Communication Driver KEBCO

Driver for serial communication with KEB-DIN 66019 Protocol

Index

1 Introduction

The KEBCO driver enables communication between Studio system and KEB frequency inverters using KEB-DIN 66019 Protocol, in accordance with the characteristics covered in this document.

This document contains 8 parts, as follow:

- Introduction: Provides an overview of the driver documentation.
- General characteristics: Provides information necessary to identify all the required components (hardware and software) necessary to implement the communication and global characteristics about the communication.
- Installation: Explains the procedures that must be followed to install the software and hardware required for the communication.
- Driver configuration: Provides the required information to configure the communication driver such as the different permutations for configuration and its default values.
- Execution: Explain the steps to test whether the driver was correctly installed and configured.
- Troubleshooting: Supplies a list of the most common error codes for this protocol and the procedures to fix them.
- Application Sample: Provides a sample application for testing the configuration the driver.
- History of versions: Provides a log of all the modifications done in driver.

☞ Note: This document presumes that the user has read the chapter *Driver Configuration* of the Studio's Technical reference manual.

2 General Characteristics

2.1 Device Characteristics

- Manufacturer: KEB
- Compatible Equipment
  - Frequency inverters that use KEB-DIN 66019 protocol

☞ Note: Please refers to section 2.4 to see the Equipment used in the standard conformance tests for this driver.

2.2 Link Characteristics

- Device communication port: RS232 port
- Physical protocol: RS232/RS485
- Logic protocol: KEB-DIN 66019
- Device Runtime software: None
- Specific PC Board: None
- Adapters / Converters: RS232/RS485 Converter

2.3 Driver Characteristics

- Operating System:
  - Windows 9x
  - Windows 2000
  - Windows NT
  - Windows CE

☞ Note: Please refer to section 2.4 to see the Operating System used in the conformance tests for this driver.

The driver is composed of the following files:

- KEBCO.INI: Internal file of the driver, it should not be modified by the user.
- KEBCO.MSG: This file contains the error messages for each error code. It is an internal file of the driver, the user should not modify it.
- KEBCO.PDF: This document provides detailed documentation about the driver.
- KEBCO.DLL: This is the compiled library for the driver.

☞ Note: All the files above must to be in the subdirectory /DRV of the Studio's installation directory.

2.4 Information about conformance testing

- Equipment:

Configuration:

PLC program:
Baud Rate:
Protocol: KEB-DIN 66019 Protocol
Communication Asynchronous Half-Duplex
Data Bits: 7
Stop Bits: 1
Parity: Even
COM port: COM1

- Cable:
- Operating System (development):
- Operating System (target):
- Studio Version:
- Driver version: 1.00 – Beta 1

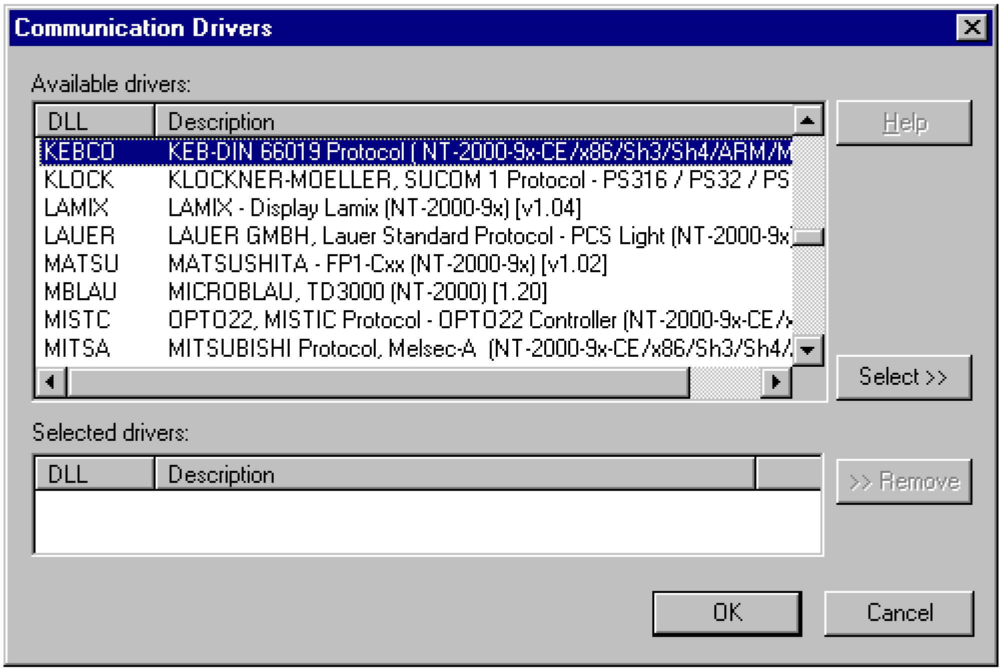
3 Installation

3.1 Installing the Driver

When you install the Studio v3.0 or higher, the communication drivers are already installed. You need now to select the driver at the applications where it will be used.

The steps to select the driver inside an application are:

1. Execute the Studio and select the proper application.
2. Select the menu *Insert + Driver...*
3. In the column **Available Drivers**, select the **KEBCO Driver** and push the button **Select >>** (the driver KEBCO must appear in the column **Selected Drivers**).
4. Press **OK**.



Available drivers:

DLL	Description
KEBCO	KEB-DIN 66019 Protocol (NT-2000-9x-CE/x86/Sh3/Sh4/ARM/M
KLOCK	KLOCKNER-MOELLER, SUCOM 1 Protocol - PS316 / PS32 / PS
LAMIX	LAMIX - Display Lamix (NT-2000-9x) [v1.04]
LAUER	LAUER GMBH, Lauer Standard Protocol - PCS Light (NT-2000-9x)
MATSU	MATSUSHITA - FP1-Cxx (NT-2000-9x) [v1.02]
MBLAU	MICROBLAU, TD3000 (NT-2000) [1.20]
MISTC	OPTO22, MISTIC Protocol - OPTO22 Controller (NT-2000-9x-CE/x
MITSA	MITSUBISHI Protocol, Melsec-A (NT-2000-9x-CE/x86/Sh3/Sh4/

Selected drivers:

DLL	Description

3.2 Other software requirements

It is not necessary to install any other software in the PC to enable the communication between the host and the Device.

☞ Note: Special precautions must be taken when installing the physical hardware. Refer to the hardware manufacturer documentation for specific instructions in this area.

4 Driver Configuration

After the driver is installed and selected in the Studio (see section 3.1), you should proceed to the driver configuration.

The driver configuration is two parts:

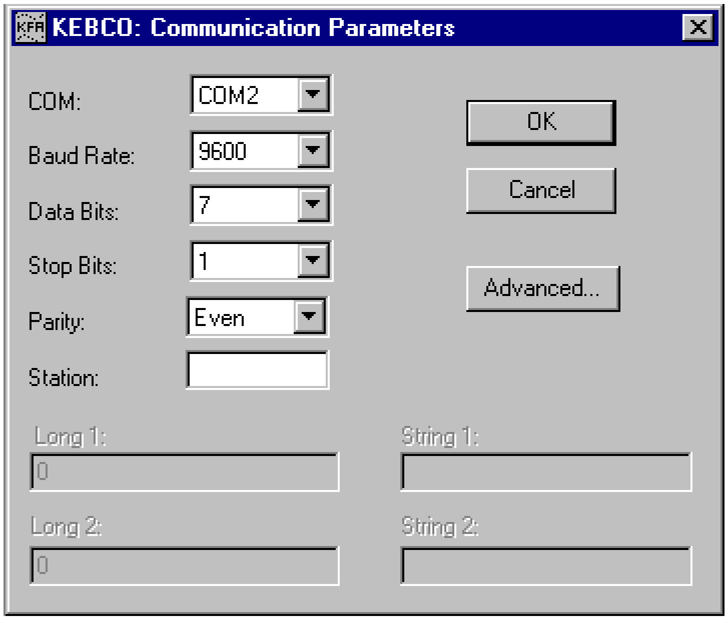
The Settings or Communication parameters, it is only one configuration to the whole driver, them you have the communication tables or Driver Worksheets, where the communication tags are defined.

4.1 Settings - Communication Parameters

These parameters are valid for all driver worksheets configured in the system. To open the window for configuring the **Communication parameters**, follow these steps:

1. In the **Workspace** of the Studio environment, select the **Comm** table.
2. Expand the folder **Drivers** and select the subfolder **KEBCO**.
3. Right click on the **KEBCO** subfolder and select the option **Settings**.

When selecting the Settings, there is the following dialog to configure:



KEBCO: Communication Parameters

COM: COM2

Baud Rate: 9600

Data Bits: 7

Stop Bits: 1

Parity: Even

Station:

Long 1: 0

Long 2: 0

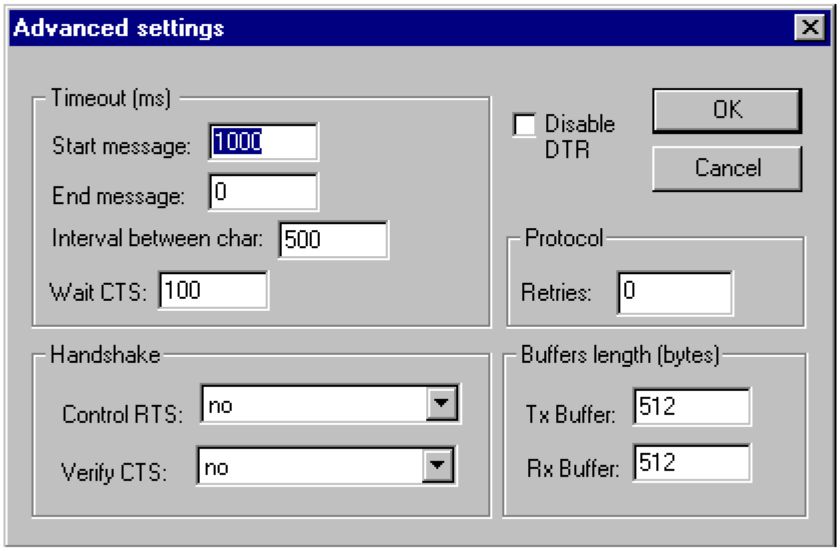
String 1:

String 2:

Parameter	Default Value	Valid values	Description
COM	COM2	COM1 to COM8	Serial port of the PC used to communication with the device
Baud Rate	9600	110 to 57600bps	Communication rate of data
Data Bits	7	5 to 8	Number of data bits used in the protocol
Stop Bits	1	1or 2	Number of stop bits used in the protocol
Parity	Even	even, odd, none, space or mark	Parity of the protocol
Station	0	0	Not used for the driver

☞ Note: These Parameters must be just the same as the configured on the KEB device. The KEB Manual indicates the following serial communication settings: Baud Rate: 9600, Data bits: 7, Stop bits: 1, Parity: Even

By clicking on the button **Advanced…** in the window **Communication Parameters,** you open additional communication parameters.



Advanced settings

Timeout (ms)

Start message: 1000

End message: 0

Interval between char: 500

Wait CTS: 100

□ Disable DTR

Protocol

Retries: 0

Handshake

Control RTS: no

Verify CTS: no

Buffers length (bytes)

Tx Buffer: 512

Rx Buffer: 512

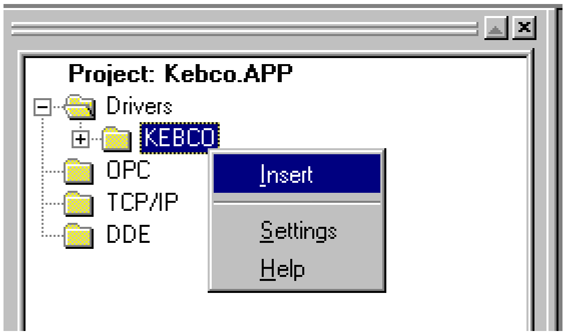
The Advanced setting parameters are explained at the Studio Technical Reference Manual, and you should keep the default values to all fields. Only the field described at the next table should be configured:

Parameter	Default Value	Valid values	Description
Control RTS	No	no, yes or yes + echo	Define if the handshake signal of RTS (Request to Send) is set before communication and if there is an echo in the communication. If we are using Windows 95 or CE with the right RS 232 – RS 485 Converter (without RTS Control), choose the option “**no**”. But, if we are under Windows NT and if the KEBCOer Hammer RS232 – 485 adapter is used, the choice must be “**yes**”. **Important**: Wrong settings on this field will not let the driver work, having the Timeout waiting start a message error.

4.2 Driver Worksheet

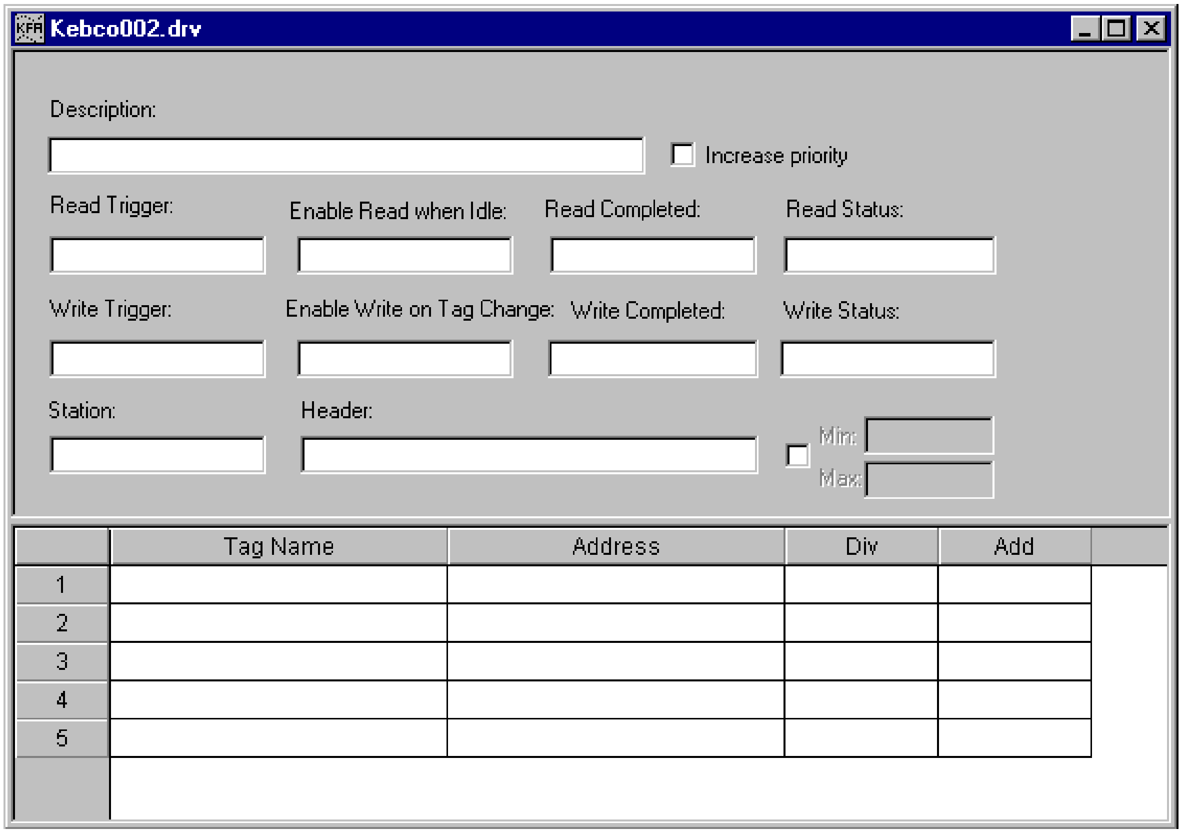
It is possible to configure many driver worksheets, each one will be composed of a Header and Body. To create a new driver worksheet, follow these steps:

1. In the **Workspace** of the Studio environment, select the table **Comm**.
2. Expand the folder **Drivers** and select the subfolder **KEBCO**.
3. Right click on the **KEBCO** subfolder and select the option **Insert**.



☞ Note: To optimize communication and ensure better performance for the system, it is important to tie the tags in different driver sheets according to the events that must trigger the communication of each group of tags and the periodicity for which each group of tags must be written or read. In addition, it is recommended to configure the addresses of communication in sequential blocks.

When creating a communication table, you have the following window:



	Tag Name	Address	Div	Add
1				
2				
3				
4				
5				

All entries at the Driver Worksheet, exception by the **Station**, **Header** and **Address** are standard to all communication drivers. You should refer to Studio Communication Driver documentation about the configuration of the standard fields. This document describes the Station, Header and Address fields, which are specific to each communication driver.

4.3 Station and Header configuration

Parameter	Default Value	Valid values	Description
Station	-	0 - 239	The Device's Address.
Header	None	-	Not used for this driver

4.4 Address Configuration

The body of the driver worksheet allows you to associate each tag to its respective address in the device. In the column **Tag Name,** you must type the tag from your application database. This tag will receive or send values from or to an address on the device.

The address cells complies to the following syntax:

<type><Address>

Address: the **Address** is a number that defines an address in the device.

Type: the **Type** is a charactere that defines the variable type. Use 'I' for integer values and 'U' for unsigned values (default).

Sample of Addressing Configuration	
Address on the Device	**Address Field**
100 (Integer)	I100
100 (Unsigned)	U100
200 (Integer)	I200
201 (Integer)	I201
400 (Integer)	I400
500 (Unsigned)	U500
600 (Unsigned)	U600

4.5 Device Configuration

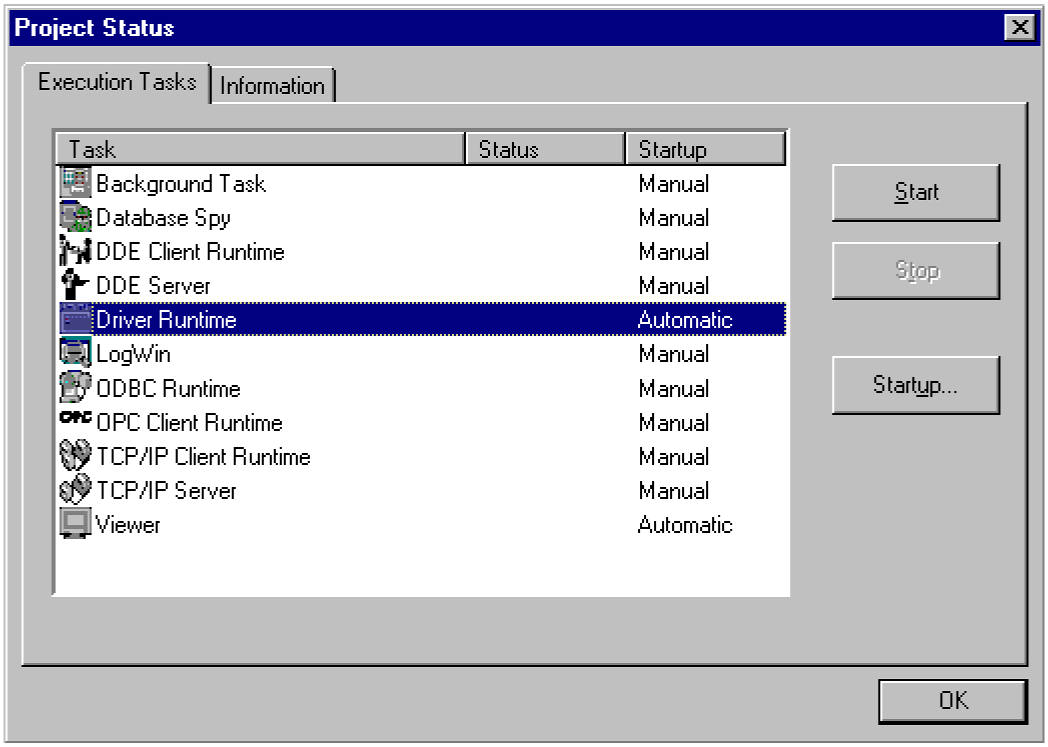
The KEB Manual indicates the following serial communication settings:

Baud Rate: 9600
Data bits: 7
Stop bits: 1
Parity: Even

Besides that, it is required to select the Inverter Address Number

5 Execution

- When installing the driver, it is automatically selected to execute when you start-up the Runtime Environment. To verify the if the driver is correctly enabled to start, use the menu option **Project** + **Status…,** and verify the task Driver Runtime



Task	Status	Startup
Background Task		Manual
Database Spy		Manual
DDE Client Runtime		Manual
DDE Server		Manual
Driver Runtime		Automatic
LogWin		Manual
ODBC Runtime		Manual
OPC Client Runtime		Manual
TCP/IP Client Runtime		Manual
TCP/IP Server		Manual
Viewer		Automatic

6 Troubleshooting

After each attempt to communicate using this driver, the tag configured in the field **Read Status** or **Write Status** will receive the error code regarding the kind of failure that occurred. The error messages are:

Error Code	Description (*)	Possible causes	Procedure to solve
0	OK	Communication without problems	-
10	Invalid Response	The device has sent and invalid response or the communication settings are not matching with the device settings	Verify the communication settings, see section 4.1
20	Invalid Address	Wrong address syntax specified.	Change the address to the correct format.
11	Invalid BCC	The device has sent and invalid BCC or the communication settings are not matching with the device settings	Verify the communication settings, see section 4.1
-15	Timeout waiting start a message.	- Disconnected cables - PLC turned off, or in Stop or error mode - Wrong Station number - Wrong RTS/CTS control settings.	- Check the cable wiring - Check the PLC state. It must be RUN - Check the station number. - Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.
-17	Timeout between rx char.	- PLC in stop or error mode - Wrong station number - Wrong parity - Wrong RTS/CTS configuration settings	- Check the cable wiring - Check the PLC state. It must be RUN - Check the station number. - Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.

☞ Note: The results of the communication may be verified in the **output** Window of the Studio's environment. To set a log of events for **Field Read Commands**, **Field Write Commands** and **Serial Communication** click with the right button of the mouse on the output window and chose the option setting to select these log events. When testing under a Windows CE target, you can enable the log at the unit (Tools/Logwin) and verify the file celog.txt created at the target unit.

When you are not able to establish the communication with the PLC, you should first of all establish the communication between the PLC Programming Tool and the PLC. Very frequently the communication it is not possible due to a hardware or cable problem, or due an error or lack of configuration at the PLC. Only after the communication between the PLC Programming Software and the PLC is working fine, you can test again the supervisory.

When testing the communication with the Studio, you should first use the application sample described at item 7, instead of the new application that you are creating.

If is required to contact technical support, please have the following information available:

- Operating System (type and version): To find this information use the Tools/System Information option
- Project information: It is displayed using the option Project/Status from the Studio menu
- Driver version and communication log: Available from Studio Output when running the driver
- Device model and boards: please refer to hardware manufacture's documentation

7 Application Sample

The Studio contains a configured project to test the driver. It is strongly recommended to do some tests with this application before beginning the configuration of the customized project, for the follow reasons:

- To understand better the information covered in section 4 of this document.
- To verify that your configuration is working.
- To certify that the hardware used in the test (device + adapter + cable + PC) is in working conditions before beginning the configuration of the applications.

☞ Note: The Application Sample is not available for all drivers.

The Studio application is in the directory: /COMMUNICATION EXAMPLES/<Driver Name>

To perform the test, you need to follow these steps:

- Configure the device communication parameters using manufacturer programmer software..
- Open the application /COMMUNICATION EXAMPLES/<Driver Name>
- Execute the application
- To display the following screen with some information about the communication, please execute the Viewer module in the Studio.

☞ Note: The application for testing may be used like a maintenance screen for the custom application.

8 History of Versions

Version	By	Date	Description of changes
1.00 – b1	Lourenço	16-Nov-2000	▪ First driver version
1.01	Plínio M. Santana	Mar-19-2007	▪ Fixed bug to writing operation. ▪ Correction made to validate the Address field.