the KEB device to the CAN bus.



- Application
  - Library Manager
  - PLC_PRG (PRG)
  - Task Configuration
    - MainTask
- CANbusMulti (CANbusMulti)
- SoftMotion General Drive Pool

Add Device

Name: KEB_CanOpenSlave

Action: ● Append device ○ Insert device ○ Plug device ○ Update device

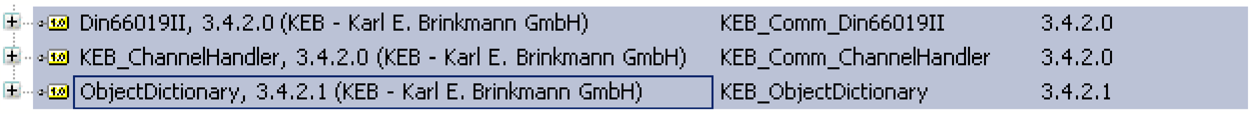
Device:

Vendor: KEB - Karl E. Brinkmann GmbH

Name	Vendor	Version
KEB CanOpen Slave	KEB - Karl E. Brinkmann GmbH	3.4.0.0

Now the difficult part: setting up the ObjectDictionary. It is highly recommended to use the CANopenDevice sample project which is installed with Combivis studio 6 but you can find the needed source code in the appendix.
In the sample project folder there are also two parameter description files which you need to install in your Combivis studio 6 (Select “Tools”-“Import KEB parameter description file” and add the two files from the sample directory).

To get the ObjectDictionary working you must add 3 more Librarys to the project:



Din66019II, 3.4.2.0 (KEB - Karl E. Brinkmann GmbH)	KEB_Comm_Din66019II	3.4.2.0
KEB_ChannelHandler, 3.4.2.0 (KEB - Karl E. Brinkmann GmbH)	KEB_Comm_ChannelHandler	3.4.2.0
ObjectDictionary, 3.4.2.1 (KEB - Karl E. Brinkmann GmbH)	KEB_ObjectDictionary	3.4.2.1

When these Libraries are included you now must do the following steps in your PLC_PRG:

1. Call the Channelhandler every Cycle
2. Do setup in first cycle:
   a. Enable ObjectDicionary for PDO Parameters
   b. Add ObjectDicionary to CAN slave
   c. Enable synchronous mode (if you want)
   Hint: When using sync mode the tasks cycle time must be the same as the can sync time.
   d. Register the DIN slave at Channelhandler
   e. Register CAN slave as master at Channelhandler

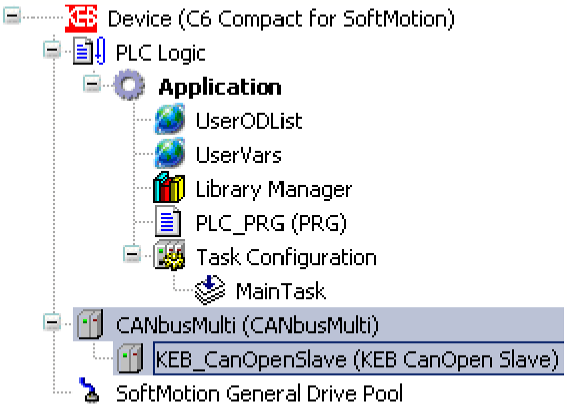
To enable synchronization you must set some input values in the setup step. Change the input **SyncPhase** of the Slave to zero (0) to activate sync mode. To turn it off you must set it to 30000. When **SyncPhase** is zero then you may change the **CycleCorrectionVal**. Depending on the busload and the jitter on the bus you may have to increase the value to get CAN in synchronous mode.
The output **CanInSync** shows the synchronization status.

```
KEB_CanOpenSlave.SyncPhase := 0;
KEB_CanOpenSlave.CycleCorrectionVal := 100;
```

Attention: When you are using a C6 Compact and download a trace or monitor many variables it is possible that the CAN bus leaves the synchronous mode.

Sample Code

This is a sample code from the sample project that is installed with Combivis studio 6.
Remember to add the devices to your project.


Device (C6 Compact for SoftMotion)
PLC Logic
Application
UserODList
UserVars
Library Manager
PLC_PRG (PRG)
Task Configuration
MainTask
CANbusMulti (CANbusMulti)
KEB_CanOpenSlave (KEB CanOpen Slave)
SoftMotion General Drive Pool

The process data is mapped to the User Object Dictionary. You can use the Arrays for your PDO data.

PLC_PRG

```
PROGRAM PLC_PRG
VAR
        started: BOOL;
        CH : KebChannelHandler;
        SL : Din_Slave_Udp;
END_VAR



CH();                                   //ALWAYS run the channelhandler
IF NOT started THEN
        started := TRUE;

        //User OD and CanOpen slave 1
        UserOd.pParaList := ADR(OdParas);
        UserOd.listSize := SIZEOF(OdParas) / SIZEOF(OD_ITEM);
        KEB_CanOpenSlave.UserOD := ADR(UserOD);

        //Enable Sync:
        KEB_CanOpenSlave.SyncPhase := 0;
        KEB_CanOpenSlave.CycleCorrectionVal := 100;

        //the Din66019 UDP slave
        CH.RegisterSlave(SL);
        //The Canopen slave as master to access the CAN-Od's
        CH.RegisterMaster(KEB_CanOpenSlave,1,1);
END_IF
```

GVL UserODList

```
VAR_GLOBAL
//Automatically generated from xml file <Paras> section with Paralist Editor V3.1.0.8
//==============================================================

OdParas : ARRAY[0..6] OF OD_ITEM := [
//RxPdo8
(
 Index:=16#2010,                    //Index
 SubIdxMod:=16#0000,                   //SubIdxMod
 Itype:=16#1000 OR
 OD_TBL_DATATYPE_UINT8 OR
 OD_TBL_OBJTYPE_ARRAY,                      //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RW OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                        //Prop1
 ReadFuncIdx:=0,                     //ReadFuncIdx
 WriteFuncIdx:=0,                    //WriteFuncIdx
 LowerLim:=0,                       //LowerLim
 UpperLim:=65535,                     //UpperLim
 DefValue:=0,                       //DefValue
 pData:=ADR(Para2010[0])                  //pData
),
//RxPdo16
(
 Index:=16#2011,                    //Index
 SubIdxMod:=16#0000,                   //SubIdxMod
 Itype:=16#1000 OR
 OD_TBL_DATATYPE_UINT16 OR
 OD_TBL_OBJTYPE_ARRAY,                      //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RW OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                        //Prop1
 ReadFuncIdx:=0,                     //ReadFuncIdx
 WriteFuncIdx:=0,                    //WriteFuncIdx
 LowerLim:=0,                       //LowerLim
 UpperLim:=65535,                     //UpperLim
 DefValue:=0,                       //DefValue
 pData:=ADR(Para2011[0])                  //pData
),
//RxPdo32
(
 Index:=16#2012,                                   //Index
 SubIdxMod:=16#0000,                                 //SubIdxMod
 Itype:=OD_TBL_ITEM_ENTRIES_8 OR
 OD_TBL_DATATYPE_UINT32 OR
 OD_TBL_OBJTYPE_ARRAY,                                    //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RW OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                                      //Prop1
 ReadFuncIdx:=0,                                   //ReadFuncIdx
 WriteFuncIdx:=0,                                  //WriteFuncIdx
 LowerLim:=0,                                     //LowerLim
 UpperLim:=-1,                                    //UpperLim
 DefValue:=0,                                     //DefValue
 pData:=ADR(Para2012[0])                               //pData
),
//TxPdo8
(
 Index:=16#2013,                                   //Index
 SubIdxMod:=16#0000,                                 //SubIdxMod
 Itype:=16#1000 OR
 OD_TBL_DATATYPE_UINT8 OR
 OD_TBL_OBJTYPE_ARRAY,                                    //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RO OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                                      //Prop1
 ReadFuncIdx:=0,                                   //ReadFuncIdx
 WriteFuncIdx:=0,                                  //WriteFuncIdx
 LowerLim:=0,                                     //LowerLim
 UpperLim:=65535,                                   //UpperLim
 DefValue:=0,                                     //DefValue
 pData:=ADR(Para2013[0])                               //pData
),
//TxPdo16
(
 Index:=16#2014,                                   //Index
 SubIdxMod:=16#0000,                                 //SubIdxMod
 Itype:=16#1000 OR
 OD_TBL_DATATYPE_UINT16 OR
 OD_TBL_OBJTYPE_ARRAY,                                    //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
```

```
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RO OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                  //Prop1
 ReadFuncIdx:=0,               //ReadFuncIdx
 WriteFuncIdx:=0,             //WriteFuncIdx
 LowerLim:=0,                 //LowerLim
 UpperLim:=65535,              //UpperLim
 DefValue:=0,                //DefValue
 pData:=ADR(Para2014[0])          //pData
),
//TxPdo32
(
 Index:=16#2015,               //Index
 SubIdxMod:=16#0000,             //SubIdxMod
 Itype:=OD_TBL_ITEM_ENTRIES_8 OR
 OD_TBL_DATATYPE_UINT32 OR
 OD_TBL_OBJTYPE_ARRAY,              //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RO OR
 OD_TBL_SS_FIX OR
 OD_TBL_PD_YES,                  //Prop1
 ReadFuncIdx:=0,               //ReadFuncIdx
 WriteFuncIdx:=0,             //WriteFuncIdx
 LowerLim:=0,                 //LowerLim
 UpperLim:=-1,                //UpperLim
 DefValue:=0,                //DefValue
 pData:=ADR(Para2015[0])          //pData
),
//cfgId
(
 Index:=16#160F,               //Index
 SubIdxMod:=16#0000,             //SubIdxMod
 Itype:=OD_TBL_ITEM_ENTRIES_0 OR
 OD_TBL_DATATYPE_UINT32 OR
 OD_TBL_OBJTYPE_VAR,               //Type
 Prop1:=OD_TBL_PWLVWR_0 OR
 OD_TBL_PWLVRD_0 OR
 OD_TBL_DEFVAL_FIX OR
 OD_TBL_ULIM_FIX OR
 OD_TBL_LLIM_FIX OR
 OD_TBL_RO OR
 OD_TBL_PD_NO,                   //Prop1
 ReadFuncIdx:=0,               //ReadFuncIdx
 WriteFuncIdx:=0,             //WriteFuncIdx
 LowerLim:=1001001,              //LowerLim
 UpperLim:=1001001,              //UpperLim
```

```
  DefValue:=1001001,                    //DefValue
  pData:=ADR(ParaCfgId)                   //pData
)
];

END_VAR
```

GVL UserVars

```
VAR_GLOBAL
        Para2010 : ARRAY [0..15] OF BYTE;     //RxPdo8
        Para2011 : ARRAY [0..15] OF WORD;     //RxPdo16
        Para2012 : ARRAY [0..7] OF DWORD;     //RxPdo32
        Para2013 : ARRAY [0..15] OF BYTE;     //TxPdo8
        Para2014 : ARRAY [0..15] OF WORD;     //TxPdo16
        Para2015 : ARRAY [0..7] OF DWORD;     //TxPdo32
        ParaCfgId : UDINT := 1001001;         //fixed value for cfg id
        UserOd : ObjectDictionary;
END_VAR
```

SDO Parameters for C6 Slave

Index	Subindex	Name	Bitlength	Default value
16#1006	16#00	communication cycle period	32	0
16#100C	16#00	guard time	16	0
16#100D	16#00	Life time factor	8	0
16#1014	16#00	COB ID EMCY	32	$NODEID+16#80
16#1016	16#01	Consumer Heartbeat time	32	0
16#1017	16#00	Producer Heartbeat time	16	0
16#1400		1st receive PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#200
	16#02	transmission type	8	254
16#1401		2nd receive PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#300
	16#02	transmission type	8	254
16#1402		3rd receive PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#400
	16#02	transmission type	8	254
16#1403		4th receive PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#500
	16#02	transmission type	8	254
16#1600		1st receive PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20110110
	16#02	2nd PDO Mapping	32	16#20110210
	16#03	3rd PDO Mapping	32	16#20110310
	16#04	4th PDO Mapping	32	16#20110410
16#1601		2nd receive PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20110510
	16#02	2nd PDO Mapping	32	16#20110610
	16#03	3rd PDO Mapping	32	16#20110710
	16#04	4th PDO Mapping	32	16#20110810
16#1602		3rd receive PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20110910
	16#02	2nd PDO Mapping	32	16#20110A10
	16#03	3rd PDO Mapping	32	16#20110B10
	16#04	4th PDO Mapping	32	16#20110C10
16#1603		4th receive PDO Mapping		

Index	Subindex	Name	Bitlength	Default value
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20110D10
	16#02	2nd PDO Mapping	32	16#20110E10
	16#03	3rd PDO Mapping	32	16#20110F10
	16#04	4th PDO Mapping	32	16#20111010
16#1800		1st transmit PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#180
	16#02	transmission type	8	254
16#1800		2nd transmit PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#280
	16#02	transmission type	8	254
16#1800		3rd transmit PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#380
	16#02	transmission type	8	254
16#1800		4th transmit PDO Parameter		
	16#01	COB-ID used by PDO	32	$NODEID+16#480
	16#02	transmission type	8	254
16#1A00		1st transmit PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20140110
	16#02	2nd PDO Mapping	32	16#20140210
	16#03	3rd PDO Mapping	32	16#20140310
	16#04	4th PDO Mapping	32	16#20140410
16#1A01		2nd transmit PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20140510
	16#02	2nd PDO Mapping	32	16#20140610
	16#03	3rd PDO Mapping	32	16#20140710
	16#04	4th PDO Mapping	32	16#20140810
16#1A02		3rd transmit PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20140910
	16#02	2nd PDO Mapping	32	16#20140A10
	16#03	3rd PDO Mapping	32	16#20140B10
	16#04	4th PDO Mapping	32	16#20140C10
16#1A03		4th transmit PDO Mapping		
	16#00	number of mapped objects	8	4
	16#01	1st PDO Mapping	32	16#20140D10
	16#02	2nd PDO Mapping	32	16#20140E10

Index	Subindex	Name	Bitlength	Default value
	16#03	3rd PDO Mapping	32	16#20140F10
	16#04	4th PDO Mapping	32	16#20141010
16#2010		RxPdo8		
	16#01	RxPdo8sub1	8	
	16#02	RxPdo8sub2	8	
	16#03	RxPdo8sub3	8	
	16#04	RxPdo8sub4	8	
	16#05	RxPdo8sub5	8	
	16#06	RxPdo8sub6	8	
	16#07	RxPdo8sub7	8	
	16#08	RxPdo8sub8	8	
	16#09	RxPdo8sub9	8	
	16#0A	RxPdo8sub10	8	
	16#0B	RxPdo8sub11	8	
	16#0C	RxPdo8sub12	8	
	16#0D	RxPdo8sub13	8	
	16#0E	RxPdo8sub14	8	
	16#0F	RxPdo8sub15	8	
	16#10	RxPdo8sub16	8	
16#2011		RxPdo16		
	16#01	RxPdo16sub1	16	
	16#02	RxPdo16sub2	16	
	16#03	RxPdo16sub3	16	
	16#04	RxPdo16sub4	16	
	16#05	RxPdo16sub5	16	
	16#06	RxPdo16sub6	16	
	16#07	RxPdo16sub7	16	
	16#08	RxPdo16sub8	16	
	16#09	RxPdo16sub9	16	
	16#0A	RxPdo16sub10	16	
	16#0B	RxPdo16sub11	16	
	16#0C	RxPdo16sub12	16	
	16#0D	RxPdo16sub13	16	
	16#0E	RxPdo16sub14	16	
	16#0F	RxPdo16sub15	16	
	16#10	RxPdo16sub16	16	
16#2012		RxPdo32		
	16#01	RxPdo32sub1	32	
	16#02	RxPdo32sub2	32	

Index	Subindex	Name	Bitlength	Default value
	16#03	RxPdo32sub3	32	
	16#04	RxPdo32sub4	32	
	16#05	RxPdo32sub5	32	
	16#06	RxPdo32sub6	32	
	16#07	RxPdo32sub7	32	
	16#08	RxPdo32sub8	32	
	16#09	RxPdo32sub9	32	
	16#0A	RxPdo32sub10	32	
	16#0B	RxPdo32sub11	32	
	16#0C	RxPdo32sub12	32	
	16#0D	RxPdo32sub13	32	
	16#0E	RxPdo32sub14	32	
	16#0F	RxPdo32sub15	32	
	16#10	RxPdo32sub16	32	
16#2013		TxPdo8		
	16#01	TxPdo8sub1	8	
	16#02	TxPdo8sub2	8	
	16#03	TxPdo8sub3	8	
	16#04	TxPdo8sub4	8	
	16#05	TxPdo8sub5	8	
	16#06	TxPdo8sub6	8	
	16#07	TxPdo8sub7	8	
	16#08	TxPdo8sub8	8	
	16#09	TxPdo8sub9	8	
	16#0A	TxPdo8sub10	8	
	16#0B	TxPdo8sub11	8	
	16#0C	TxPdo8sub12	8	
	16#0D	TxPdo8sub13	8	
	16#0E	TxPdo8sub14	8	
	16#0F	TxPdo8sub15	8	
	16#10	TxPdo8sub16	8	
16#2014		TxPdo16		
	16#01	TxPdo16sub1	16	
	16#02	TxPdo16sub2	16	
	16#03	TxPdo16sub3	16	
	16#04	TxPdo16sub4	16	
	16#05	TxPdo16sub5	16	
	16#06	TxPdo16sub6	16	
	16#07	TxPdo16sub7	16	

Index	Subindex	Name	Bitlength	Default value
	16#08	TxPdo16sub8	16	
	16#09	TxPdo16sub9	16	
	16#0A	TxPdo16sub10	16	
	16#0B	TxPdo16sub11	16	
	16#0C	TxPdo16sub12	16	
	16#0D	TxPdo16sub13	16	
	16#0E	TxPdo16sub14	16	
	16#0F	TxPdo16sub15	16	
	16#10	TxPdo16sub16	16	
16#2015		TxPdo32		
	16#01	TxPdo32sub1	32	
	16#02	TxPdo32sub2	32	
	16#03	TxPdo32sub3	32	
	16#04	TxPdo32sub4	32	
	16#05	TxPdo32sub5	32	
	16#06	TxPdo32sub6	32	
	16#07	TxPdo32sub7	32	
	16#08	TxPdo32sub8	32	
	16#09	TxPdo32sub9	32	
	16#0A	TxPdo32sub10	32	
	16#0B	TxPdo32sub11	32	
	16#0C	TxPdo32sub12	32	
	16#0D	TxPdo32sub13	32	
	16#0E	TxPdo32sub14	32	
	16#0F	TxPdo32sub15	32	
	16#10	TxPdo32sub16	32	

Disclaimer

KEB Automation KG reserves the right to change/adapt specifications and technical data without prior notification. The safety and warning reference specified in this manual is not exhaustive. Although the manual and the information contained in it is made with care, KEB does not accept responsibility for misprint or other errors or resulting damages. The marks and product names are trademarks or registered trademarks of the respective title owners.

The information contained in the technical documentation, as well as any user-specific advice in verbal or in written form are made to the best of our knowledge and information about the application. However, they are considered for information only without responsibility. This also applies to any violation of industrial property rights of a third-party.

Inspection of our units in view of their suitability for the intended use must be done generally by the user. Inspections are particular necessary, if changes are executed, which serve for the further development or adaption of our products to the applications (hardware, software or download lists). Inspections must be repeated completely, even if only parts of hardware, software or download lists are modified.

Application and use of our units in the target products is outside of our control and therefore lies exclusively in the area of responsibility of the user.

KEB Automation KG
Südstraße 38 • D-32683 Barntrup
fon: +49 5263 401-0 • fax: +49 5263 401-116
net: www.keb.de • mail: info@keb.de