ABSOLUTE ROTARY ENCODER WITH DEVICENET INTERFACE USER MANUAL

DeviceNet Interface

User Manual

1. Introduction

Absolute rotary encoders provide a definite value for every possible position. All these values are reflected on one or more code discs. The beams of infrared LEDs are sent through code discs and detected by Opto-Arrays. The output signals are electronically amplified and the resulting value is transferred to the interface.

The absolute rotary encoder has a maximum resolution of 65536 steps per revolution (16 Bit). The Multiturn version can detect up to 16384 revolutions (14 Bit). Therefore the largest resulting resolution is 30 Bit = 1.073.741.824 steps. The standard Singleturn version has 12 Bit, the standard Multiturn version 24 Bit.

The integrated CAN-Bus interface of the absolute rotary encoder supports all of the DeviceNet functions. The following modes can be programmed and enabled or disabled:

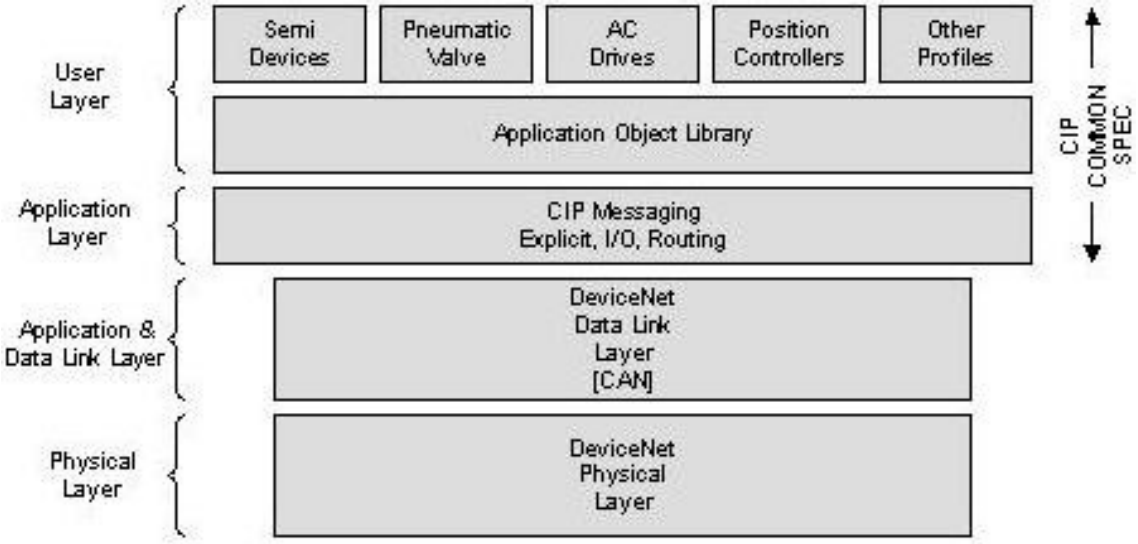
- Polled Mode
- Change of State

The protocol supports the programming of the following additional functions:

- Code sequence (Complement)
- Resolution per revolution
- Total resolution
- Preset value
- Baudrate
- MAC-ID

The general use of absolute rotary encoders with DeviceNet interface is guaranteed.

1.1 Control and Information Protocol (CIP)



The DeviceNet specification defines the Application Layer and the Physical Layer. The Data Link layer is based on the CAN-specification. For the optimal industrial control will be defined two different messaging types. I/O messaging (Implicit Messaging) and explicit messaging.With Implicit Messaging becoming I/O data exchanged in realtime and with Explicit Messaging becoming data exchanged to configure a device.

CIP (Common Industrial Protocol) make for the user available four essential functions:

- Unique control service
- Unique communication service
- Unique allocation of messaging
- Common knowledge base

1.2 Object modell

DeviceNet describes all data and functions of a device considering as object model. By means of that object-oriented description a device can be defined complete with single objects. A object is defined across the centralization by associated attributes (e.g. processdata), his functions (read- or write access of a single attribute) as well as by the defined behaviour.

DeviceNet distinction is drawn between three different objects:

- Communication object
  Define the exchange messages over DeviceNet and becoming designated as Connection Objects. (DeviceNet Object, Message Router Object, Connection Object, Acknowledge Handler Object)
- System objects
  Define common DeviceNet-specific data and functions. (Identity Object, Parameter Object)
- Applications-specific objects
  Define device-specific data and functions. (Application Object, Assembly Object)

2. Data Transmission

The data transmission in the DeviceNet network is realised by message telegrams. Basically, these telegrams can be divided into the CAN-ID and 8 following bytes as shown in the table below:

CAN-ID	Message Header	Message Body
11 Bit	1 Byte	7 Byte

2.1. The Object Dictionary

Instance Attribute of the Position Sensor Objects

Class Code: 23 hex

Attribute ID	Access	Name	Data Type	Description
1 hex	Get	Number of Attributes	USINT	Number of supported Attributes
2 hex	Get	Attribute	Array of USINT	List of supported Attribute
3 hex	Get	Position value	DINT	current position
0B hex	Get / Set	Code sequence	Boolean	Controls the code sequence clockwise or counterclockwise
2C hex	Get / Set	resolution per revolution	INT	resolution for one revolution
2D hex	Get / Set	total resolution	DINT	total measurable resolution
2E hex	Get / Set	preset value	DINT	setting a defined position value
6E hex	Get / Set	Baudrate		Adjustment of the Baudrate
6F hex	Get / Set	MAC ID		Adjustment of the MAC ID

Get / Set: : read, write

2.2 Definition of the CAN-ID

DeviceNet is based on the standard CAN-protocol and used a 11Bit (2048 specifiable messages) messages identifier. For the identification of a device in a DeviceNet network are 6Bit enough because a network belongs 64 nodes. That nodes will be call MAC-ID. The CAN-Identifier consists of the Message Group, Message ID and the MAC ID of the device.
By our absolute rotary encoder it is a matter of a Group 2 Messages. In the table below a user can see the importance CAN-IDs for a certain communication type.
.

10	9	8	7	6	5	4	3	2	1	0	Identity Usage	Hex Range
0	Group 1 Message ID				Source MAC ID						GROUP 1 Message	000-3ff
0	1	1	0	1	Source MAC ID						Slave's I/O Change of State or Cyclic Message	
0	1	1	1	1	Source MAC ID						Slave's I/O Poll Response or Change of State/Cyclic Acknowledge Message	
1	0	MAC ID						Group 2 Message ID			GROUP 2 Messages	400 - 5ff
1	0	Destination MAC ID						0	1	0	Master's Change of State or Cyclic Acknowledge Message	
1	0	Source MAC ID						0	1	1	Slave's Explicit/Unconnected Response Messages	
1	0	Destination MAC ID						1	0	0	Master's Explicit Request Message	
1	0	Destination MAC ID						1	0	1	Master's I/O Poll Command/Change of State/Cyclic Message	
1	0	Destination MAC ID						1	1	0	Group 2 Only Unconnected Explicit Request Message (reserved)	
1	0	Destination MAC ID						1	1	1	Duplicate MAC ID Check Messages	

3. Programmable Parameters

3.1. Encoder parameters

3.1.1. Operating Parameter

The operating parameter can be used to select the code sequence.

Attribute ID	Default value	Value range	Data Type
0 b hex	1 hex	0 hex - 1hex	Boolean

The parameter code sequence (complement) defines the counting direction of the process value **as seen on the shaft** whether clockwise or counter clockwise. The counting direction is defined in the attribute 0b hex:

Bit 0	Drehrichtung	Ausgabecode
1	CW	Steigend
0	CCW	Fallend

3.1.2. Resolution per revolution

The parameter resolution per revolution is used to program the encoder to set a desired number of steps per revolution. Each value between 1 and the maximum (see type shield) can be realised

Attribute ID	Default value	Value range	Data Type
2C hex	(*)	0hex - 2000hex	Unsigned Integer16

(*) see type shield, Maximum resolution:
12/24 Bit Encoder: 1,000 hex (4096)
13/25 Bit Encoder: 2,000 hex (8192)

When the value is set larger than 4096 (8192 for a 13/25 Bit encoder), the process value of the encoder will not be single stepped and values will be skipped while rotating the shaft. So, it is recommended, to keep the measuring steps per revolution below 4096 (8192) measuring steps.

3.1.3. Total resolution

This value is used to program the desired number of measuring steps over the total measuring range. This value must not exceed the total resolution of the encoder with 24 bit = 16,777,216 steps (25 bit = 33,554,432 steps). Please note the value written on the type shield.

Attribute ID	Default value	Value range	Data Type
2D hex	(*)	0h - 2,000,000h	Unsigned Integer 32

(*) see type shield
Maximum total resolution
24 Bit Encoder: 1,000,000 hex
25 Bit Encoder: 2,000,000 hex

Attention:
The following formula letters will be used:

- PGA Physical total resolution of the encoder (see type shield)
- PAU Physical resolution per revolution (see type shield)
- GA Total resolution (customer parameter)
- AU Resolution per revolution (customer parameter)

If the desired resolution per revolution is less than the physical resolution per revolution of the encoder, then the total resolution must be entered as follows:

Total resolution
GA = PGA * AU / PAU, if AU < PAU
Example: Customer requirement: AU = 2048, Encoder type shield: PGA=24 bit, PAU=12 bit

GA = 16777216 * 2048 / 4096
GA = 8388608

If the total resolution of the encoder is less than the physical total resolution, the parameter total resolution must be a multiple of the physical total resolution:

- k = PGA / GA
- k = integer

3.1.4. Preset value

The preset value is the desired position value, which should be reached at a certain physical position of the axis. The position value of the encoder is set to the desired process value by the parameter preset. The preset value must not exceed the parameter total measuring units

Attribute ID	Default value	Value range	Data Type
2E hex	0 hex	0hex - total measuring range	Unsigned Integer 32

3.1.5. MAC-ID

Attribute ID	Default value	Value range	Data length
6F hex	0 hex	0hex – 3Fhex	BYTE

Each node in a DeviceNet network is identified using a MAC-ID (Media Access Control Identifier). Every device needs an explicit and unique MAC-ID. A DeviceNet netwok supports 64 nodes. The MAC-ID can only be adjusted via explicit messaging. The default MAC-ID is setting on d63.

3.1.6. Baudrate

Attribute ID	Default value	Value range	Data length
6E hex	0 hex	0hex - 2hex	BYTE

DeviceNet supports three different baurates that are being showed in the below table. The baudrate can be changed via explicit messages and stored in the EEPROM with a save command. It is to insure that the selective baudrate has to be the same as the DeviceNet network baudrate. The default baudrate is setting 125kBaud.

0x	Baudrate in kBaud
0	125
1	250
2	500

4. Operating Mode

4.1. Polled Mode

For switching the polled mode on the following telegrams are needed. Further it is assumed a master MAC ID of 0A hex and a slave MAC ID of 03 hex in the following example.

Allocate Master / Slave Connection Set

1. Allocate Polling

Byte Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Frag [0]	XID	MAC ID					
1	R/R [0]	Service [4B]						
	Class ID [03]							
	Instance ID [01]							
	Allocation Choice [03]							
	0	0	Allocator MAC ID					

Definition CAN ID

10	9	8	7	6	5	4	3	2	1	0	Identity Usage	Hex Range
1	**0**	**Destination MAC ID**						**1**	**1**	**0**	**Group 2 Only Unconnected Explicit Request Message (reserved)**	

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
41E	0A	4B	03	01	03	0A

1. Setting the Expected_packet_rate of the Explicit Message Connection on 0:

Definition CAN-ID

10	9	8	7	6	5	4	3	2	1	0	Identity Usage	Hex Range
1	**0**	**Destination MAC ID**						**1**	**0**	**0**	**Master´s Explicit Request Message**	

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
41C	0A	10	05	01	09	00	00

1. Setting the Expected_packet_rate of the Polling Connection on 0:n:

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
41C	0A	10	05	02	09	00	00

Release Master / Slave Connection Set

Release Polling

Byte Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Frag [0]	XID	MAC ID					
1	R/R [0]	Service [4C]						
	Class ID [03]							
	Instance ID [01]							
	Release Choice [03]							

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
41E	0A	4C	03	01	03

4.2. Change of State Mode

The absolute rotary encoder sends data, without any request from the host, when the actual process value is changing. No telegram will occur when the position value is not changing. This results in a reduced bus loading.

Allocate Master / Slave Connection Set

Allocate COS

Byte Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Frag [0]	XID	MAC ID					
1	R/R [0]	Service [4B]						
	Class ID [03]							
	Instance ID [01]							
	Allocation Choice [51]							
	0	0	Allocator MAC ID					

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
41E	0A	4B	03	01	51	0A

2. Setting Expected_packet_rate of the Explicit Message Connection on 0:

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
41C	0A	10	05	01	09	00	00

3. Setting Expected_packet_rate of the Change of State Connection on 0:

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
41C	0A	10	05	04	09	00	00

Release Master / Slave Connection Set

Release COS

Byte Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Frag [0]	XID	MAC ID					
1	R/R [0]	Service [4C]						
	Class ID [03]							
	Instance ID [01]							
	Release Choice [51]							

Example:

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
41E	0A	4C	03	01	51

4.3. Saving Parameter

The parameters of the absolute rotary encoder are saved in a non-volatile FLASH memory. Because of a limited number of writing cycles (≈ 1,000), it is useful to transmit the modified parameter in the first step only in the RAM area. After adjusting and examination, those values can be saved in the FLASH memory. After successful saving of the parameter the encoder sends his MAC-ID on the bus. To get the process value a new allocation of the slave is required.

Byte Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Frag [0]	XID	MAC ID					
1	R/R [0]	Service [32]						
	Class ID [23]							
	Instance ID [01]							

Example:
(MAC-ID Master: 0A hex, MAC-ID Slave: 03 hex)

CAN-ID	Byte 0	Byte 1	Byte 2	Byte 3
41C	0A	32	23	01

5. Transmission of the actual position

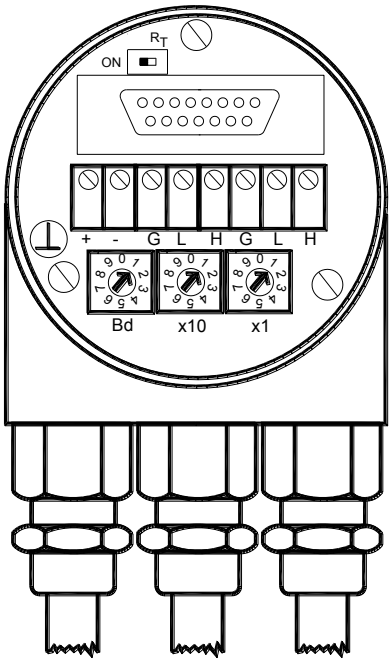
The process value is transmitted according to the following table.

CAN-ID	process value			
11 Bit	Byte 0	Byte 1	Byte 2	Byte 3
	2^7 to 2^0	2^{15} to 2^8	2^{23} to 2^{16}	2^{31} to 2^{24}

6. Installation

6.1. Electrical connection

The rotary encoder is connected by three cables. The power supply is achieved with a two-wire connection cable through one PG 9. Each one of the twisted-pair and shielded bus lines are guided in and out through two PG 9 on the right side (as seen on clamps)



Clamp	Description
⊥	Ground
+	24 V Supply voltage
-	0 V Supply voltage
CG	CAN Ground
CL	CAN Low
CH	CAN High
CG	CAN Ground
CL	CAN Low
CH	CAN High

There is a resistor provided in the connection cap, which must be used as a line termination on the last device

Resistor:

The setting of the node number is achieved by 2 turn-switches in the connection cap. Possible addresses lie between 0 and 63 whereby every address can only be used once. 2 LEDs on the backside of the connection cap show the operating status of the encoder.

DeviceNet Devices BCD coded rotary switches	
x1 x10	Device adress 0...63 Setting CAN-node number
xBd	Setting of the baud-rate

6.2. Setting of the baudrate

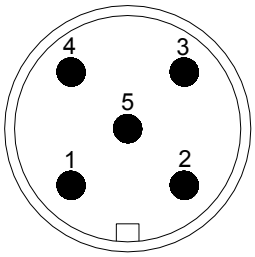
Baudrate in kBit/s	BCD coded rotary switches
125	0
250	1
500	2
125	3
reserved	4...9

6.3 Cable

Pin	Signal	Description	Color
1	V-	GND	Black
2	CAN-L	CAN Bus signal (dominant low)	Blue
3	CAN-H	CAN Bus signal (dominant high)	White
4	V+	External voltage supply Vcc	Red

6.3 Connector

Pin	Signal	Description	Color
2	V+	External voltage supply Vcc	Red
3	V-	GND	Black
4	CAN-H	CAN Bus signal (dominant high)	White
5	CAN-L	CAN Bus signal (dominant low)	Blue



5 pin connector

7. Power On

7.1. Operating Mode

After power on the absolute rotary encoder sends two times his MAC ID telegram on the bus.

7.2. Programming

If some parameters should not be modified you can skip over this chapter.
The following numbers are given in hexadecimal format. In the examples, the CAN ID and MAC ID are 0A (hex) and for the slave 03 (hex).
The changeable values are written in Italic.

7.2.1. Operating Parameter

Master to absolute rotary encoder: Set-Parameter

CAN ID	MAC ID	Service Code	Class ID	Instance ID	Attribute ID	Data		
	Byte 0	Byte1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*0A*	10	23	01	0b	X	-	-

X: 1 hex for CW (Default)

0 hex for CCW

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID	Service Code
	Byte 0	Byte 1
41B	*0A*	90

7.2.2. Resolution per revolution

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Service Code	Class ID	Instance ID	Attribute ID	Data		
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*0A*	10	23	01	2C	X	*X*	-

X: desired resolution per revolution

Absolute rotary encoder to master: Confirmation

CAN ID	MAC ID	Service Code
	Byte0	Byte1
41B	0A	90

7.2.3. Total resolution

A fragmented transmission is needed, when the total resolution must be sent to the encoder.
So here are more messages necessary.

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Fragment	Service Code	Class ID	Instance ID	Attribute ID		
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*8A*	00	10	23	01	2D	X	X

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID		
	Byte0	Byte 1	Byte 2
41B	8A	C0	00

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Fragment						
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*8A*	81	X	X	-	-	-	-

X: desired total resolution

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID		
	Byte0	Byte 1	Byte 2
41B	*8A*	C1	00

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID	Service Code
	Byte0	Byte1
41B	*0A*	90

7.2.4. Preset Value

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Fragment	Service Code	Class ID	Instance ID	Attribute ID		
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*8A*	00	10	23	01	2E	X	X

X: desired preset value

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID		
	Byte0	Byte 1	Byte 2
41B	*8A*	C0	00

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Fragment						
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	*8A*	81	X	X	-	-	-	-

X: desired preset value

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID		
	Byte0	Byte 1	Byte 2
41B	*8A*	C1	00

Absolute Rotary Encoder to Master: Confirmation

CAN ID	MAC ID	Service Code
	Byte0	Byte1
41B	*0A*	90

7.2.5. MAC-ID

Master to encoder: Set-Parameter

CAN ID	MAC ID	Service Code	Class ID	Instance ID	Attribute ID	Data		
	Byte0	Byte1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	0A	10	23	01	6F	X	-	-

X: Value of the MAC-ID

Encoder to Master: Confirmation

CAN ID	MAC ID	Service Code
	Byte0	Byte1
41B	0A	90

7.2.6. Baudrate

Master to encoder: Set-Parameter

CAN ID	MAC ID	Service Code	Class ID	Instance ID	Attribute ID	Data		
	Byte0	Byte1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
41C	0A	10	23	01	6E	X	-	-

X: Value of the Baudrate

X	Baudrate
o	125kbaud
1	250kbaud
2	500kbaud

Encoder to Master: Confirmation

CAN ID	MAC ID	Service Code
	Byte0	Byte1
41B	0A	90

7.2.7. Parameter Saving

Master to Absolute Rotary Encoder: Set-Parameter

CAN ID	MAC ID	Service Code	Class ID	Instance ID
	Byte0	Byte1	Byte 2	Byte 3
		32	23	01

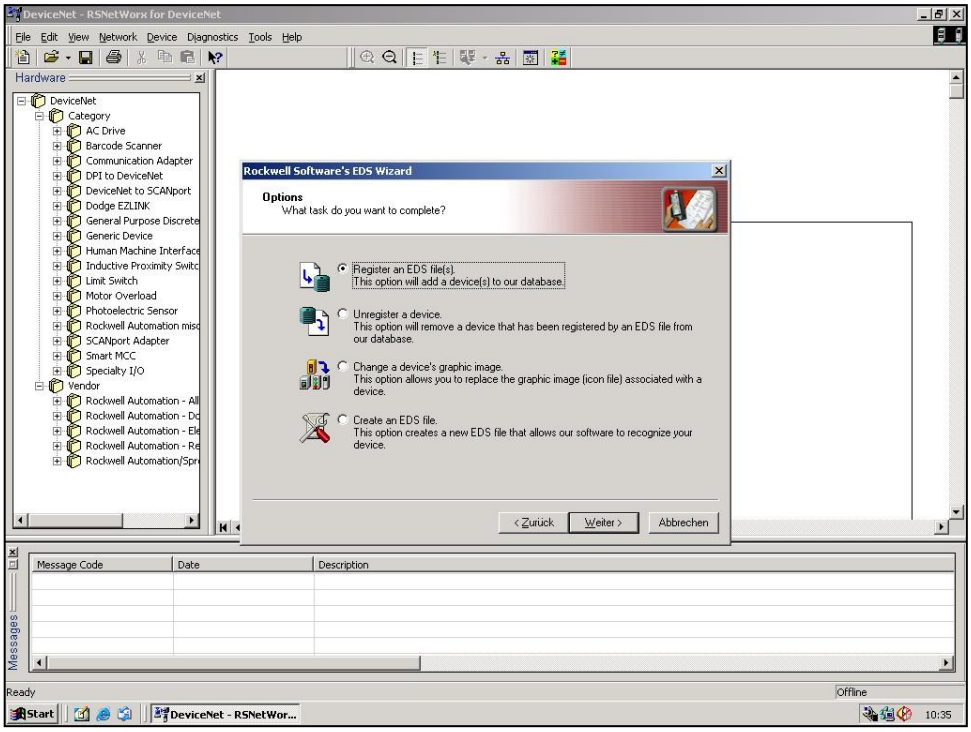
If the transfer has been successful, the absolute rotary encoder responds after 3-4s with the Duplicate MAC-ID. After that the master must reallocate the slave.

If the transfer is not successful, an error message will be sent. The service code used to save the parameter set is manufacturer specific.

8. RsNetworx

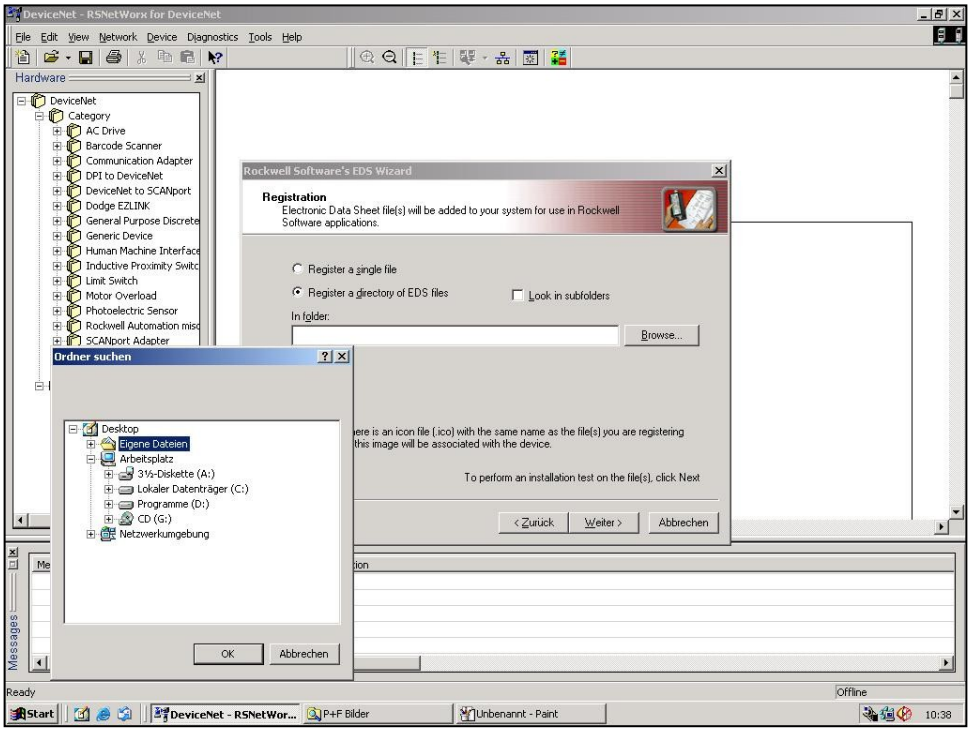
8.1. EDS Wizard

The EDS File contains information about device specific parameters as well as possible operating modes of the encoder. With this file you have a data sheet in an electronic format, which can be used to configure the device in the network, for example with RsNetworx from Rockwell.



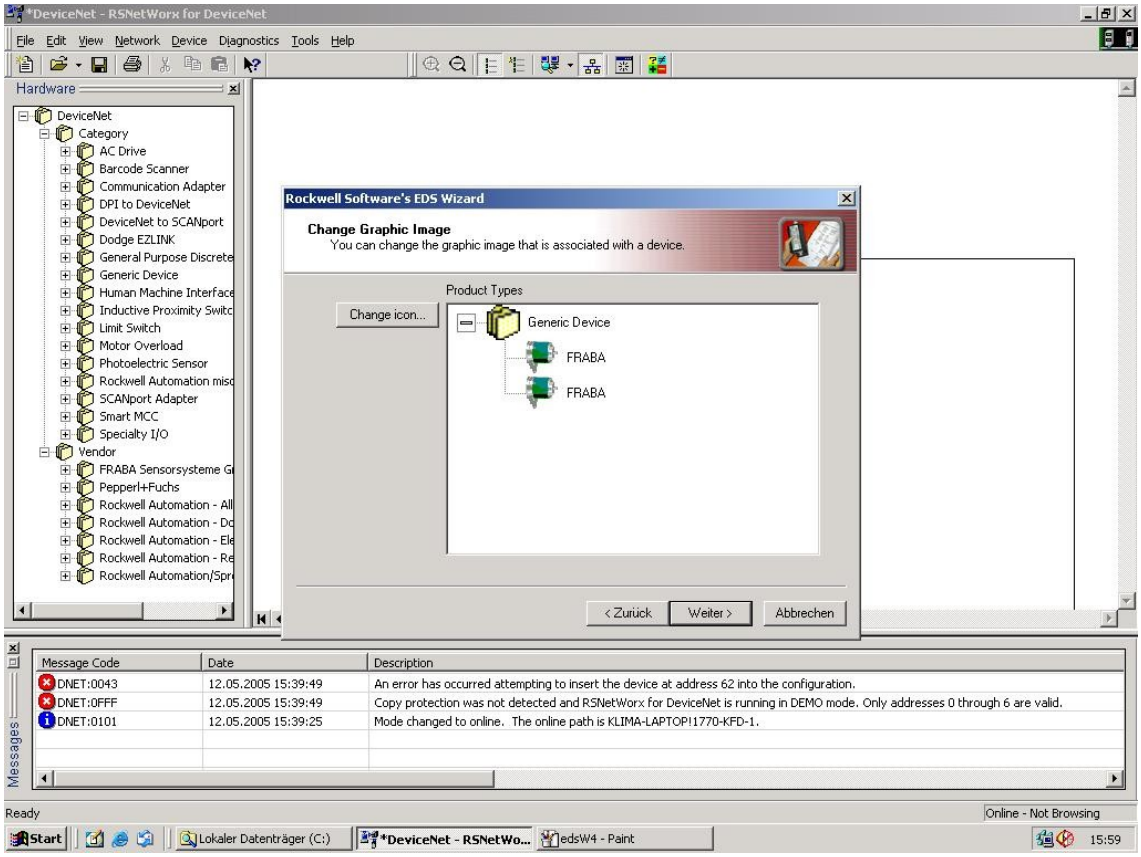
1.1 EDS Wizard

To install the EDS file the EDS Wizard has to be started, that can be done in the menu *Tools/EDS Wizard.* If the EDS Wizard is activated successfully the *Register an EDS File(s)* has to be chosen and after that the button *weiter*. In the next step the *Register a directory of EDS files* has to be chosen and with *Browse* the path of the EDS file(s). That is indicated in picture 1.2.



1.2 EDS Wizard

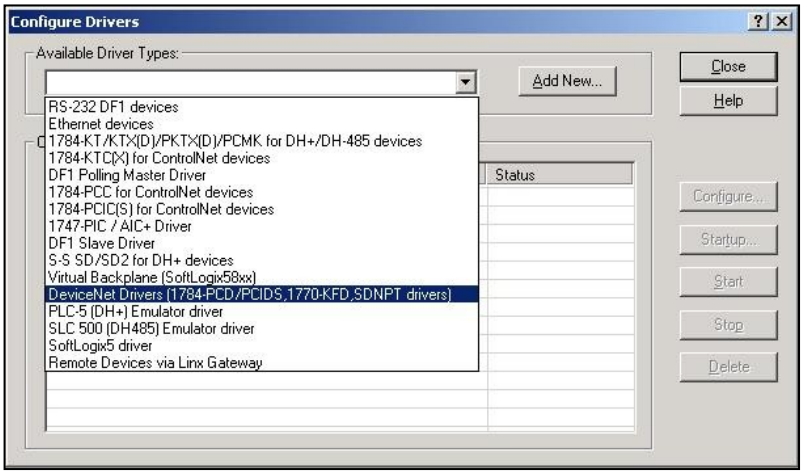
The Wizard finds all EDS files that are discarded in the choosing path and operates a test to check the EDS files on errors. In the next step (see picture 1.3) pictures can be selected for the using nodes. With the button *weiter* the installation can be continued and finished.



1.3 EDS Wizard

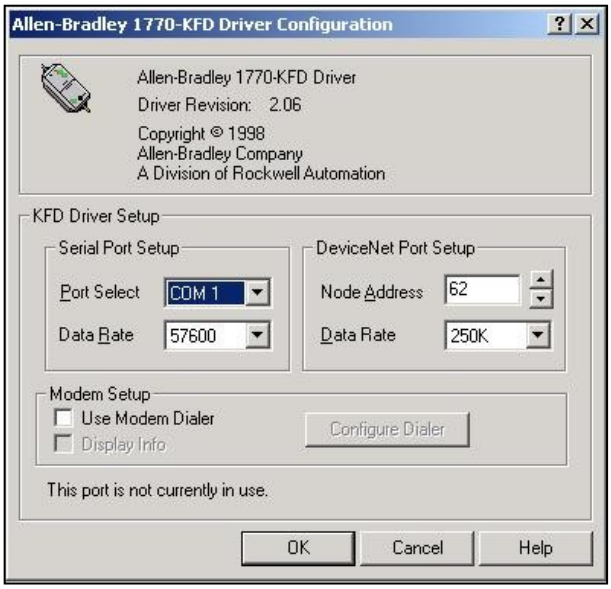
8.2 Driver Configuration

After a successful installing of the EDS file the next step is to choose the suitable driver. With *Start/Programme/Rockwell Software/RSLinx* in the menu the programm RSLinx can be started. With this programm the suitable driver can be chosen. For this example the driver typ 1770-KFD is being used. In the next step the window *Configure Drivers* in the menu *Communications/ Configure Drivers* has to be started. In the drop down Menü *Available Driver Types* the driver typ 1770-KFD has to be chosen and confirmed with the button *Add New.* (See picture 1.4)



1.4 Configure Drivers

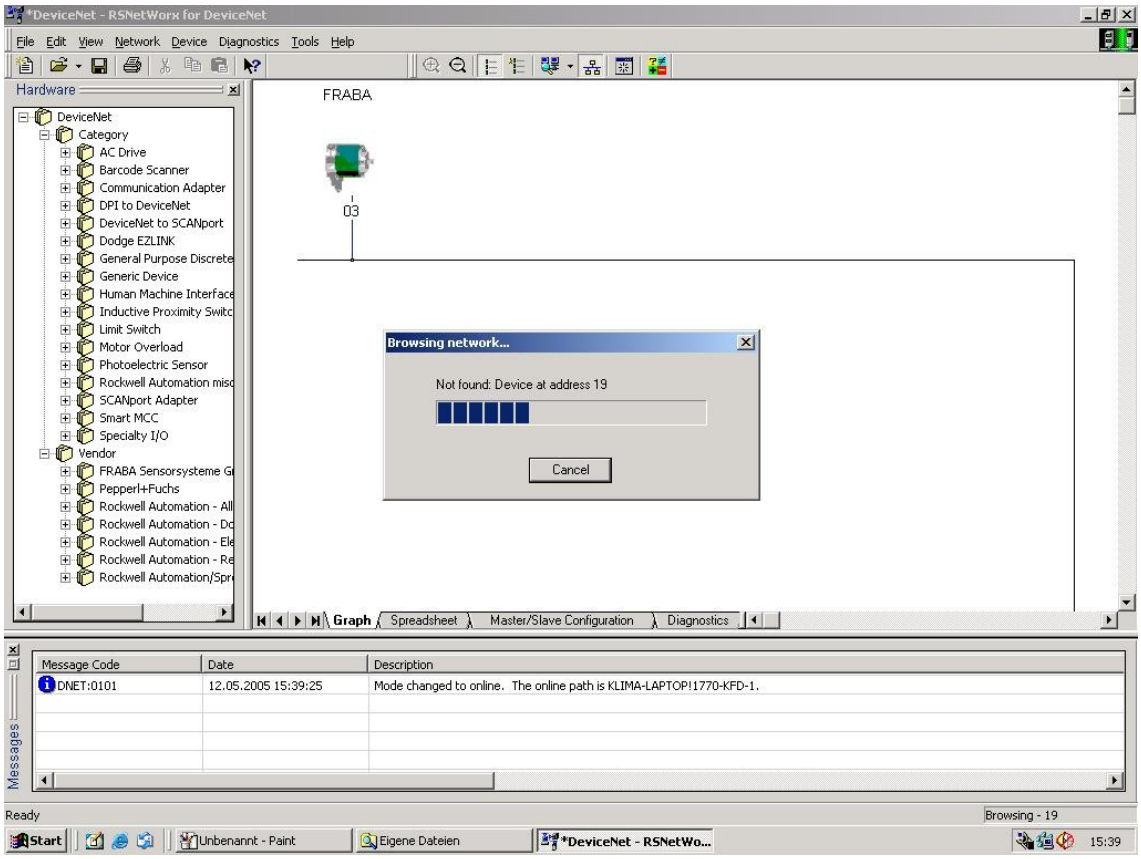
If the suitable driver is chosen it can be configured in the window *Driver Configuration.* In this step the correct baudrate has to be registered (picture 1.5). In the next step a requested name can be registered.



1.5 Driver Configuration

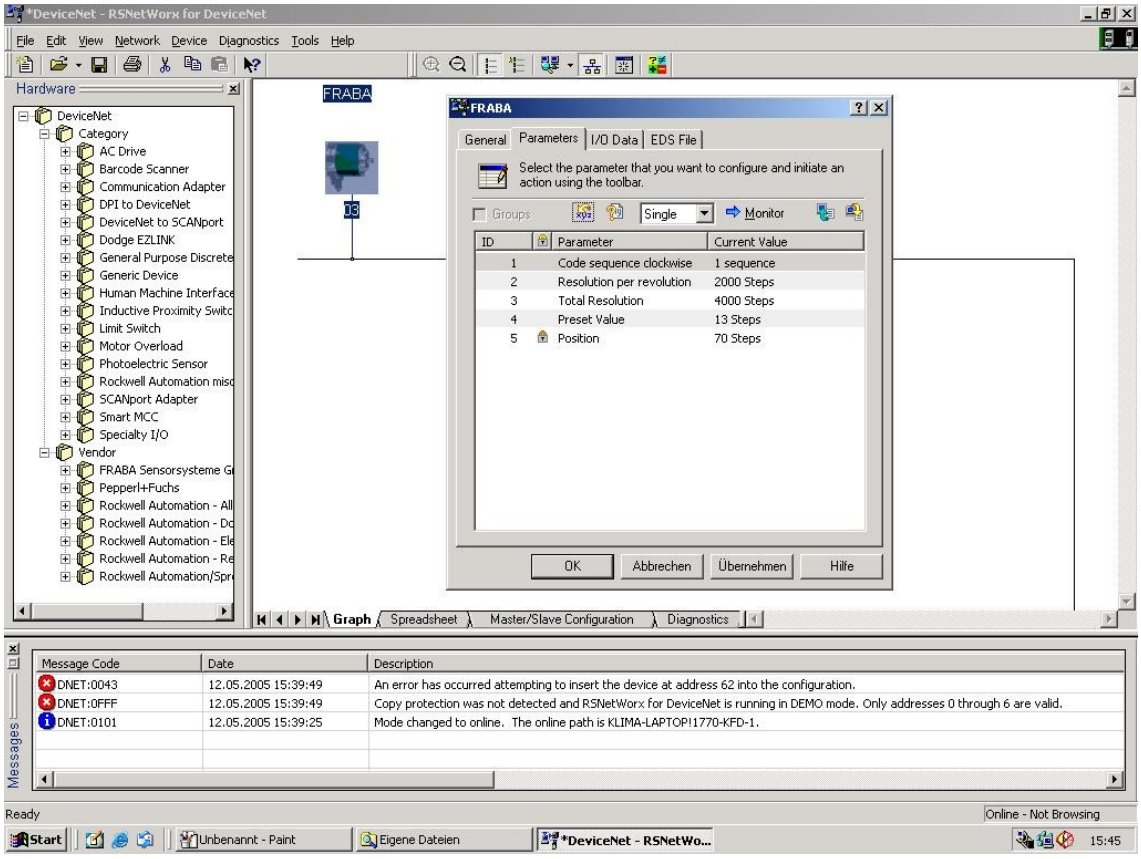
8.3 Network Connection

This chapter will explain how to switch a network online and how to parametrise a encoder. In the menu *Network/ Online* the window *Browse for network* will be opened. If the driver *1770-KFD* has been choosen, this is explained in chapter 6.2, the network is online. After that RsNetworx searches in the network for connecting nodes. That is also being showed in picture 1.6.



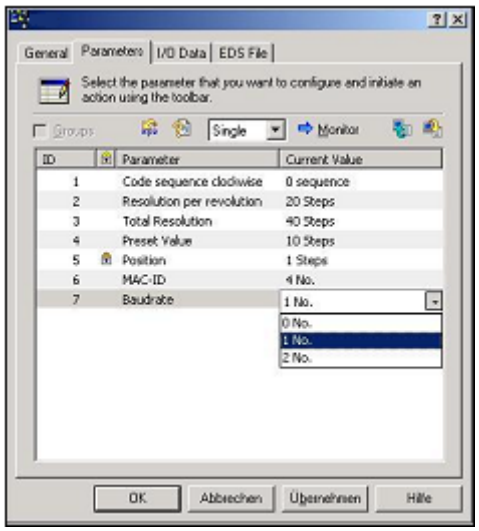
1.6 Browsing Network

To cofigure the encoder the configuration window in the menu *Device/Properties* has to be opened. By pushing *Parameters* an upload of the encoder parameter is realized.

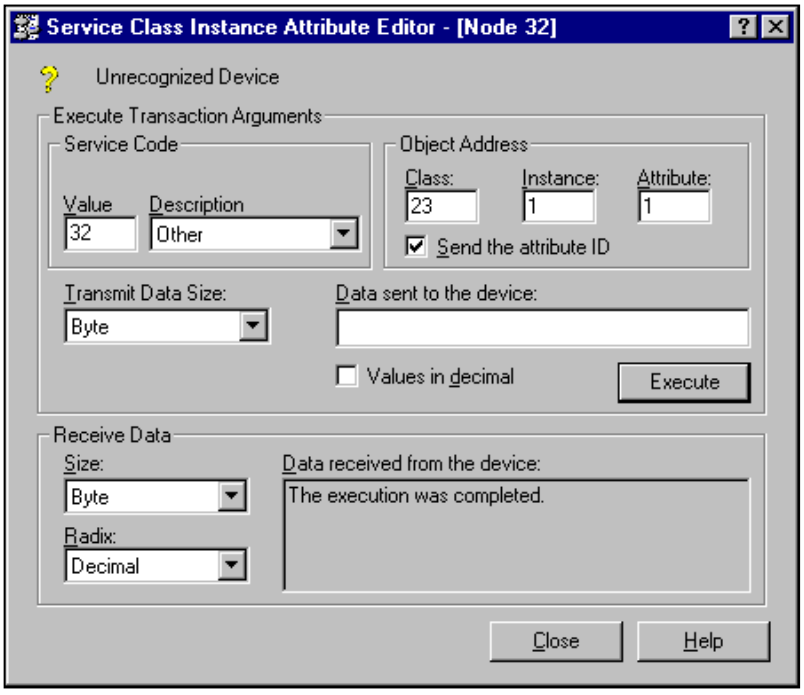


1.7 Upload Parameter

After a successful upload of the parameters, those can be configured as the picture 1.8 below shows. A download of the configured parameters can be realized with the yellow arrow that is showing down and is placed at the top right in the configuration window. An upload can be realized with the arrow beside the download arrow which is showing up. To show the position value the button *Monitor* has to be pushed. It should be noticed that the configuration parameters are not stored in the EEPROM. To store the parameters in the EEPROM the window in the menu *Device/Class Instance Editor* has to be opened. The entries that are necessary to store the parameters are being showed in the picture 1.9 below. At last the button *execute* has to be executed to store the parameters in the EEPROM.



1.8 Configure Parameters



1.9 Service Class Instance Attribute Editor

9. Technical Data

9.1 Electrical Data

Interface	Transceiver according ISO/DIS 11898, up to 64 nodes galvanically isolated by opto-couplers
Transmission rate	150 kBaud, 250 kBaud, 500kBaud
Device addressing	Adjustable by rotary switches in connection cap
Supply voltage	10 - 30 V DC (absolute limits)
Current consumption	max. 230 mA with 10 V DC, max. 100 mA with 24 V DC
Power consumption	max. 2.5 Watts
Step frequency LSB	800 kHz
Accuracy of division	± ½ LSB (12 bit), ± 2 LSB (16 bit)
EMC	Emitted interference: EN 61000-6-4
	Noise immunity: EN 61000-6-2
Electrical lifetime	> 10^5 h

9.2 Mechanical Data

Housing	Aluminum, optional stainless steel
Lifetime	Dependent on shaft version and shaft loading – refer to table
Max. shaft loading	Axial 40 N, radial 110 N
Inertia of rotor	≤ 30 gcm^2
Friction torque	≤ 3 Ncm (without shaft sealing)
RPM (continuous operation)	Singleturn: max. 12,000 RPM
	Multiturn: max. 6,000 RPM
Shock (EN 60068-2-27)	≤ 30 g (halfsine, 11 ms)
Permanent shock (EN 60028-2-29)	≤ 10 g (halfsine, 16 ms)
Vibration (EN 60068-2-6)	≤ 10 g (10 Hz ... 1,000 Hz)
Weight (standard version)	Singleturn: ≈ 550 g
	Multiturn: ≈ 600 g
Weight (stainless steel version)	Singleturn: ≈ 1,100 g
	Multiturn: ≈ 1,200 g

Flange	Synchro (S)		Clamp (C)	Hollow shaft (B)
Shaft diameter	6 mm	10 mm	10 mm	15 mm
Shaft length	10 mm	20mm	20 mm	-
hollow shaft depth min. / max.	-	-	-	15 mm / 30 mm

9.3 Minimum (mechanical) lifetime

Flange	Lifetime in 10^8 revolutions with F_a / F_r		
	40 N / 60 N	40 N / 80 N	40 N / 110 N
C10 (Clamp flange 10 x 20)	247	104	40
S10 (Synchro flange 10 x 20)	262	110	42
S6 (Synchro flange 6 x 10) without shaft sealing	822	347	133

S6 (Synchro flange 6 x 10) with shaft sealing: max. 20 N axial, 80 N radial

9. 4 Environmental Conditions

Operating temperature	- 40 .. + 85°C
Storage temperature	- 40 .. + 85 °C
Humidity	98 % (without liquid state)
Protection class (EN 60529)	Casing side: IP65
	Shaft side: IP64 (optional with shaft sealing: IP66)