CAN-8123/CAN-8223/
CAN-8423/CAN-8823
CANopen Slave Device

User's Manual

Warranty

Without contrived damage, all products manufactured by ICP DAS are warranted in one year from the date of delivery to customers.

Warning

ICP DAS revises the manual at any time without notice. However, no responsibility is taken by ICP DAS unless infringement act imperils to patents of the third parties.

Copyright

Copyright © 2007 is reserved by ICP DAS.

Trademark

The brand name ICP DAS as a trademark is registered, and can be used by other authorized companies.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Contents

1 Introduction 4
1.1 Overview 4
1.2 Hardware Features 6
1.3 CAN-8x23 Features 7
1.4 Utility Features 8
2 Hardware Specification 9
2.1 Hardware Structure 9
2.2 Layout Structure 11
2.3 Wire Connection 12
2.4 CAN Connector 13
2.5 Terminal Resistor Jumper and Initial Switch 15
2.6 Power LED 15
2.7 CANopen Status LED 16
2.7.1 The RUN LED 16
2.7.2 The ERR LED 17
2.8 The Node ID & the Baud rate Rotary Switch 19
2.9 I/O Pair-connection Mode 20
2.10 Module Support 21
3 CANopen Application 22
3.1 CANopen Introduction 22
3.2 SDO Introduction 29
3.3 PDO Introduction 31
3.4 EMCY Introduction 43
3.5 NMT Introduction 44
3.5.1 Module Control Protocols 45
3.5.2 Error Control Protocols 46
4 Configuration & Getting Start 49
4.1 CAN-8123/CAN-8223 Configuration Flowchart 49
4.2 CAN-8423/CAN-8823 Configuration Flowchart 51
4.3 CANopen Slave Utility Overview 53
4.4 Configuration with the CANopen Slave Utility 54
4.5 CAN-8123/8223 Configuration (Off-line mode) 55
4.6 CAN-8423/8823 Configuration (On-line mode) 60
5 CANopen Communication Set 65
5.1 SDO Communication Set 66
5.1.1 Upload SDO Protocol 66
5.1.2 SDO Block Upload Protocol 75

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1.3 Download SDO Protocol 86
5.1.4 SDO Block Download 91
5.1.5 Abort SDO Transfer Protocol 99
5.2 PDO Communication Set 102
5.2.1 PDO COB-ID Parameters 102
5.2.2 Transmission Type 104
5.2.3 PDO Communication Rule 105
5.3 EMCY Communication Set 144
5.3.1 EMCY COB-ID Parameter 144
5.3.2 EMCY Communication 145
5.4 NMT Communication Set 153
5.4.1 Module Control Protocol 153
5.4.2 Error Control Protocol 156
5.5 Special Functions for CAN-8x23 161
6 Object Dictionary of CAN-8x23 169
6.1 Communication Profile Area 169
6.2 Manufacturer Specific Profile Area 176
6.3 Standardized Device Profile Area 178
6.4 Object of Counter/Frequency Modules 182
6.5 Object of PWM Module (Only for I-8088W) 184
Appendix A: Type Code Table 186
Appendix B: DIO Type Define of I-8050 Modules 199

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

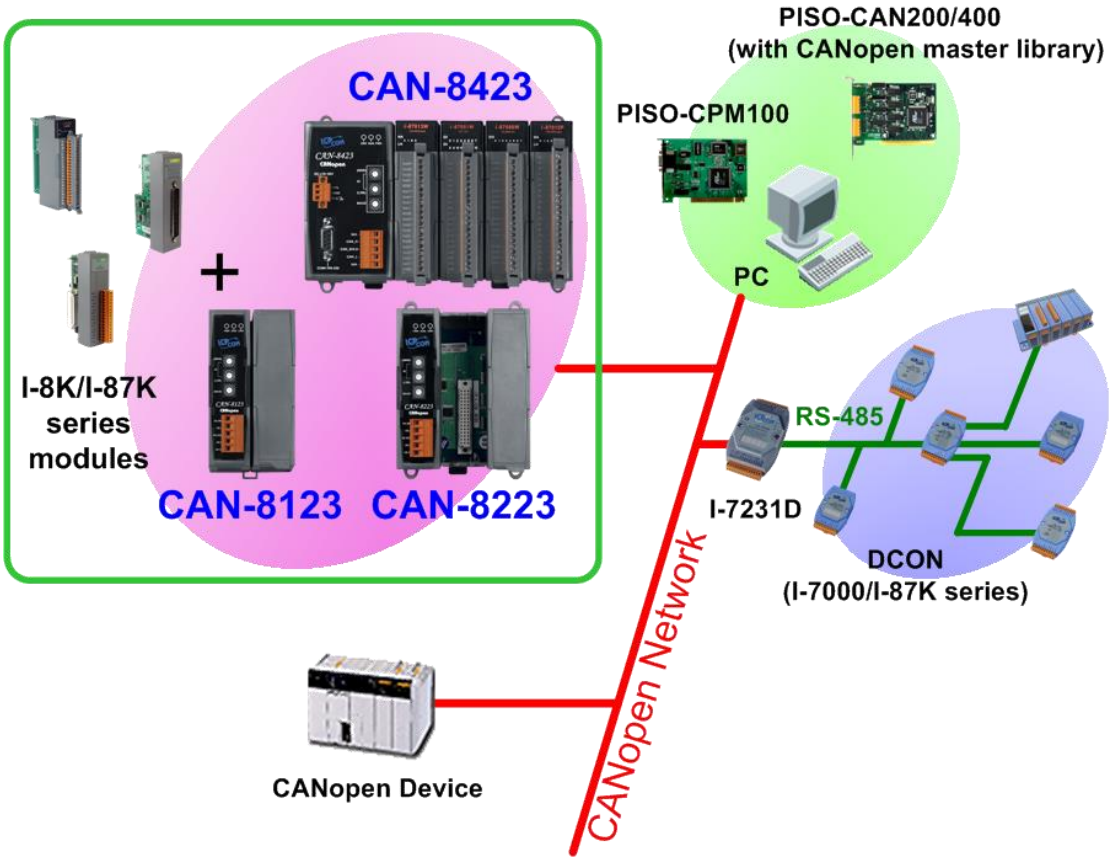
1 Introduction

1.1 Overview

CANopen, a kind of communication protocols, is an intelligent field bus (CAN bus). It has been developed as a standard embedded network with a high flexible configuration. It provides a standard communication protocol transmitting real-time data in PDO (**P**rocess **D**ata **O**bjects), configuration data in SDO (**S**ervice **D**ata **O**bjects), and network management data (NMT message, and Error Control), even supports the special functions (Time Stamp, Sync message, and Emergency message). Nowadays, CANopen is used on many applications and in specific fields, such as medical equipment, off-road vehicles, maritime electronics, public transportation, automation and so on.

The main control units CAN-8123/CAN-8223/CAN-8423/CAN-8823 (CAN-8x23 in general) are specially designed for the slave device of the CANopen protocols. In order to expand the I/O channel, and make it more flexible, the CAN-8x23 supports up to 8 expansion slots for users to increase applications by adding I/O channels. Users can choose either the I-87K or the I-8000 series DI/DO/AI/AO slot modules for their application purposes. The CAN-8123/CAN-8223 has one and two expansion slots respectively, and the CAN-8423/CAN-8823 supports four and eight expansion slots. Each expansion slot can insert in one I-87K or I-8000 series I/O module. All of these main control units follow the CANopen Spec DS-301 V4.01 and DS-401 V2.1, and supply a great deal of features to users, such as dynamic PDO, EMCY object, error output value, SYNC cyclic and acyclic and so forth. In addition, the CAN Slave Utility is also provided to allow users to create EDS files dynamically. EDS files based on the CANopen DS-306 is compatible with other CANopen master interface made by different manufacturers, also supporting the EDS files. The general application for the CAN-8x23 CANopen slave device architecture is as follows.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1.2 Hardware Features

- CPU:80186, 80MHz
- Philip SJA1000 CAN controller
- Philip 82C250 CAN transceiver
- SRAM:512 Kbytes
- Flash Memory:512 Kbytes
- EEPROM:2 Kbytes
- NVRAM: 32 bytes
- Real Time Clock
- Built-in Watchdog Timer
- 16-bit Timer
- Power LED, RUN LED, and ERR LED
- Support 1/2/4 expansion I/O slots
- 2500 Vrms isolation at CAN side
- 120Ω terminal resister selected by jumper(s)
- CAN bus interface: ISO/IS 11898-2, 5-pin screw terminal with on-board optical isolators' protection.
- Power Supply: 20W. Unregulated from +10VDC ~ +30VDC
- Operating Temperature:-25°C ~ +75°C
- Storage Temperature:-30°C ~ +85°C
- Humidity:5%~95% RH

COM1

- RS-232: TXD,RXD,RTS,CTS,GND
- Communication speed: 115200 bps.
- Configure tool connection

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1.3 CAN-8x23 Features

- NMT: Slave
- Error Control: Node Guarding、Heartbeat Producer
- Node ID: Setting by Rotary Switch
- No. of PDOs: 16 Rx, 16Tx
- PDO Modes: Event-triggered, remotely requested, cyclic and acyclic SYNC
- PDO Mapping: variable
- No of SDOs: 1 server, 0 client
- Emergency Message: Yes
- CANopen Version: CiA-301 v4.02
- Device Profile: CiA-401 v2.1
- Produce EDS file dynamically
- Baud Rate Selection : 10K, 20K, 50K, 125K, 250K, 500K, 800K and 1M bps
- Power LED, RUN LED, and ERR LED indicators
- Support I-8000 and I-87K I/O expansion slot:
  - CAN-8123: 1 slot
  - CAN-8223: 2 slots
  - CAN-8423: 4 slots
  - CAN-8823: 8 slots
- Provide a friendly Utility to configure the I-8000 and I-87K series modules

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1.4 Utility Features

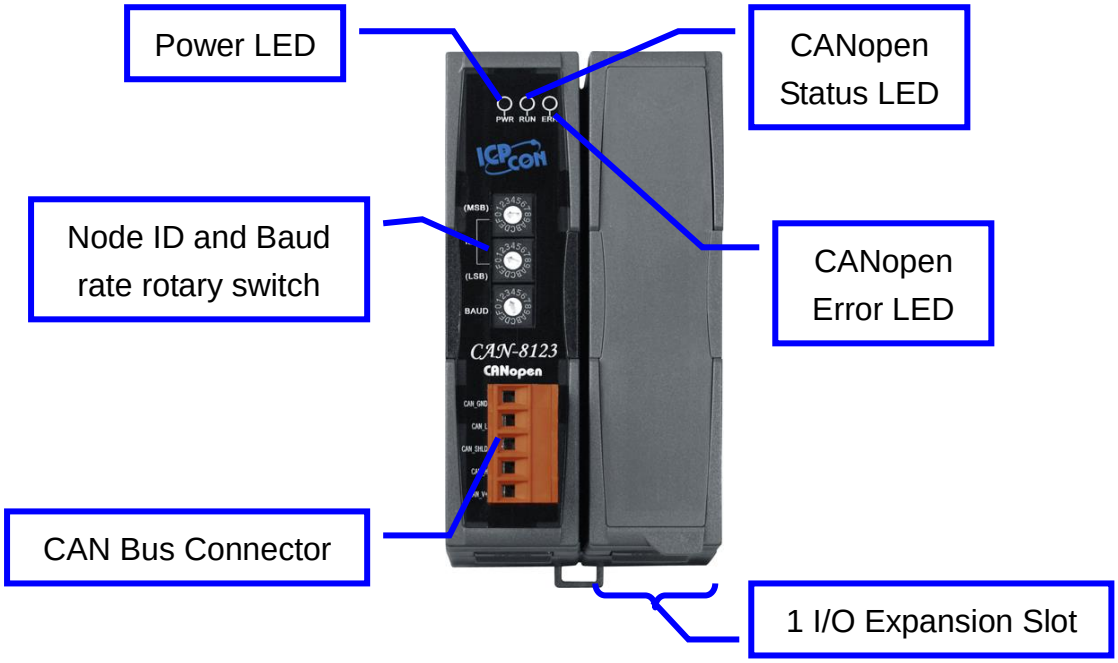
- Support parameter configuration on the I-8000 and I-87K modules
- Provide to show Application and Device Object information
- Provide to show Rx and Tx PDO mapping
- Support EDS file creation

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

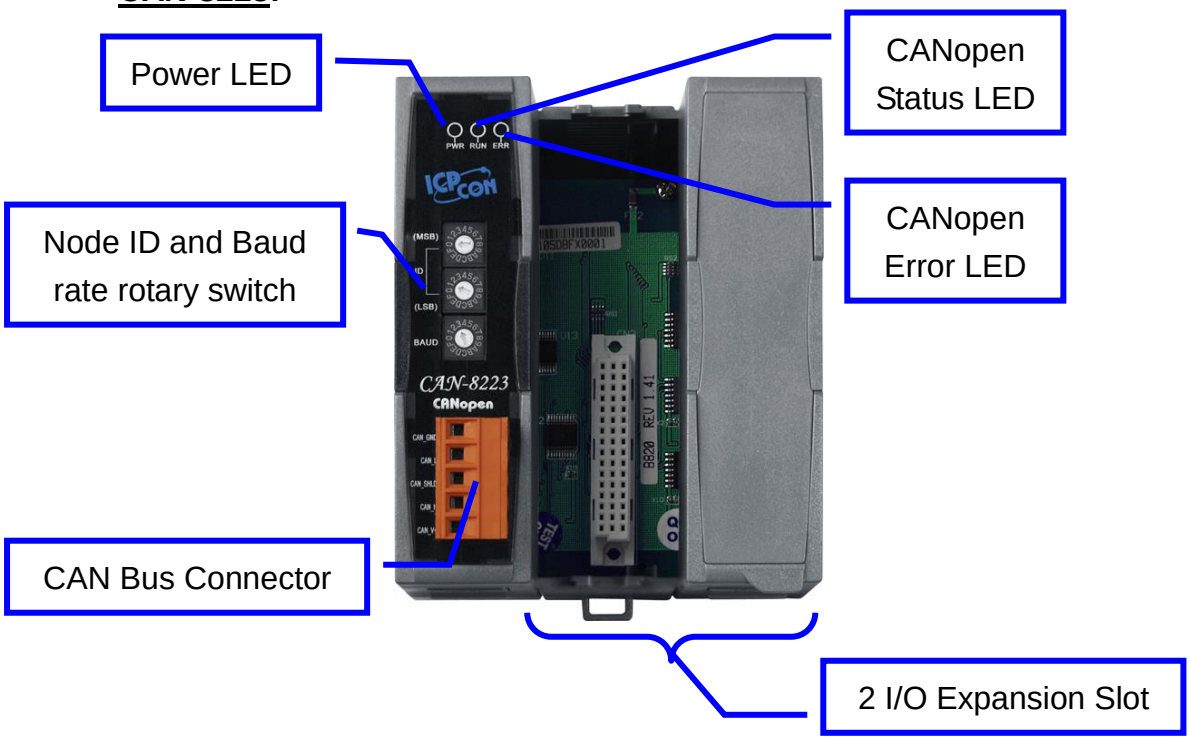
2 Hardware Specification

2.1 Hardware Structure

CAN-8123:

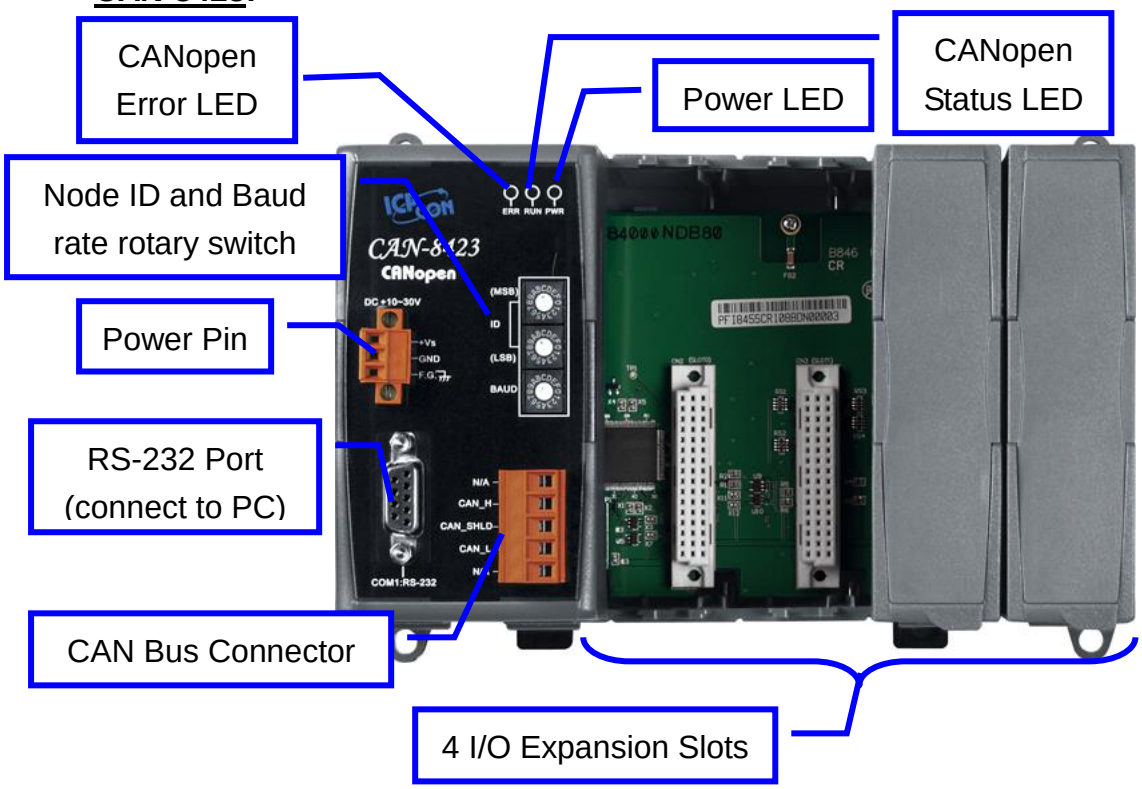


CAN-8223:

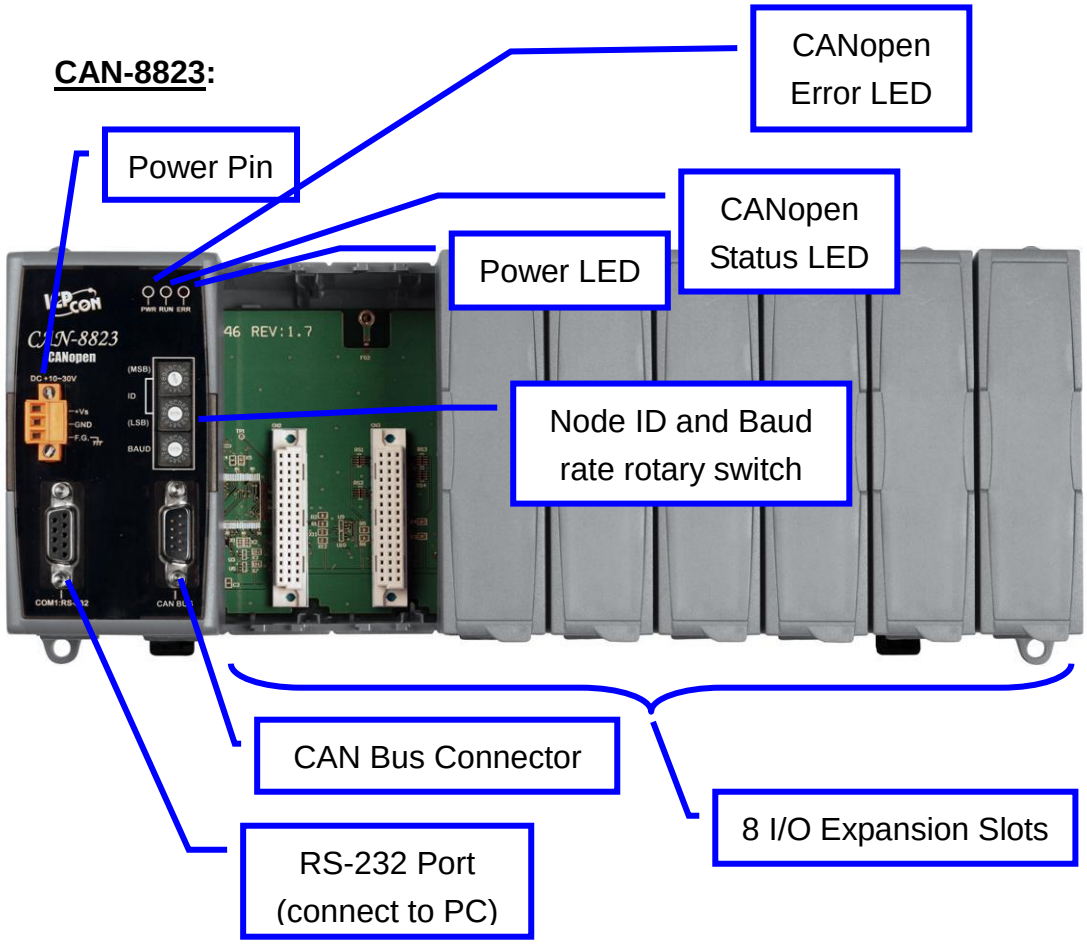


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

CAN-8423:



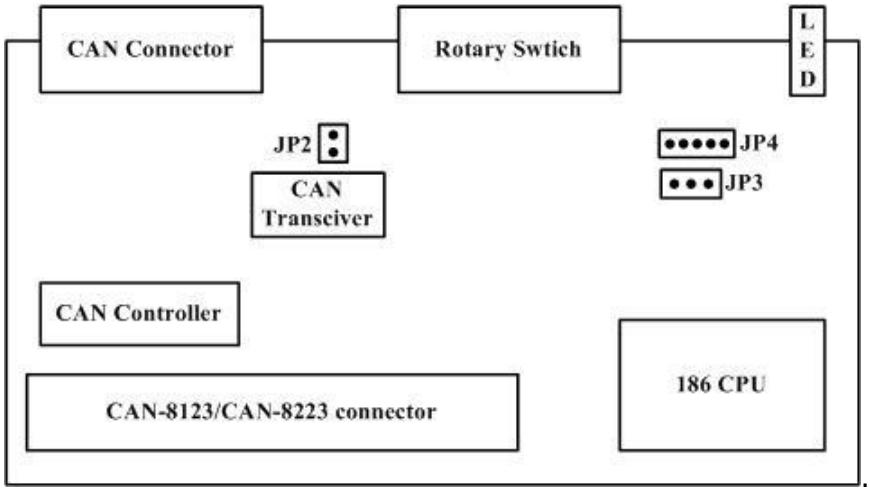
CAN-8823:



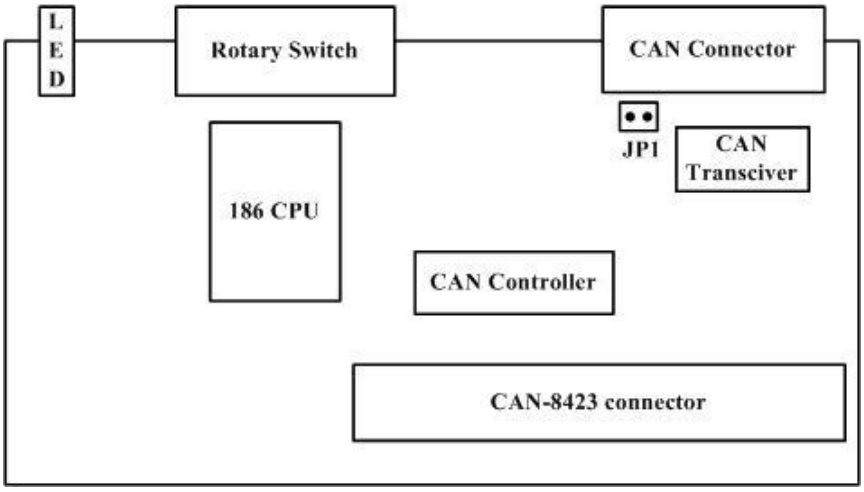
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.2 Layout Structure

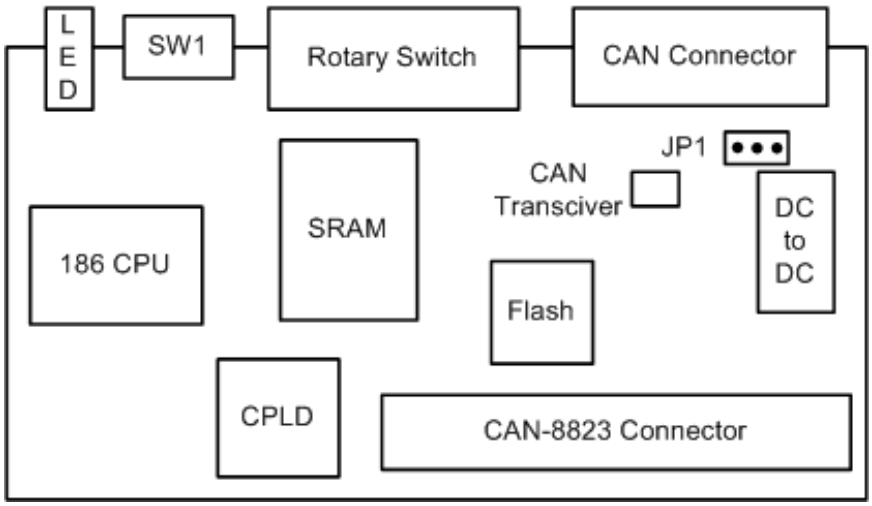
CAN-8123/CAN-8223:



CAN-8423:



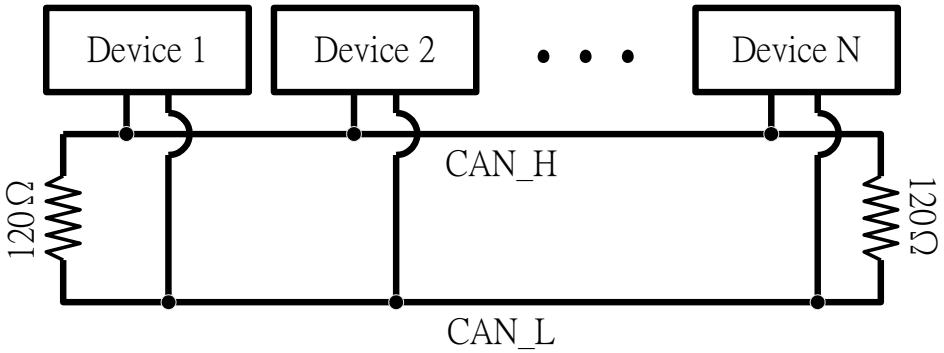
CAN-8823:



LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.3 Wire Connection

In order to minimize the reflection on the CAN bus line, the CAN bus line has to be terminated at both ends by two terminal resistances as shown in the following. According to the ISO 11898-2 spec, each terminal resistance is 120Ω (or other between 108Ω~132Ω). The length related resistance has to reach 70mΩ/m. At this circumstance, users would better check the resistances of the CAN bus before installing a new CAN network.



Moreover, to minimize the voltage drop, value of the terminal resistance must be higher than the one defined in the ISO 11898-2. The following table is for users' reference.

Bus Length (meter)	Bus Cable Parameters		Terminal Resistance (Ω)
	Length Related Resistance (mΩ/m)	Cross Section (Type)	
0~40	70	0.25(23AWG)~ 0.34mm^2(22AWG)	124 (0.1%)
40~300	< 60	0.34(22AWG)~ 0.6mm^2(20AWG)	127 (0.1%)
300~600	< 40	0.5~0.6mm^2 (20AWG)	150~300
600~1K	< 20	0.75~0.8mm^2 (18AWG)	150~300

In the CAN-8x23, the 120Ω terminal resistance is supplied as a standard accessory. About enable/disable the 120Ω terminal resistance jumps, please refer to section **2.5 "Terminal Resistor Jumper and Initial Switch"**.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

The bus length determines the CAN bus baud rate. In the following the table provides users a relationship between the baud rate and the bus length.

Baud rate (bit/s)	Max. Bus length (m)
1 M	25
800 K	50
500 K	100
250 K	250
125 K	500
50 K	1000
20 K	2500
10 K	5000

Note: When the bus length is greater than 1000m, the bridge or repeater devices may be needed.

2.4 CAN Connector

The pin descriptions of the CAN bus connectors on the CAN-8x23 are shown below.

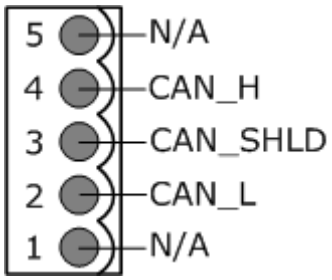
CAN-8123/CAN-8223:



Pin No.	Signal	Description
1	CAN_GND	Ground (0V)
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_SHLD	Optional CAN Shield
4	CAN_H	CAN_H bus line (dominant high)
5	CAN_V+	CAN external positive supply (+10V ~ +30V)

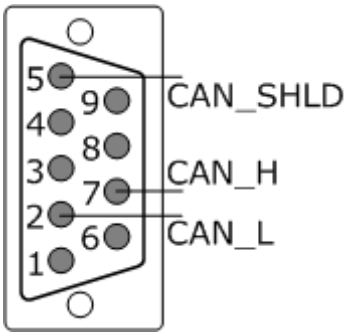
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

CAN-8423:



Pin No.	Signal	Description
1	N/A	N/A
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_SHLD	Optional CAN Shield
4	CAN_H	CAN_H bus line (dominant high)
5	N/A	N/A

CAN-8823:



Pin No.	Signal	Description
2	CAN_L	CAN_L bus line (dominant low)
5	CAN_SHLD	Optional CAN Shield
7	CAN_H	CAN_H bus line (dominant high)
Others	N/A	N/A

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.5 Terminal Resistor Jumper and Initial Switch

The jumpers enable/disable of the terminal resistor show as follow:

Device	Jumper	Enable	Disable
CAN-8123 / CAN-8223	JP2		
CAN-8423	JP1		
CAN-8823	JP1		

Before updating firmware or using the utility tool to configure the CAN-8423 and the CAN-8823, the initial mode is needed. For more detail configuration, please refer to the cheaper 4. Since the CAN-8123/CAN-8223 doesn't support RS-232 COM Port, the utility tool in the off-line mode takes the place to get the EDS file.

Following shows the initial switch of CAN-8423 and CAN-8823 **(CAN-8123/CAN-8223 not support the initial function)**.

Device	Switch	Initial Mode	Run Mode
CAN-8423	Baud Rotary Switch	Switch to "9"	Switch to "0" ~ "7"
CAN-8823	SW1	Switch to "Init"	Switch to "Run"

2.6 Power LED

The CAN series products (CAN-8x23) need 10 to 30 VDC power supplies, (Please note that other slot modules, inserted in, will also consume part of the inputted power). Under a normal connection, a good power supply and a correct voltage selection, as the unit is turned on, the LED will light up in yellow. If it can't work, please check with local agents or resellers for more help. CANopen Status LED

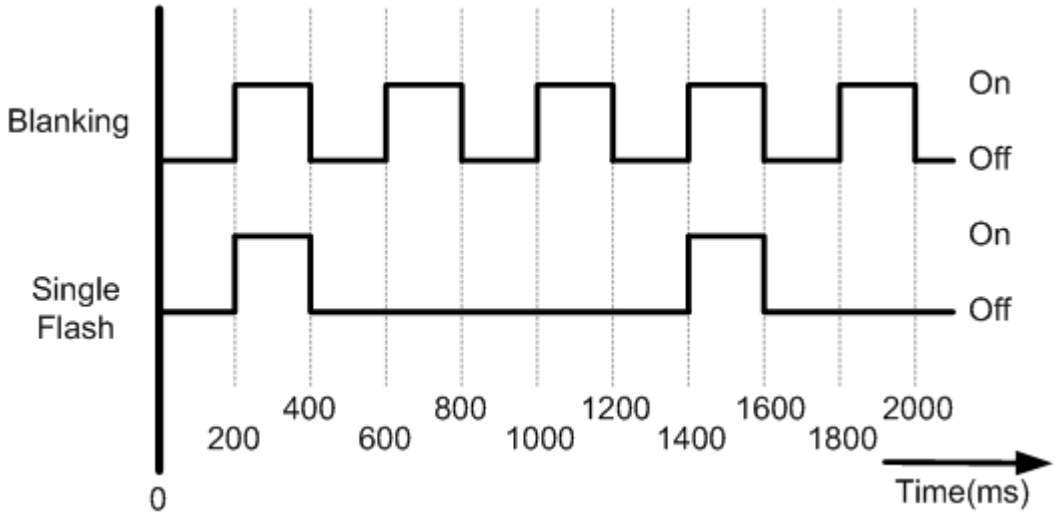
LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.7 CANopen Status LED

Each one CAN-8x23 has two LED indicators. One is the Error LED (lighting in red) and the other one is the RUN (Performing) LED (lighting in green). The Error LED and the Run (Performing) LED information are presented in the CANopen specifications. When the CANopen communication carries out, these indicators will glitter in different time. The following descriptions will show meanings of the glittering signal as these indicators are being triggered.

2.7.1 The RUN LED

The RUN LED relates to the physical mechanism on the CANopen that will be discussed later. The data state and the signal state description are respectively shown in the following figure and table.

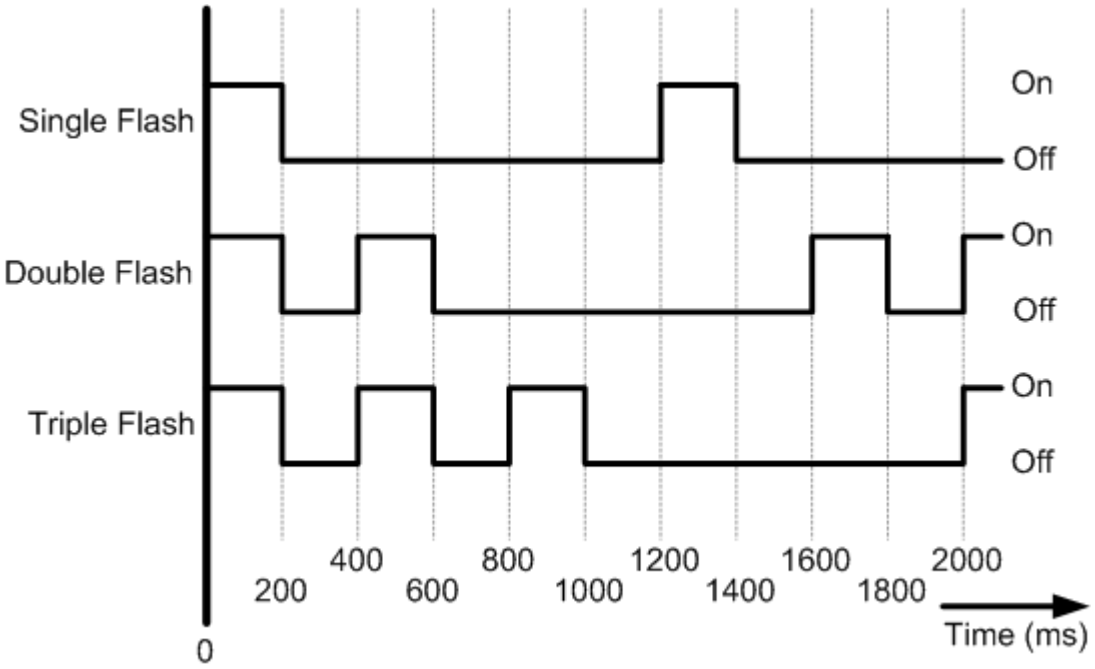


No.	Signal	State	Description
1	No Light	Non-operation	Malfunction or Power Supply /Connection not ready
2	Single Flash	Stopped	The device is in Stopped state
3	Blinking	Pre-operation	The device is in the pre-operational state
4	Continuing Light	Operation	The device is in the operational state
5	Blinking rapidly	Module Error	I/O module is removed when running or CAN-8x23 detects the module different from before. Users can use utility to reset it.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.7.2 The ERR LED

The ERR LED relates to the state of missing messages at the CAN physical layer (These missing messages might be SYNC or Guard messages). The data state and the signal state description are respectively shown in the following figure and table.



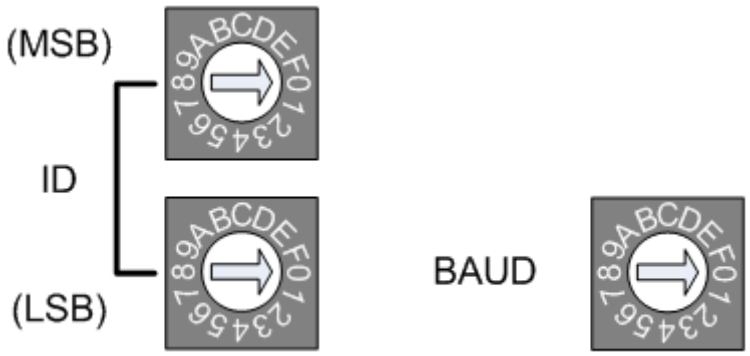
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

No.	Signal	State	Description
1	No Light	No error	The device is in working condition.
2	Single Flash	Error Reminding when Warning Level is Reached	At least one of the error counters of the CAN controller has reached or exceeded the warning level (too many error frames).
3	Double Flash	Error Reminding when Events happen.	A guard event (NMT-Slave or NMT-master) or a heartbeat event (Medical Application) has occurred.
4	Triple Flash	SYNC Error	The SYNC message has not been received within the specific communication cycle before time out (see Object Dictionary Entry 0x1006).
5	Continuing Light	Bus Off	The CAN controller is in a bus off condition.

Note: If several errors occur at the same time, the most severe error will have high priority to show its signal first. For example, if NMT Error (No. =3) and Sync Error (No. =4) occur, the SYNC error signal will indicate.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.8 The Node ID & the Baud rate Rotary Switch



The first two rotary switches (MSB & LSB) control the CAN-8x23 node ID. MSB (Most Significant Bit) means the high nibble of the node ID, and LSB (Least Significant Bit).

ID Rotary Switch	Status
0x01 ~ 0x7F	Normal CANopen ID
0x81 ~ 0xFF	I/O Pair-connection CANopen ID

The last rotary switch (BAUD) handles the CAN-8x23 baud rate. The relationship between the rotary switch value and the practical baud rate is presented in the following table.

Rotary Switch Value	Baud rate (K BPS)
0	10
1	20
2	50
3	125
4	250
5	500
6	800
7	1000
9	Initial Mode (Only for CAN-8423)
Others	N / A

Furthermore, when users apply the CAN-8x23 the CANopen firmware will automatically check these rotary switches. Any illegal value for these rotary switches will cause the boot-up failure.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

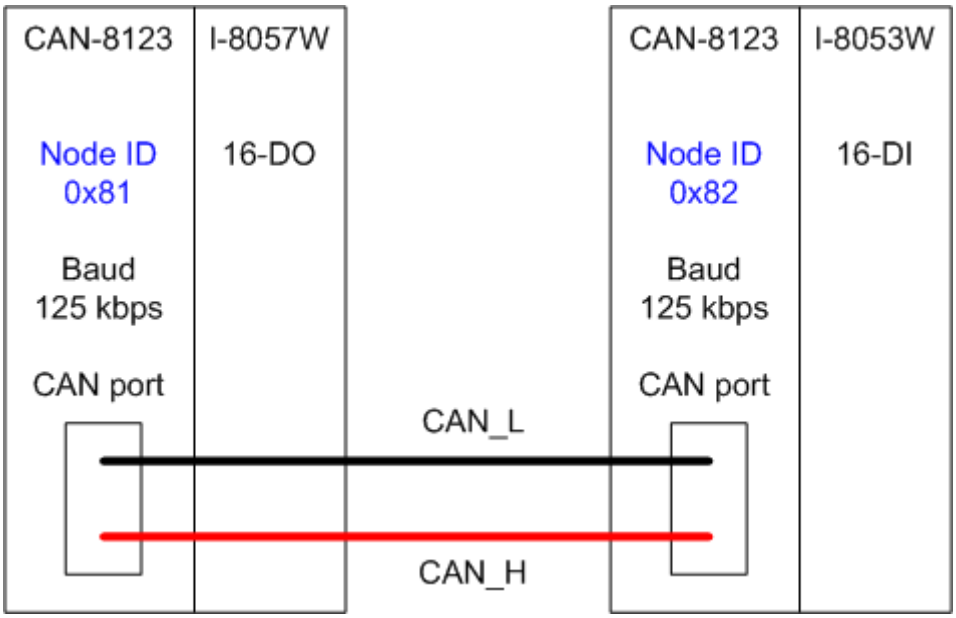
2.9 I/O Pair-connection Mode

The CAN-8x23 provides the I/O pair-connection function. Before using this function, you need to prepare two CAN-8x23s with DI and DO I/O modules (such as I-8057W and I-8053W). When applying this function, the DI channels and the DO channels are mapping with each other. That is to say that when the DI channels of one CAN-8x23 get the ON signal, the corresponding DO channels of the other one will be turned on.

When you completed the connection of these two CAN-8x23s by CAN bus, you need to set the **ID rotary switch** of these two modules to 0x81 ~ 0xFE by the special rule. Set the node ID to be odd for one module, and set the node ID of another module to be the value which is equal to the node ID increased one of the former. Therefore, they are the couple as the following figure.

1st CAN-8x23 Node-ID		2nd CAN-8x23 Node-ID
0x81	← A Couple →	0x82
0x83	← A Couple →	0x84
0x85	← A Couple →	0x86
.....	.....	.....
0xFD	← A Couple →	0xFE

For example, user uses a CAN-8123 with I-8057W and a CAN-8123 with I-8053W to do I/O pair-connection. The connection structure is as follows.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

The node ID of left CAN-8123 is “0x81”, and the node ID of right CAN-8123 is “0x82”. Both of these two module’s node ID switch are selected to I/O pair-connection mode node ID, and these two modules will into Operational state automatically. When the DI module, right CAN-8123, receives a DI ON-signal, the DO module, left CAN-8123, will output the ON-signal at the corresponding DO channels.

2.10 Module Support

The CAN-8x23 supports many kinds of DI, DO, AI and AO types across the I-8000/I-87K series modules. When users want to apply these modules on the CANopen network, they only insert these modules into the CAN-8x23 I/O expansion slots. Then, the CANopen firmware built in the CAN-8x23 will automatically search them, and apply the corresponding CANopen objects. The following table shows the information of the IO types and module names which can be supported by the CAN-8x23.

IO Type	Module Name	IO Type	Module Name
AI	I-8014/I-8014C/I-8017H/ I-8017HC/I-8017HS/I-8017HW I-87005/I-87013/I-87015/ I-87015P/I-87016/I-87017/ I-87017R/I-87017RC/I-87017A5/ I-87018/I-87018R/I-87018Z/ I-87019R	AO	I-8024/I-8026/ I-8024U/I-8028U I-87022/I-87024/I87024R/ I-87026/I-87026P
DO	I-8037/I-8041/I-8041A/I-8041P I-8056/I-8057/I-8057P/ I-8060/I-8064/I-8065/ I-8066/I-8068/I-8069 I-87041/I-87041P/I-87057/ I-87057P/I-87064/I-87065/ I-87061P/I-87066/I-87068 I-87068-2A/I-87069/I-87069P	DI	I-8040/I-8040P/I-8040A1/ I-8046/I-8048/I-8051/ I-8052/I-8053/I-8053P/ I8053A1/I-8058/ I-87040/I-87040P/I-87046/ I-87051/ I-87052/I-87053/ I-87053A2/I-87053P/ I-87053A5/I-87053E5/ I-87058/I-87059
DO&DI	I-8042/I-8050/I-8054/I-8055/ I-8063/I-8077 I-87042/I-87054/I-87055/I-87063	Counter Frequency	I-8080/I-8084/I-8088

Note: All modules are supported with “High profile” (W) and “Low profile”. But only the “High profile” I-87K modules support hot-swap function.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3 CANopen Application

The CANopen is a kind of network protocols evolving from the CAN bus, used on car control system in early days, and has been greatly used in various applications, such as vehicles, industrial machines, building automation, medical devices, maritime applications, restaurant appliances, laboratory equipment & research.

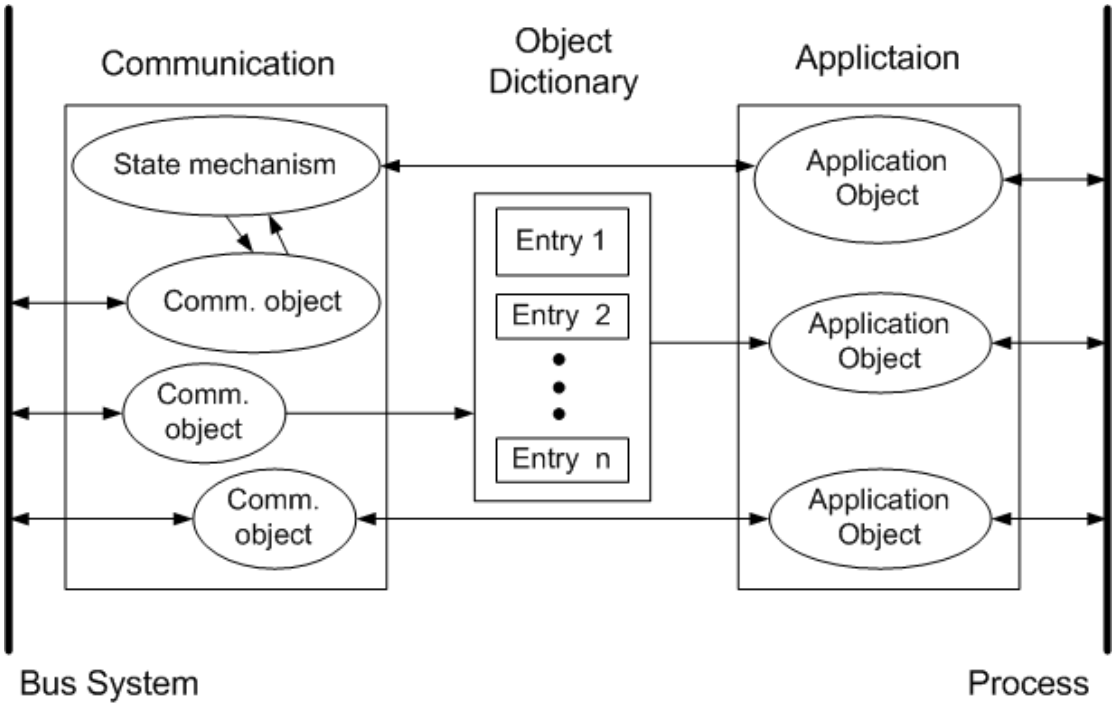
3.1 CANopen Introduction

CANopen provides not only the broadcasting function but also the peer-to-peer data exchange function between every CANopen node. The network management function instructed in the CANopen simplifies the program design. In addition, users can also implement and diagnose the CANopen network, including network start-up, and error management by standard mechanisms (CANopen device), i.e. the CANopen device can effectively access the I/O values and detect node states of other devices in the same network. Generally, a CANopen device can be modeled into three parts.

- Communication
- Object Dictionary
- Application program

The functions and general concepts for each part are shown as follows.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Communication

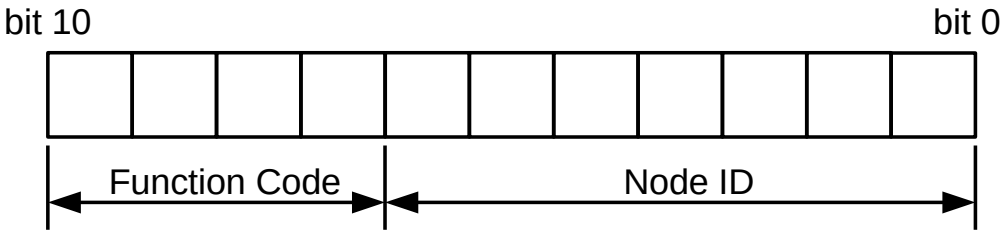
The communication part provides several communication objects and appropriate functionalities to transmit CANopen messages via the network structure. These objects include PDO (Process Data Object), SDO (Service Data Object), NMT (Network Management Objects), SYNC (Synchronous Objects)…etc. Each communication object has its relative communication model and functionality. For example, the communication objects for accessing the device object dictionary is SDO, using the Client/Server structure as its communication model (section 3.2). Real-time data or I/O values can be accessed quickly without any protocol by means of PDO communication objects. The PDO's communication model follows the Producer/Consumer structure. It is also named the Push/Pull model (section 3.3). NMT communication objects are used for controlling and supervising the state of the nodes in the CANopen network, and it follows a Master/Slave structure (section 3.5). No matter which kind of communication object is used, the transmitted message will comply with the data frame defined in the CAN 2.0A spec. Generally, it looks like the following figure.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ID	RTR	Data Length	8-byte Data

11-bit data is limited in the ID field. It is useful in the arbitration mechanism. The RTR, limited in 1-bit data, is used for remote-transmitting requests as the value is set to 1. The data length, limited in 4-bit data, shows the valid data number stored in the 8-byte data field. The last field, 8-byte data, is applied to store the message data.

In the CANopen specifications the 4-bit function code and 7-bit node ID are assumed to combine the 11-bit ID of CAN message, and named the communication object ID (COB-ID). The COB-ID structure is displayed below.



The COB-IDs are used for recognizing where the message comes from or where the message is sent to, as well deciding the priority of the message transmission around node network. According to the arbitration mechanism rule of the CAN bus, the CAN message with the lower COB-ID will get the higher priority to be transmitted. In the CANopen specifications some COB-IDs are reversed for specific communication objects, and can't be defined arbitrarily by users. The following list shows these reversed COB-IDs.

Reversed COB-ID (Hex)	Used by object
0	NMT
1	Reserved
80	SYNC
81~FF	EMERGENCY
100	TIME STAMP
101~180	reversed
581~5FF	Default Transmit-SDO
601~67F	Default Receive-SDO
6E0	reversed
701~77F	NMT Error Control
780~7FF	reversed

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

In addition, the other COB-IDs shown in the following table can be used if necessary.

(Bit10~Bit7) (Function Code)	(Bit6~Bit0)	Communication object Name
0000	0000000	NMT
0001	0000000	SYNC
0010	0000000	TIME STAMP
0001	Node ID	EMERGENCY
0011/0101/0111/1001	Node ID	TxPDO1/2/3/4
0100/0110/1000/1010	Node ID	RxPDO1/2/3/4
1011	Node ID	SDO for transmission (TxSDO)
1100	Node ID	SDO for reception (RxSDO)
1110	Node ID	NMT Error Control

Note: For the CAN-8x23, all communication objects are supported except the TIME STAMP.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

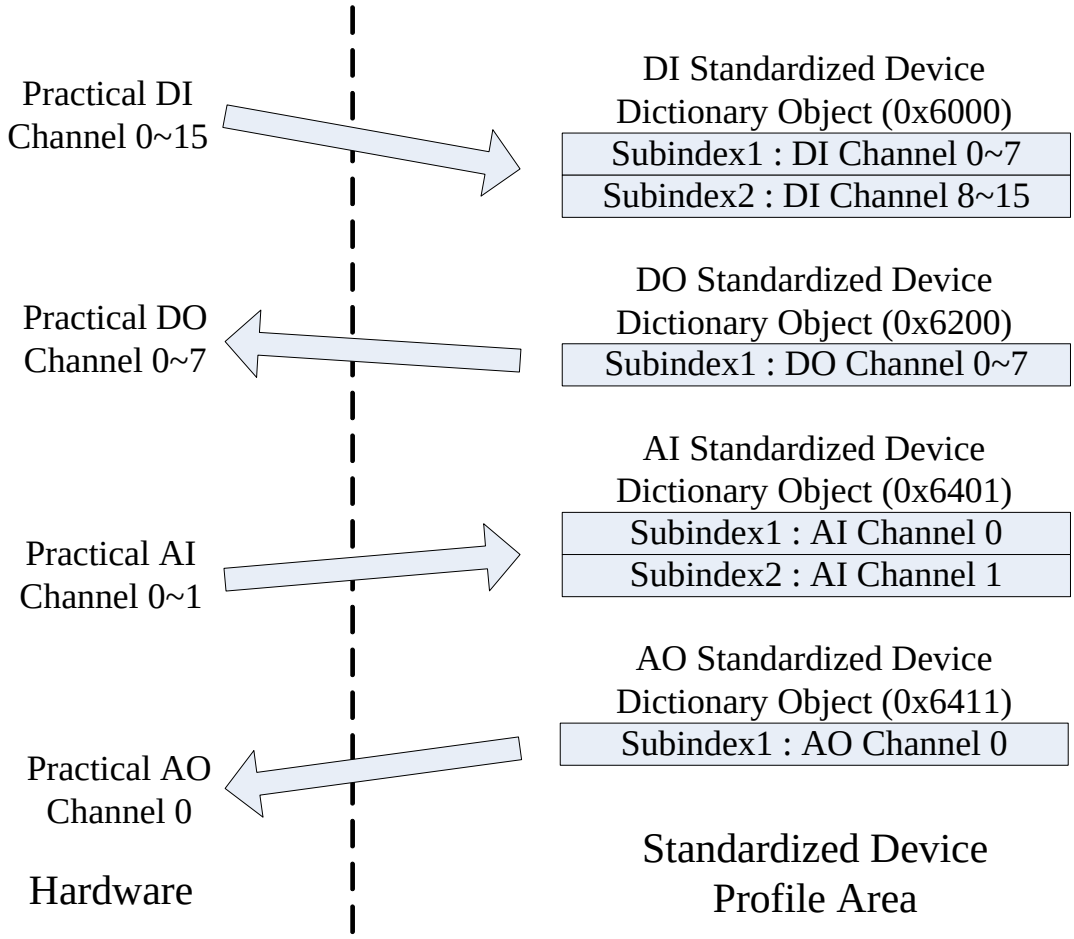
Object Dictionary

The object dictionary collects a lot of important information which can affect device's reaction, such as the data accessing through I/O channels, the communication values and the network states. Essentially, the object dictionary consists of a group of entry objects, and these entries can be accessed via the node network in a pre-defined method. Each object entry within the object dictionary has its own function, for example communication parameters, device profile, data type (ex. 8-bit Integer, 8-bit unsigned…), and access type (read only, write only …). All of them are addressed in a 16-bit index and an 8-bit sub-index. The overall profile of the standard object dictionary is shown below.

Index	Object
0x0000	Reserved
0x0001 - 0x001F	Static Data Types
0x0020 - 0x003F	Complex Data Types
0x0040 - 0x005F	Manufacturer Specific Complex Data Types
0x0060 - 0x007F	Device Profile Specific Static Data Types
0x0080 - 0x009F	Device Profile Specific Complex Data Types
0x00A0 - 0x0FFF	Reserved for further use
0x1000 - 0x1FFF	Communication Profile Area
0x2000 - 0x5FFF	Manufacturer Specific Profile Area
0x6000 - 0x9FFF	Standardized Device Profile Area
0xA000 - 0xBFFF	Standardized Interface Profile Area
0xC000 - 0xFFFF	Reserved for further use

Take the standardized device profile area as an example. Assume that a CANopen device has 16 DI, 8 DO, 2 AI and 1 AO channels. The values of these channels will be stored in the Standardized Device Profile Area, especially the entries with indexes 0x6000, 0x6200, 0x6401, and 0x6411. When the CANopen device obtains the input value, these values will be stored in the 0x6000 and 0x6401 indexes. Furthermore, the values stored in the 0x6200 and 0x6411 indexes will also output to the DO and AO channels. The basic concept is presented as follows.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Take the CAN-8423 as an example. There are some I-8000 or I-87K series modules inserted in the CAN-8423 I/O expansion slots. The related information for each module is shown below.

Module Name	Slot No	DO (ch)	AO (ch)	DI (ch)	AI (ch)
I-8063	0	4	0	4	0
I-87053	1	0	0	16	0
I-8053	3	0	0	16	0

When the CAN-8423 is powered on, all device modules inserted in the CAN-8423 channels will be scanned, as well the I/O values of these channels will be arranged into proper object entries one by one with the minimum data size 1 byte, If the DI and DO channels, which can't reach one byte, will be automatically regarded as one byte.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

By objects with the index 0x6000, the CAN-8423 can store the input values of DI channel, i.e. the I/O values of DO, AI, and AO channels are put into the object with the indexes 0x6200, 0x6401, and 0x6411 respectively. When values are resulted through these I/O, and correspond to the specific object, the device will follow the rules below.

- The I/O channel values of the I-8000/I-87K series modules with lower slot numbers will have priority to be placed into the object dictionary. After the CAN-8423 has filled the all I/O channels in one module, then the CAN-8423 will go to the next slot number to continue.
- Each analog channel will be stored in 2 bytes.
- The values of digital channels of one module, which can't be divided by 8, will be stored in 1 byte.
- After using the rule described above, the result of the object format is as follows.

Index / sub-index	0x6000 (for DI)	0x6200 (for DO)	0x6401 (for AI)	0x6411 (for AO)
0x00	9	1	9	4
0x01	DI0~DI3 (Slot:0)	DO0~DO3 (Slot:0)		
0x02	DI0~DI7 (Slot:1)			
0x03	DI8~DI15 (Slot:1)			
0x04	DI0~DI7 (Slot:3)			
0x05	DI8~DI15 (Slot:3)			

The information described above can also be viewed by using the CAN Slave Utility. For more details about the object dictionary and how to use the CAN Slave Utility, please refer to the chapter 5 and chapter 6.

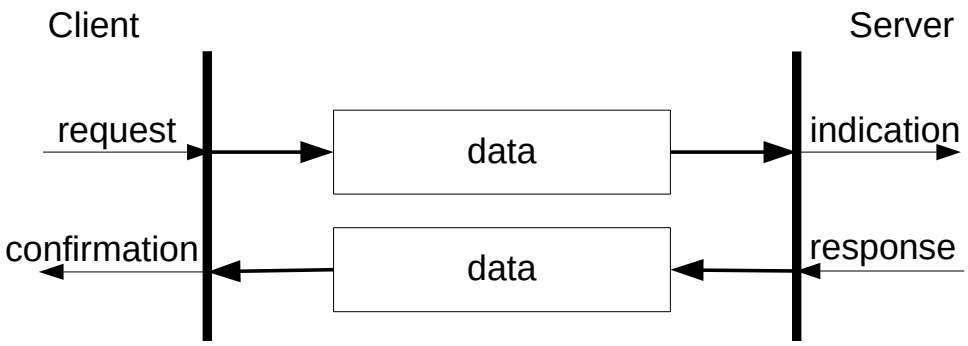
Application

The application objects control all of the device functions, related to the interaction with the process environment. It's just like a medium between the object dictionary and practical process, such as the analog I/O, digital I/O….

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2 SDO Introduction

In order to access the entries in a device object dictionary, service data objects (SDOs) are provided. By means of the SDO communication method, a peer-to-peer communication bridge between two devices is established, and its transmission follows the client-server relationship. The general concept is shown in the figure below.



The SDO has two kinds of the COB-IDs, RxSDOs and TxSDOs. They can be viewed in the CANopen device. For example, users send a SDO message to the CAN-8x23 by using RxSDO. On the contrary, the devices CAN-8x23 transmit a SDO message by using TxSDOS.

Before the SDO has been used, only the client can take the active requirement for a SDO transmission. When the SDO client starts to transmit a SDO, it is necessary to choose a proper protocol.

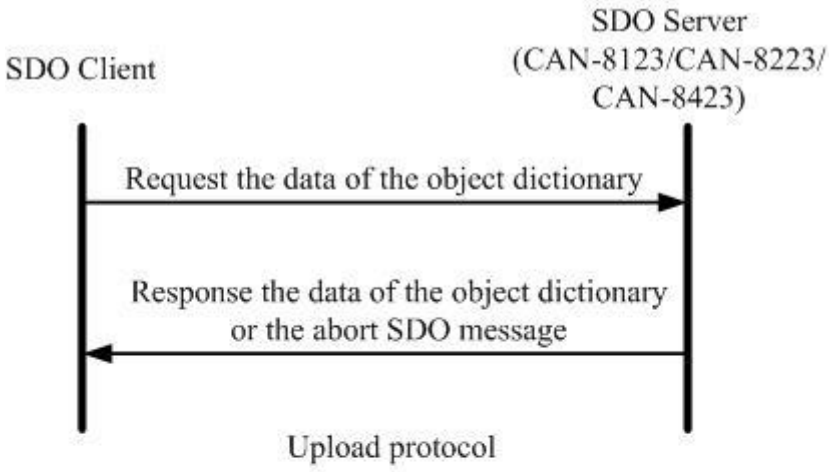
If the SDO client has to get the information from the device object dictionary and from the SDO server, the segment upload protocol or block upload protocol will be applied.

It is worth to be mentioned, the front protocol is used for transmitting fewer data; the latter protocol is used for transmitting larger data. Both the segment download protocol and block download protocol will work when the SDO client wants to modify the object dictionary to the SDO server. The differences between the segment download protocol and the block download protocol are similar to the differences between the segment upload protocol and the block upload protocol. Because of the different access types in the object dictionary, not all accessing action of the object dictionary via the SDO transmission is allowed. If the SDO client trends to modify the entries of the SDO server object dictionary which uses the read-only access type, the abort SDO transfer

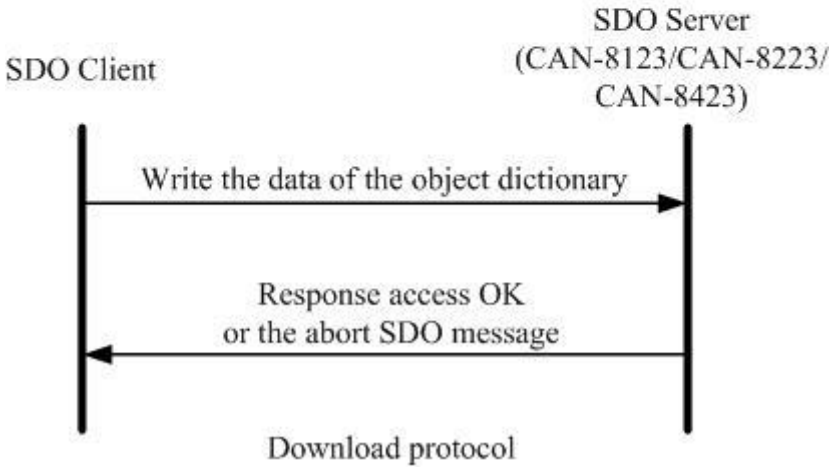
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

protocol will be given, and the SDO transmission will also be stopped.

The CAN-8x23 only supports the SDO server. Therefore, it can be passive and wait for requests from clients. The general concept figure of the upload and download protocol with the CAN-8x23 is shown as follows.



Upload protocol



Download protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

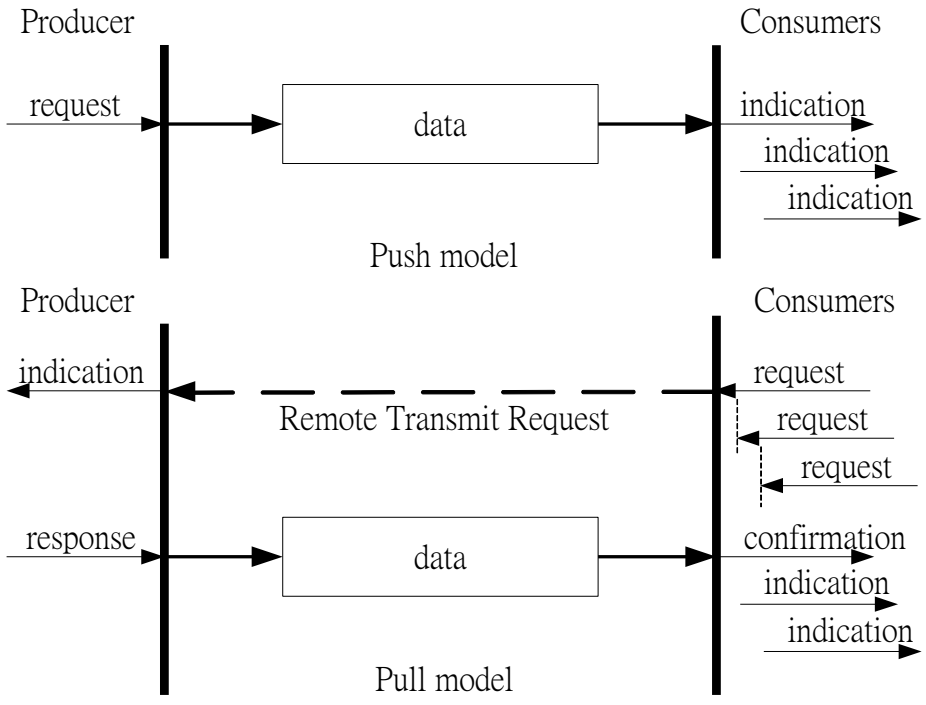
3.3 PDO Introduction

Based on the transmission data format of the CAN bus, the PDO can transmit eight bytes of process data at one time. Because of the PDO messages without overheads, it is more efficient than other communication objects within CANopen and therefore used for real-time data transfer, such as DI, DO, AI, AO, etc.

Communication Modes for the PDO

PDO reception or transmission is implemented via the producer/consumer communication model (also called the push/pull model). When starting to communicate in the PDO push mode, it needs one CANopen device to play the role of PDO producer, and non device or more than one device to play the role of PDO consumer.

The PDO producer sends out the PDO message after it reached the CAN bus arbitration. Afterwards, each PDO consumer will receive this PDO message respectively, and then message is processed by each device to check whether it is needed or not (be dropped). In the PDO pull mode, one of the PDO consumers needs to send out a remote transmit request to the PDO producer. According to this remote request message, the PDO producer responds the corresponding PDO message for each PDO consumer in the CAN bus. The PDO communication structure figure is shown below.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

For the CANopen device, the TxPDO specializes in transmitting data, and is usually applied on DI/AI channels. The COB-ID of the PDO for receiving data is RxPDO COB-ID, and it is usually applied on DO/AO channels. Take the CAN-8x23 as an example. If a PDO producer sends a PDO message to the CAN-8x23, it needs to use the RxPDO COB-ID of the CAN-8x23 because it is a PDO reception action viewed from the CAN-8x23. Inversely, when some PDO consumers send remote transmit requests to the CAN-8x23, it must use the TxPDO COB-ID of the CAN-8x23 because it is a PDO transmission action viewed from the CAN-8x23.

Trigger Modes Of PDO

For PDO producers, PDO transmission messages can be trigged by three conditions. They are the event driven, timer driven and remote request conditions. All of them are described below.

Event Driven

PDO transmission can be triggered by a specific driven event, including the following conditions. Under the cyclic synchronous transmission type, the event is driven by the expiration of the specified transmission period, synchronized by the reception of the SYNC message.

Moreover, under the acyclic synchronous or asynchronous transmission type, the PDO transmission can also be triggered or driven by a device-specified event in the CANopen specification DS-401 v2.1, i.e. by following this spec, the PDO will be triggered by any change in the DI-channel states when the transmission type of this PDO is set to acyclic synchronous or asynchronous.

Timer Driven

PDO transmissions are also triggered by a specific time event, even if a specified time elapsed without occurrence of an event. For example, the PDO transmission of the CAN-8x23 can be triggered by the event timer of the PDO communication parameters, which is set by users.

Remote Request

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

The PDO transmission can be triggered by receiving a remote request from any other PDO consumer with under the asynchronous or RTR setting.

PDO Transmission Types

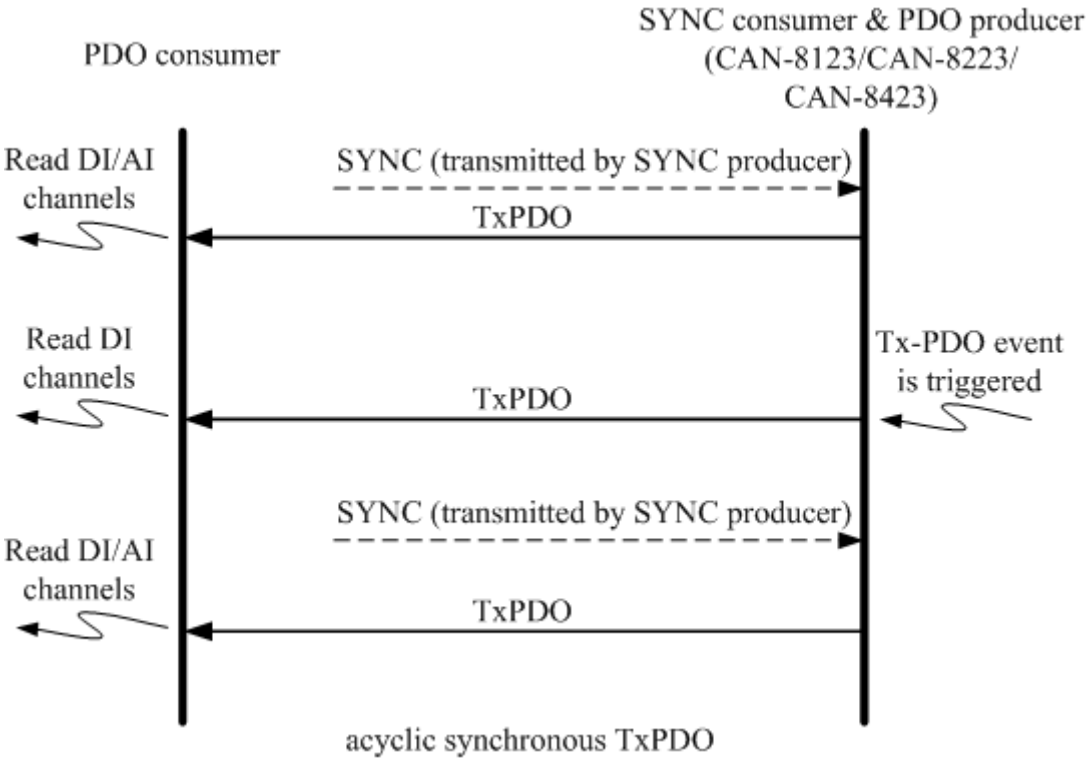
Generally there are two kinds of PDO transmission modes, synchronous and asynchronous. For the PDO in a synchronous mode, it must be triggered by the reception of a SYNC message.

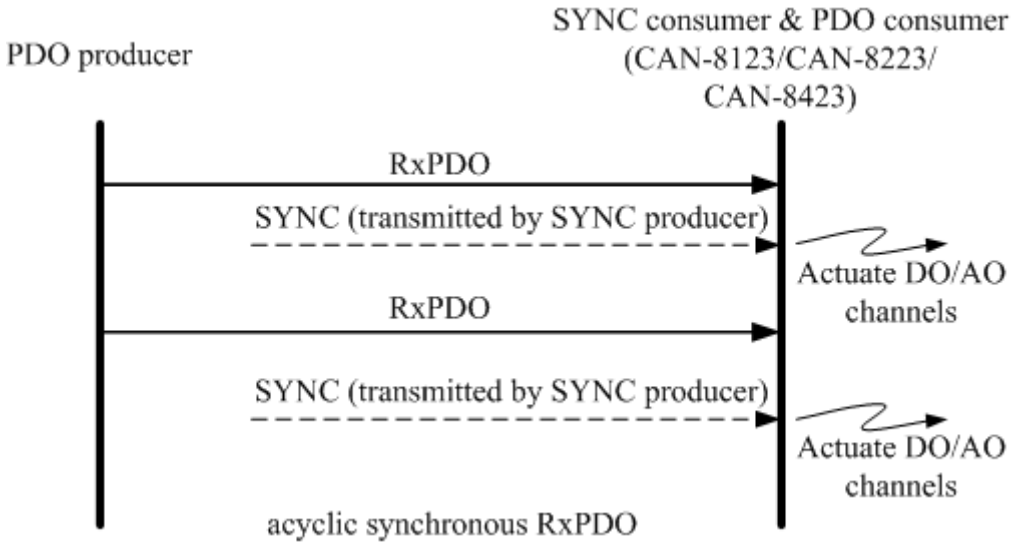
The synchronous mode can be further distinguished into three kinds of transmission(s), acyclic synchronous, cyclic synchronous and RTR-only synchronous. The acyclic synchronous can be triggered by both the reception of a SYNC message and the driven event mentioned above.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Acyclic synchronous

For the TxPDO object, after receiving an object from the SYNC producer, the CAN-8x23 will respond with a pre-defined TxPDO message to the PDO consumers. For the RxPDO object, the CAN-8x23 needs to receive the SYNC objects to actuate the RxPDO object, which is received before the SYNC object. The following figures indicate how the acyclic synchronous transmission type works on the RxPDO and the TxPDO.

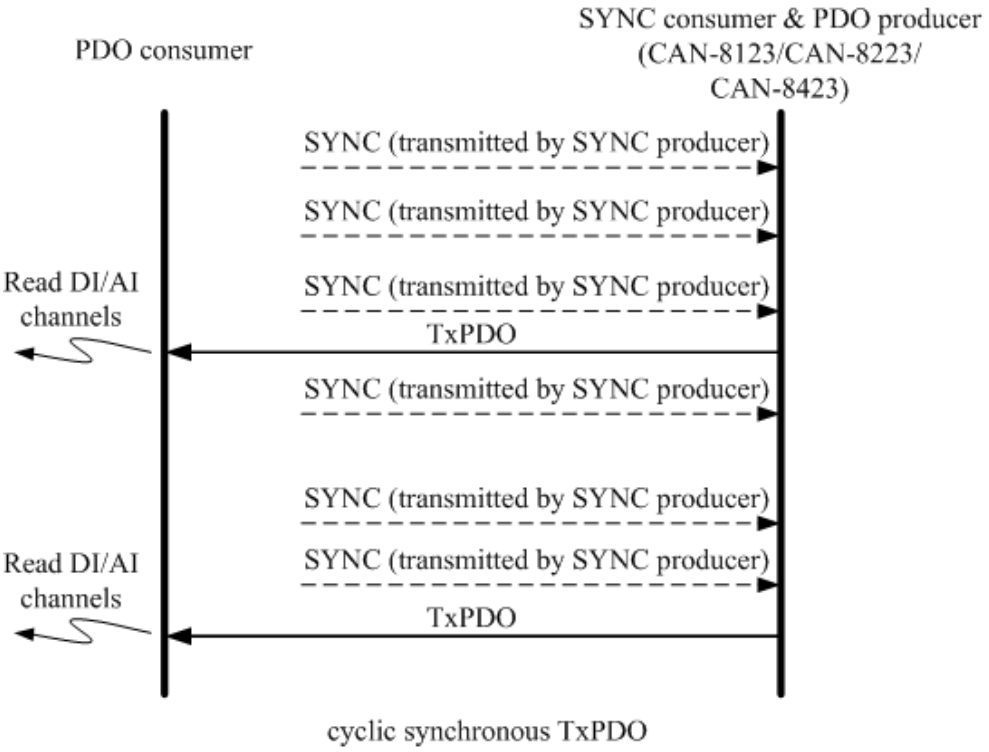




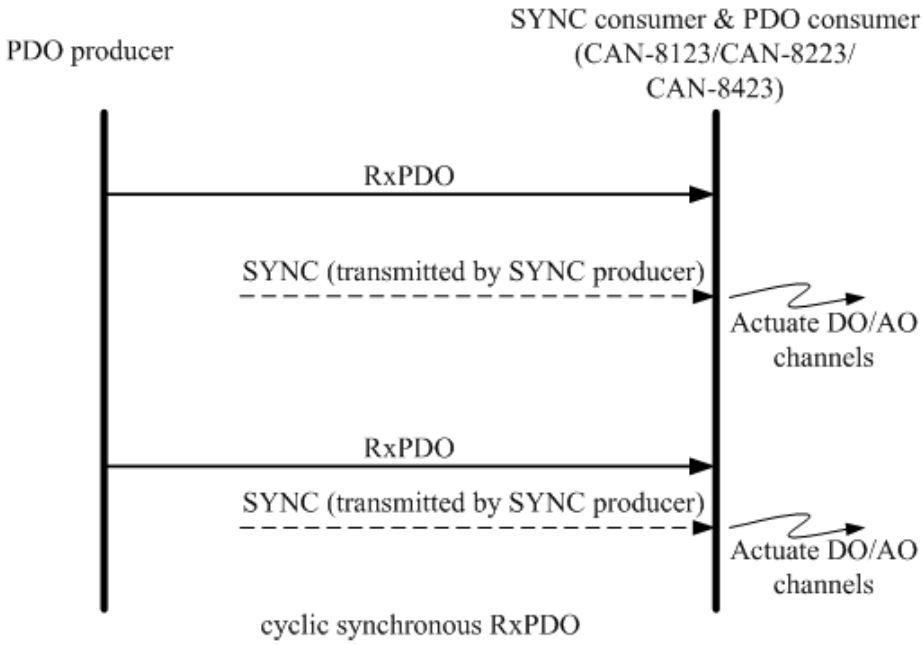
Cyclic synchronous

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Inversely, the cyclic synchronous transmission mode is triggered by the reception of an expected number of SYNC objects, and the max number of expected SYNC objects can be 240. For example, if the TxPDO is set to response when receiving 3 SYNC objects, the CAN-8x23 will feed back the TxPDO object according to the set. For the RxPDO, actuating the DO/AO channels by the RxPDO is independent of the number of SYNC objects. These concepts are shown in the figures below.



cyclic synchronous TxPDO

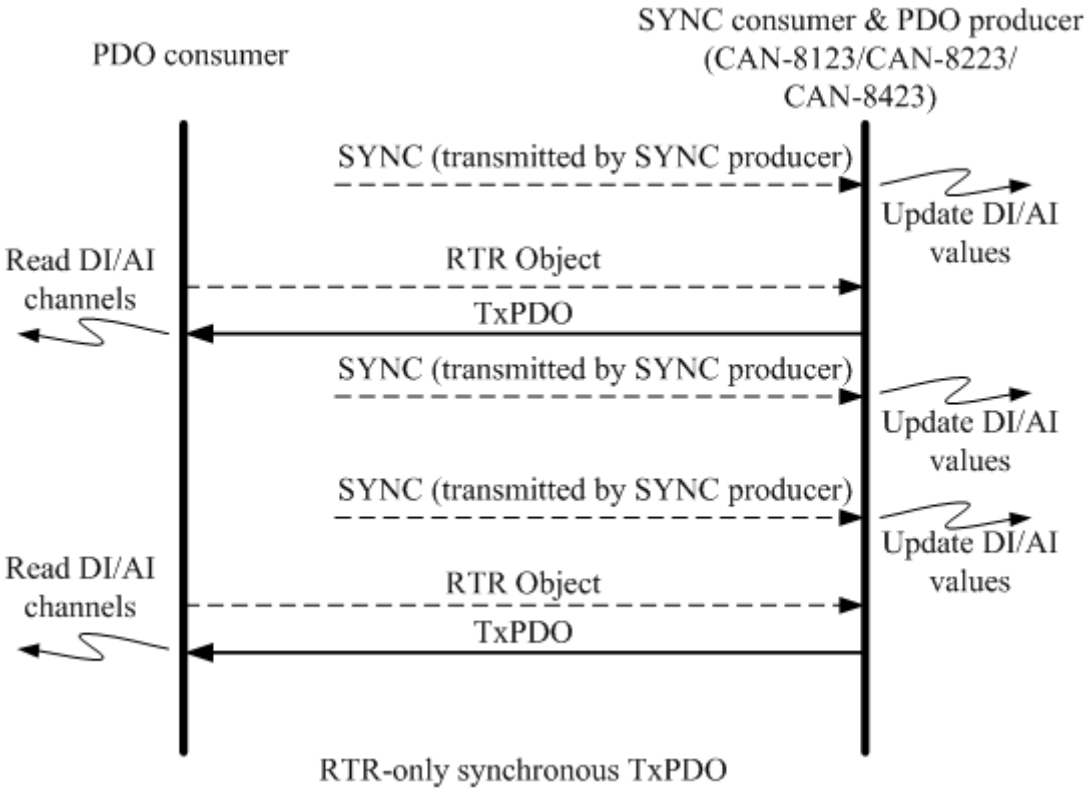


cyclic synchronous RxPDO

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

RTR-only synchronous

The RTR-only synchronous mode is activated when receiving a remote-transmit-request message, i.e. SYNC objects. This transmission type is only useful for TxPDO. In this situation, the CAN-8x23 will update the DI/AI value when receiving any SYNC object. And, if the RTR object is received, the CAN-8x23 will respond to the TxPDO object. The following figure shows the mechanism of this transmission type.

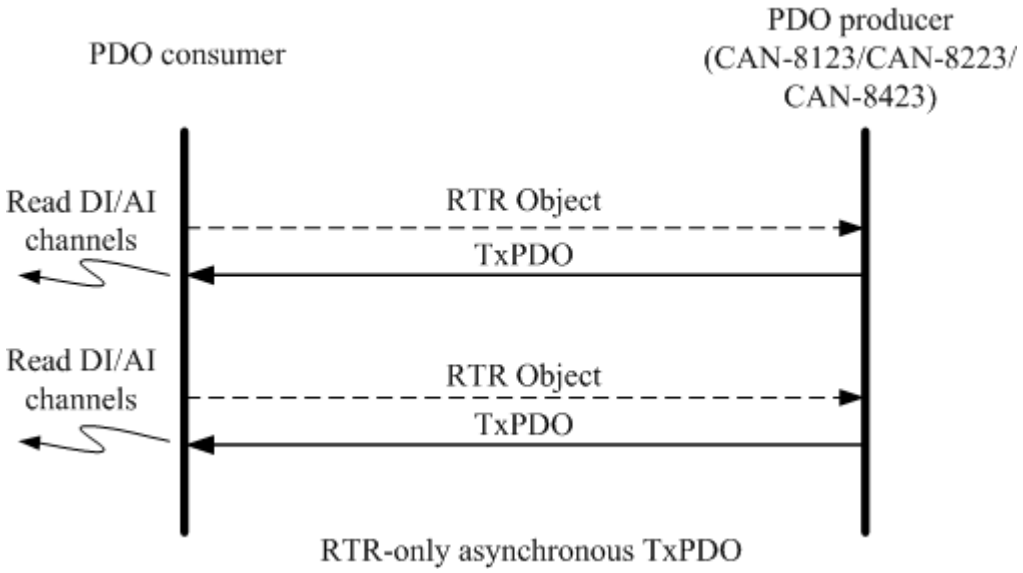


RTR-only synchronous TxPDO

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

RTR-only asynchronous

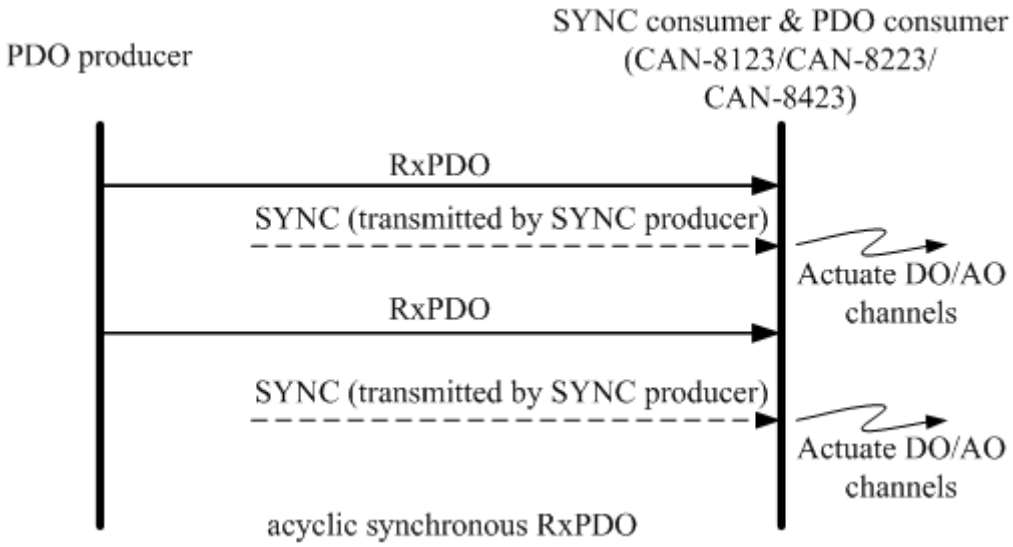
The asynchronous mode is independent of the SYNC object. This mode can also be divided into two parts. There are RTR-only asynchronous transmission type and asynchronous transmission type. The RTR-only transmission type is only for supporting TxPDO transmissions, only triggered by receiving the RTR object from the PDO consumer. This action is depicted below.

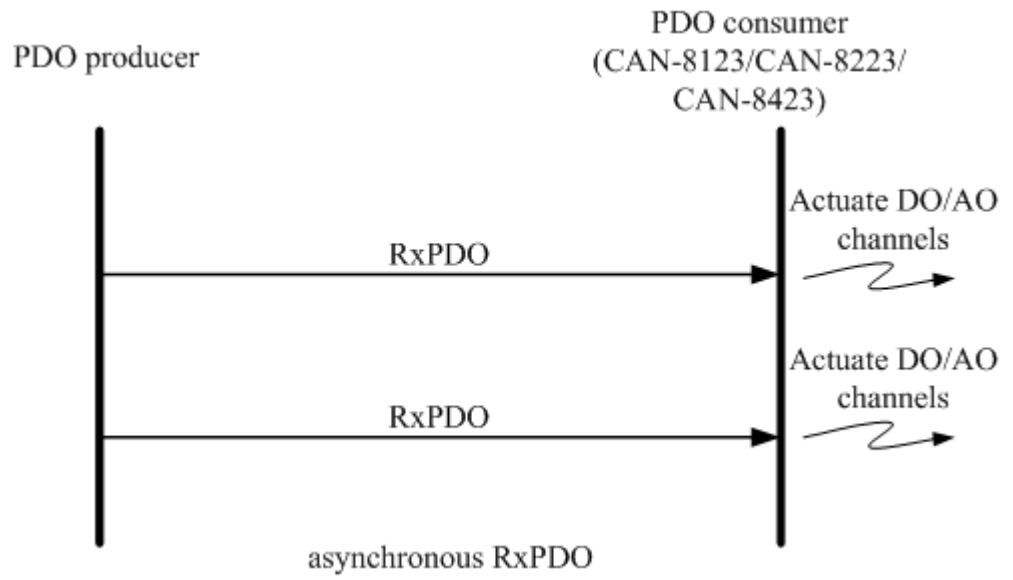


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Asynchronous

The other part is the asynchronous transmission type. Under this type, the TxPDO message can be triggered by receiving the RTR object and the device-specified event mentioned in the event driven paragraph. Furthermore, the DO/AO channels can act directly by receiving the RxPDO object. This transmission type is the default value when the CAN-8x23 boots up. The concept of the asynchronous type is shown as follows.





Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Inhibit Time

Because of the arbitration mechanism of the CAN bus, the CANopen communication object ID in small size has a higher transmission priority than the bigger one. For example, there are two nodes on the CAN bus, the one needs to transmit the CAN message with the COB-ID 0x181, and the other has to transmit the message with COB-ID 0x182. When these two nodes transmit the CAN message to the CAN bus simultaneously, only the message containing COB-ID 0x181 can be successfully sent to the CAN bus because of the higher transmission priority. So the message with COB-ID 0x182 will be held to transmit until the message with COB-ID 0x181 is successfully transmitted. This arbitration mechanism can guarantee the successful transmission for one node when a transmission conflict occurs.

However, if the message with COB-ID 0x181 is continually transmitted, the message with COB-ID 0x182 will be postponed to be transmitted. In order to avoid the occupation of the transmission privilege by the message with the lower COB-ID, the inhibit time parameters for each of the PDO objects are supported to define a minimum time interval between each PDO message transmission, which has a multiple of 100us. During this time interval, the PDO message will be inhibited from transmission.

Event Timer

This parameter setting on the event timer is only used for TxPDO. If the parameter of the event timer is not equal to 0 under the transmission type in asynchronous mode, the expiration of this time value can be just considered to be an event. This event will cause the TxPDO message transmission. The event timer parameter is defined as a multiple of 1ms.

PDO Mapping Objects

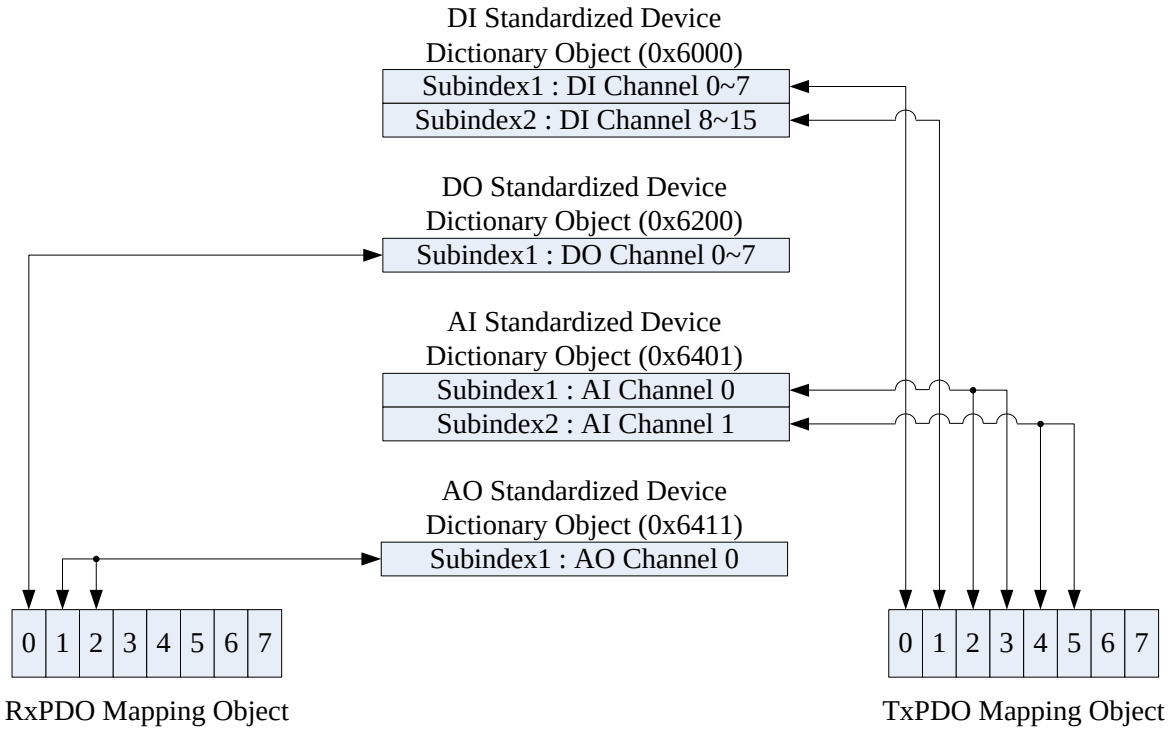
The PDO mapping objects are provided to the interface which is for PDO messages and real I/O data in the CANopen device. They define the meanings for each byte in the PDO message, and may be changed by using a SDO message. All of the PDO mapping objects are arranged in the Communication Profile Area. In the CANopen spec (see DS 401), RxPDO and TxPDO default mapping objects will specify something as follows:

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- There shall be up to 4 TxPDO mapping objects and up to 4 RxPDO mapping objects with default mappings.
- The 1st RxPDO and TxPDO mapping objects are used for digital outputs and inputs to each other.
- The 2nd, 3rd, and 4th RxPDO and TxPDO mapping objects are respectively assigned to record the value of analog outputs and inputs.
- If a device supports too many digital input or output channels over 8 channels, the related analog default PDO mapping objects remaining the additional unused digital I/Os will use its additional objects. This rule with the same concept is used on the additional analog channels. Take the RxPDO as an example; there are 11 DO and 13 AO object entries in the object dictionary. Generally in the CAN-8x23, the first 8 DO object entries will be mapped to the first RxPDO mapping object because one DO object entry needs one byte space. The last 3 DO object entries will be assigned to the 5th RxPDO according to the above rules the 2nd and the 3rd. Furthermore, one AO object entry needs 2 bytes of space. Therefore, the second RxPDO mapping object has been occupied by the first 4 AO object entries. The following 4 AO object entries will be assigned to the third RxPDO mapping object, as well to the 4th RxPDO mapping object. Because the 5th RxPDO mapping object has been occupied by the DO object entries, the last AO object entry shall be assigned into the 6th RxPDO mapping object.

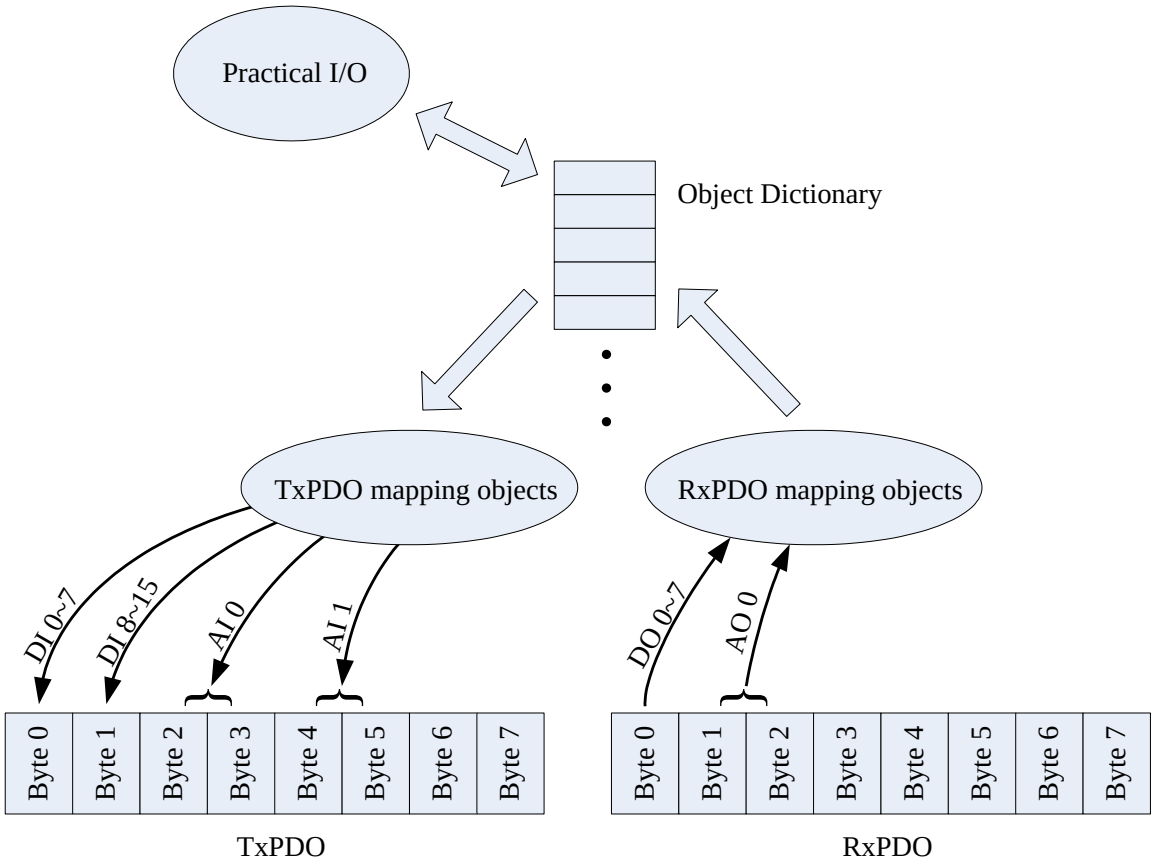
Before applying the PDO communications, the PDO producer and the PDO consumers must have mutual PDO mapping information. On the one hand, the PDO producers need PDO mapping information to decide how to assign the expected practical I/O data to PDO messages. Besides, PDO consumers need the PDO mapping information to recognize each byte of received PDO message, i.e. when a PDO producer transmits a PDO object to PDO consumers, the consumers will contrast this PDO message with PDO mapping entries, previously obtained from the PDO producer, and then interpret the meanings of these values from the received PDO object. For example, if a CANopen device has 16 DI, 8 DO, 2 AI, and 1 AO channels. The input or output values of these channels will be mutually stored into several specific entries.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



According to the PDO mapping objects in the figure above, if this CANopen device gets the RxPDO message in three bytes, the first byte is for the output value from the DO channels 0~7, and the following two bytes are for the analog output value. After interpreting the data of the RxPDO message, the device will actuate the DO and AO channels by the received RxPDO message. It is worth to mention that TxPDO also operate in the same procedure as RxPDO message. When the TxPDO trigger events occur, the CANopen device will send the TxPDO message to the PDO consumers. The values of the bytes assigned in the TxPDO message will follow the TxPDO mapping object as shown in the above figure. The first two bytes of the TxPDO message are respectively for the values from the DI channels 0~7 and channel 8~15. The following third and forth bytes of the TxPDO message is for the value 0 of the AI channel. And the fifth and sixth bytes are for the value 1 of the AI channel. The relationships among the object dictionary, the PDO mapping object and the PDO message are given below.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



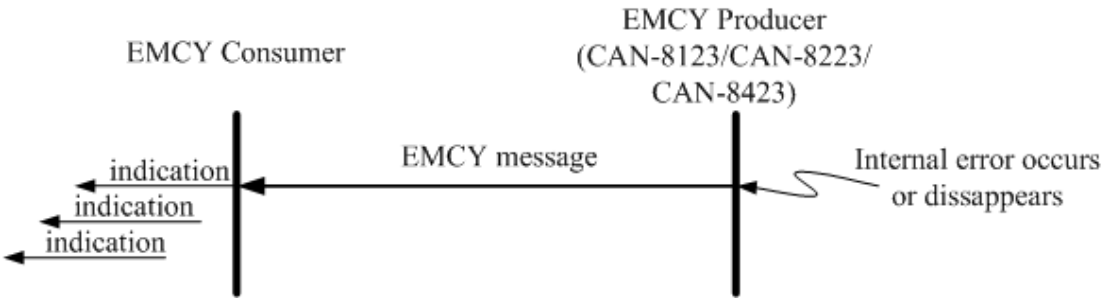
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.4 EMCY Introduction

EMCY messages are triggered when a device internal error occurs, i.e. after a CANopen device detects the internal error, an emergency message will be transmitted to the EMCY consumers per time per error event. But the EMCY message will not be transmitted again if the same error repeatedly occurs. When error reasons are gone, an emergency message containing the emergency error code “00 00” will only respond to the specific error fields. So, by checking the EMCY message, users can understand what happened in the CAN-8x23, and then do something about the error event.

Please note that only the emergency consumers can receive the EMCY object, and only the CAN-8x23 can support functions of the emergency producer.

The general concept regarding EMCY communications is shown below.



An emergency message containing 8-byte of data is called emergency object data. The abbreviated diagram is shown below, and all fields in the emergency object data will be described in section 5.3.

Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code		Error register	Manufacturer specific Error Field				

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.5 NMT Introduction

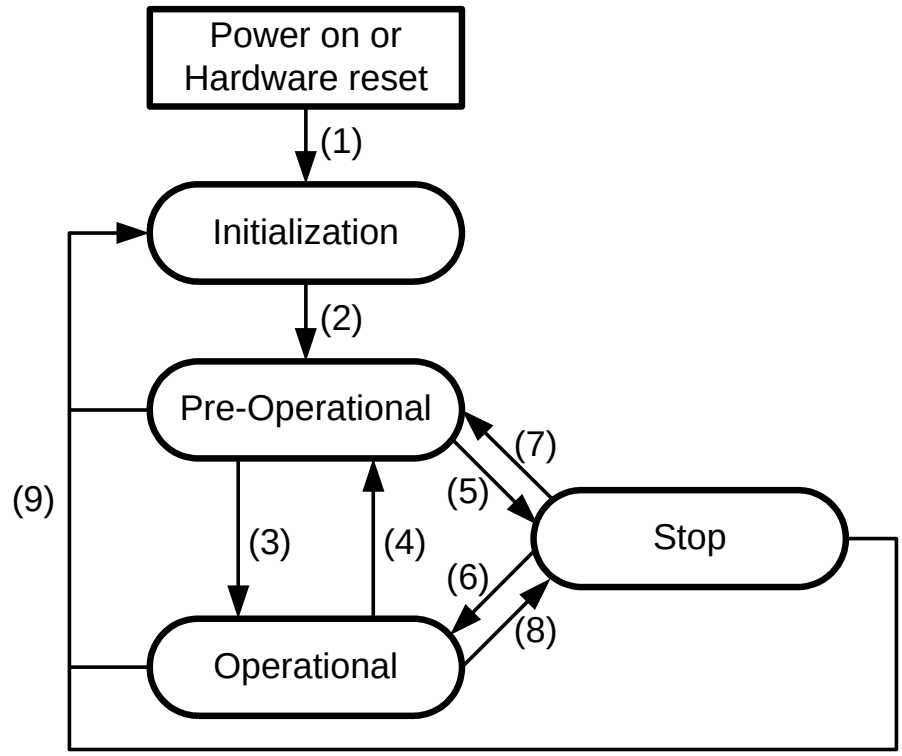
The Network Management (NMT) follows the node-oriented structure and the master-slave relationship. In the same CAN bus network, only one CANopen device is allowed to execute the function of NMT master. Each CANopen node is regarded as a unique NMT slave identified by its node ID from 1 to 127.

The NMT service supplies two protocols, the module control protocol and the error control protocol. Through the module control protocol, the nodes can be controlled to several kinds of status, such as installing, pre-operational, operational, and stopped. According to the NMT slave can present in different statuses, it has different privileges to carry out the communication protocol. Through the error control protocol, users are enable to detect the remote error in the network in order to confirm whether the node still works or not.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.5.1 Module Control Protocols

Before introducing the modules control protocols, the architecture of the NMT state mechanism needs to be mentioned. The diagram shows the process and the relationships among each NMT state and the mechanism.



State Mechanism Diagram

(1)	Under “Power on” or “Hardware Reset”, the initialization state will be loaded automatically.
(2)	As the Initialization accomplished, Pre-Operational state will be entered automatically
(3),(6)	Indication of starting remote node
(4),(7)	Indication of entering Pre-Optional State
(5),(8)	Indication of stopping remote node
(9)	Indication of the “Reset Node” or the “Reset Communication”

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Devices will directly lead to the Pre-Operational state after finishing the device initialization. Then, the nodes will be switched into different state by receiving a specific indication. By the way, each different NMT state will consider a specific communication method. For example, the PDO message can only do the transmission and receiving in the operational state. In the following table, the relationship among each NMT state and communication objects is given.

	Installing	Pre-operational	Operational	Stopped
PDO			**O**	
SDO		**O**	**O**	
SYNC Object		**O**	**O**	
Time Stamp Object		**O**	**O**	
EMCY Object		**O**	**O**	
Boot-Up Object	**O**			
NMT		**O**	**O**	**O**

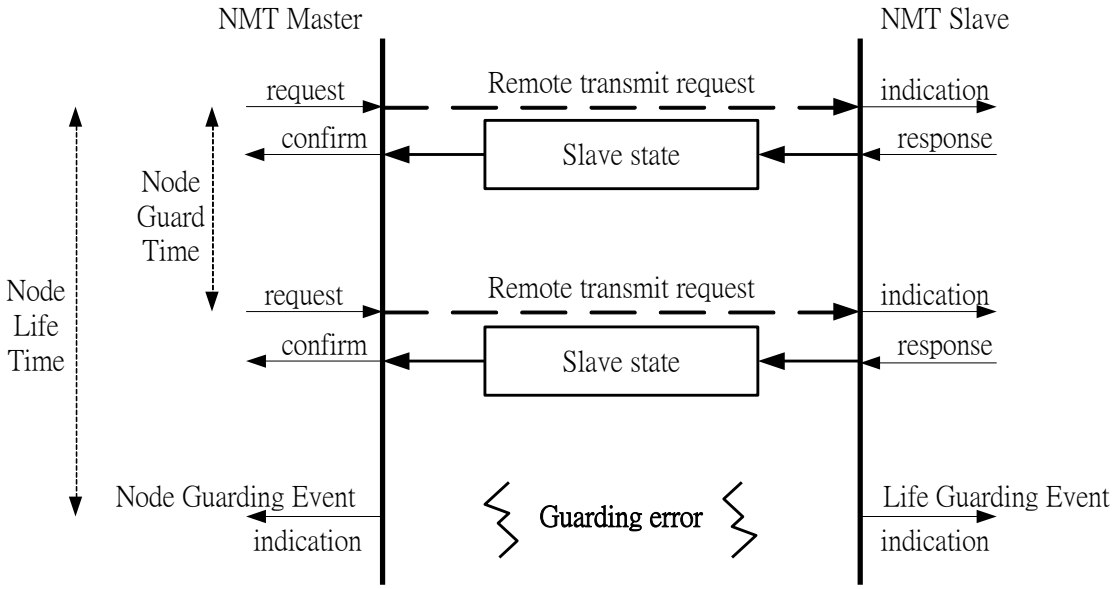
3.5.2 Error Control Protocols

There are two kinds of protocols defined in the error control protocol. According to the CANopen spec, one device is not allowed to use the following error control mechanisms at the same time, Node Guarding Protocol and Heartbeat Protocol. In addition, the CAN-8x23 provides the salve function of the Node Guarding Protocol for practical applications. Therefore, only node guarding protocols will be highlighted here, and described below.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Node Guarding Protocol

The Node Guarding Protocol follows the Master/Slave relationship. It helps users monitoring the node in the CAN bus. The communication method of node guarding protocol is defined as follows.



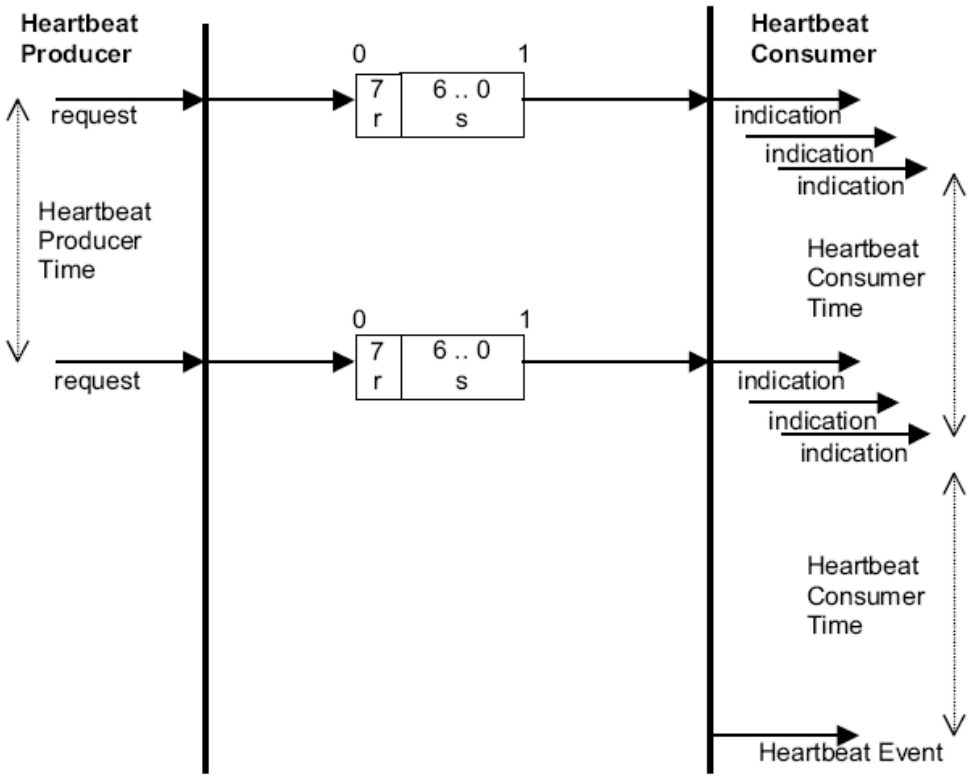
The NMT master will inspect each NMT slave at regular time intervals. This time-interval is called the node guard time, given by the “guard time * life time factor”, and may be different from each NMT slave. And the response of the NMT slave contains the state of that NMT slave, which may be in a "**Stopped**", "**Operational**", or "**Pre-operational**" state. The node life time factor can also be different for each NMT slave. If the NMT slave has not been inspected during its life time, a remote node error will be given, and indicate through the "Life Guarding Event" service.

In addition, the reported NMT slave state, which does not match the expected state, will also produce the “Node Guarding Event”. This event may occur in the DO and AO channels, and output the error mode value, recorded in the object with index 0x6207 and index 0x6444. Moreover, the object with index 0x6206 and 0x6443 can control the error mode value of the DO or AO channels in enabling or disabling when the “Lift Guarding Event” has been indicated. For more information about objects with index 0x6206, 0x6207, 0x6443, and 0x6444, please refers to the chapter 6.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Heartbeat Protocol

The Heartbeat Protocol follows the Producer/Consumer relationship. It provides a way to help uses monitor the node in the CAN bus. The communication method of heartbeat protocol is defined as follows.

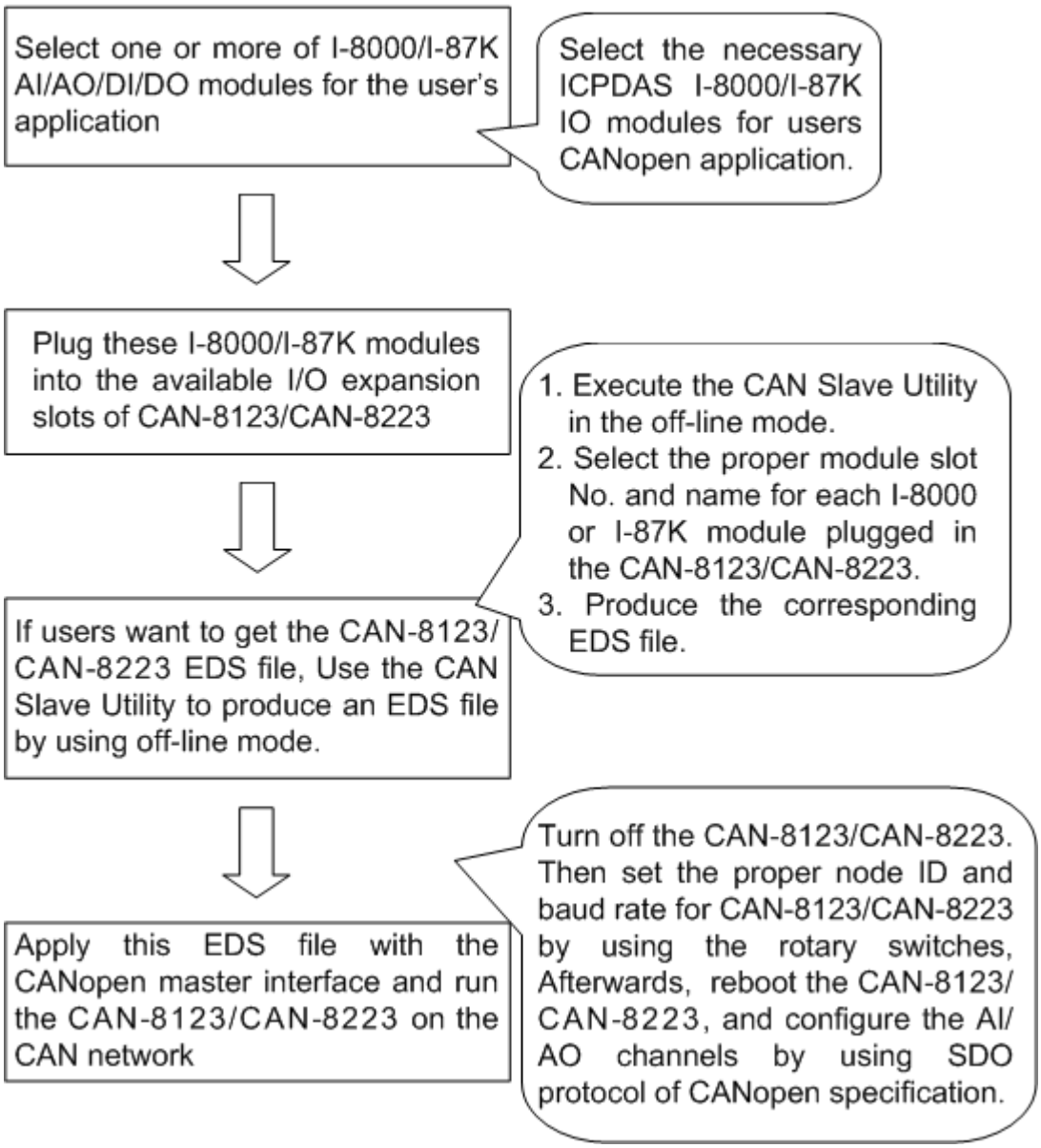


The Heartbeat Protocol defines an Error Control Service without need for remote frames. A Heartbeat Producer transmits a Heartbeat message cyclically. One or more Heartbeat Consumer receive the indication. The relationship between producer and consumer is configurable via the object dictionary. The Heartbeat Consumer guards the reception of the Heartbeat within the Heartbeat Consumer Time. If the Heartbeat is not received within the Heartbeat Consumer Time a Heartbeat Event will be generated.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

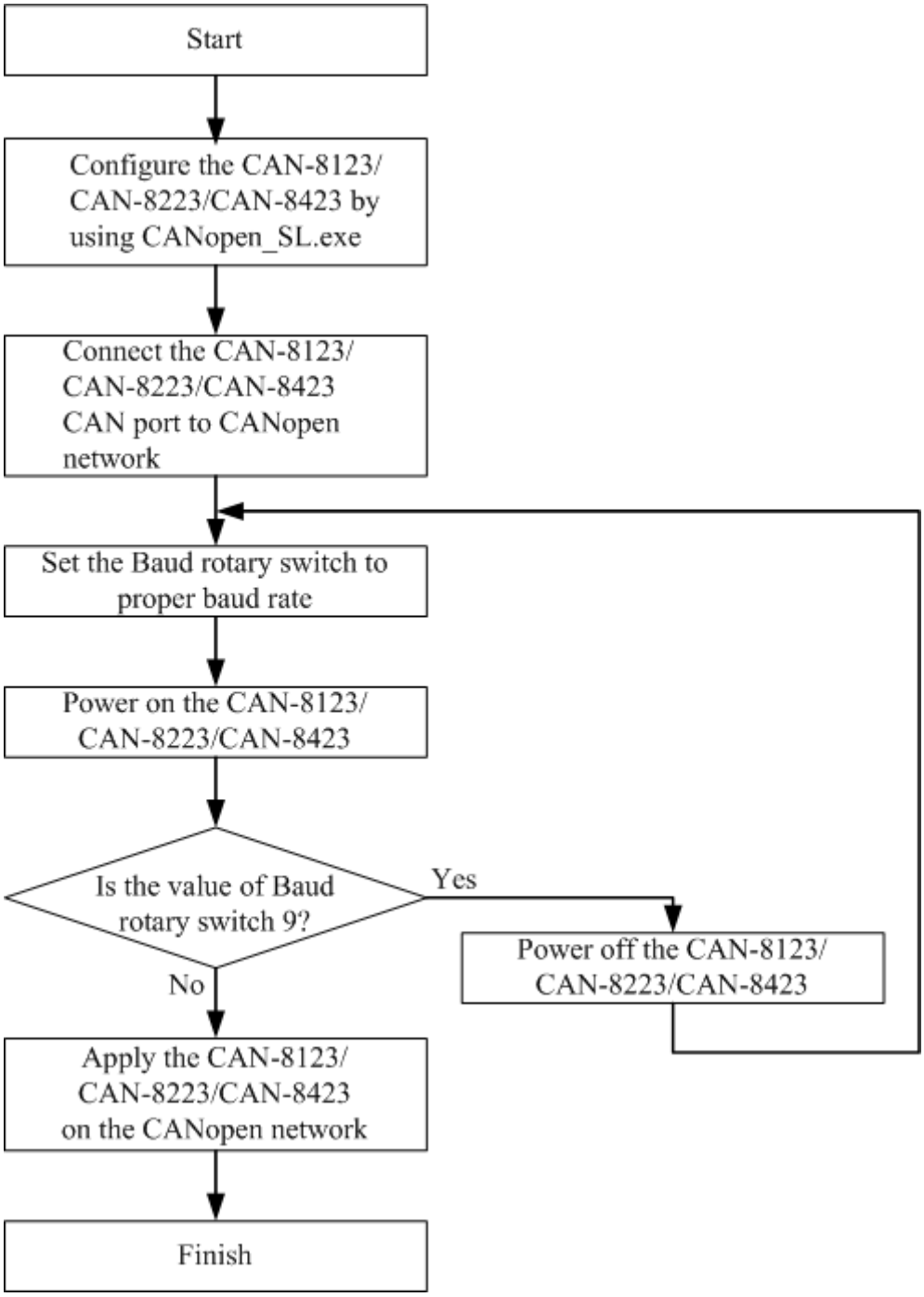
4 Configuration & Getting Start

4.1 CAN-8123/CAN-8223 Configuration Flowchart



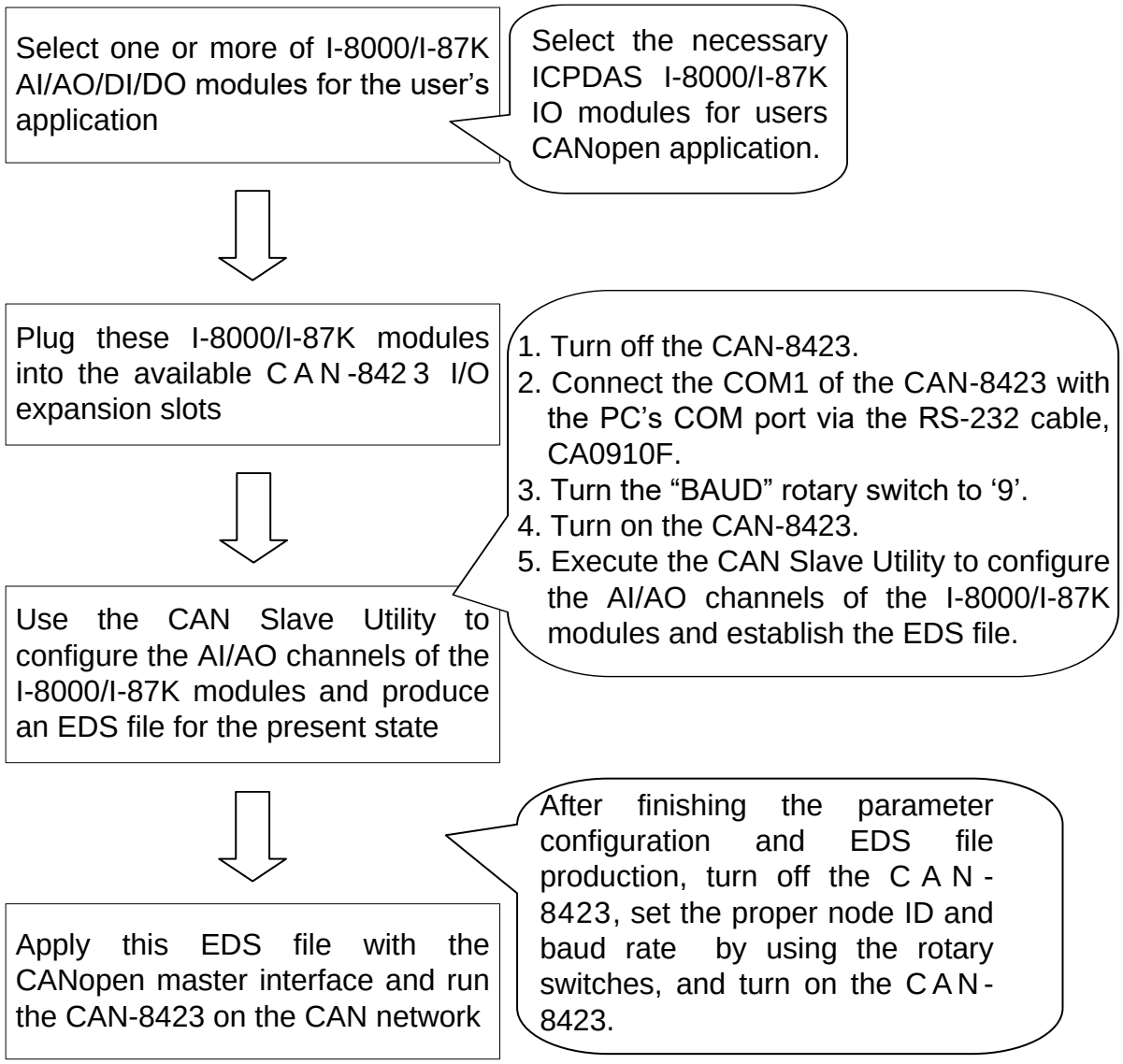
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

The following procedure is the general concept for the off-line mode. This procedure can be applied in the CAN-8x23.



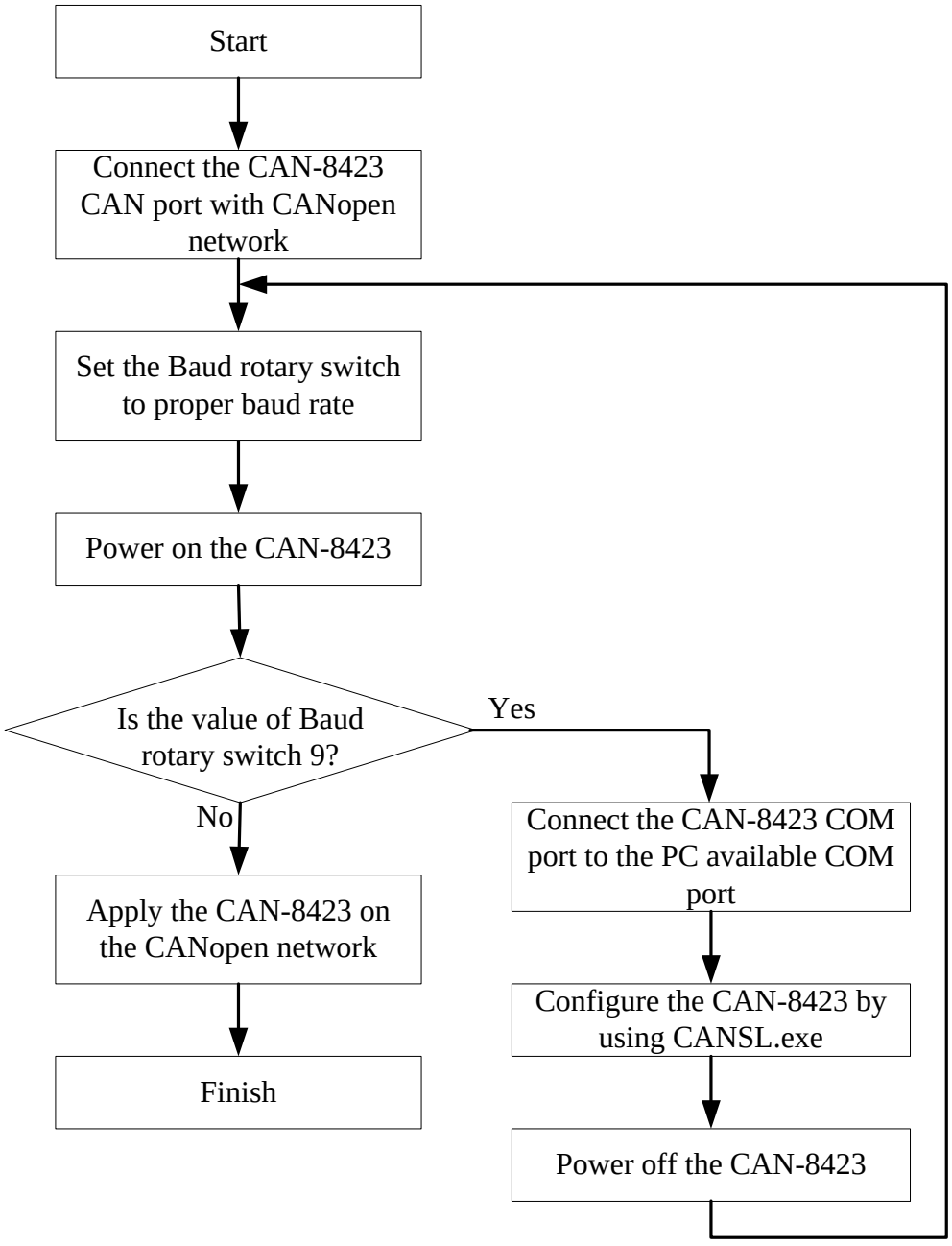
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.2 CAN-8423/CAN-8823 Configuration Flowchart



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

The following procedure is the general concept for the on-line mode. This procedure can be applied only in the CAN-8423 and CAN-8823.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.3 CANopen Slave Utility Overview

The CANopen Slave Utility is designed for the devices CAN-8x23. It provides following functions.

- Allows configuring the input range of the I-8000 and I-87K AI/AO modules plugged in CAN-8423 and CAN-8823.
- Supports to create EDS files to match the scan result in the on-line mode after scanning the I-8000 or I-87K modules in CAN-8423 and CAN-8823.
- Supports to produce the EDS file by using the off-line method for CAN-8x23.
- Shows the important information which is useful in the CANopen network. Such as the PDO communication objects, and the standardized device objects and manufacturer specific objects defined in the CAN-8x23.

Because all parameters configuration of the I-8000/I-87K AI/AO can be done by using SDO protocol, complying with the CANopen specifications, the CAN-8x23 can work directly without using the CANopen Slave Utility if users don't need the CAN-8x23 EDS file created under the on-line mode, i.e. users can turn on the CAN-8x23 and directly apply it in the CANopen network. If the EDS file is requested, users can get the EDS file by using CAN Slave Utility. If the AI/AO channels configuration is also requested, users can apply SDO protocol to modify the AI/AO parameter configurations. For more detail information, please refer to the chapters 5.5 and 6.2.

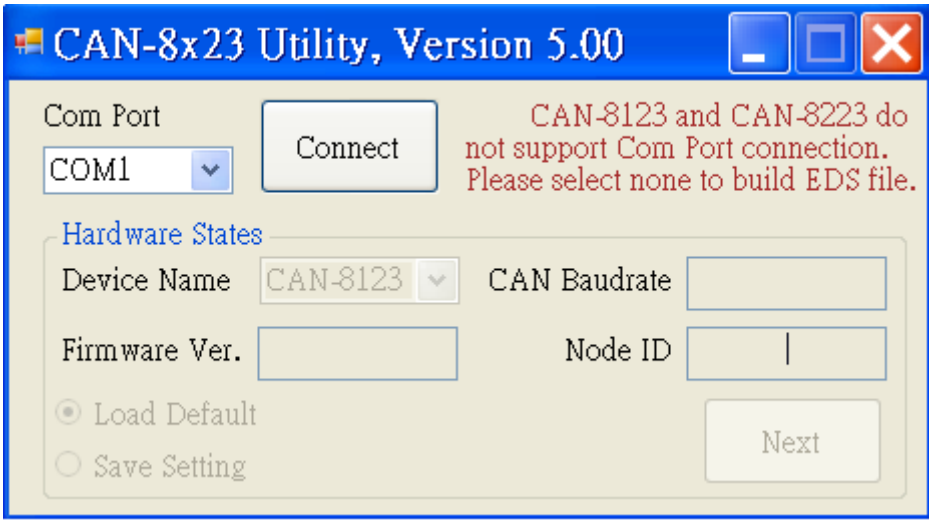
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.4 Configuration with the CANopen Slave Utility

CANopen Slave Utility

Step 1: Download the CANopen Slave Utility file from the web site http://www.icpdas.com/products/Remote_IO/can_bus/can-8423.htm or http://www.icpdas.com/products/Remote_IO/can-8123.htm or CD-ROM disk via the following path of “CD:\CANopen\Slave\CAN-8x23\Utility\”.

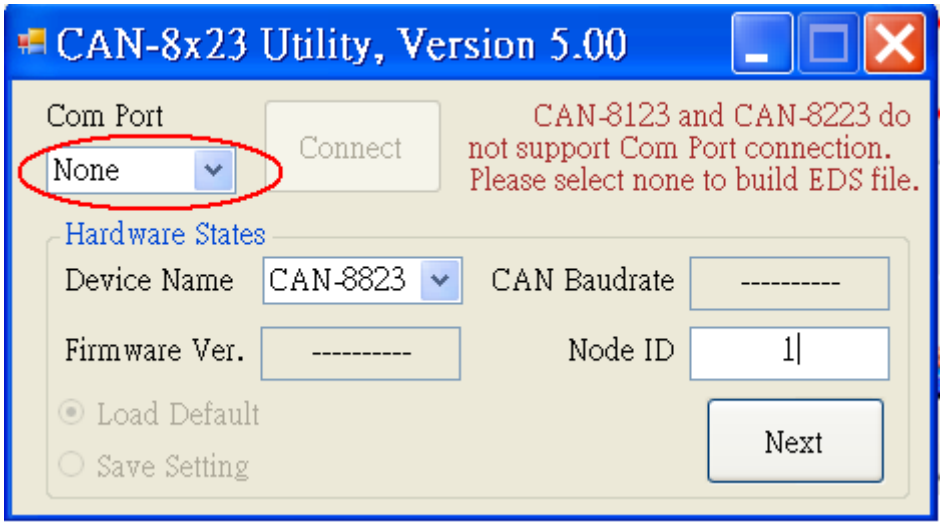
Step 2: Execute the CANopen_SL.exe file to start the CANopen Slave Utility.



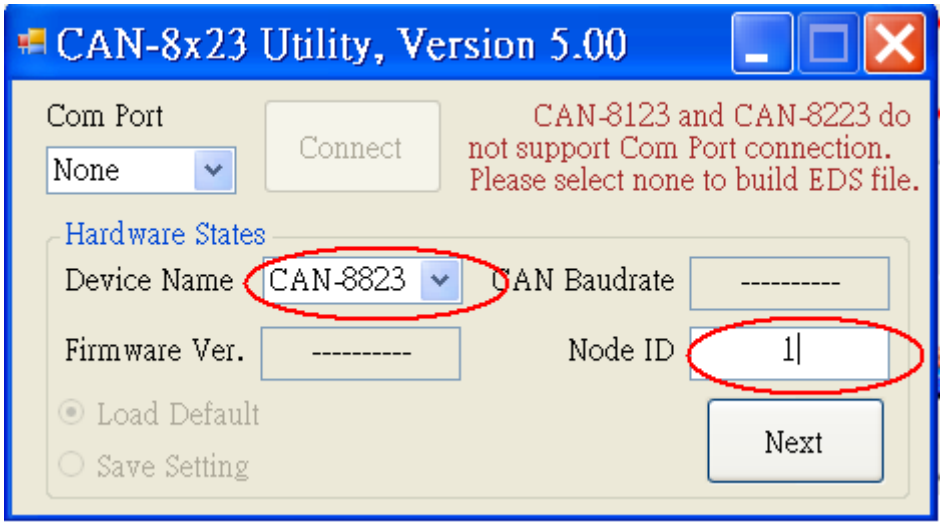
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.5 CAN-8123/8223 Configuration (Off-line mode)

Step 1: Select “None” in the “COM Port” area.



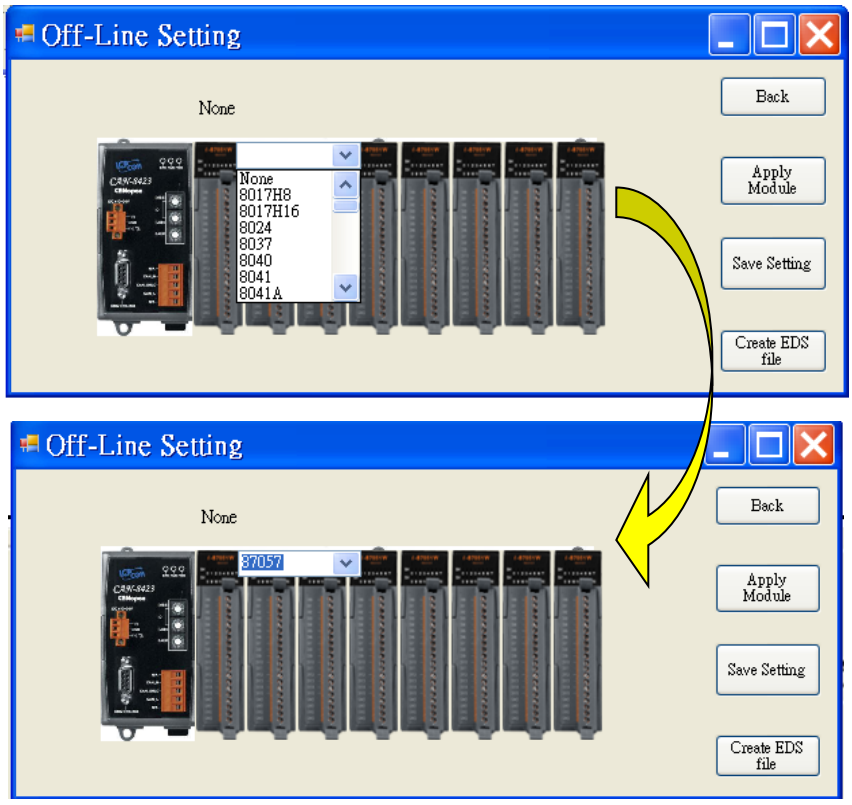
Step 2: Take the CAN slave device (CAN-8823 with node ID 1) as an example, Users have to fill in “NODE ID” with 1 and choose “Device Name” with CAN-8823. Then, press “Next” button.



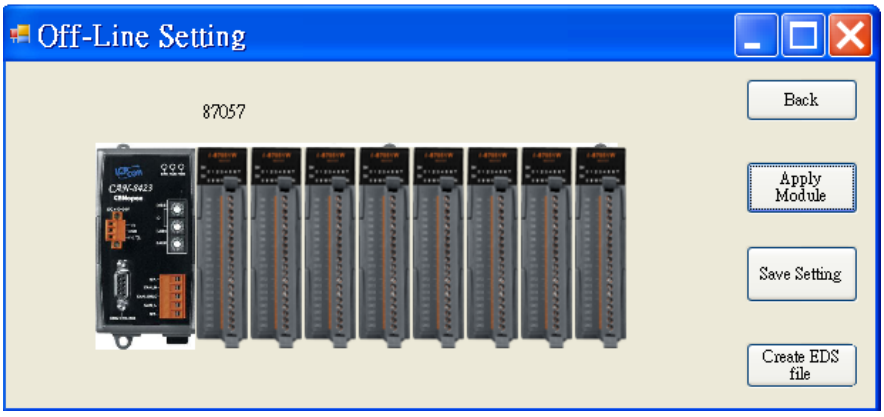
Step 3: Then, select a specific device presented in the “Off line Setting” frame, and choose a correct slot module inserted.

For example, if the I-87057 and I-8051 modules are inserted in slot 0 and slot 1 respectively, please select 87057 in the list box, and click “Apply Module” to save the configuration.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

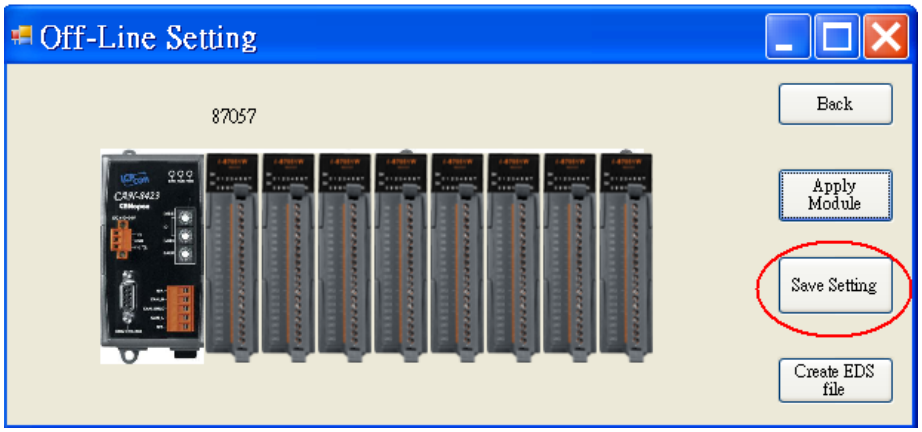


Step 4: After finishing the configuration, users can one-left click on the slot module in the “Off Line Setting” frame if need to change the configuration. If the configuration is successful, users can see the correct module name when mouse moving in, for example 87057 on the top of the slot module.

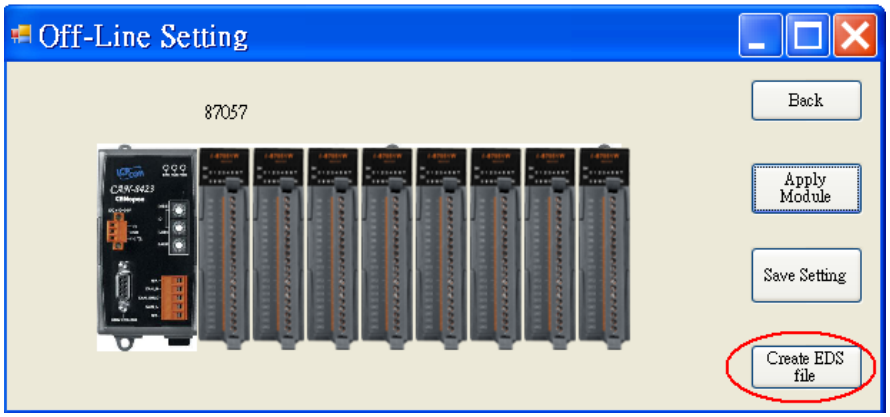


Step 5: Then, repeat the step 3~4 to configure the slot 1 to I-8042 module. Then, click “Save Setting” button to finish the off-line parameter settings.

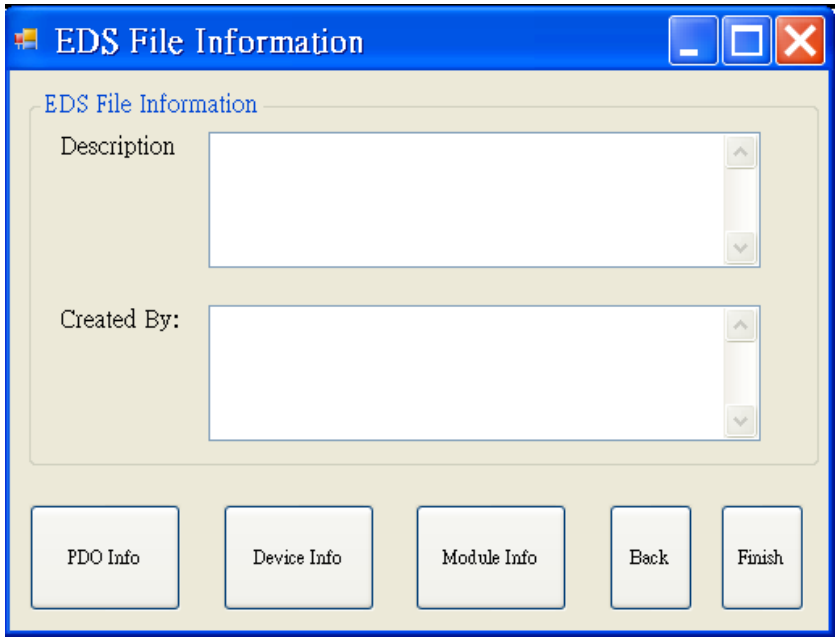
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Step 6: Then users can press button “Create EDS Module” for create CANopen slave EDS file.

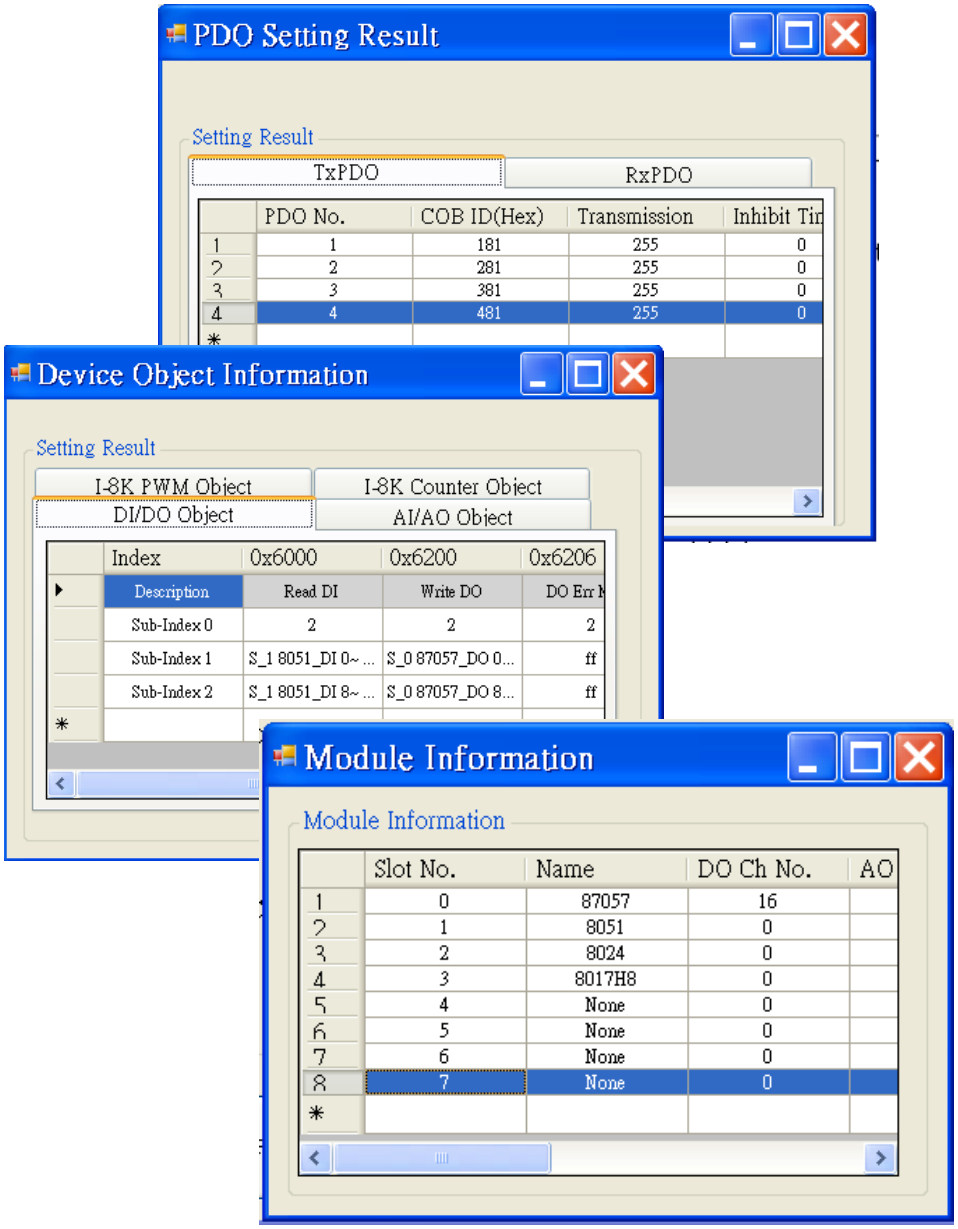


Step 7: The two fields, “description” and “create by”, can help users to do some notes in EDS files. If these two fields are empty, the “ICPDAS CANopen I/O Slave Device” and “ICPDAS” will be used as the default value when creating the EDS file.



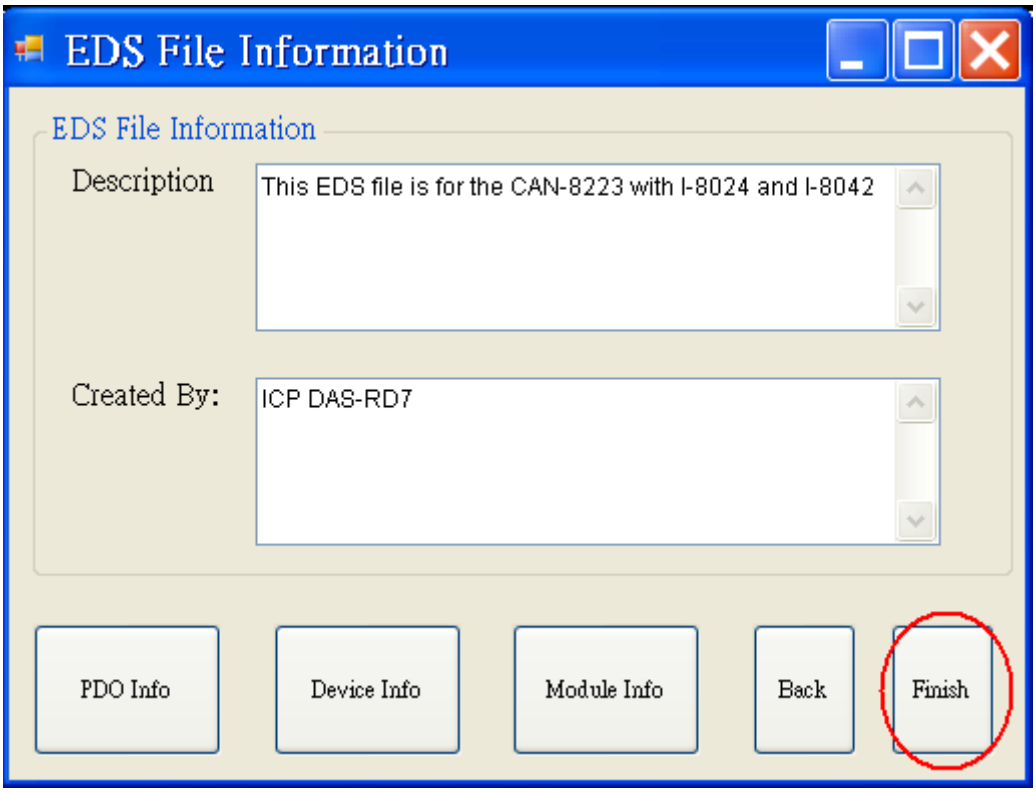
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 8: Users can select the “PDO Info”, the “Device Info“ and the “Module Info” button for purpose to view the PDO objects, device profile and slot module configuration information. These information dialogs are shown below.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

If everything is ok, click the “Finish” button to create the EDS file.



Note: If users use off-line method to get the EDS file, the objects, used to record the input/output range of the analog modules, will be described to default value in the EDS file. However, the I-87K slot modules will keep the input/output range parameter settings in their own EEPROM. As a matter of fact, it may cause the mismatch between real input/output range setting and EDS file. By the way, CAN-8123/CAN-8223 needs to configure the input/output range settings by using CANopen SDO protocol. For more detail, please refer to the section 5.5.

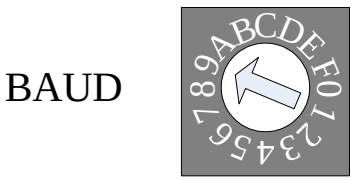
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.6 CAN-8423/8823 Configuration (On-line mode)

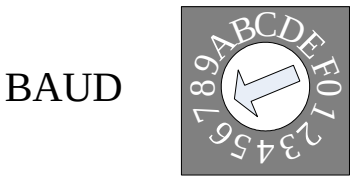
Before using the CAN Slave utility in the On-line mode of the CAN-8423/8823, please make sure that all connections are ready, from the CAN-8423/8823 to your PC via COM port. The architecture figure is displayed in the following. Take the following application as an example, the CAN-8423 and slot modules, I-87057, I-8051, I-8024 and I-8017 are inserted in the slot 0, 1, 2, 3 respectively.



Step 1: To turn off the CAN-8423 is the beginning. Then, users can set the “Baud” rotary switch of CAN-8423 to 9. Then, turn on the CAN-8423.

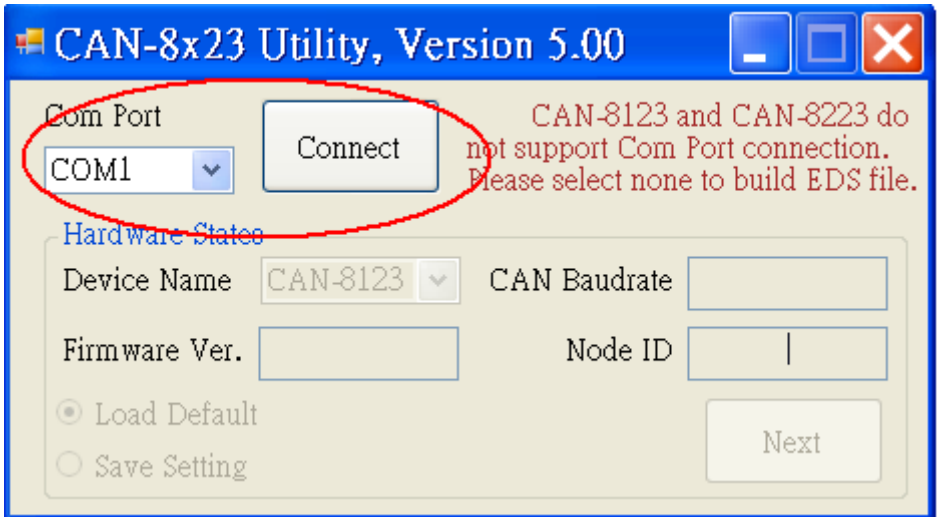


Step 2: Please use the “Baud” rotary switch again to set the baud rate for the CAN-8423, i.e. if users want to set baud rate in 1000Kbps, they have to adjust the “Baud” rotary switch to 7.

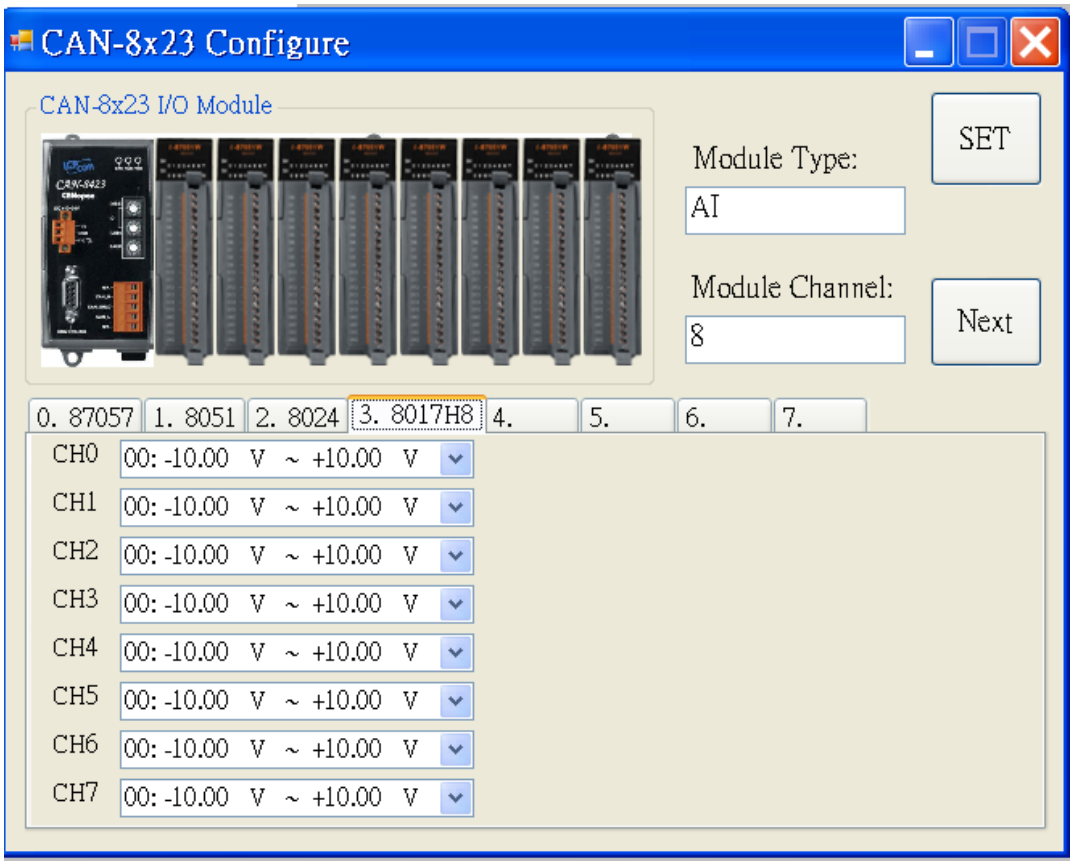


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3: To execute the CAN_SL.exe file, and to display the figure, users have to connect a PC COM port and the CAN-8423 or CAN-8823 well. Here, take the PC COM 1 as an example. Click “Connect” button to get the information stored in the CAN-8823.



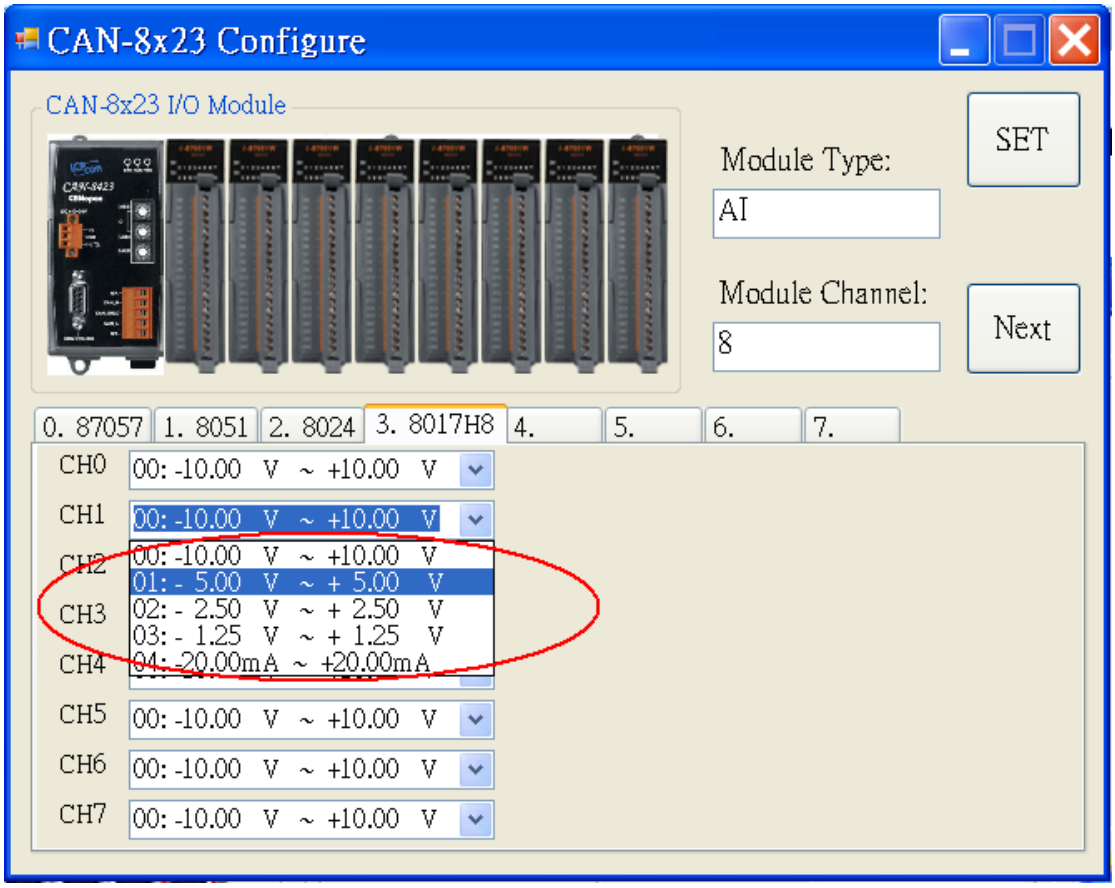
Step 4: Then, users can set the slot information of CAN-8823 in the below of “CAN-8x23 Configure” frame.



Step 5: Please select the slot module 3 in the control tab area, and choose the

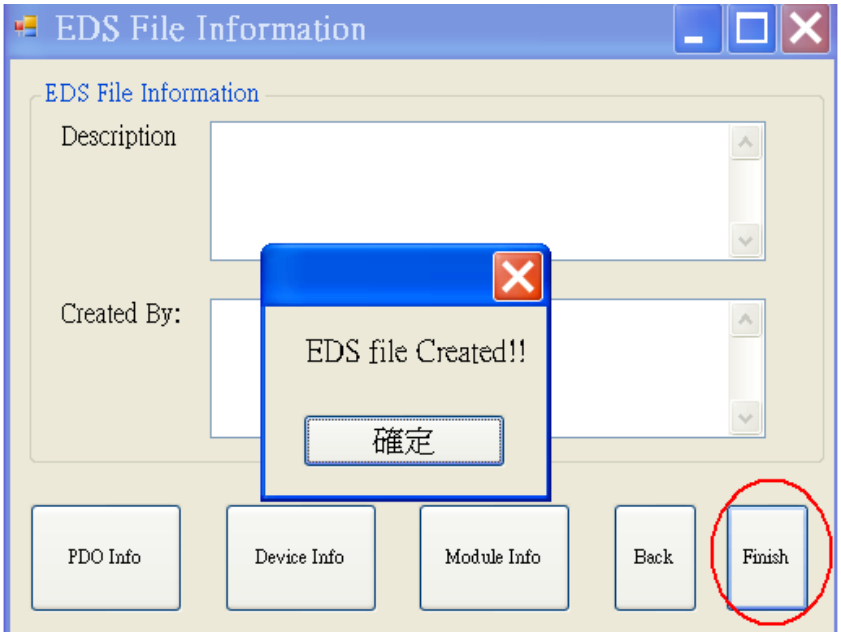
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

output range in the channel area. Here, take the selection -5.00V~+5.00V as an example. Because of the feature of I-8017H8 slot module, output range on each channel will be changed in the same way after users select the output range in one of the channels.



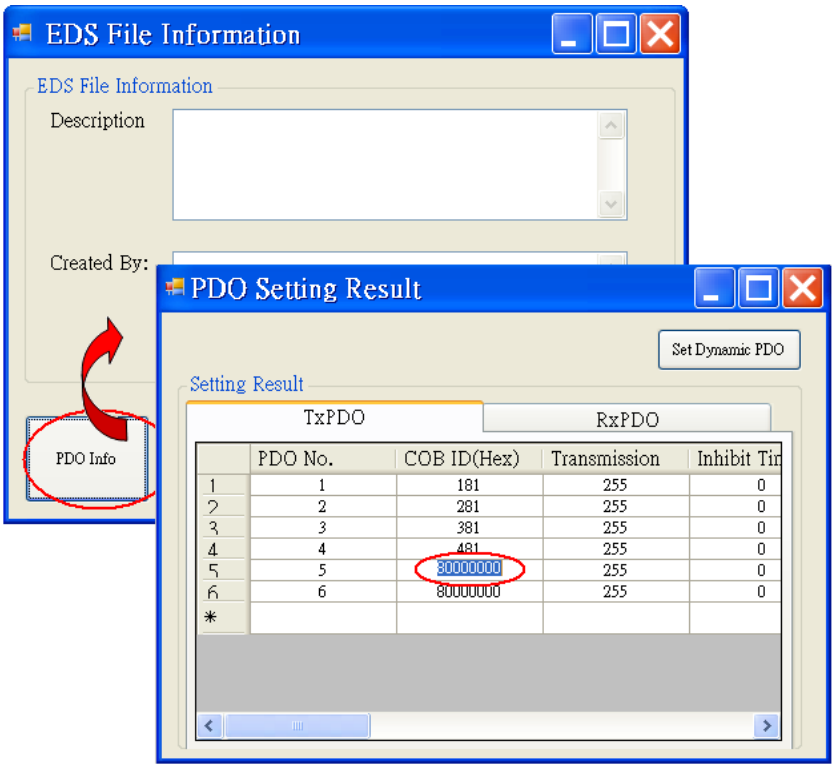
Step 6: After setting the proper output range, users can click “Set” button to store the configuration. If all of slot module configurations are finished, click “Next” button to next step.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



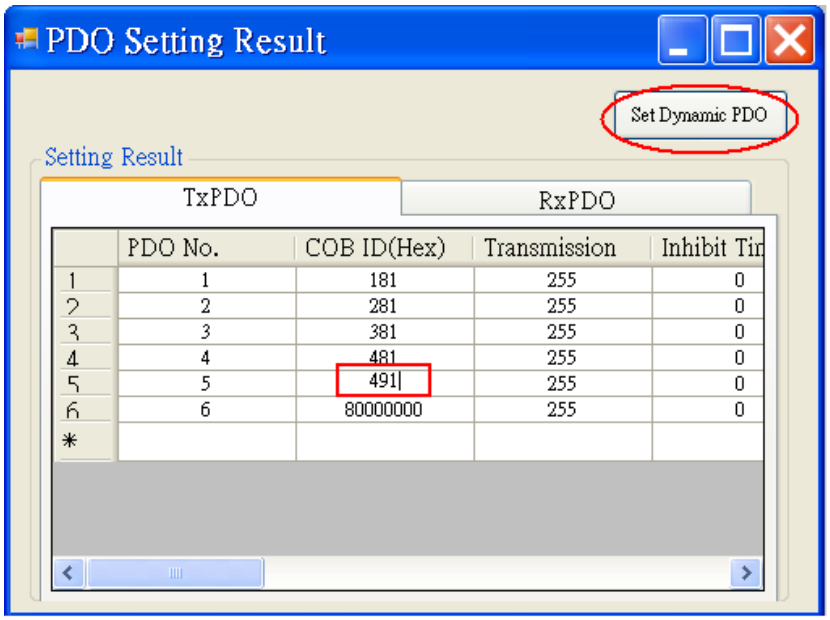
Step 7: Then, “EDS File Information” window will pop out. Users can fill the “Description” and “Create by” fields for the EDS file. Also, users can see the CANopen objects information and modules information by clicking the buttons. For more detail information, please refer to the Step 7 and 8 in section 4.5.

If User wants to set dynamic PDO COB-ID, input the COB-ID into the field of “PDO setting Result” window.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Then press button “ Set Dynamic PDO” to store the dynamic PDO COB-ID.



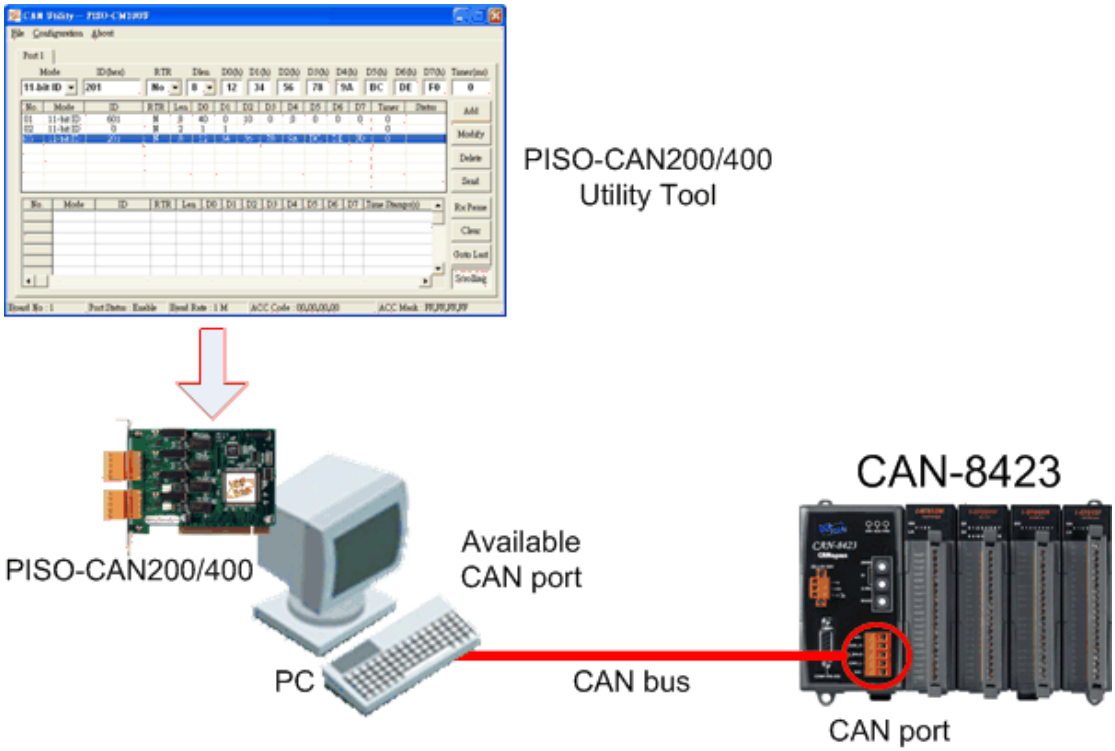
Note1: The CAN-8423/8823 can also create the EDS file by using off-line mode, and set the analog input range or analog output range by using the CANopen SDO protocol.

Note2: The function, dynamic PDO setting, is only supported on-line mode.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5 CANopen Communication Set

In the following section, several CANopen communication protocols are described. Each protocol description has one corresponding example. Because the communication methods in the CAN-8123/CAN-8223/CAN-8823 are similar to the one in CAN-8423, only the example for CAN-8423 is given. Before the example, users must have one CAN interface to send out the CAN command. Therefore, the PISO-CAN200/400 CAN interface card with a 2/4 CAN port PCI will be requested. It provides an easy-to-use utility tool to sending the CAN 2.0A or 2.0B command. The relationship between the software and the hardware is shown as follows.



Please refer to the PISO-CAN200/400 user manual to know how to use its Utility Tool.

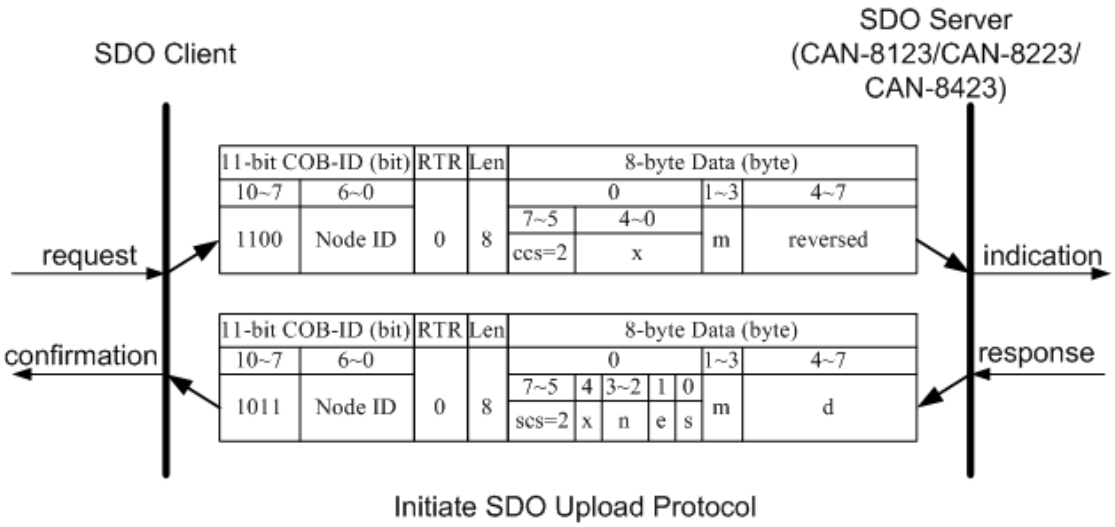
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1 SDO Communication Set

5.1.1 Upload SDO Protocol

Initiate SDO Upload Protocol

Before transferring the SDO segments, the client and server need to communicate with each other by using the initiate SDO upload protocol. Via the initiate SDO upload protocol, the SDO client will inform the SDO server what object the SDO client wants to request. As well, the initiate SDO upload protocol is permitted to transmit up to four bytes of data. Therefore, if the data length of the object, which the SDO client can read, is equal to or less than the permitted data amount, the SDO communication will be finished only by using the initial SDO upload protocol, i.e. if the data upload is less enough to be transmitted in the initiate SDO upload protocol, then the upload SDO segment protocol will not be used. The communication process of this protocol is shown as follows.



Initiate SDO Upload Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specified
2: initiate upload request

scs : server command specified
2: initiate upload response

n : Only valid if **e** = 1 and **s** = 1, otherwise 0. If valid, it indicates the number of bytes in **d** that do not contain data. Bytes [8-**n**, 7] do not contain segment data.

e : transfer type
0: normal transfer
1: expedited transfer
If the **e**=1, it means that the data of the object are equal or less than 4 bytes, and only initiate SDO upload protocol is needed. If **e**=0, the upload SDO segment protocol is necessary.

s : size indicator
0: Data set size is not indicated.
1: Data set size is indicated.

m : multiplexer
It represents the index/sub-index of the data to be transfer by the SDO. The first two bytes are the index value and the last byte is the sub-index value.

d : data
e=0, **s**=0: **d** is reserved for further use.
e=0, **s**=1: **d** contains the number of bytes to be uploaded, and byte 4 contains the least significant bit, and byte 7 contains the most significant bit.
e=1, **s**=1: **d** contains the data of length 4-**n** to be uploaded, the encoding depends on the type of the data referenced by index and sub-index.
e=1, **s**=0: **d** contains unspecified number of bytes to be uploaded.

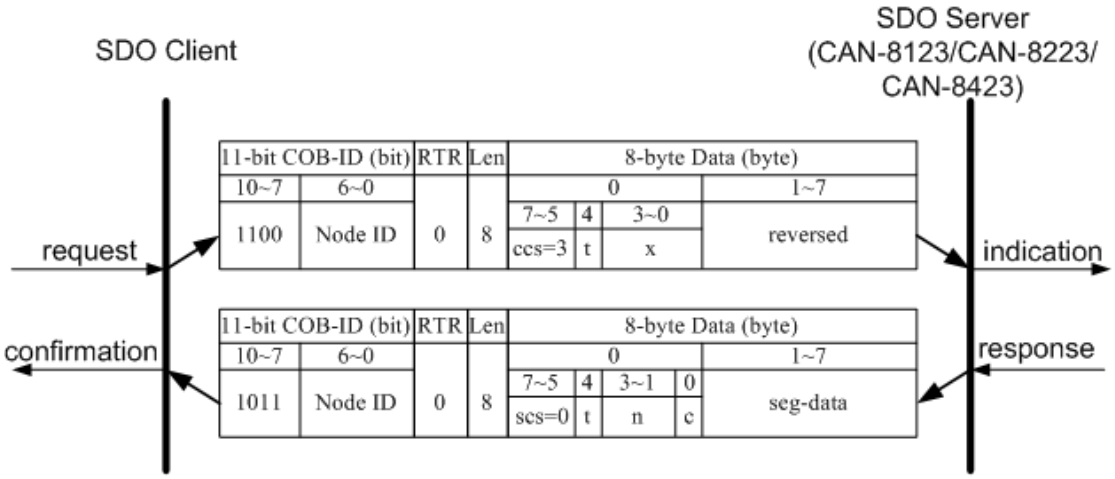
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Upload SDO Segment Protocol

When the upload data length is over 4 bytes, the upload SDO segment protocol will be needed. After finishing the transmission of the initiate SDO upload protocol, the SDO client will start to upload the data. The upload SDO segment protocol will comply with the process shown below.



Upload SDO Segment Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specified
3: upload segment request

scs : server command specified
0: upload segment response

t : toggle bit
This bit must alternate for each subsequent segment that is uploaded. The first segment will have the toggle bit set to 0. The toggle bit will be equal for the request and the response message.

c : indicates whether there are still more segments to be uploaded
0: more segments to be uploaded.
1: no more segments to be uploaded.

seg-data : It is at most 7 bytes of segment data to be uploaded. The encoding depends on the type of the data referenced by index and sub-index.

n : It indicates the number of bytes in **seg-data** that do not contain segment data. Bytes [8-**n**, 7] do not contain segment data. **n** = 0 if no segment size is indicated.

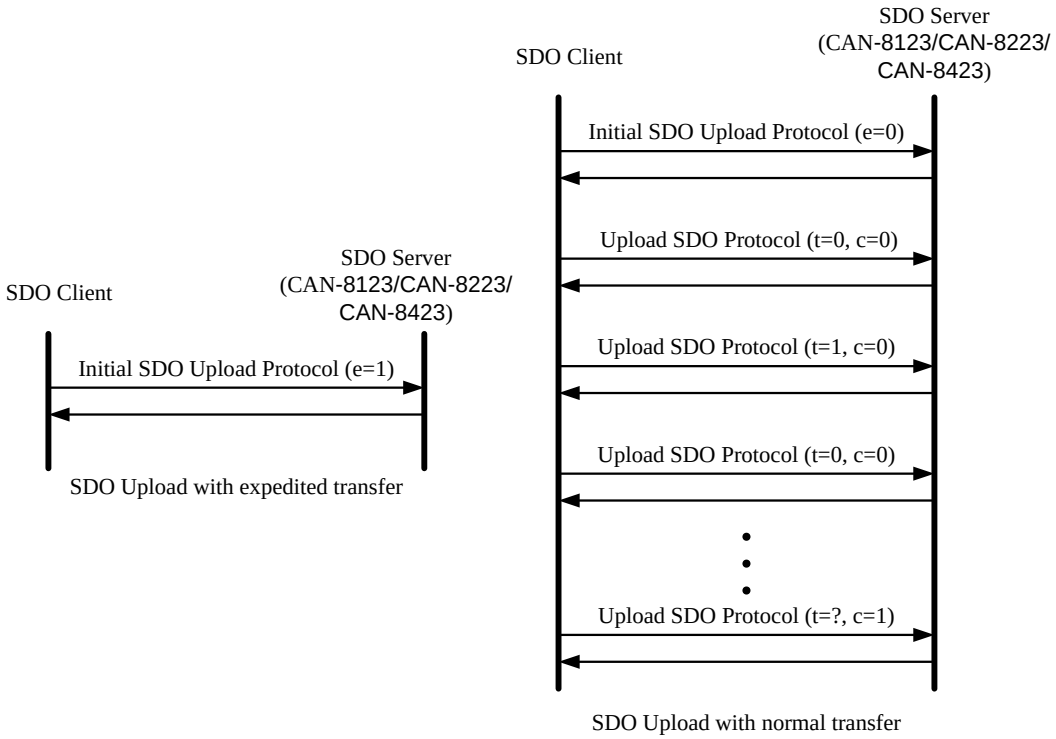
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SDO Upload Example

The practical application of the SDO upload is illustrated as below.



In the following paragraph, both expedited transfer and normal transfer are given according to the procedure described above. In addition, the method of how to get the value stored in the object dictionary is also presented. As to the initiate SDO upload protocol, users can obtain how many sub-indexes the object with index 0x1400 can support. This information is in the object with index 0x1400 with sub-index 00. As well, users can get the string in the object with index 0x1008 via the initiate SDO upload protocol and the upload SDO segment protocol.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Example for expedited transfer**

Step 1. SDO message will be sent to the CAN-8423 to obtain the object entry with index 0x1400 and sub-index 00 stored in the communication profile area. The message structure is as follows. Moreover, the node ID of the CAN-8423 set to 1, and the information about the object entry with index 0x1400 will be described in the chapter 6.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	00	14	00	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 2

m : 00 14 00

According to the low byte has the higher transferred sequence, the first byte “00” will get the priority than the second byte “14”. Here the last byte “00” means the sub-index 00.

Step 2. The CAN-8423 will reply to the data stored in the object entry with index 0x1400 and sub-index 00.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	00	14	00	02	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 2

n : 3

e : 1

s : 1

m : 00 14 00

d : 02 00 00 00

Because of the **n**=3, only the 4th byte is valid. Therefore, the feedback value is 02.

-

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Example for normal transfer**

Step 1. Send the RxSDO message to the CAN-8423 to obtain the object entry with index 0x1008 and sub-index 00 stored in the communication profile area. The message structure is as follows. Moreover, the node ID for the CAN-8423 set to 1, and the information about object entry with index 0x1008 will be described in the chapter 6.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	08	10	00	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 2
m : 08 10 00

Step 2. The CAN-8423 will respond to the SDO message with the indication of how many bytes will be uploaded from the CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	41	08	10	00	09	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 2
n : 0
e : 0
s : 1
m : 08 10 00
d : 09 00 00 00

Because of the **e**=0 and **s**=1, the **d** means how many data users will upload from the CAN-8423. The byte “09” is the lowest byte in the data length with a long format. Therefore, the data “09 00 00 00” means that users will upload 9 bytes data from CAN-8423.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. The CAN-8423 is requested to start the data transmission.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	60	00	00	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 3
t : 0

Step 4. The CAN-8423 will respond to the first 7 bytes in the index 0x1008 and sub-index 00 object entries.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	00	43	41	4E	2D	38	34	32

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 0
t : 0
n : 0
c : 0
seg-data : 43 41 4E 2D 38 34 32

Users can check the chapter 6 to know that the object entry with index 0x1008 and sub index 00 has the data type “VISIBLE_STRING”. Therefore, users need to transform these data values into the corresponding ASCII character. After transformation, they become “CAN-842”.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 5. The CAN-8423 is requested to transmit the rest of the data.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	70	00	00	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 3
t : 1

Step 6. Tthe rest of the data will be received from the SDO server.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	1B	33	00	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

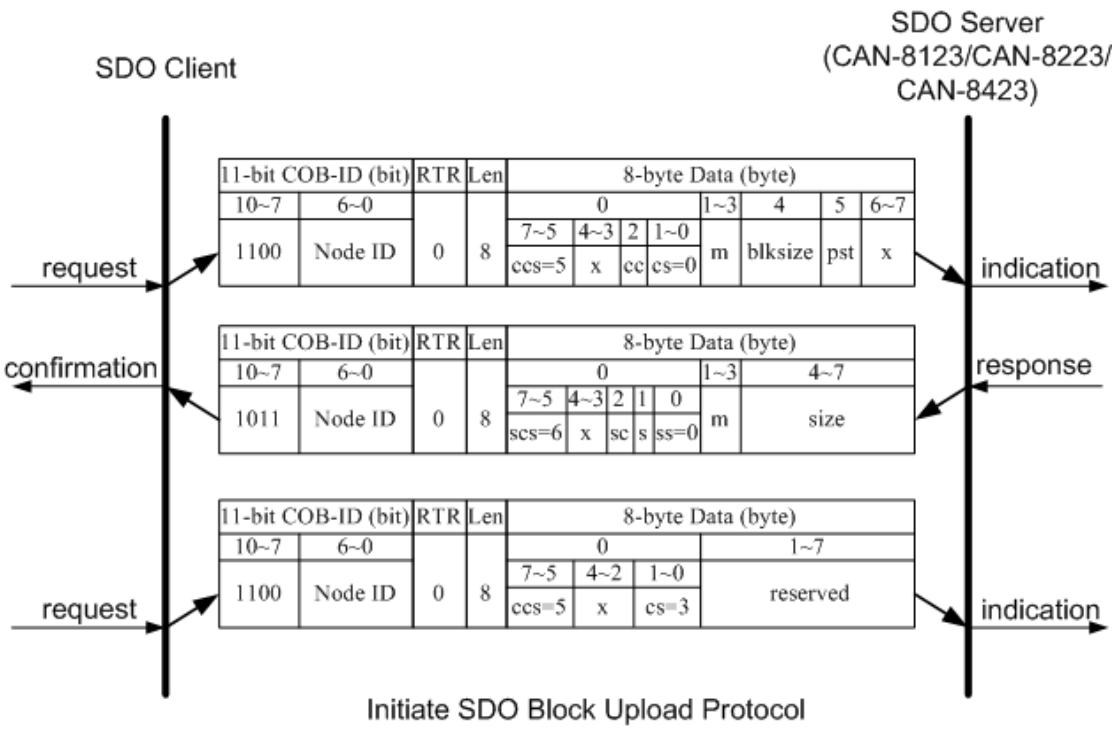
scs : 0
t : 1
n : 5
c : 1
seg-data : 33 00 00 00 00 00 00
Because of the **n**=5, and only the first two bytes are valid, the value of 0x33 and 0x00 will be transferred to the corresponding ASCII character. After transformation, it became “3 ”.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1.2 SDO Block Upload Protocol

Initiate SDO Block Upload Protocol

The SDO Block Upload is usually used for the large data transmission. At the beginning of the SDO Block Upload, the Initiate SDO Block Upload protocol is needed. This protocol is described below.



Initiate SDO Block Upload Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specified
5: block upload

scs : server command specified
6: block upload.

cs : client subcommand
0: initiate upload request
3: start upload

ss : server subcommand
0: initiate upload response

m : multiplexor
It represents the index/sub-index of the data to be transfer by the SDO.

cc : client CRC support
cc=0: Client does not support generating CRC on data.
cc=1: Client supports generating CRC on data.

sc : server CRC support
sc=0: Server does not support generating CRC on data.
sc=1: Server supports generating CRC on data.

pst : Protocol Switch Threshold in bytes to change the SDO transfer protocol
pst=0: change of transfer protocol not allowed
pst>0: If the size of the data in bytes that has to be uploaded is less or equal **pst**, the server can optionally switch to the 'SDO Upload Protocol' by transmitting the server response of the 'SDO Upload Protocol'.

s : size indicator
0: Data set size is not indicated.
1: Data set size is indicated.

size : upload size in byes
s=0: **size** is reserved for further use, always 0.
s=1: **size** contains the number of bytes to be uploaded. Byte 4 contains the LSB and byte 7 is the MSB.

blksize : number of segments per block with 0 < **blksize** < 128

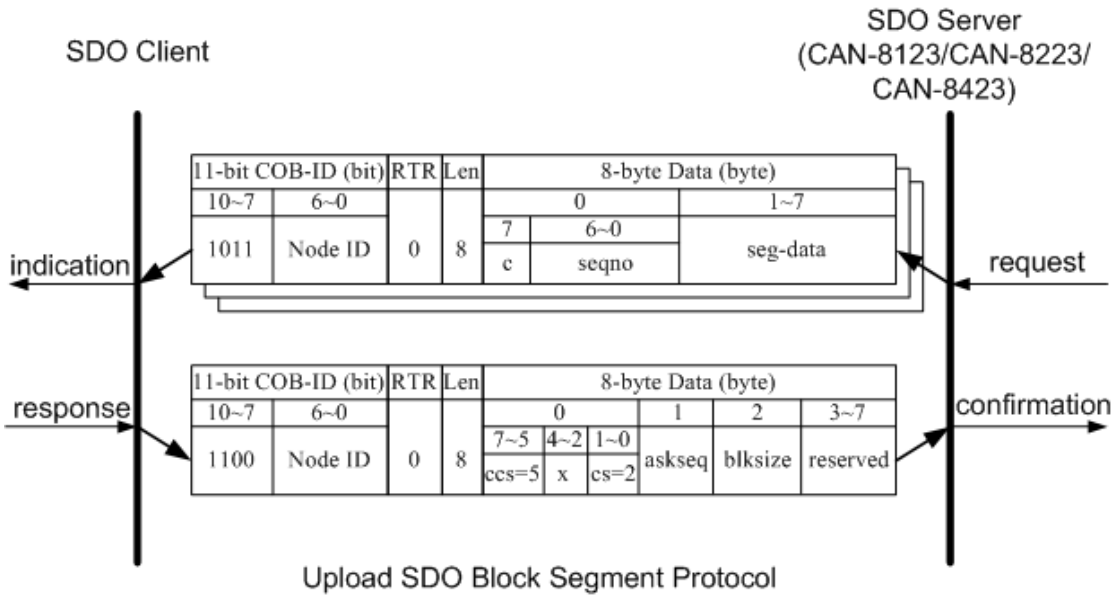
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Upload SDO Block Segment Protocol

After finishing the Initiate SDO Block Upload protocol, the SDO server starts to respond to the data by using the Upload SDO Block Segment protocol. Each block contains 1 segment for the minimum and 127 segments for the maximum. One segment consists of 1~7 bytes. And only one block can be transmitted during an Upload SDO Block Segment protocol. The SDO server can send a maximum of 127 blocks by using 127 Upload SDO Block Segment protocols. The following figure is the structure for the Upload SDO Block Segment protocol.



Upload SDO Block Segment Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specifier
5: block upload

cs : client subcommand
2: block upload response

c : It indicates whether there are still more segments to be uploaded.
0: more segments to be uploaded
1: no more segments to be uploaded , enter ‘End block upload’ phase

seqno : sequence number of segment, 0 < **seqno** < 128

seg-data : It is at most 7 bytes of segment data to be uploaded.

ackseq : sequence number of last segment that was successfully received during the last block upload
If **ackseq** is set to 0, the client will indicate that the segment with the sequence number 1 was not received correctly and all segments have to be retransmitted by the server.

blksize : number of segments per block that has to be used by server for the following block upload with 0 < **blksize** < 128

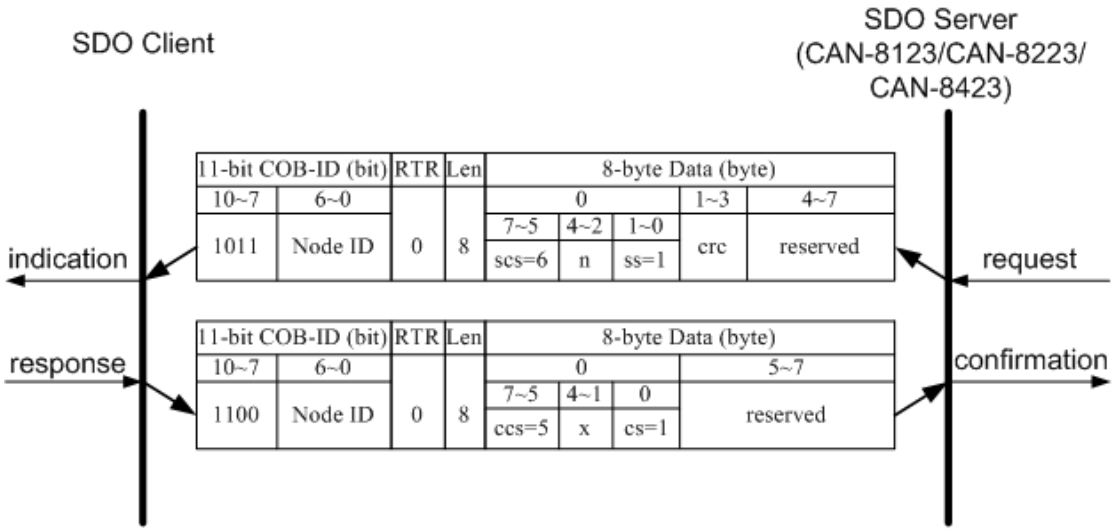
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

End SDO Block Upload Protocol

The End SDO Block Upload protocol is used for finishing the SDO Block upload, and is shown in the following figure.



End SDO Block Upload Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specifier
5: block upload

scs : server command specifier
6: block upload

cs : client subcommand
1: end block upload request

ss : server subcommand
1: end block upload response

n : It indicates the number of bytes in the last segment of the last block that do not contain data. Bytes [8-**n,**7] do not contain segment data.

crc : 16 bit Cyclic Redundancy Checksum (CRC) for the whole data set.
The algorithm for generating the CRC is as follows.

x^16+x^12+x^5+1

CRC is only valid if in Initiate Block Upload **cc** and **sc** are set to 1. Otherwise **crc** has to be set to 0. For CAN-8123/CAN-8223/CAN-8423, it is not support CRC check mechanism.

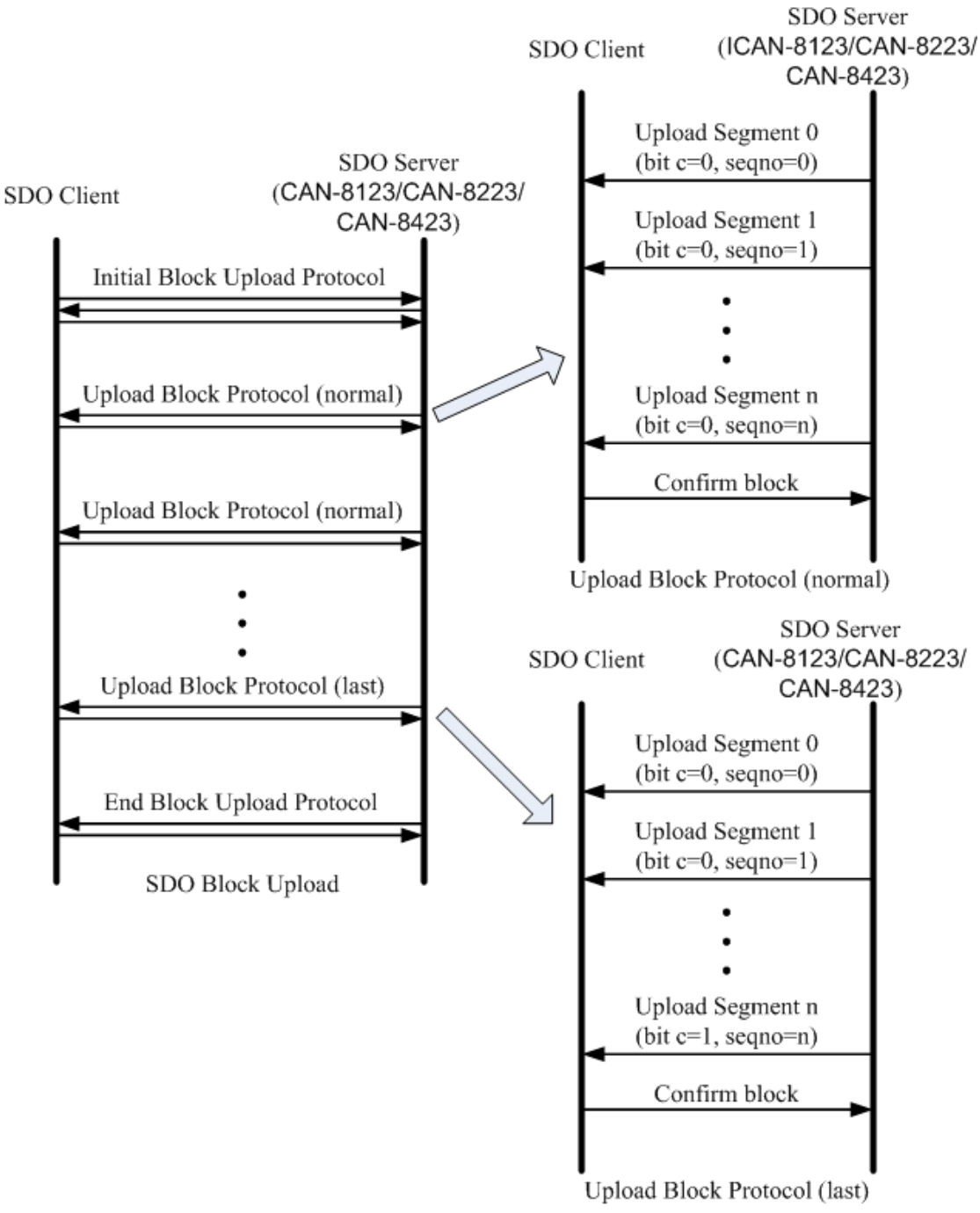
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SDO Block Upload Example

The following figure shows the general procedure of applying the SDO Block upload.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

By the following procedure, an example is provided to obtain a value of the index 0x1008 and sub-index 00 object entries.

Step 1. The CAN-8423 is requested to transmit the data by using the SDO Block Upload method.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A0	08	10	00	7F	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 5
cc : 0
cs : 0
m : 08 10 00
blksize : 7F
Each block contains 127 segments.
pst : 00

Step 2. The CAN-8423 will confirm the requirement with the Initiate SDO Block Upload protocol.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	C2	08	10	00	09	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 6
sc : 0
s : 1
ss : 0
m : 08 10 00
size : 09 00 00 00
The CAN-8123 will response 9 bytes data during the SDO Block Upload.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. The message is sent to finish the Initiate SDO Block Upload protocol, and will actuate the CAN-8423 to start the data transmission.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A3	00	00	00	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 5
cs : 3

Step 4. The CAN-8423 will responds to the first 7 bytes of data by using the Upload SDO Block Segment protocol.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	1	43	41	4E	2D	38	34	32

SDO client ← **SDO server (CAN-8x23)**

c : 0
seqno : 1
seg-data : 43 41 4E 2D 38 34 32

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 5. The CAN-8423 will transmit the rest of the data.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	82	33	00	00	00	00	00	00

SDO client ◄—————————— **SDO server (CAN-8x23)**

c : 1
seqno : 2
seg-data : 33 00 00 00 00 00 00
Because this segment is the last one, not all of the data in the **seg-data** filed is useful. The valid data length will be indicated when the CAN-8423 send a message to finish the Block Upload protocol. Please refer to the value of **n** in the step 7.

Step 6. Then, users will send a message to confirm the received data transmitted from the CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A2	02	7F	00	00	00	00	00

SDO client ——————————► **SDO server (CAN-8x23)**

ccs : 5
cs : 2
ackseq : 2
blksize : 7F

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 7. When the reception is confirmed, the CAN-8423 will send a message to enter the End SDO Block Upload protocol.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	D5	00	00	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 6

n : 5

This value means the invalid data in the last segment are from [8-5] to 7, i.e. only the first 3 bytes are valid.

ss : 1

crc : 00 00

Step 8. Users will send a message to finish the End SDO Block Upload protocol.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	A1	00	00	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 5

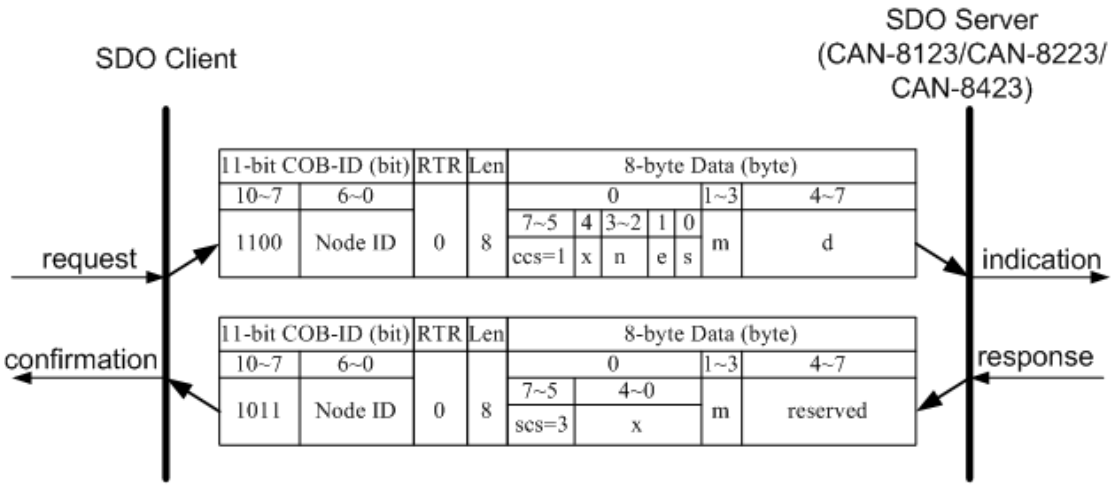
cs : 1

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1.3 Download SDO Protocol

Initiate SDO Download Protocol

The download modes are similar to the upload modes, but different in some parameters of the SDO messages. They are also separated into two steps. If the download data length is less than 4 bytes, the download action will finish in the download initialization protocol. Otherwise, the download segment protocol will be needed. These two protocols are shown below.



Initiate SDO Download Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs : client command specified
1: initiate download request

scs : server command specified
3: initiate download response

n : Only valid if **e** = 1 and **s** = 1, otherwise 0. If valid, it indicates the number of bytes in **d** that do not contain data. Bytes [8-**n**, 7] do not contain segment data.

e : transfer type
0: normal transfer
1: expedited transfer
If the **e**=1, it means that the data of the object are equal or less than 4 bytes, and only initiate SDO download protocol is needed.
If **e**=0, the download SDO protocol is necessary.

s : size indicator
0: data set size is not indicated
1: data set size is indicated

m : multiplexer
It represents the index/sub-index of the data to be transfer by the SDO.

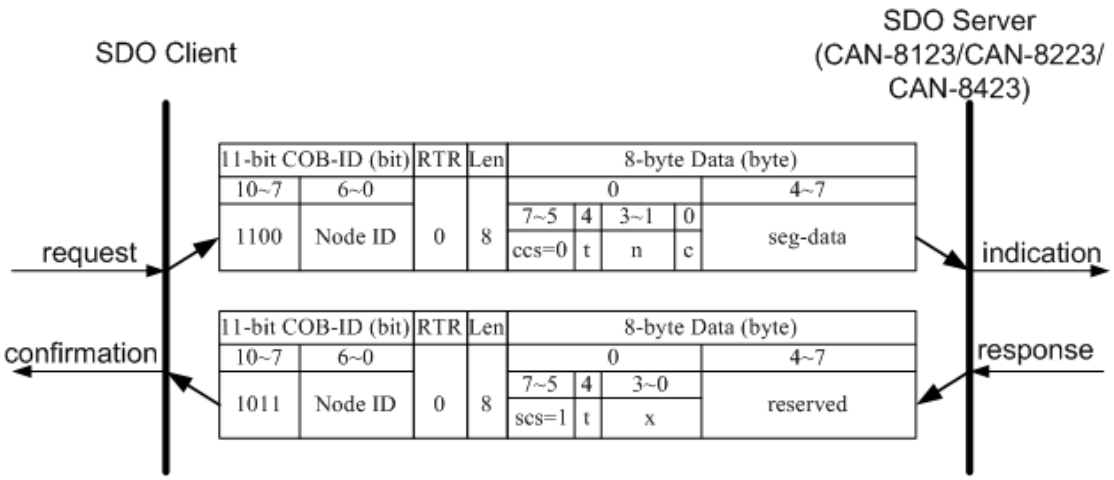
d : data
e=0,**s**=0: **d** Is reserved for further use.
e=0,**s**=1: **d** contains the number of bytes to be downloaded, and byte 4 contains the least significant bit, and byte 7 contains the most significant bit.
e=1,**s**=1: **d** contains the data of length 4-**n** to be downloaded, the encoding depends on the type of the data referenced by index and sub-index.
e=1,**s**=0: **d** contains unspecified number of bytes to be downloaded.

x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Download Segment Protocol



Download SDO Segment Protocol

ccs : client command specified
0: download segment request

scs : server command specified
1: download segment response

seg-data : It is at most 7 bytes of segment data to be downloaded. The encoding depends on the type of the data referenced by index and sub-index.

n : It indicates the number of bytes in **segment data** that do not contain segment data. Bytes [8-**n**, 7] do not contain segment data. **n** = 0 if no segment size is indicated.

c : It indicates whether there are still more segments to be downloaded.
0 more segments to be downloaded
1: no more segments to be downloaded

t : toggle bit
This bit must alternate for each subsequent segment that is downloaded. The first segment will have the toggle-bit set to 0. The toggle bit will be equal for the request and the response message.

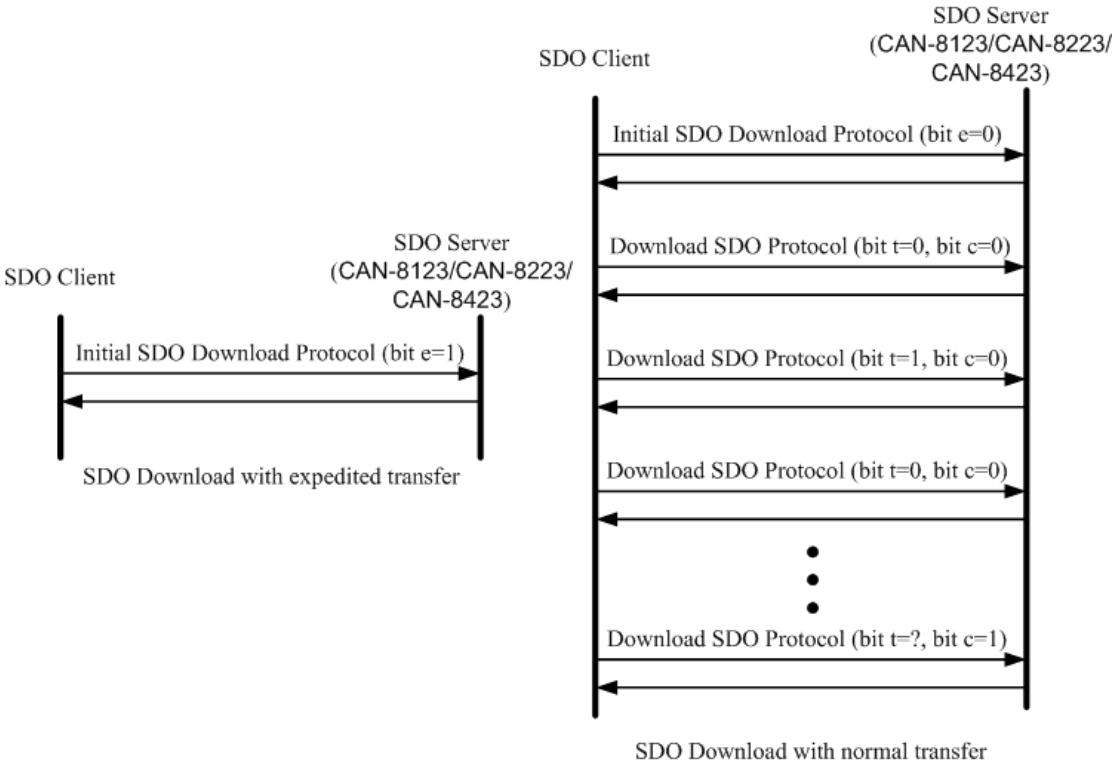
x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SDO Download Example

When the SDO download example has been applied, the procedure in the below figure may be applied.



Since all of those object entries, which can be written, in the CAN-8123/CAN-8223/CAN-8423 are equal or less than 4 bytes, we can only provide the example for expedited transfer.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Example for expedited transfer**

Step 1. The Rx SDO message is sent to the CAN-8423 to access the object entry with index 0x1400 and sub-index 02 stored in the communication profile area. For example, the value of this object entry is changed to 5, as the node ID for the CAN-8423 is set to 1.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	05	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 14 02
d : 05 00 00 00
Because the **n**=3, only the 4th byte is valid. Therefore, the feedback value is 05.

Step 2. The CAN-8423 will reply with the message to finish the data download. Then, users can use the upload methods to read back the value.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

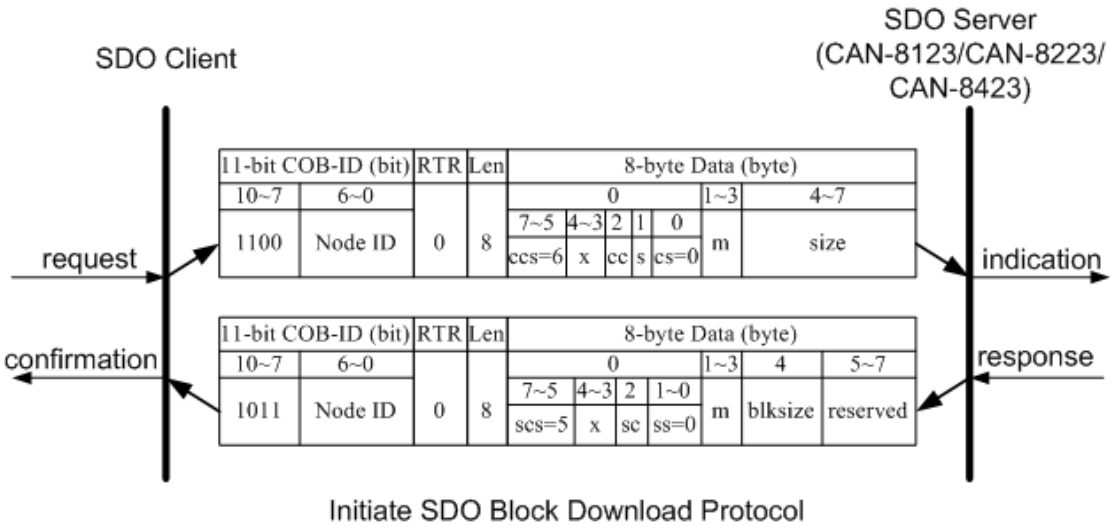
scs : 3
m : 00 14 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1.4 SDO Block Download

The procedure of SDO Block Download is similar to the SDO Block Upload. There are three steps during the SDO Block Download. The Initiate SDO Block Download protocol is the beginning protocol for SDO Block Download. In this protocol, the SDO server and SDO client will mutually communicate. Afterwards, the SDO Block Download protocol will also be used. And, data will be sent to SDO server by SDO client. After finishing the data transmission, the client and server will use the End SDO Block protocol to terminate the SDO Block Download. The following figures are the structures for the three protocols.

Initiate SDO Block Download Protocol



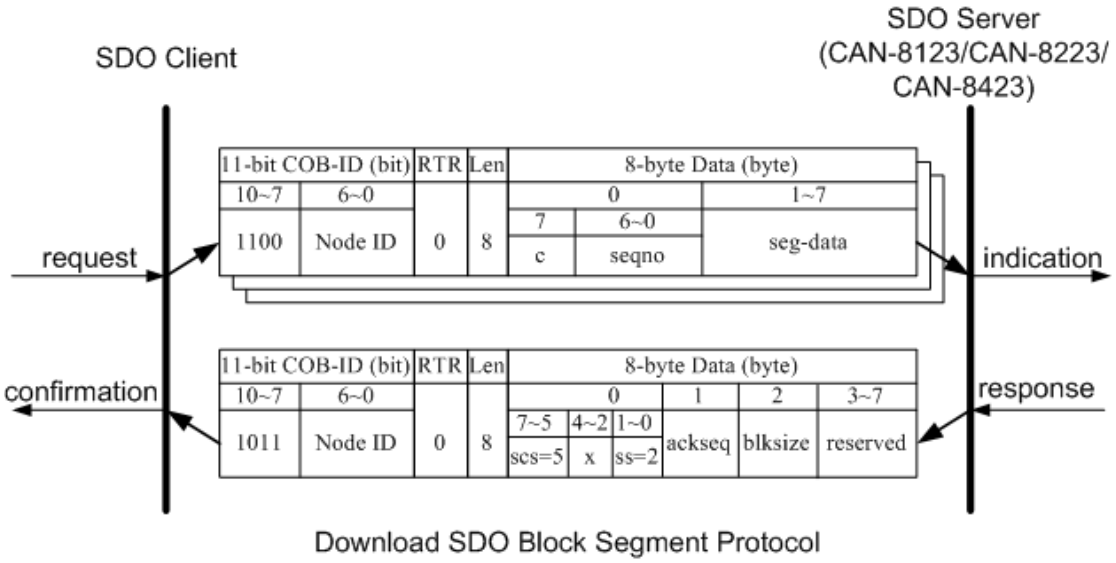
Initiate SDO Block Download Protocol

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ccs	:	client command specified 6: block download
scs	:	server command specified 5: block download
s	:	size indicator 0: Data set size is not indicated. 1: Data set size is indicated.
cs	:	client subcommand 0: initiate download request
ss	:	server subcommand 0: initiate download response
cc	:	client CRC support **cc**=0: Client does not support generating CRC on data. **cc**=1: Client supports generating CRC on data.
sc	:	server CRC support **sc**=0: Server does not support generating CRC on data. **sc**=1: Server supports generating CRC on data.
m	:	multiplexor It represents the index/sub-index of the data to be transfer by the SDO.
size	:	download size in byes **s**=0: Size is reserved for further use, always 0. **s**=1: Size contains the number of bytes to be downloaded. Byte 4 contains the LSB and byte 7 is the MSB.
blksize	:	number of segments per block with 0 < **blksize** < 128
x	:	not used, always 0
reserved	:	reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Download SDO Block Segment Protocol



Download SDO Block Segment Protocol

scs : server command specified
5: block download

ss : server subcommand
0: initiate download response

c : It indicates whether there are still more segments to be downloaded.
0: more segments to be downloaded
1: no more segments to be downloaded , enter ‘End block download’ phase

seqno : sequence number of segment, 0 < **seqno** < 128

seg-data : It is at most 7 bytes of segment data to be downloaded.

ackseq : sequence number of last segment that was received successfully during the last block download
If **ackseq** is set to 0, the server indicates the client that the segment with the sequence number 1 was not received correctly and all segments have to be retransmitted by the client.

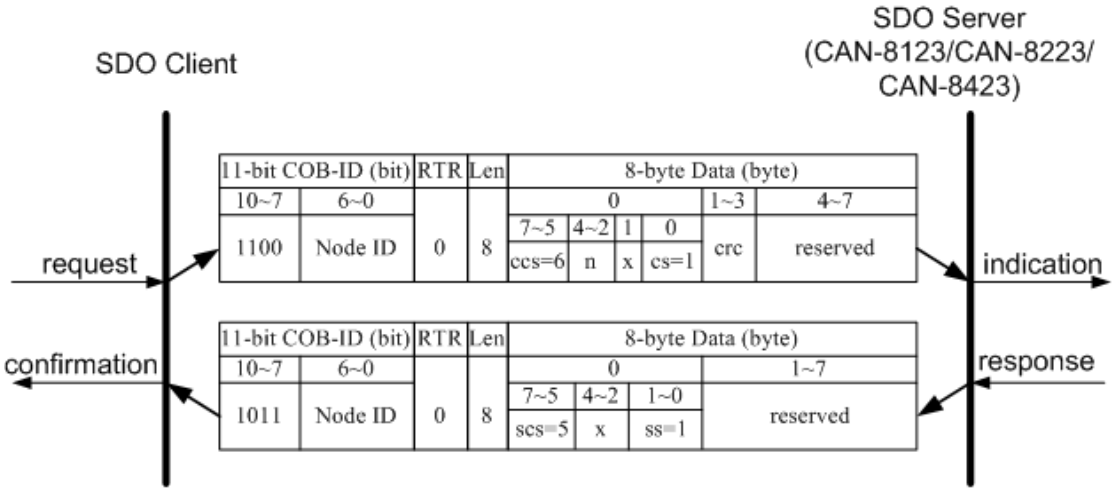
blksize : number of segments per block that has to be used by client for the following block download with 0 < **blksize** < 128

x : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

End SDO Block Download Protocol



End SDO Block Download Protocol

ccs : client command specified.
6: block download

scs : server command specified.
5: block download

cs : client subcommand
1: end block download request

ss : server subcommand
1: end block download response

n : It indicates the number of bytes in the last segment of the last block that do not contain data. Bytes [8-**n,**7] do not contain segment data.

crc : 16 bit Cyclic Redundancy Checksum (CRC) for the whole data set.
The algorithm for generating the CRC is as follows.

x^16+x^12+x^5+1

CRC is only valid if in Initiate Block Download cc and sc are set to 1. Otherwise, crc has to be set to 0. For CAN-8123/CAN-8223/CAN-8423, it is not support CRC check mechanism.

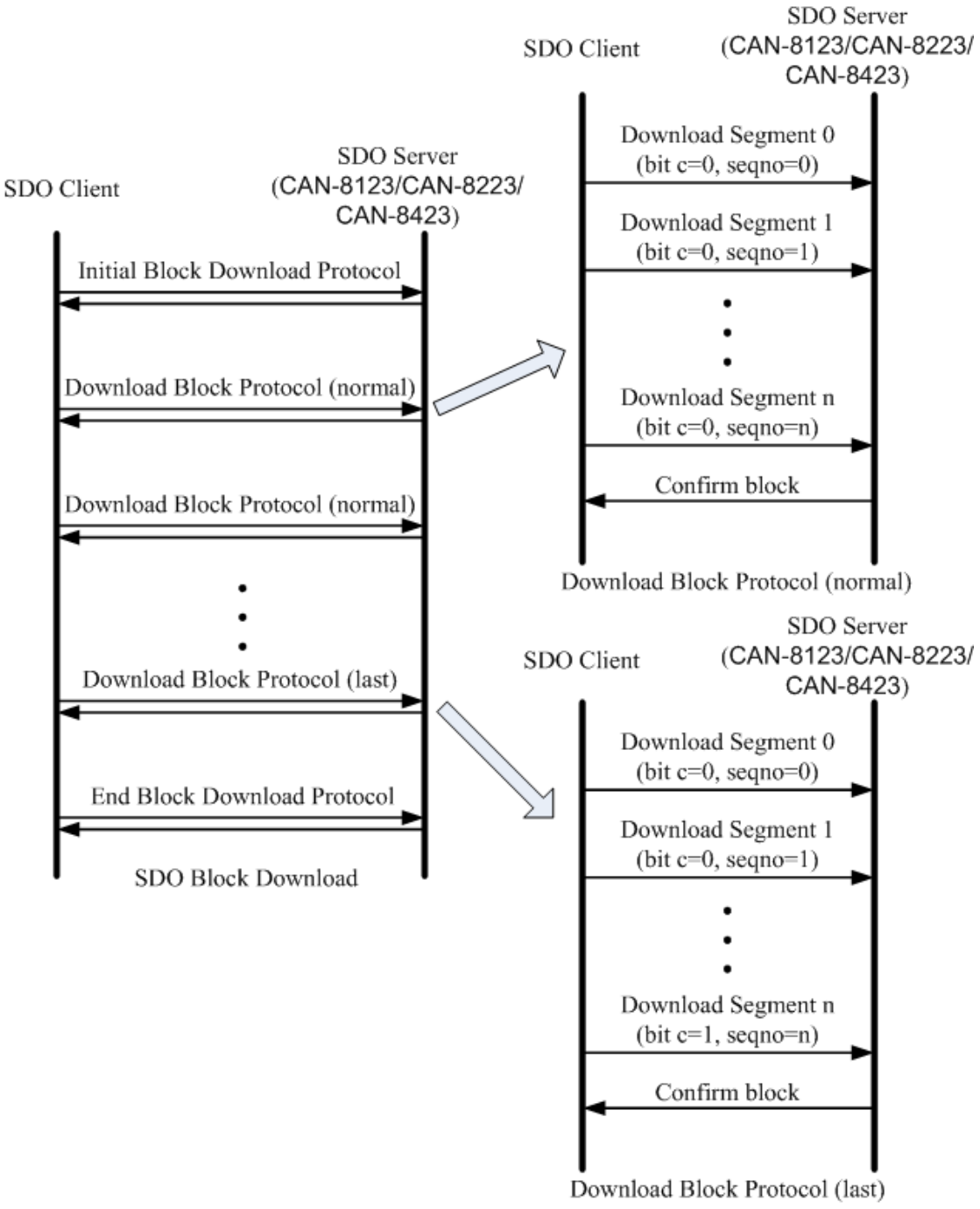
X : not used, always 0

reserved : reserved for further use , always 0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SDO Block Download Example

In this example, the value of the object entry with index 0x1400 and sub-index 0x02 will be changed to 5 by using the SDO Block Download communication method. When the SDO Block Download is actuated, the procedure will be as follows.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 1. When the Initiate SDO Block Download protocol is carried out, the CAN-8423 will be informed with the value of the object entry with index 0x1400 and sub-index 02 modified by the method of the SDO Block Download,

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	C0	00	14	02	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 6
cc : 0
s : 0
cs : 0
m : 00 14 02
size : 00 00 00 00
Because the value of **s** is 0, the **size** is not used.

Step 2. The CAN-8423 will reply to the message by using the Initiate SDO Block Download protocol. Then, the SDO client will download the object‘s data with index 0x1400 and sub-index 02 to CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A0	00	14	02	7F	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 5
sc : 0
s : 0
ss : 0
m : 00 14 02
blksize : 7F

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. The SDO client will transmit the data of the object entry index 0x1400 and sub-index 02 by using the Download SDO Block Segment protocol. The following description shows that the data length of the value is less than the maximum data length of one block, the SDO Block Segment Download protocol is just implemented once.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	81	05	00	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

c : 1
seqno : 1
seg-data : 05 00 00 00 00 00 00
Because this segment is the last one, not all of the data in the **seg-data** filed is useful. The valid data length will be indicated when the users send a message to finish the Block Download protocol. Please refer to the value of **n** in the step 5.

Step 4. The CAN-8423 will reply to the message in order to check whether the transmission is successful or not. If not, this block transmission will be requested again. After finishing the data transmission, the Download SDO Block Segment protocol will be terminated.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A2	01	7F	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 5
ss : 2
ackseq : 01
blksize : 7F

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 5. The SDO client will send the ending message to finish the SDO Block Download.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	D9	00	00	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 6

n : 6

This value means the useless data in the last segment are from [8-6] to 7, i.e. only the first 2 bytes are valid.

cs : 1

crc : 00 00

Step 6. The CAN-8423 will reply to the message, and terminate the End SDO Block Download protocol.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	A1	00	00	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

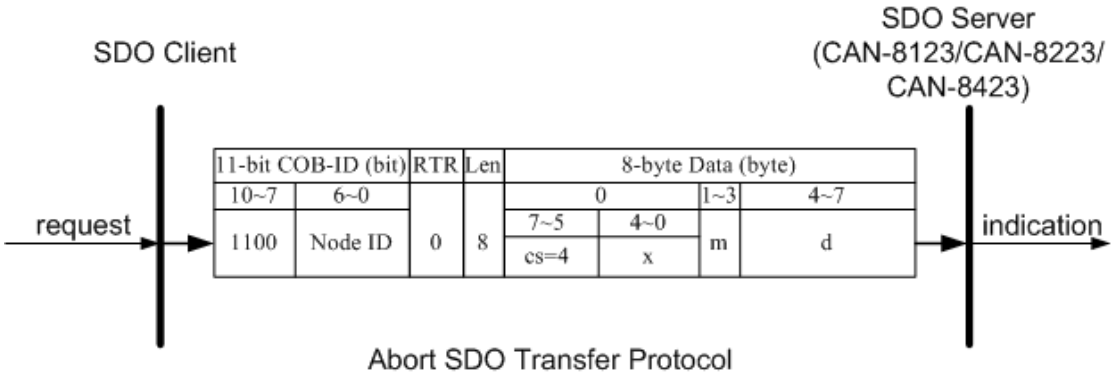
scs : 5

ss : 1

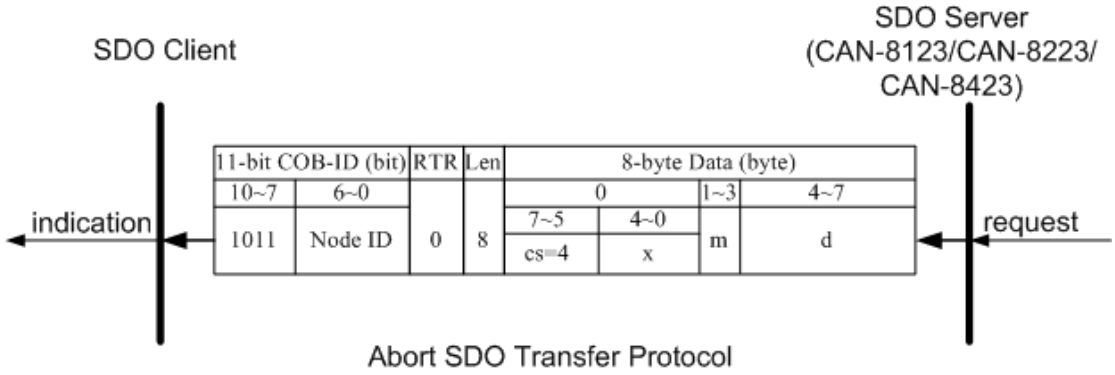
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.1.5 Abort SDO Transfer Protocol

In some conditions, the SDO client or SDO server will terminate the SDO transmission. For example, the value of entries that users want to modify does not exist or is read-only, even users wouldn't continue the uncompleted SDO protocol under some special situations. When these conditions occur, both the client and the server can be activated to send the Abort SDO Transfer message. The Abort SDO Transfer protocol is shown below.



Abort SDO Transfer Protocol



Abort SDO Transfer Protocol

cs : command specified
4: abort transfer request

x : not used, always 0

m : multiplexer
It represents index and sub-index of the SDO

d : contains a 4-byte "Abort Code" about the reason for the abort

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Abort Code	Description
0503 0000h	Toggle bit not alternated.
0504 0000h	SDO protocol timed out.
0504 0001h	Client/server command specified not valid or unknown.
0504 0002h	Invalid block size (block mode only).
0504 0003h	Invalid sequence number (block mode only).
0504 0004h	CRC error (block mode only).
0504 0005h	Out of memory.
0601 0000h	Unsupported access to an object.
0601 0001h	Attempt to read a write only object.
0601 0002h	Attempt to write a read only object.
0602 0000h	Object does not exist in the object dictionary.
0604 0041h	Object cannot be mapped to the PDO.
0604 0042h	The number and length of the objects to be mapped would exceed PDO length.
0604 0043h	General parameter incompatibility reason.
0604 0047h	General internal incompatibility in the device.
0606 0000h	Access failed due to an hardware error.
0607 0010h	Data type does not match, length of service parameter does not match
0607 0012h	Data type does not match, length of service parameter too high
0607 0013h	Data type does not match, length of service parameter too low
0609 0011h	Sub-index does not exist.
0609 0030h	Value range of parameter exceeded (only for write access).
0609 0031h	Value of parameter written too high.
0609 0032h	Value of parameter written too low.
0609 0036h	Maximum value is less than minimum value.
0800 0000h	General error.
0800 0020h	Data cannot be transferred or stored to the application.
0800 0021h	Data cannot be transferred or stored to the application because of local control.
0800 0022h	Data cannot be transferred or stored to the application because of the present device state.
0800 0023h	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of an file error).

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Abort SDO Transfer Example

The object index 0x1008 doesn't support the sub-index 01 entry. Therefore, if users read the object entry with index 0x1008 and sub-index 01, the CAN-8x23 will reply the Abort SDO Transfer message. The example is figured as follows.

Step 1. The Rx SDO message will be sent to the CAN-8423 in order to get the object entry with index 0x1008 and sub-index 01. The following example is assumed that the node ID for the CAN-8423 is set to 1.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	08	10	01	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 2
m : 08 10 01

Step 2. The CAN-8423 will reply to the Abort SDO message as shown below.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	80	08	10	01	11	00	09	06

SDO client ⟵ **SDO server (CAN-8x23)**

cs : 4
m : 08 10 01
d : 11 00 09 06
According to the low byte data have the transferring priority, the data will be converted to "06 09 00 11". Therefore, after searching the Abort Code table described above, this Abort Code can be interpreted as "Sub-index does not exist".

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.2 PDO Communication Set

5.2.1 PDO COB-ID Parameters

Before the real-time data are transmitted by the PDO, it is necessary to check the COB-ID parameter of this PDO in the PDO communication objects. This parameter setting controls the COB-ID of the PDO communication, which is in 32 bits, and each bit with its meaning is given in the table follow.

Bit Number	Value	Meaning
31 (MSB)	0	PDO exits (PDO is valid)
	1	PDO does not exist (PDO is not valid)
30	0	RTR allowed on this PDO
	1	No RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	If bit 29=0
	x	If bit 29=1: 28-11 bits of 29-bit COB-ID
10-0 (LSB)	x	10-0 bits of COB-ID

Note: Only CAN-8123/CAN-8223/CAN-8423 supports CAN 2.0A.

In the following table, it's regarding the default PDO COB-ID parameters.

Number of PDO	Default COB-ID of PDO	
	Bit10~Bit7 (Function Code)	Bit6~Bit0
TxPDO1	0011	Node ID
TxPDO2	0101	Node ID
TxPDO3	0111	Node ID
TxPDO4	1001	Node ID
RxPDO1	0100	Node ID
RxPDO2	0110	Node ID
RxPDO3	1000	Node ID
RxPDO4	1010	Node ID

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Note: 1. Users can also define the PDO COB-ID by themselves. Actually, all COB-ID can be defined by users except the reserved COB-ID described in the table of the section 3.1. It is important to avoid the conflict with the defined COB-ID used in the same node.

2. The PDO COB-ID parameters cannot be changed if the PDO is valid (bit 31 =0).

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.2.2 Transmission Type

The transmission type is one of the several parameters defined in PDO communication objects with sub-index 02. Each PDO has its own transmission type. The transmission type can indicate the transmission or reception character for its corresponding PDO. The following table describes the relationship between the value of the transmission type and the PDO character. For example, if users used transmission type 0 for the first TxPDO, the CANopen device will follow the rule of the acyclic and synchronous PDO transmission.

Transmission Type	PDO Transmission method				
	cyclic	**acyclic**	**synchronous**	**asynchronous**	**RTR only**
0		**O**	**O**		
1-240	**O**		**O**		
241-251	----------**reversed**----------				
252			**O**		**O**
253				**O**	**O**
254				**O**	
255				**O**	

Note:

1. The transmission type 1-240 indicates how many SYNC objects the TxPDO will be triggered. The RxPDO is always triggered by the following SYNC upon reception of data independent of the transmission types 0-240.
2. The transmission type 252 and 253 are only used for TxPDO. The transmission type 252 means that the data is updated (but not sent) immediately after reception of the SYNC object. For these two transmission types, the PDO is only transmitted on remote transmission requests.
3. For the transmission types 254 and 255, the event timer will be used in the TxPDO. The PDO, including the DI value, will be sent when the DI value is changed. And both transmission types will directly trigger an update of the mapped data when receiving the RxPDO.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

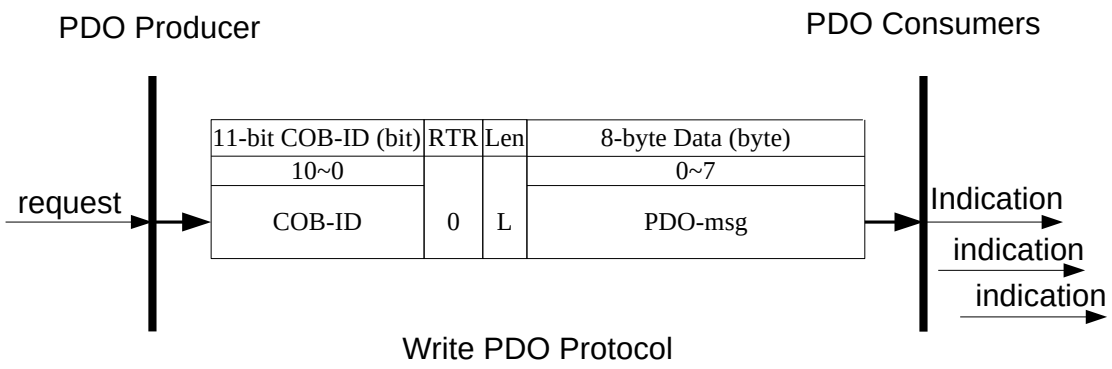
5.2.3 PDO Communication Rule

The PDO related objects are indicated from index 0x1400 to 0x1BFF. For the CAN-8x23, RxPDO communication objects are from index 0x1400 to index 0x140F, and RxPDO mapping objects are from index 0x1600 to index 0x160F. The ranges of the TxPDO communication objects and the mapping objects are from index 0x1800 to index 0x180F and from index 0x1A00 to index 0x1A0F respectively. Moreover, each PDO communication object has its own PDO mapping object.

For example, the first RxPDO communication object is stored in the entry with index 0x1400, and the corresponding mapping object is stored in an entry with index 0x1600. The object with index 0x1401 and the object with index 0x1601 are a group, and so on. The TxPDO also follows the same rules. The first TxPDO communication object is stored in the entry with 0x1800, and the corresponding mapping object is in the 0x1A00 entry, and so on. Therefore, before users access the practical I/O channels via PDO communication, each parameter for the PDO communications and mapping objects must be controlled.

Besides, only PDO communications can be used in the NMT operational state. Users can use the NMT module control protocol to change the NMT state of the CAN-8x23. It is described in the section 5.3. Besides, during communication via the PDO messages, the data length of the PDO message must match with the PDO mapping object. If the data length ‘L’ of the PDO message exceeds the total bytes ‘n’ of the PDO mapping object entries, only the first 'n' bytes of the PDO message are used by the PDO consumer. If ‘L’ is less than 'n', the PDO message will not be disposed by the PDO consumer, and an Emergency message with error code 8210h will be transmitted to the PDO producer. The PDO communication set is shown as follows.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

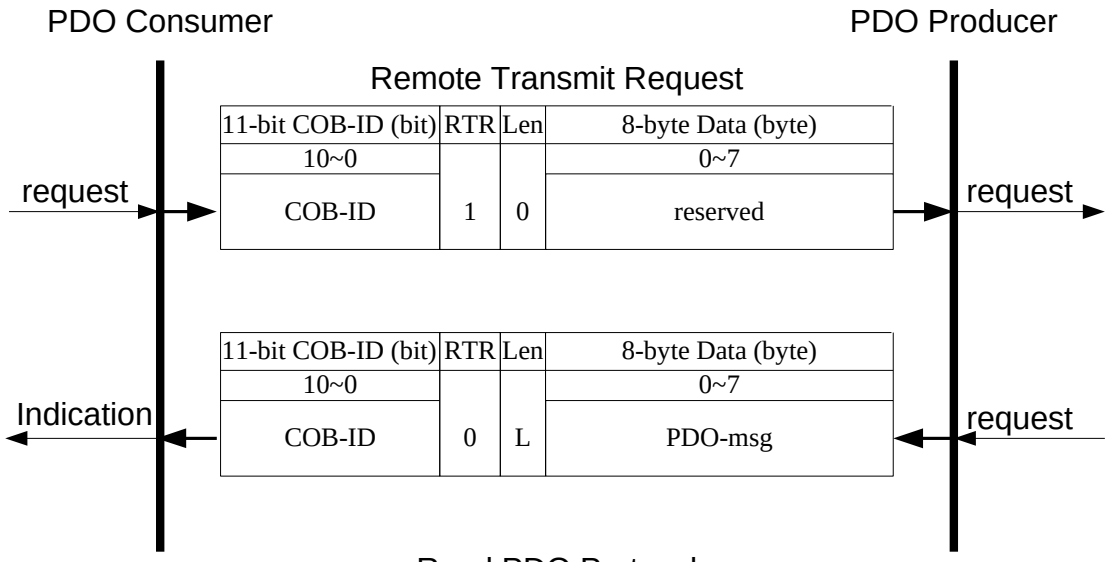


Write PDO Protocol

COB-ID : the default PDO COB-ID, or the PDO COB-ID can be defined by user

L : the data length about how many bytes the PDO message has

PDO-msg : the real-time data or the data which can be mapped into the PDO mapping objects



Read PDO Protocol

COB-ID : the default PDO COB-ID, or the PDO COB-ID defined by users

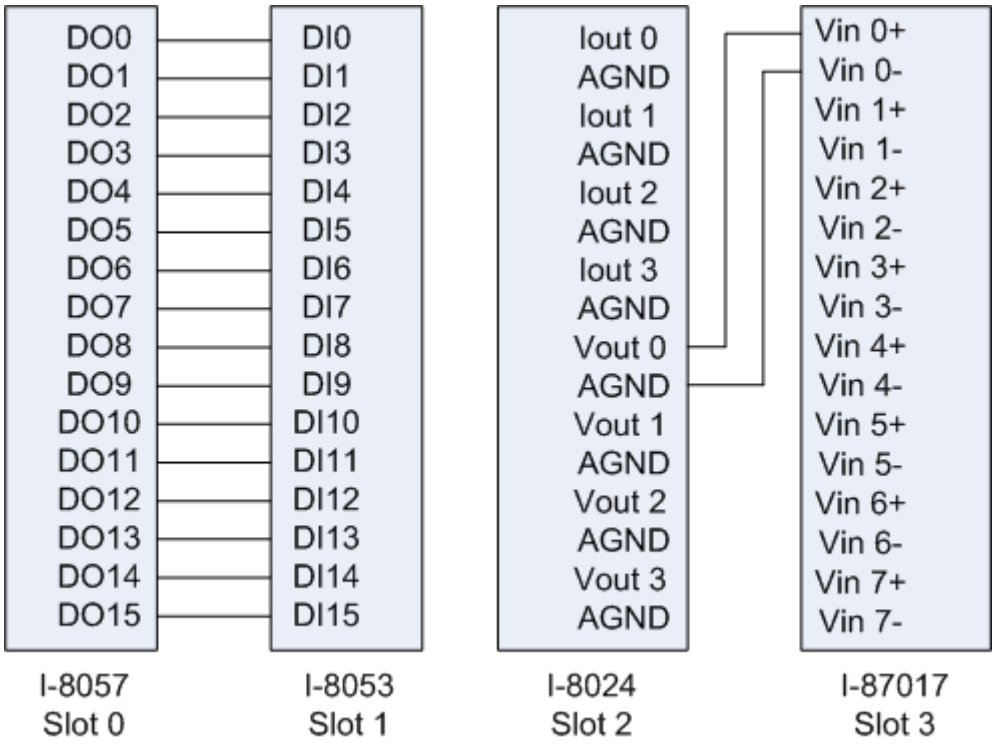
L : the data length about how many bytes the PDO message has

PDO-msg : the real-time data or the data which can be mapped into the PDO mapping objects

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

PDO Communication Example

To take a look at a PDO communication demo, some I-8000 slot modules will be needed. They are I-8057, I-8053, I-8024 and I-87017. Connect each I/O channels for these modules as following figure.



Please use the CAN-8423 rotary switch to set the node ID to 1, and CAN bus baud rate to 125Kbps. Moreover, use CAN Slave Utility to set the I-8024 and I-87017 input/output range to -10V~+10V. When using the CAN Slave Utility, the following information can be as a reference. (Note: CAN-8123/CAN-8223 can't be used with the on-line mode to set the channel input/output range. Therefore, users have to refer to the section 5.5 to know how to use the SDO protocol to set the channel input/output range)

PDO NO.	COB-ID (Hex)	Transmission	Inhibit Time	Event Timer	Mapping 0	Mapping 1	Mapping 2
1	201	255	0	0	s 0c00~07	s 0c08~15	s--c--
2	301	255	0	0	s 2c 0	s 2c 0	s 2c 1
3	401	255	0	0	s--c--	s--c--	s--c--
4	501	255	0	0	s--c--	s--c--	s--c--

Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6	Mapping 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s 2c 1	s 2c 1	s 2c 2	s 2c 2	s 2c 3	s 2c 3
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--

RxPDO Information

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

PDO NO.	COB-ID (Hex)	Transmission	Inhibit Time	Event Timer	Mapping 0	Mapping 1	Mapping 2
1	181	255	0	0	s 1c00~07	s 1c08~15	s--c--
2	281	255	0	0	s 3c 0	s 3c 0	s 3c 1
3	381	255	0	0	s 3c 4	s 3c 4	s 3c 5
4	481	255	0	0	s--c--	s--c--	s--c--

Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6	Mapping 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--
s 3c 1	s 3c 1	s 3c 2	s 3c 2	s 3c 3	s 3c 3
s 3c 5	s 3c 5	s 3c 6	s 3c 6	s 3c 7	s 3c 7
s--c--	s--c--	s--c--	s--c--	s--c--	s--c--

TxPDO Information

Index	0x6000	0x6200	0x6206	0x6207
Description	Read DI	Write DO	DO Err Mode	DO Err Value
Sub-Index 0	2	2	2	2
Sub-Index 1	8053_DI 0~ DI 7	8057_DO 0~ DO 7	FF	0
Sub-Index 2	8053_DI 8~ DI F	8057_DO 8~ DO F	FF	0

Index	0x6401	0x6411	0x6443	0x6444
Description	Read AI	Write AO	AO Err Mode	AO Err Value
Sub-Index 0	8	4	4	4
Sub-Index 1	87017_AI 0	8024_AO 0	1	0 V
Sub-Index 2	87017_AI 1	8024_AO 1	1	0 V
Sub-Index 3	87017_AI 2	8024_AO 2	1	0 V
Sub-Index 4	87017_AI 3	8024_AO 3	1	0 V
Sub-Index 5	87017_AI 4			
Sub-Index 6	87017_AI 5			
Sub-Index 7	87017_AI 6			
Sub-Index 8	87017_AI 7			

Standardized Device Profile Area Information

After concluding the above preparations, the several functions of PDO communication will be introduced as follows.

- The function of accessing digital I/O & analog I/O with asynchronous PDO.
- The function by using Event Timer to obtain the input value.
- The function of the acyclic and synchronous RxPDO.
- The function of the acyclic and synchronous TxPDO.
- The function of the cyclic and synchronous TxPDO.
- The function of the synchronous and RTR-only TxPDO.
- The function of the asynchronous and RTR-only RxPDO.
- The function of the dynamic PDO mapping for DI/AI/DO/AO channels

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Before describing the example, the step0 must be checked. And the default COB-ID for each communication object is assumed to be being used.

Step0: The following message must be sent in order to change the NMT state of the CAN-8423 first, because only the PDO communication can run under the NMT Operational state.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	01	00	00	00	00	00	00

NMT master → **NMT slave (CAN-8x23)**

cs : 1

Node ID : 1

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Access Digital I/O & Analog I/O**

Step 1. In order to change the DO value of the I-8057 to be 0x1234, users must send the PDO message by using the first RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	34	12	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201

L : 8

PDO-msg : 34 12 00 00 00 00

Only the first two bytes are valid, even if the **L** is set to 8, because the data in the first RxPDO contains only two bytes. According to the PDO mapping table shown above, the first byte is for the DO0~DO7 channel values of the I-8057. The second byte is for the DO8~DO15 channel values of the I-8057.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 2. Because of the change of the DI-channel status, the TxPDO is transmitted automatically when the transmission type is 255, based on the CANopen spec 401. Then, users will receive the 1st TxPDO message.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	34	12	00	00	00	00	00	00

PDO consumer ← **PDO consumer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : 34 12 00 00 00 00 00 00

Because the data length is 2, only the first two bytes are valid. The DI value will be 1 if the DI is OFF, according to the character of the I-8053 DI channels. Therefore, the first byte indicates that the DI2, DI4, and DI5 of the I-8053 are in ON state. The second byte shows that the DI9 and DI12 of the I-8053 are in ON state.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. In order to output 5V to the AO0 of the I-8024, users must send the PDO message by using the 2nd RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	1	0	0	0	0	0	0	0	1	0	8	FF	3F	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x301

L : 8

PDO-msg : FF 3F 00 00 00 00 00 00

The first two bytes are for AO channel 0, and the others are for AO channel 1, 2 and 3. Users need to transfer the float value to hex format because only the CAN-8123/CAN-8223/CAN-8423 supports the hex format. The output range of the I-8024 is -10V~10V. According to the transformation table stored in the appendix table, the mapping hex-format range is from 0x8000 (-32768) to 0x7FFF (32767). Therefore, the 5V is mapped to the 0x3FFF by applying following equation.

$$HexValue = \left(\frac{5V - (-10V)}{10V - (-10V)} \right) * \left(32767 - (-32768)\right) + (-32768)$$
$$= 16383.25 \approx 16383 = 0x3FFF$$

The first two bytes of the PDO message will be filled with “FF” and “3F”. All the other AI channels are set to 0V. Then, the value “00 00” will be given for these channels. For more details about how to transfer the value between the hex and float, please refer to the section 6.3.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 4. Even the AI input value has been changed according the AO value, the RxPDO will not respond automatically in the CAN-8423. Therefore, users need to use the RTR message from the 2nd TxPDO to read back the AI value.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer → **PDO consumer (CAN-8x23)**

COB-ID : 0x281

Step 5. The feedback value for AI is 5V.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FD	FF	FD	FF	FD	FF

PDO consumer ← **PDO consumer (CAN-8x23)**

COB-ID : 0x281

L : 8

PDO-msg : 00 40 FD FF FD FF FD FF

The first two bytes are for AI channel 0. The others are for AI channel 1, 2, and 3. The feedback AI0 value is 0x4000. All the other AI channels are 0xFFFD. Users need to transfer this AI0 value to float. The I-87017's input float range is set to -10V ~ +10V and the input hex range is from 0x8000 (-32768) to 0x7FFF (32767). The value 0x4000 (16384) can be transferred by using the following equation.

$$FloatValue = \left(\frac{16384-(-32768)}{32767-(-32768)}\right) * (10V-(-10V)) + (-10V)$$

$$\approx 5.001V$$

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Event Timer Functionality**

Step 6. Users can use the SDO to change the event timer of the 2nd RxPDO to 1000, stored in index 0x1801 with sub-index 5. In addition, the value 1000 means 1 second according to the event timer is ms,

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	00	18	05	E8	03	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 2
e : 1
s : 1
m : 00 18 05
d : E8 03 00 00
The value 0x03E8 is equal to 1000. Because the **n**=2, the last two bytes “00 00” is useless.

Step 7. The CAN-8423 will response the message to finish the data download.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	05	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 00 18 05

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 8. After changing the value of the event timer, the AI value will be automatically transmitted per second. The example below shows that at the first time the 2n TxPDO message is received.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FD	FF	FF	FF	FF	FF

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x281
L : 8
PDO-msg : 00 40 FD FF FF FF FF FF

Step 9. The following example shows that at the second time the 2nd TxPDO message is received.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	06	40	FF	FF	FF	FF	FF	FF

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x281
L : 8
PDO-msg : 06 40 FF FF FF FF FF FF

The value of 0x4006 is equal to 5.002V. The AI value is changed because of the noise disturbance or other factors.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 10. It shows that at the third time for the 2nd TxPDO message is received.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	00	40	FF	FF	FD	FF	FF	FF

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x281
L : 8
PDO-msg : 00 40 FF FF FD FF FF FF

Step 11. Users can set the event timer to 0 to finish the event timer test.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	00	18	05	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 2
e : 1
s : 1
m : 00 18 05
d : 00 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	05	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 00 18 05

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Transmission Type 0 for the first RxPDO**

Step 12. Users can set the transmission type of the first RxPDO to 0.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 14 02
d : 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 00 14 02

Step 13. Change the DO value of the I-8057 to be 0x5678 by using the 1st RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	78	56	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201
L : 8
PDO-msg : 78 56 00 00 00 00 00 00

Step 14. The DO value isn't changed immediately according to the character of the transmission type 0. Meanwhile, the SYNC message is needed to trigger the action of the 1st RxPDO.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0	00	00	00	00	00	00	00	00

SYNC producer → **SYNC consumer (CAN-8x23)**

COB-ID : 0x80

The message of the SYNC object is always fixed as the format described above. The COB-ID of the SYNC object can be changed arbitrarily. It complies with the producer/consumer relationship.

Step 15. After transmitting the SYNC object, the 1st RxPDO is triggered. The DI value is also changed at the same time. Hence, users can receive the 1st TxPDO from CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	78	56	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : 78 56 00 00 00 00 00 00

Step 16. Users can set the transmission type of the first RxPDO to 255 to finish the test.

11-bit COB-ID (bit)		RTR	Data Length	8-byte Data (byte)
Func Code	Node ID			

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	14	02	FF	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 14 02
d : FF 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	14	02	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 00 14 02

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Transmission Type 0 for the first TxPDO**

Step 17. Users can set the transmission type of the first TxPDO to 0.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 18 02
d : 00 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO server (CAN-8x23) → **SDO client**

scs : 3
m : 00 18 02

Step 18. Users can change the DO value of the I-8057 to be 0x90AB by using

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

the first RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	AB	90	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201
L : 8
PDO-msg : AB 90 00 00 00 00 00 00

Step 19. The first TxPDO will not be transmitted immediately even if the DI value is changed according to the character of the transmission type 0. In addition, the SYNC message is needed to trigger the action of the first TxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0	00	00	00	00	00	00	00	00

SYNC producer → **SYNC consumer (CAN-8x23)**

COB-ID : 0x80

Step 20. After transmitting the SYNC object, the 1st TxPDO is triggered, and users can receive the 1st TxPDO from CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	90	AB	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181
L : 2
PDO-msg : 90 AB 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 21. Users can send the SYNC message again.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0	00	00	00	00	00	00	00	00

SYNC producer → **SYNC consumer (CAN-8x23)**

SYNC COB-ID : 0x80

Step 22. Nothing happened because the DI values were not changed. This is the main difference between transmission type 0 and 1. (Under the transmission type 1, the TxPDO is always transmitted no matter whether the DI values are changed or not, when the CAN-8423 receives the SYNC object.)

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Transmission Type 3 for the first TxPDO**

Step 23. Users can set the transmission type of the first TxPDO to 3.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	03	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 18 02
d : 03 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x423)**

scs : 3
m : 00 18 02

Step 24. Users can change the DO value of the I-8057 to be 0xCDEF by using the first RxPDO.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	EF	CD	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201

L : 8

PDO-msg : EF CD 00 00 00 00 00 00

Step 25. The SYNC message has to be transmited in 3 times according to the character of transmission type 3.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0	00	00	00	00	00	00	00	00

SYNC producer → **SYNC consumer (CAN-8x23)**

COB-ID : 0x80

Step 26. After finishing the transmission of the three SYNC objects, the first TxPDO will be triggered, and users will receive the first TxPDO from CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	EF	CD	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : EF CD 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Transmission Type 252 for the first TxPDO**

Step 27. Users can set the transmission type of the first TxPDO to 252.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FC	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 18 02
d : FC 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 00 18 02

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 28. Users can change the DO value of the I-8057 to be 0x1234 by using the first RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	34	12	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201
L : 8
PDO-msg : 34 12 00 00 00 00 00 00

Step 29. The fist TxPDO will not be transmitted immediately according to the transmission type 252. Meanwhile, it will send the RTR message of the first TxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer → **PDO producer (CAN-8x23)**

COB-ID : 0x181

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 30. The feedback DI values are out-of-date. (Users can see the LEDs status on the I-8053 to confirm the practical DI values).

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	34	12	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181
L : 2
PDO-msg : 34 12 00 00 00 00 00 00

Step 31. Transmit a SYNC message.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	0	0	0	00	00	00	00	00	00	00	00

SYNC producer → **SYNC consumer (CAN-8x23)**

COB-ID : 0x80

Step 32. Users can send the RTR message of the first TxPDO again.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer → **PDO producer (CAN-8x23)**

COB-ID : 0x181

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 33. The feedback DI values will be the real DI values.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	34	12	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : 34 12 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Transmission Type 253 for the first TxPDO**

Step 34. Users can set the transmission type of the first TxPDO to 253.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FD	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 18 02
d : FD 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 00 18 02

Step 35. Users can change the DO value of the I-8057 to be 0x5678 by using the first RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	8	78	56	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x201
L : 8
PDO-msg : 78 56 00 00 00 00 00 00

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 36. According to the transmission type 253, only the first TxPDO can be transmitted when receiving the RTR message. So, users can send the RTR message to get DI values. Then, the CAN-8423 will reply with the I-8053 digital input status.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer ⟵ **PDO producer (CAN-8x23)**

COB-ID : 0x181

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	78	56	00	00	00	00	00	00

PDO consumer ⟵ **PDO producer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : 78 56 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 37. Set the transmission type of the 1st TxPDO to 255 to finish the test.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	00	18	02	FF	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 00 18 02
d : FF 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	00	18	02	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 00 18 02

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- **Dynamic PDO Mapping for DI/AI/DO/AO Channels**

Step 38. Users can use the 5th TxPDO to create a new PDO communication with PDO COB-ID 0x182, which is useless for the CAN-8423. Before setting the COB-ID of a PDO, users have to check the bit 31 of the COB-ID first. Only the COB-ID with the value 0 on the bit 31 can be changed. So the COB-ID can be configured directly according to the 5th TxPDO is invalid.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	18	01	82	01	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 18 01
d : 82 01 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	18	01	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 05 18 01

Step 39. Users can create a new PDO mapping object for the 5th TxPDO. Before getting the device objects into the index 0x1A05, users have to check the value of the index 0x1A05 with sub-index 00. If the value is not equal to 0, any modification will be rejected. In this case, it is necessary to have the value in 0. Therefore, users have to fill the DI0~DI7 of the I-8053 into the index 0x1A05 with sub-index 01.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	1A	01	08	01	00	60

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 1A 01
d : 08 01 00 60

The value “08 01 00 60” means the mapped object is stored in the index 0x6000 with sub-index 01. It is an 8-bit data unit. Users can check this object in the Standardize object mapping table described above. It is mapped according to the DI0~DI7 of the I-8053.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	1A	01	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 1A 01

Step 40. According to the purposes, users have to fill the DI8~DI15 of the

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-8053 and AI0 of the I-87017 into the index 0x1A05 with sub-index 02 and 03 respectively.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	1A	02	08	02	00	60

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 1A 02
d : 08 02 00 60

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	1	0	1	1	1	0	1	0	8	60	05	1A	02	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 05 1A 02

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	1A	03	10	01	01	64

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 1A 03
d : 10 01 01 64

The value “10 01 01 64” means that the mapped object is stored in the index 0x6401 with sub-index 01. It is a 16-bit data unit. User can check this object in the Standardize object mapping table described above. It is mapped according to AI0 of the I-87017. In CAN-8123/ CAN-8223/CAN-8423, all analog channels are presented by 16-bit value.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	1A	03	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 1A 03

Step 41. In order to use this PDO mapping object normally, the value of the

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

index 0x1A05 with sub-index 00 must be changed to 3. The value 3 means there are 3 objects mapped to the 5th TxPDO. They are the index 0x6000 with sub-index 01, index 0x6000 with sub-index 02, and index 0x6401 with sub-index 01.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	05	1A	00	03	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 05 1A 00
d : 03 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	1A	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 1A 00

Step 42. Users can use the 5th RxPDO to create a new PDO communication

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

with PDO COB-ID 0x202, and create the RxPDO mapping object in the index 0x1605 because the COB-ID 0x202 is not available for the CAN-8423. This procedure is similar to the steps 37 to 40.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	14	01	02	02	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 14 01
d : 02 02 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	14	01	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 14 01

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	16	01	08	01	00	62

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 16 01
d : 08 01 00 62

The value “08 01 00 62” means the mapped object is stored in the index 0x6200 with sub-index 01. It is an 8-bit data unit. Users can check this object in the Standardize object mapping table described above. It is mapped to the DO0~DO7 for I-8057.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	16	01	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 16 01

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	16	02	08	02	00	62

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 16 02
d : 08 02 00 62

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	16	02	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 05 16 02

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	23	05	16	03	10	01	11	64

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 05 16 03
d : 10 01 11 64

The value “10 01 11 64” means the mapped object is stored in the index 0x6411 with sub-index 01. It is a 16-bit data unit. Users can check this object in the Standardize object mapping table described above. It is mapped according to the AO0 of the I-8024.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	16	03	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 05 16 03

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	05	16	00	03	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 05 16 00
d : 03 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	05	16	00	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 05 16 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 43. Transform the DO0~DO15 of I-8057 and AO0 of I-8024 to be 0x90AB and 0V respectively.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	1	0	0	8	AB	90	00	00	00	00	00	00

PDO consumer → **PDO producer (CAN-8x23)**

COB-ID : 0x202

PDO-msg : AB 90 00 00 00 00 00 00

The first two bytes are assigned to the value 0x90AB of the DO0~DO15 of the I-8057. The 3rd and 4th bytes are assigned to the value 0x0000 for the AO0 of the I-8024. Total bytes of this PDO message are 4.

Step 44. Users will receive the 1st TxPDO and 5th TxPDO simultaneously because the DI values have been changed.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0	0	1	0	2	AB	90	00	00	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x181

L : 2

PDO-msg : AB 90 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	1	0	1	0	4	AB	90	FF	FF	00	00	00	00

PDO consumer ← **PDO producer (CAN-8x23)**

COB-ID : 0x185

L : 4

PDO-msg : AB 90 FF 3F 00 00 00 00

The first two bytes are assigned to the value 0x90AB for the DI0~DI15 of the I-8053. The 3rd and 4th bytes are assigned to the value 0xFFFF for the AI0 of the I-87017. After transferring, the input value of the AI0 is -0.001V.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.3 EMCY Communication Set

5.3.1 EMCY COB-ID Parameter

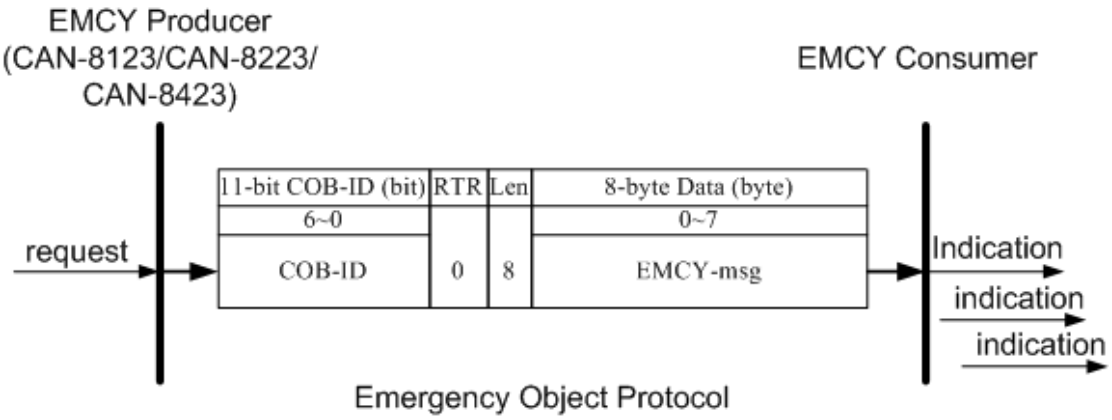
The EMCY COB-ID is similar to the PDO COB-ID. It can be a default value or can be the value defined by users via SDO communication methods. This COB-ID is stored in the object 0x1014, and the data format is shown in the following table. Before using the EMCY mechanism, bit 31 of the EMCY COB-ID needs to be confirmed.

Bit Number	Value	Meaning
31 (MSB)	0	EMCY exits (EMCY is valid)
	1	EMCY does not exist (EMCY is not valid)
30	0	reserved (always 0)
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	If bit 29=0
	x	If bit 29=1: 28-11 bits of 29-bit COB-ID
10-0 (LSB)	x	10-0 bits of COB-ID

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.3.2 EMCY Communication

The EMCY message is triggered when some internal error occurs. After the transmission of one EMCY message, the object with index 0x1003 will record this EMCY event. Therefore, users can track the error's occurrences. The CAN-8x23 supports the maximum of 5 records stored in the index 0x1003 object. The sub-index 1 of this object will store the last EMCY event, and sub-index 5 will record the most previous EMCY event. The EMCY communication set is given below.



COB-ID : the EMCY COB-ID
The EMCY COB-ID can be defined by users. This situation is similar to the PDO COB-ID. The default value is 4-bit function code "0001" with 7-bit node ID.

EMCY-msg : record the type or the class of the occurrence error

The data format of the emergency object data complies with the structure bellows.

Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code		Error register	Manufacturer specific Error Field				

Each bit on the error register is defined as follows. Only the CAN-8x23 supports bit 0, bit 4 and bit 7.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Bit	Meaning
0	generic error
1	current
2	voltage
3	temperature
4	communication error (overrun, error state)
5	device profile specific
6	reserved (always 0)
7	manufacturer specific

The emergency error codes and the error register are specified in the following table.

Emergency Error Code		Error Register	Manufacturer Specific Error Field			Description
High Byte	Low Byte		First Two Byte		Last Three Byte	
00	00	00	00	00	00 00 00	Error Reset or No Error
10	00	81	01	00	00 00 00	CAN Controller Error Occur
50	00	81	02	00	00 00 00	EEPROM Access Error
50	00	81	03	00	00 00 00	COM Port Access Error
81	10	11	04	00	00 00 00	Soft Rx Buffer Overrun
81	10	11	05	00	00 00 00	Soft Tx Buffer Overrun
81	10	11	06	00	00 00 00	CAN Controller Overrun
81	30	11	07	00	00 00 00	Lift Guarding Fails
81	40	11	08	00	00 00 00	Recover from bus off
82	10	11	09	00	00 00 00	PDO Data length Error
FF	00	80	0A	00	00 00 00	Request to reset Node or communication

After producing the EMCY message, the emergency object data will be saved to the object with index 0x1003, and the error register of the emergency object data will be mapped to object 0x1001. Therefore, users can use these two objects to view what happened in the CAN-8x23 and check the error history.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

EMCY Communication Example

Before starting the example, CAN-8423 with I-8057, I-8053, I-8024 and I-87017 slot module are needed. Here, the same hardware configuration shown in the PDO example is used for the EMCY communication.

Step 1. In order to generate the emergency event, it's necessary to send the data to RxPDO1 with data length 1.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	1	00	00	00	00	00	00	00	00

PDO consumer ⟶ **PDO producer (CAN-8x23)**

COB-ID : 0x201
L : 1
PDO-msg : 00

Step 2. Then, the CAN-8423 will reply to an emergency message based on the PDO data length of TxPDO1 doesn't correspond to the value defined in the PDO mapping object.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	1	0	8	10	82	11	09	00	00	00	00

EMCY consumer ⟵ **EMCY producer (CAN-8x23)**

COB-ID : 0x81
EMCY-msg : 10 82 11 09 00 00 00 00

The first two bytes "10 82" are for the emergency error codes. The 3rd byte "11" is for the error register, i.e. the CAN-8423 has either a communication or a generic error. The last five bytes "09 00 00 00 00" are for the manufacturer specific errors. This emergency message means that the data length of TxPDO doesn't correspond to the value defined in the PDO mapping object.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. After recognizing the 0x1003 object with sub-index 01, users will get emergency error codes of the emergency object data recording in this object.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	01	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 2
m 03 10 01

Step 4. The CAN-8423 will reply to the ending message.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	01	10	82	09	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 2
n : 0
e : 1
s : 1
m : 03 10 01
d : 10 82 09 00

Step 5. Users have to check the object 0x1001, and make sure that the communication and generic errors on the error register are indicated.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	01	10	00	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 2
m 01 10 00

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 6. The communication and generic errors on the error register are indicated in the received message.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	01	10	00	11	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 2
n : 3
e : 1
s : 1
m : 01 10 00
d : 11 00 00 00

Step 7. Users can send the data to RxPDO1 with data length 2. Then, the EMCY message containing the error reset information will be received. Because the value of TxPDO is the same with the previous one, the DO channels will not change.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	0	1	0	2	00	00	00	00	00	00	00	00

PDO consumer → **PDO producer (CAN-8x23)**

COB-ID : 0x201
L : 2
PDO-msg : 00 00 00 00 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	0	0	1	0	8	00	00	00	00	00	00	00	00

NMT master ⟵ **NMT slaver (CAN-8x23)**

EMCY-msg : 00 00 00 00 00 00 00 00

The data “00 00 00 00 00 00 00 00” are for the error reset EMCY message, i.e. CAN-8423 has no error now.

Step 9. Users have to check the index 0x1003 with sub-index 01 again. Then, the error reset emergency code should be recorded.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	01	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 2
m 03 10 01

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	01	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 03 10 01
d : 00 00 00 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 10. Users have to check the index 0x1003 with sub-index 02. Then, the received emergency error code had been recorded in the emergency object data.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	03	10	02	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 2
m : 03 10 02

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	43	03	10	02	10	82	09	00

SDO client ← **SDO server (CAN-8x23)**

ccs : 1
n : 0
e : 1
s : 1
m : 03 10 02
d : 10 82 09 00

Step 11. Users have to confirm the error register stored in index 0x1001. The value should be 0 now.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	01	10	00	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 2
m 01 10 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	01	10	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

ccs : 1
n 2
e 1
s 1
m : 01 10 00
d : 00 00 00 00

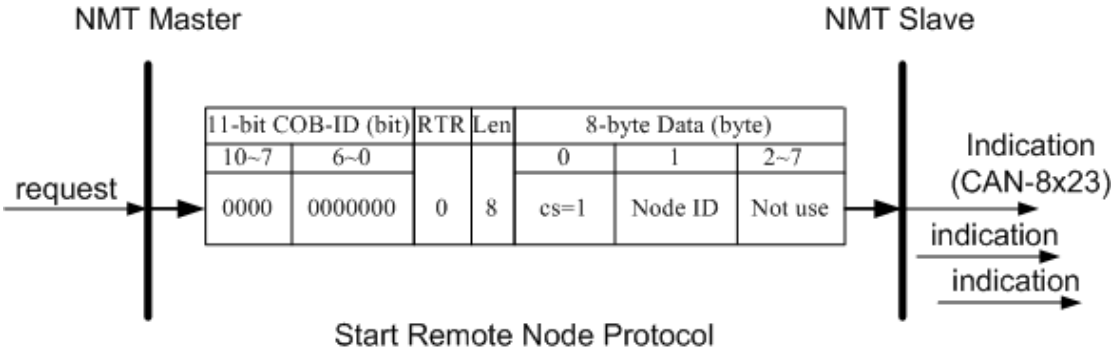
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.4 NMT Communication Set

5.4.1 Module Control Protocol

The NMT communication set can be applied for changing the NMT slave status. The following figure shows how to change the different NMT statuses for the CAN-8x23.

Start Remote Node Protocol

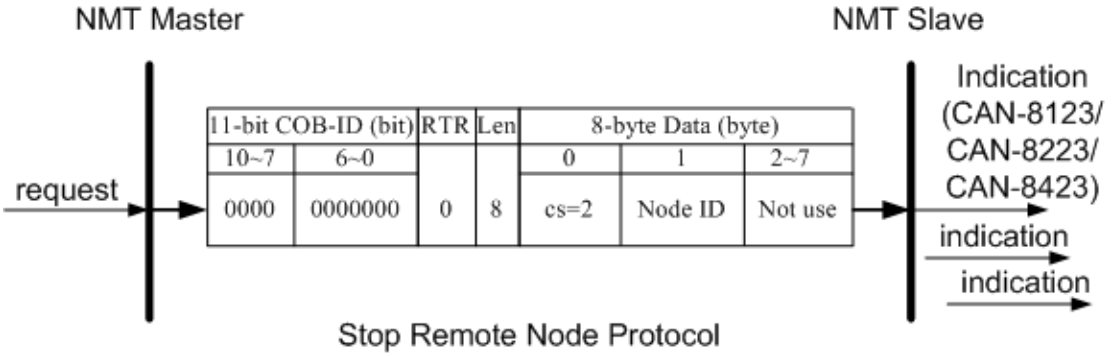


Start Remote Node Protocol

cs : NMT command specified
1: start

Node ID : the node ID of the NMT slave device

Stop Remote Node Protocol



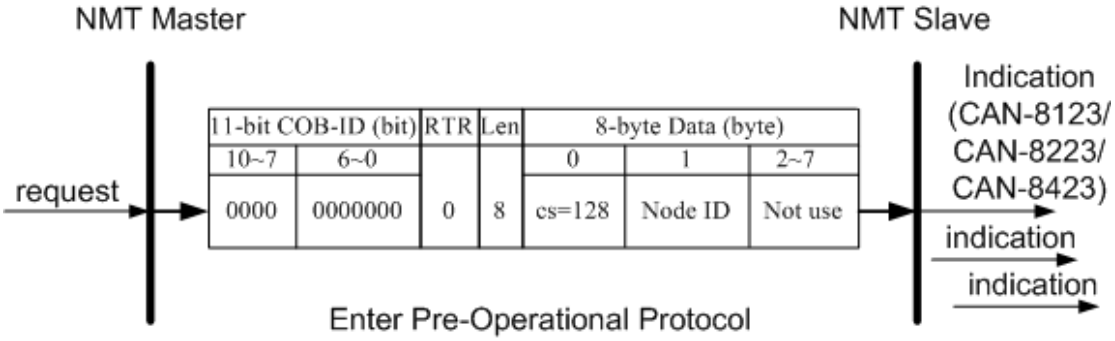
Stop Remote Node Protocol

cs : NMT command specified
2: stop

Node ID : the node ID of the NMT slave device

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

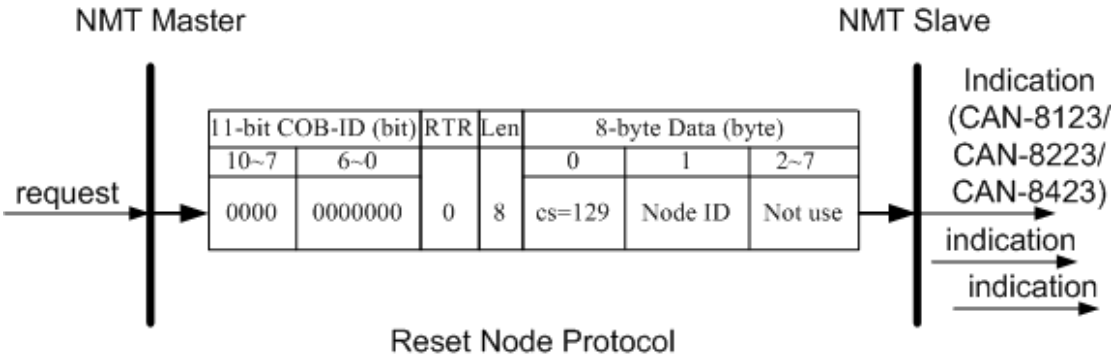
Enter Pre-Operational Protocol



cs : NMT command specified
128: enter PRE-OPERATIONAL

Node ID : the node ID of the NMT slave device

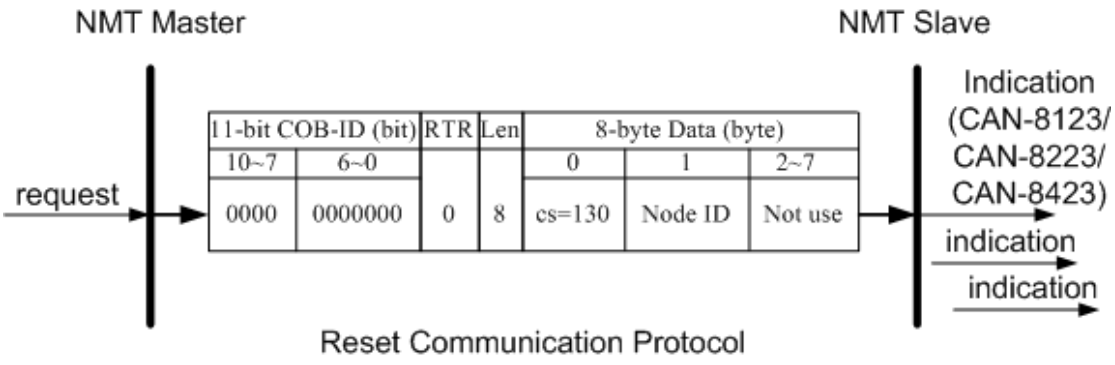
Reset Node Protocol



cs : NMT command specified
129: Reset_Node

Node ID : the node ID of the NMT slave device

Reset Communication Protocol



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

cs : NMT command specified
130: Reset_Communication
Node ID : the node ID of the NMT slave device

Module Control Protocol Example

If the CAN-8423 node ID is set to 5 as an example, the following steps would be…

Step1. Turn off the CAN-8423.

Step2. Then, turn it on. After the initialization, the CAN-8423 will automatically enter the Pre_Operational state. Users will note the RUN LED flashing twice per second.

Step3. Users can send the NMT module control protocol, and control the CAN-8423 to enter the operational state.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	05	00	00	00	00	00	00

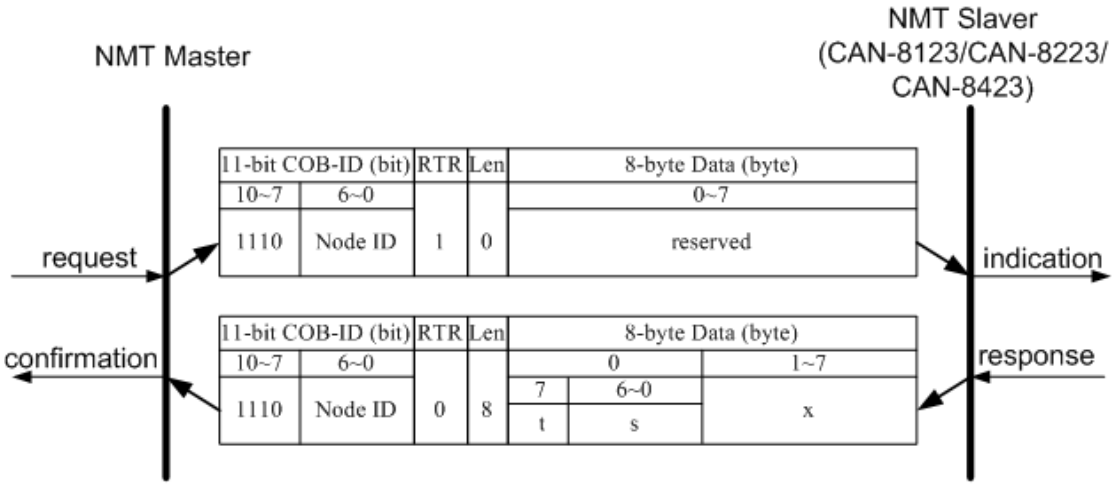
NMT master → **NMT slave (CAN-8x23)**

cs : 1
Node ID : 5

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.4.2 Error Control Protocol

Error Control Protocol is a kind of the solution to check whether the CANopen device is still alive or not. And its related objects include 0x100C and 0x100D. The 0x100C is the guard time, and the 0x100D is the life time factor. The node life time is the guard time multiplied by the life time factor. The Node Guarding timer of the CAN-8x23 will start to count after receiving the first RTR message for the guarding identifier. The communication set of the Error Control protocol is displayed below.



Node Guarding Protocol

t : toggle bit
The value of this bit will be alternatively changed between two consecutive responses from the NMT slave. After the node Guarding protocol becomes active, the value of the toggle-bit of the first response will be 0.

s : the state of the NMT Slave
4: STOPPED
5: OPERATIONAL
127: PRE-OPERATIONAL

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Error Control Protocol Example

The default EMCY function code and the node ID 1 for the CAN-8423 are used as an example on the error control protocol. The steps will be as follows.

Step 1. Turn off the CAN-8423. Then, turn it on. The CAN-8423 will be in the pre_operational state.

Step 2. Users can set the guard time value to 250. This value will be stored in index 0x100C with sub-index 00.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2B	0C	10	00	FA	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 1
n : 2
e : 1
s : 1
m : 0C 10 00
d : FA 00 00 00

Step 3. The CAN-8423 will reply with the ending message.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	0C	10	00	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 3
m : 0C 10 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 4. Users can set the life-time factor value to 4. This value will be stored in the index 0x100D with sub-index 00. Then, the ending message from CAN-8423 will be received.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	0D	10	00	04	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 0D 10 00
d : 04 00 00 00

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	0D	10	00	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 0D 10 00

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 5. Users can send the node guarding protocol to start the mechanism of the node guard. The life time here is equal to 1000 ms (guard time * life time factor =250*4=1000),

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	1	0	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

NMT master ⟶ **NMT slaver (CAN-8x23)**

COB-ID : 0x701

Step 5. Then, users will receive the message, recording the NMT state of the CAN-8423. For the reason that life time is equal to 1000 ms (guard time * life time factor =250*4=1000), users will transmit the node guarding protocol again.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	1	0	0	0	0	0	0	0	1	0	8	7F	00	00	00	00	00	00	00

NMT master ⟵ **NMT slaver (CAN-8x23)**

COB-ID : 0x701

t : 1

s 7F

The value 7F means that the CAN-8423 is in the NMT pre_operational state.

Step 6. Since the life time is equal to 1000 ms (guard time * life time factor =250*4=1000), users will transmit the node guarding protocol again.

Step 7. If the transmission is not available, an error event will be triggered, and an EMCY message for guarding failure will be received. Moreover, all values from the output channels will be changed according to index 0x6206, index 0x6207, index 0x6443, and index 0x6444.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	1	0	0	0	0	1	0	1	0	8	30	81	11	07	00	00	00	00

EMCY consumer ← **EMCY producer (CAN-8x23)**

EMCY-msg : 30 81 11 07 00 00 00 00

The first two bytes “30 81” are for the emergency error code. The 3rd byte “11” is for the error register. The last five bytes “07 00 00 00 00” are for the manufacturer specific error values. This emergency message indicates a life guard error.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.5 Special Functions for CAN-8x23

Analog Modules Input/Output range Entry

The CAN-8x23 Manufacturer in the Specific Profile Area defines some entries, which are useful for the analog input/output range. On the other hand, the object with index 0x2004~0x200B will map to the input/output range for the slot 0 to 8, and the entry is dynamic. For the entries information in the Manufacturer Specific Profile Area, it will be described as follows. If users use the CAN-8123, there is only one entry, “0x2004” provided. If users the use CAN-8423, there are 4 entries, “0x2004”, “0x2005”, “0x2006” and “0x2007” provided. And each index of these entries only has two subindex, such as subindex 00, and subindex 01.

The subindex 00 is used to distinguish in which one slot the module inserted. If there is no module or digital module inserted in the slot, the subindex 00 will indicate a number 0. If a slot has been inserted by any module, the subindex 00 will indicate a number more than one. Furthermore, if a slot module supports 4 analog output channels, there should be 4 subindexes, “01”, “02”, “03”, and “04”. They have different, but simply functions, for example the subindex 01 records the channel 1 output range code. The subindex 02 records the channel 2 output range code, and so forth. If users use CAN-8x23 without any slot module, no entry will appear in the Manufacturer Specific Profile Area.

For example, there are 4 slot modules inserted in the CAN-8423, such as I-8057, I-8053, I-8024 and I-87017, and they are respectively inserted in the slot 0, slot1, slot2 and slot3. Then, CAN-8423 will automatically create the following 4 entries, index 0x2004, 0x2005, 0x2006 and 0x2007. The values of the subindex 0 for the first two entries, 0x2004 & 0x2005, are 0 because the first two slot modules belong to the digital I/O modules, and the values for the last two entries, 0x2006 & 0x2007, are 1. Moreover, the values of subindex 1 for 0x2006 are 4 because of 4 analog output channels of I-8024. Under the default situation, the values for the subindex 1 to 4 are 00. These values indicate that all AO channel output ranges of the I-8024 are -10V~+10V. The values of subindex 1 for 0x2007 are 8 because of 8 analog input channels of I-87017. Under the default situation, the values for the subindex 1 to 8 are 00. These values indicate that all AI channel input ranges of the I-87017 are -10V~+10V. Users can refer to the appendix B for more detail information

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

about the input/output range of different analog I/O modules. In the following paragraph, a simple example will be given. Please note that the hardware and wire connection are the same as the situation used in the PDO example.

Step1: Users can send the NMT message to set the NMT operational state on the CAN-8423.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0	0	0	8	01	01	00	00	00	00	00	00

NMT master → **NMT slave (CAN-8x23)**

cs : 1
Node ID : 1

Step 2. Users can send the SDO message to confirm the output range value of the I-8024 AO channel 0.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	06	20	01	00	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 2
m : 06 20 01

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 3. The CAN-8423 will reply with the output range. For example, when the I-8024 is under the default situation, and the value is 0, the output range of the I-8024 AO channel 0 will be -10V~+10V.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	06	20	01	00	00	00	00

SDO client ⟵ **SDO server (CAN-8x23)**

scs : 2
n : 3
e : 1
s : 1
m : 06 20 01
d : 00 00 00 00
Because of the **n**=3, only the 4th byte is valid. Therefore, the feedback value is 00.

Step 4. Users can send the SDO message to confirm the input range value of the I-87017 AI channel 0.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	40	07	20	01	00	00	00	00

SDO client ⟶ **SDO server (CAN-8x23)**

ccs : 2
m : 07 20 01

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 5. The CAN-8423 will reply with the input range. For example when the I-87017 is under the default situation, and the value is 0, the input range of the I-87017 AI channel 0 is -10V~+10V.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	4F	07	20	01	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 2
n : 3
e : 1
s : 1
m : 07 20 01
d : 00 00 00 00
Because of the **n**=3, only the 4th byte is valid. Therefore, the feedback value is 00.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 6. In order to output 7V to the AO0 of the I-8024, users must send the PDO message by using the 2nd RxPDO.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	1	0	0	0	0	0	0	0	1	0	8	98	59	00	00	00	00	00	00

PDO producer → **PDO consumer (CAN-8x23)**

COB-ID : 0x301

L : 8

PDO-msg : 98 59 00 00 00 00 00 00

The first two bytes are valid for AO0. The other bytes are for AO1 to AO3. And the output range of the I-8024 is -10V~10V. According to the transformation table stored in the appendix table, the 7V is mapped to the 0x5998 by applying following equation.

$$HexValue = \left(\frac{7V - (-10V)}{10V - (-10V)} \right) * \left(32767 - (-32768)\right) + (-32768)$$
$$= 22936.75 \approx 22936 = 0x5998$$

The first two bytes of the PDO message will be filled in “98” and “59”. For more details about how to transfer the value between the hex and float, please refer to the section 6.3.

Step 7. Although the AI input value has been changed according the AO value, the RxPDO will not automatically reply in the CAN-8423. Therefore, users have to use the RTR message from the 2nd TxPDO to read back the AI value.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer → **PDO consumer (CAN-8x23)**

COB-ID : 0x281

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 8. The feedback value for AI is 6.992V.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	80	59	FD	FF	FD	FF	FD	FF

PDO consumer ← **PDO consumer (CAN-8x23)**

COB-ID : 0x281

L : 8

PDO-msg : 80 59 FD FF FD FF FD FF

The feedback AI0 value is 0x5980. Users have to transfer this value to be a float one with the input range from -10V to +10V, and a hex one with the input range from 0x8000 (-32768) to 0x7FFF (32767). The value 0x5980 (22912) can be transferred by using the following equation.

$$FloatValue = \left(\frac{22912 - (-32768)}{32767 - (-32768)} \right) * (10V - (-10V)) + (-10V)$$
$$\approx 6.992V$$

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 9. Users can send the Rx SDO message to the CAN-8423 to access the object entry with index 0x1400 and sub-index 02 stored in the communication profile area. Here, users can also change the value of this object entry to 5. For example, the node ID for the CAN-8423 is set to 1.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	1	0	8	2F	07	20	01	01	00	00	00

SDO client → **SDO server (CAN-8x23)**

ccs : 1
n : 3
e : 1
s : 1
m : 07 20 01
d : 01 00 00 00

Step 10. The CAN-8423 will reply with the message to finish the data download. Then, users can use upload methods mentioned before to read back the value for confirmation.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0	0	0	1	0	8	60	07	20	01	00	00	00	00

SDO client ← **SDO server (CAN-8x23)**

scs : 3
m : 07 20 01

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 11. Although the AI input value has been changed according the AO value, the RxPDO will not automatically reply in the CAN-8423. Therefore, users have to use the RTR message from the 2nd TxPDO to read back the AI value.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	1	0	00	00	00	00	00	00	00	00

PDO consumer → **PDO consumer (CAN-8x23)**

COB-ID : 0x281

Step 12. The feedback value for AI is 5V.

11-bit COB-ID (bit)											RTR	Data Length	8-byte Data (byte)							
Func Code				Node ID																
10	9	8	7	6	5	4	3	2	1	0			0	1	2	3	4	5	6	7
0	1	0	1	0	0	0	0	0	0	1	0	8	FF	7F	FF	FF	FD	FF	FF	FF

PDO consumer ← **PDO consumer (CAN-8x23)**

COB-ID : 0x281

L : 8

PDO-msg : FF 7F FF FF FD FF FF FF

The feedback AI0 value is 0x7FFF, i.e. this value is in max limit for the range. The AI0 value will still be 5V, even the input 7V is over the max input range,

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6 Object Dictionary of CAN-8x23

6.1 Communication Profile Area

The following tables are regarding each entry of the communication profile area is defined in CAN-8x23. For the convenient purpose, all communication entries are divided into several tables. They are “General Communication Entries”, “RxPDO Communication Entries”, “RxPDO Mapping Communication Entries”, “TxPDO Communication Entries”, and “TxPDO Mapping Communication Entries”.

Please note that In the table header with “Idx”, “Sidx” and “Attr” represent “index”, “sub-index”, and “attribute” respectively. The sign “---” in the default field means that the default is not defined or can be defined conditionally by the firmware built in CAN-8x23. In the table, the number accompanying letter “h” indicates that this value is in the hex format.

General Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1000h	0h	device type	UNSIGNED 32	RO	---
1001h	0h	error register	UNSIGNED 8	RO	---
1003h	0h	largest sub-index supported for “predefine error field”	UNSIGNED 8	RO	0h
	1h	actual error (the newest one)	UNSIGNED 32	RO	---
	...	...	...	...	---
	5h	actual error (the oldest one)	UNSIGNED 32	RO	---
1005h	0h	COB-ID of Sync message	UNSIGNED 32	RW	80h
1008h	0h	manufacturer device name	VISIBLE_STRING	RO	CAN-8123/ CAN-8223/ CAN-8423/ CAN-8823/
1009h	0h	manufacturer hardware version	VISIBLE_STRING	RO	---
100Ah	0h	manufacturer software version	VISIBLE_STRING	RO	---
100Ch	0h	guard time	UNSIGNED 16	RW	0
100Dh	0h	life time factor	UNSIGNED 8	RW	0

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1010h	0h	largest sub-index supported for “store parameters”	UNSIGNED 8	RO	3h
	1h	Save all parameter	UNSIGNED 32	RW	---
	2h	Save communication parameter	UNSIGNED 32	RW	---
	3h	Save application parameter	UNSIGNED 32	RW	---
1011h	0h	largest sub-index supported for “restore parameters”	UNSIGNED 8	RO	3h
	1h	Restore all default parameters	UNSIGNED 32	RW	---
	2h	Restore communication default parameters	UNSIGNED 32	RW	---
	3h	Restore application default parameters	UNSIGNED 32	RW	---
1014h	0h	COB-ID of EMCY	UNSIGNED 32	RW	80h+Node-ID
1015h	0h	Inhibit time of EMCY	UNSIGNED 16	RW	0
1018h	0h	largest sub-index supported for “identity object”	UNSIGNED 8	RO	1
	1h	Vendor ID	UNSIGNED 32	RO	0x0000013C
	2h	Product code	UNSIGNED 32	RO	---
	3h	Revision number	UNSIGNED 32	RO	---
	4h	Serial number	UNSIGNED 32	RO	---

Note: 1. The object with index 0x1000 has the following data format:

Additional information		General Information
bit 31~ bit 24	bit 23 ~ bit16	bit 15 ~ bit 0
Specific functionality	I/O functionality	Device profile number

For CAN-8x23, the specific function is always in 0. The I/O function defines what kind of device the CAN-8x23 is. Bit 16, 17, 18, 19 present the DI, DO, AI, AO respectively. For example, if bit 16 is 1, it means that the CAN-8x23 has DI channels. If both bit 16 and 17 are 1, the CAN-8x23 will have both DI and DO channels. Bit 23 ~ bit 19 is always in 0. The general information is 0x191 (0x191=401), it means that the CAN-8x23 complies with the CANopen spec DS401.

2. About the objects with index 0x1001 and 0x1003, please refer to the section 5.3.2.

3. The object with index 0x1005 stores the SYNC COB-ID. In the CAN-8x23, this object is used to receive the SYNC COB-ID. The following table shows the data format of the

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SYNC.

Bit Number	Value	Meaning
31 (MSB)	x	do not care
30	0	Device does not generate SYNC message
	1	Device generates SYNC message
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	If bit 29=0
	x	If bit 29=1: 28-11 bits of 29-bit COB-ID
10-0 (LSB)	x	10-0 bits of COB-ID

The CAN-8x23 doesn't support the SYNC generation, therefore 29-bit ID, bit 30 and bit 31 are always in 0.

4. The object with index 0x1008, 0x1009 and 0x100A records the CAN-8423 product information. When interpreting these objects, the ASCII table will be needed.

5. The range of the 0x100c is from 0 to 32767 in CAN-8x23. For more information of the object with index 0x100C and 0x100D, please refer to the section 5.4.2.

6. The object 0x1010/0x1011 supports the saving/restoring of parameters in EEPROM. There 3 parameter groups are distinguished:
   Subindex 1 saves/restores all PDO communication and I/O application parameters.
   Subindex 2 saves/restores all PDO communication parameters.
   Subindex 3 saves/restores all I/O application parameters (ex: safe value, AI inverter, PWM setting, and so on).

7. For the object with index 0x1014, please refer to the section 5.3.1.

8. The object with index 0x1015 stores the inhibit time period between two EMCY message. The function of this object is similar to the PDO communication object with sub-index 04. It is valid for avoiding the large loading on the CAN bus when transmitting a lot of EMCY messages. This parameter range is from 0 to 32767 for the CAN-8x23, and the unit of EMCY inhibit time is ms.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

SDO Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1200h	0h	largest sub-index supported for “server SDO parameter”	UNSIGNED 8	RO	2
	1h	COB-ID form client to server (RxSDO)	UNSIGNED 32	RO	600h+Node-ID
	2h	COB-ID form server to client (TxSDO)	UNSIGNED 32	RO	580h+Node-ID

RxPDO Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1400h	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	2
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	200h+Node-ID
	2h	transmission type	UNSIGNED 8	RW	FFh
1401h	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	5
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	300h+Node-ID
	2h	transmission type	UNSIGNED 8	RW	FFh
1402h	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	5
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	400h+Node-ID
	2h	transmission type	UNSIGNED 8	RW	FFh
1403h	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	5
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	500h+Node-ID
	2h	transmission type	UNSIGNED 8	RW	FFh
1404h	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	5
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	80000000h
	2h	transmission type	UNSIGNED 8	RW	FFh
...	...	...	...	...	...
141Fh	0h	largest sub-index supported for “receive PDO parameter”	UNSIGNED 8	RO	5
	1h	COB-ID used by PDO (Rx)	UNSIGNED 32	RW	8000 0000h

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	2h	transmission type	UNSIGNED 8	RW	FFh

RxPDO Mapping Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1600h	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	8
	1h	write digital output 1h to 8h	UNSIGNED 8	RW	6200 0108h
	2h	write digital output 9h to 10h	UNSIGNED 8	RW	6200 0208h
	3h	write digital output 11h to 18h	UNSIGNED 8	RW	6200 0308h
	4h	write digital output 19h to 20h	UNSIGNED 8	RW	6200 0408h
	5h	write digital output 11h to 28h	UNSIGNED 8	RW	6200 0508h
	6h	write digital output 19h to 30h	UNSIGNED 8	RW	6200 0608h
	7h	write digital output 11h to 40h	UNSIGNED 8	RW	6200 0708h
	8h	write digital output 19h to 48h	UNSIGNED 8	RW	6200 0808h
1601h	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	4
	1h	write analog output 1h	UNSIGNED 16	RW	6411 0110h
	2h	write analog output 2h	UNSIGNED 16	RW	6411 0210h
	3h	write analog output 3h	UNSIGNED 16	RW	6411 0310h
	4h	write analog output 4h	UNSIGNED 16	RW	6411 0410h
1602h	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	4
	1h	write analog output 5h	UNSIGNED 16	RW	6411 0510h
	2h	write analog output 6h	UNSIGNED 16	RW	6411 0610h
	3h	write analog output 7h	UNSIGNED 16	RW	6411 0710h
	4h	write analog output 8h	UNSIGNED 16	RW	6411 0810h
1603h	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	4
	1h	write analog output 9h	UNSIGNED 16	RW	6411 0910h
	2h	write analog output Ah	UNSIGNED 16	RW	6411 0A10h
	3h	write analog output Bh	UNSIGNED 16	RW	6411 0B10h
	4h	write analog output Ch	UNSIGNED 16	RW	6411 0C10h
1604h	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	---
	1h	---	---	RW	---
	...	...	...	...	...

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	---	---	---	RW	---
...	...	...	...	...	...
161Fh	0h	largest sub-index supported for "receive PDO mapping"	UNSIGNED 8	RO	---
	1h	---	---	RW	---
	...	...	...	...	...
	---	---	---	RW	---

TxPDO Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1800h	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	180h+Node-ID
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0
	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0
1801h	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	280h+Node-ID
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0
	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0
1802h	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	380h+Node-ID
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0
	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0
1803h	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	480h+Node-ID
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0
1804h	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	80000000h
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0
	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0
...	...	...	...	...	...
181Fh	0	largest sub-index supported for "receive PDO parameter"	UNSIGNED 8	RO	5
	1	COB-ID used by PDO (Tx)	UNSIGNED 32	RW	80000000h
	2	transmission type	UNSIGNED 8	RW	FFh
	3	inhibit time	UNSIGNED 16	RW	0
	4	reversed	---	---	---
	5	event timer	UNSIGNED 16	RW	0

TxPDO Mapping Communication Entries

Idx	Sidx	Description	Type	Attr	Default
1A00h	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	8
	1h	read digital input 1h to 8h	UNSIGNED 8	RW	6000 0108h
	2h	read digital input 9h to 10h	UNSIGNED 8	RW	6000 0208h
	3h	read digital input 11h to 18h	UNSIGNED 8	RW	6000 0308h
	4h	read digital input 19h to 20h	UNSIGNED 8	RW	6000 0408h
	5h	read digital input 11h to 28h	UNSIGNED 8	RW	6000 0508h
	6h	read digital input 19h to 30h	UNSIGNED 8	RW	6000 0608h
	7h	read digital input 11h to 40h	UNSIGNED 8	RW	6000 0708h
	8h	read digital input 19h to 48h	UNSIGNED 8	RW	6000 0808h
1A01h	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	4
	1h	read analog input 1h	UNSIGNED 16	RW	6401 0110h
	2h	read analog input 2h	UNSIGNED 16	RW	6401 0210h

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	3h	read analog input 3h	UNSIGNED 16	RW	6401 0310h
	4h	read analog input 4h	UNSIGNED 16	RW	6401 0410h
1A02h	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	4
	1h	read analog input 5h	UNSIGNED 16	RW	6401 0510h
	2h	read analog input 6h	UNSIGNED 16	RW	6401 0610h
	3h	read analog input 7h	UNSIGNED 16	RW	6401 0710h
	4h	read analog input 8h	UNSIGNED 16	RW	6401 0810h
1A03h	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	4
	1h	read analog input 9h	UNSIGNED 16	RW	6401 0910h
	2h	read analog input Ah	UNSIGNED 16	RW	6401 0A10h
	3h	read analog input Bh	UNSIGNED 16	RW	6401 0B10h
	4h	read analog input Ch	UNSIGNED 16	RW	6401 0C10h
1A04h	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	---
	1h	---	---	RW	---
	...	...	...	...	...
	---	---	---	RW	---
...	...	...	...	...	...
1A1Fh	0h	largest sub-index supported for "transmit PDO mapping"	UNSIGNED 8	RO	---
	1h	---	---	RW	---
	...	...	...	...	...
	---	---	---	RW	---

6.2 Manufacturer Specific Profile Area

In the following table, there is information about some special functions for the CAN-8x23. The index from 0x2004 to 0x200B records the analog input/output range parameters. The number of these entries will be automatically confirmed when the CAN-8x23 boot up. For example. If the CAN-8423 is used and at least one module is inserted, there are only 4 entries created by CAN-8423 automatically. They are 0x2004, 0x2005, 0x2006, and 0x2007. For more detail about these objects, please refer to the section 5.5.

Analog Modules Input/Output range Entry

Idx	Sidx	Description	Type	Attr	Default

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2004h	0h	largest sub-index supported for “Analog Modules Input/Output Range Control”	UNSIGNED 8	RO	According to AI/AO channel number in the slot 0 module
	1h	Input/Output range of the AI/AO channel 0	UNSIGNED 8	RW	---
	...	...	...	...	...
2005h	0h	largest sub-index supported for “Analog Modules Input/Output Range Control”	UNSIGNED 8	RO	According to AI/AO channel number in the slot 1 module
	1h	Input/Output range of the AI/AO channel 0	UNSIGNED 8	RW	---
...	...	...	...	...	...
200Bh	0h	largest sub-index supported for “Analog Modules Input/Output Range Control”	UNSIGNED 8	RO	According to AI/AO channel number in the slot 8 module
	1h	Input/Output range of the AI/AO channel 0	UNSIGNED 8	RW	---
...	...	...	...	...	...
2014h	0h	largest sub-index supported for “Analog Input Max/Min Inverter”	UNSIGNED 8	RO	According to AI channel number in the slot 1 module
	1h	Input value of the AI channel 0 0: normal 1: 0x7FFF → 0x8000 2: 0x8000 → 0x7FFF	UNSIGNED 8	RW	---
...	...	...	...	...	...
…	...	...	...	...	...
201Bh	0h	largest sub-index supported for “Analog Input Max/Min Inverter”	UNSIGNED 8	RO	According to AI channel number in the slot 8 module
	1h	Input value of the AI channel 0 0: normal 1: 0x7FFF → 0x8000	UNSIGNED 8	RW	---

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

		2: 0x8000 → 0x7FFF			
	...	...	...	...	...

6.3 Standardized Device Profile Area

When the CAN-8x23 is powered on, all of device profile entries are automatically generated by the firmware built inside the CAN-8x23. These device entries will match the channel types and numbers of the slot modules inserted into the CAN-8x23. For the convenient purpose, these entries are divided into four tables, “Digital Input Devices Entries”, “Digital Output Devices Entries”, “Analog Input Devices Entries” and “Analog Output Devices Entries”. They are as follows.

Digital Input Devices Entries

Idx	Sidx	Description	Type	Attr	Default
6000h	0h	largest sub-index supported for “read digital input 8-bit”	UNSIGNED 8	RO	8
	1h	read digital input 1h to 8h	UNSIGNED 8	RO	---
	...	...	...	...	...

Digital Output Devices Entries

Idx	Sidx	Description	Type	Attr	Default
6200h	0h	largest sub-index supported for “write digital output 8-bit”	UNSIGNED 8	RO	---
	1h	write digital output 1h to 8h	UNSIGNED 8	RW	---
	...	...	...	...	...
6206	0h	largest sub-index supported for “error mode digital output 8-bit”	UNSIGNED 8	RW	---
	1h	error mode digital output 1h to 8h	UNSIGNED 8	RW	0
	...	...	...	...	---
6207	0h	largest sub-index supported for “error value digital output 8-bit”	UNSIGNED 8	RW	---
	1h	error value digital output 1h to 8h	UNSIGNED 8	RW	0

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	...	...	...	...	---

Note: When the bus-off is detected or the node guarding fails, the CAN-8x23 will check the value of the object with index 0x6206. If the bit of this value is set to 1, the CAN-8x23 will output the error mode digital output value to the corresponding DO channel. For example, if the sub-index 01 in the object with index 0x6206 and 0x6207 are 0x31 and 0xF8 respectively, and when the error events occurs, only the DO5, DO4, DO0 will be changed to error mode output value because the bit 5, bit 4 and bit 1 of the value 0x31 is 1. So, the DO5, DO4, and DO0 will be change to 1, 1, and 0 respectively. Other channels except DO5, DO4, and DO0 will do nothing.

Analog Input Devices Entries

Idx	Sidx	Description	Type	Attr	Default
6401h	0h	largest sub-index supported for “read analog input 16-bit”	UNSIGNED 8	RO	8
	1h	read analog input 1h	UNSIGNED 16	RO	---
	...	...	...	...	...

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Note: 1. Because the CAN-8x23 only supports the hex format, all AI channels have to transfer to the hex format when storing into this object. The transformation equation is shown below.

$$FloatValue = \left(\frac{HexValue - H\min}{H\max - H\min} \right) * \left(F\max - F\min \right) + F\min$$

The Float Value is the result after transformation. The Hex Value is the value which wants to be transferred. The Hmax and Hmin is the maximum and minimum values of the 2's complement hex range. The Fmax and Fmin is the maximum and minimum value of the float range. User can find out the Hmax, Hmin, Fmax, and Fmin in the appendix B. For example, The input range of the module I-87017 is set to -10V ~ +10V. According to the table in the appendix B, we can find out the range for hex format is from 0x7FFF (+36767) to 0x8000 (-32768). Therefore, if the value got from the AI channel of the I-87017 is 0x1234(+4660), the AI value with float format will be calculated as follows.

$$\left(\frac{4660 - (-32768)}{32767 - (-32768)} \right) * (10V - (-10V)) + (-10V) \approx 1.422V$$

By the way, any AI value which is great than the maximum value of the input range will be automatically set to the maximum value of the input range. And, the AI value which is less than the minimum value of the input range will be automatically set to the minimum value of the input range.

Analog Output Devices Entries

Idx	Sidx	Description	Type	Attr	Default
6411h	0h	largest sub-index supported for “write analog output 16-bit”	UNSIGNED 8	RO	---
	1h	write analog output 1h	UNSIGNED 16	RW	---
	...	...	...	...	...
6443	0h	largest sub-index supported for “error mode analog output 16-bit”	UNSIGNED 8	RW	---
	1h	error mode analog output 1h	UNSIGNED 16	RW	0
	...	...	...	...	---
6444	0h	largest sub-index supported for “error value analog output 16-bit”	UNSIGNED 8	RW	---

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	1h	error value analog output 1h	UNSIGNED 16	RW	0
	...	...	...	...	---

Note: 1. Because the CAN-8x23 doesn't support float format, users have to transfer the AO value from the float format to hex format. It is similar to the AI situation. The transformation equation is as follows.

$$HexValue = \left(\frac{FloatValue - F\min}{F\max - F\min} \right) * \left(H\max - H\min \right) + H\min$$

The Hex Value is the result after transformation. The Float Value is the value which wants to be transferred. The *Fmax* and *Fmin* is the maximum and minimum values of the float range. The *Hmax* and *Hmin* is the maximum and minimum value of the 2's complement hex range. User can find out the *Fmax*, *Fmin*, *Hmax*, and *Hmin* in the appendix B.

2. When the bus-off is detected or the node guarding fails, the CAN-8x23 will check the value of the object with index 0x6443. If this value is set to 1, the CAN-8x23 will output the error mode analog output value to the corresponding AO channel. For example, if the sub-index 01 in the object with index 0x6443 and 0x6444 are 1 and 0x0000 respectively, and when the error events occurs, this AO will be output to error mode output because the value of the object with index 0x6443 and sub-index 01 is 1. The AO output value is 0 because of the value in the object with index 0x6444 and sub-index 01.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4 Object of Counter/Frequency Modules

(Only for I-8080 and I-8084W)

Idx	S-idx	Description	Type	Attr	PDO mapping	Default
3000h	0h	largest sub-index supported for “read counter / frequency 32-bit”	UNSIGNED 8	RO	No	---
	1h	Read counter / frequency with ch1	UNSIGNED 32	RO	Yes	---
	...	...	...	...		...
3001h	0h	largest sub-index supported for “read counter overflow 16-bit”	UNSIGNED 8	RO	No	---
	1h	Read counter overflow with ch1	UNSIGNED 16	RO	Yes	---
	...	...	...	...		---
3002h	0h	largest sub-index supported for “reset counter value with writing 0”	UNSIGNED 8	RO	No	---
	1h	Reset ch1 counter value	UNSIGNED 8	WO	Yes	---
	...	...	...	...		---
3003h	0h	largest sub-index supported for “set XorRegister with 0 or 1”	UNSIGNED 8	RO	No	---
	1h	XorRegister value of ch1	UNSIGNED 8	RW	No	---
	...	...	...	...		---
3004h	0h	largest sub-index supported for “set frequency mode with 0, 1, or 2”	UNSIGNED 8	RO	No	---
	1h	Frequency mode of ch1	UNSIGNED 8	RW	No	---
	...	...	...	...		---
3005h	0h	largest sub-index supported for “set frequency update time (unit – ms)”	UNSIGNED 8	RO	No	---
	1h	Frequency update time of ch1	UNSIGNED 16	RW	No	---
	...	...	...	...		---
3006h	0h	largest sub-index supported for “enable/disable low pass filter”	UNSIGNED 8	RO	No	---
	1h	Low pass filter status of ch1	UNSIGNED 8	RW	No	---
	...	...	...	...		---
3007h	0h	largest sub-index supported for “set low pass filter time (unit - us)”	UNSIGNED 8	RO	No	---
	1h	Low pass filter time of ch1	UNSIGNED 16	RW	No	---
	...	...	...	...		---

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Owing to the configuration of object index 0x3000 to 0x3007, you may parameterize the counter modules. The object index 0x3000 records the counter value of each channel. Each sub-index is corresponding to each channel. Users can use object index 0x2004~2007 to decide the counting method. Please refer to the appendix A for more detail information. If users select Dir/Pulse, Up/Down or AB phase counting, the counter value may be negative value. Therefore, users need to transfer the UNSIGNED 32 to SIGNED32 by themselves. Object index 0x3001 records the overflow times of object index 0x3000. Therefore, the total counter value of one channel is the combination of the value of object index 0x3000 and 0x3001. For example, if users select Dir/Pulse, Up/Down or AB phase counting, there are two situations. If the object index 0x3000 with sub-index 1 is 16384 and the object index 0x3001 with sub-index 1 is 1, the total counter is 1*0x80000000+16384=2147500032. If the object index 0x3000 with sub-index 1 is -8192 and the object index 0x3001 with sub-index 1 is -1, the total counter is (-1)*0x80000000-8192=-2147491840. If users select Frequency or Up counting, there is only one situation. If the object index 0x3000 with sub-index 1 is 16384 and the object index 0x3001 with sub-index 1 is 1, the total counter is 1*0x100000000+16384=4294983680. Take a note that the Frequency counting always has no overflow value.

The object index 0x3002 can help users to clear the counter value. Write this object to 0 once will clear the counter value store in object index 0x3000 and 0x3001. The object index 0x3003 can set the count mode to high active with value 1 or low active with value 0. The object index 0x3004 is used to set the frequency mode of counter module. Value 0 is auto mode, value 1 is low frequency and value 2 is high frequency. These three frequency modes are decided by the object “update time of frequency” with index 0x3005. If users want to set the update time of frequency measurement, use object 0x3005 to reach this purpose. The object index 0x3006 decide if users want to use low-pass filter or not, set the value 1 to enable the low-pass filter, or value 0 to disable the low-pass filter. The object index 0x3007 is used to set the pulse width which will be used in the low-pass filter. If the pulse width of the signal is less than this value, this signal will be taken into account.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.5 Object of PWM Module (Only for I-8088W)

Idx	S-idx	Description	Type	Attr	PDO mapping	Default
3100h	0h	largest sub-index supported for “start to output pulse”	UNSIGNED 8	RO	No	---
	1h	Start to output pulse with ch1	UNSIGNED 8	RW	Yes	---
	...	...	...	...		...
3101h	0h	largest sub-index supported for “set burst count 16-bit”	UNSIGNED 8	RO	No	---
	1h	Set burst count with ch1	UNSIGNED 16	RW	No	---
	...	...	...	...		---
3102h	0h	largest sub-index supported for “set frequency of output pulse 32-bit”	UNSIGNED 8	RO	No	---
	1h	Set frequency of output pulse	UNSIGNED 32	RW	No	---
	...	...	...	...		---
3103h	0h	largest sub-index supported for “set pulse duty with 1 ~ 999 (‰)”	UNSIGNED 8	RO	No	---
	1h	Set pulse duty of ch1	UNSIGNED 16	RW	No	---
	...	...	...	...		---
3104h	0h	largest sub-index supported for “enable/disable hardware trig”	UNSIGNED 8	RO	No	---
	1h	Set hardware trig of ch1	UNSIGNED 8	RW	No	---
	...	...	...	...		---
3105h	0h	largest sub-index supported for “enable/disable sync channel”	UNSIGNED 8	RO	No	---
	1h	Set sync channel of ch1	UNSIGNED 8	RW	No	---
	...	...	...	...		---
3106h	0h	largest sub-index supported for “start the synchronous pulse”	UNSIGNED 8	RO	No	---
	1h	Start the synchronous pulse of the first PWM module which has lower slot No.	UNSIGNED 8	RW	Yes	---
	...	...	...	...		---

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Owing to the configuration of object index 0x3100 to 0x3106, you may parameterize the PWM modules. The object index 0x3100 can control the module to start or stop the pulse output of each channel. Each sub-index is corresponding to each channel. Users can use object index 0x2004~2007 to decide the PWM method of each slot. Please refer to the appendix A for more details. If users select Burst Counting mode, the object index 0x3101 must be set to decide how many pulse users want to output. Users can set 1 ~ 65535 to the object 0x3101 and use object 0x3100 to start or stop the pulse output. Every time when set the object 0x3100 to 1, the channel will output the specific pulses with one burst cyclic. For example, user set channel 0 to Burst Counting mode and set object index 0x3101 with sub-index 1 to 100. When user set the object 0x3100 with sub-index 1 to 1, the channel 0 will output 100 pulses. Or if users select Continue Counting mode, the object 0x3101 will useless. When users set the object 0x3100 to 1, the channel will start to output the pulse cyclically until the object is set to 0.If you want to change the frequency of pulse, you can set the value 100 ~ 5000000 with the base 0.1Hz (that is 10Hz ~ 500kHz) to object 0x3102.

Object index 0x3103 is pulse duty per mille (‰). If set the object to value 300, it means that the high duty is 300‰ and the low duty is 700‰ in one pulse width. The object 0x3104 can set the DI pin of the PWM module as hardware trigger channel. When set the value of object 0x3104 with sub-index 2 to 1, it means the DI channel 2 will loss the DI functions and become a hardware trigger pin. In this case, if there is a signal (5V~30V) into the DI channel 2, the channel 2 will start to output until the signal is clear.

Object 0x3105 and 0x3106 can control all of the channels of the PWM module to output synchronous. If user wish channel 0 ~ 3 of the PWM module output the pulse synchronously. Set the object 0x3105 with sub-index 1 ~ 4 to 1, and set the others to 0. Then, set the object 0x3106 with sub-index 1 to 1. These 4 channels (channel 0 ~ 3) will start to output pulse at the same time (their first low-to-high edge will be triggered at the same time, but the period may be different because of different pulse width). Take a note that the different sub-index of the object 0x3106 indicates the different PWM module in different slot. If there are two PWM modules on the CAN-8x23, the maximum sub-index number of the object 0x3106 is 2. The sub-index 1 is for the PWM module with lower slot No. and the sub-index 2 is for the one with higher slot No.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Appendix A: Type Code Table

In order to look up the configuration parameters of each slot module more quickly, the transformation table has separated into several parts according to the name of slot module. They are given below.

I-87K module	I-8K module
I-87013	I-8017HS/I-8017HW
I-87015/I-87015P	I-8024/I-8024W
I-87017/I-87017R/I-87017W/I-87017RW/ I-87017ZW/ I-87017ZW/ I-87017W-RMS	I-8050
I-87017RC	I-8080/I-8084W
I-87018/I-87018RW/I-87018W/I-87018Z /I-87018PW	I-8088W
I-87019RW/I-87019PW/I-87019ZW	
Thermocouple of I-87018/I-87019 Series	
I-87022	
I-87024/I-87024W	
I-87026	
I-87024C/I-87028C	
I-87028UW/ I-87028VW/I-87024UW	

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87013/ I-87015 RTD Type Definition				Back to table
Range Code (Hex)	**RTD Type**	**Data Format**	**Max Value**	**Min Value**
20 (default)	Platinum 100 a = 0.00385	Input Range	+100.00℃	-100.00℃
		2's complement HEX	7FFF	8000
21	Platinum 100 a = 0.00385	Input Range	+100.00℃	+000.00℃
		2's complement HEX	7FFF	0000
22	Platinum 100 a = 0.00385	Input Range	+200.00℃	+000.00℃
		2's complement HEX	7FFF	0000
23	Platinum 100 a = 0.00385	Input Range	+600.00℃	+000.00℃
		2's complement HEX	7FFF	0000
24	Platinum 100 a = 0.003916	Input Range	+100.00℃	-100.00℃
		2's complement HEX	7FFF	8000
25	Platinum 100 a = 0.003916	Input Range	+100.00℃	+000.00℃
		2's complement HEX	7FFF	0000
26	Platinum 100 a = 0.003916	Input Range	+200.00℃	+000.00℃
		2's complement HEX	7FFF	0000
27	Platinum 100 a = 0.003916	Input Range	+600.00℃	+000.00℃
		2's complement HEX	7FFF	0000
28	Nickel 120	Input Range	+100.00℃	-080.00℃
		2's complement HEX	7FFF	999A

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

29	Nickel 120	Input Range	+100.00℃	+000.00℃
		2's complement HEX	7FFF	0000
2A	Platinum 1000 a = 0.00385	Input Range	+600.00℃	-200.00℃
		2's complement HEX	7FFF	D556
2B*1	Cu 100 a = 0.00421	Input Range	+150.00℃	-020.00℃
		2's complement HEX	7FFF	EEEF
2C*1	Cu 100 a = 0.00421	Input Range	+200.00℃	-000.00℃
		2's complement HEX	7FFF	0000
2D*1	Cu 1000 a = 0.00421	Input Range	+150.00℃	-020.00℃
		2's complement HEX	7FFF	EEEF
2E*2	Pt 100 a = 0.00385	Input Range	+200.00℃	-200.00℃
		2's complement HEX	7FFF	8000
2F*2	Pt 100 a = 0.003916	Input Range	+200.00℃	-200.00℃
		2's complement HEX	7FFF	8000
80*2	Pt 100 a = 0.00385	Input Range	+600.00℃	-200.00℃
		2's complement HEX	7FFF	D556
81*2	Pt 100 a = 0.003916	Input Range	+600.00℃	-200.00℃
		2's complement HEX	7FFF	D556

Note :

* 1: Type 2B, 2C and 2D are only available with I-87015.
* 2: Type 2E, 2F, 80 and 81 are only available with the I-87015 firmware version A1.10 and later, I-87013 firmware version B1.3 and later.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87017 Series Type 08 to 0D Definition (not for I-87017RC)				Back to table
Range Code (Hex)	**Input Range**	**Data Format**	**Full Scale**	**Negative Full Scale**
08 (default)	-10V to +10V	Input Range	+10.000 V	-10.000 V
		2's Complement HEX	7FFF	8000
09	-5V to +5V	Input Range	+5.0000 V	-5.0000 V
		2's Complement HEX	7FFF	8000
0A	-1V to +1V	Input Range	+1.0000 V	-1.0000 V
		2's Complement HEX	7FFF	8000
0B	-500mV to +500mV	Input Range	+500.00 mV	-500.00 mV
		2's Complement HEX	7FFF	8000
0C	-150mV to +150mV	Input Range	+150.00 mV	-150.00 mV
		2's Complement HEX	7FFF	8000
0D*1	-20mA to +20mA	Input Range	+20.000 mA	-20.000 mA
		2's Complement HEX	7FFF	8000

Note:
*1: When I-87017 and I-87017R are connecting to a current source set to OD type code, an optional external 125 Ohms resistor is required.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87017RC Type 07 to 1A Definition				Back to table
Range Code (Hex)	**Input Range**	**Data Format**	**Full Scale**	**Negative Full Scale**
07	-4mA to +20mA	Input Range	+04.000 mA	+20.000 mA
		2's Complement HEX	7FFF	8000
0D (default)	-20mA to +20mA	Input Range	+20.000 mA	-20.000 mA
		2's Complement HEX	7FFF	8000
1A	+0A to +20mA	Input Range	+00.000 mA	+20.000 mA
		2's Complement HEX	7FFF	8000

Note:

1. I-87017RC has built-in 125 Ohms resistors for each channels. When connecting to a current source, no add any external resistors required.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87018 Series Type 00 to 06 Definition				Back to table
Range Code (Hex)	**Input Range**	**Data Format**	**Full Scale**	**Negative Full Scale**
00	-15mV to +15mV	Input Range	+15.000 mV	-15.000 mV
		2's Complement HEX	7FFF	8000
01	-50mV to +50mV	Input Range	+50.000 mV	-50.000 mV
		2's Complement HEX	7FFF	8000
02	-100mV to +100mV	Input Range	+100.00 mV	-100.00 mV
		2's Complement HEX	7FFF	8000
03	-500mV to +500mV	Input Range	+500.00 mV	-500.00 mV
		2's Complement HEX	7FFF	8000
04	-1V to +1V	Input Range	+1.0000 V	-1.0000 V
		2's Complement HEX	7FFF	8000
05 (default)	-2.5V to +2.5V	Input Range	+2.5000 V	-2.5000 V
		2's Complement HEX	7FFF	8000
06*1	-20mA to +20mA with 125Ω resistor	Input Range	+20.000 mA	-20.000 mA
		2's Complement HEX	7FFF	8000

Note:

*1: When I-87018 and I-87018R are connecting to a current source set to 06 type code, an optional external 125 Ohms resistor is required.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87019R Type 00 to 19 Definition				Back to table
Range Code (Hex)	**Input Range**	**Data Format**	**Full Scale**	**Negative Full Scale**
00	-15mV to +15mV	Input Range	+15.000 mV	-15.000 mV
		2's Complement HEX	7FFF	8000
01	-50mV to +50mV	Input Range	+50.000 mV	-50.000 mV
		2's Complement HEX	7FFF	8000
02	-100mV to +100mV	Input Range	+100.00 mV	-100.00 mV
		2's Complement HEX	7FFF	8000
03	-500mV to +500mV	Input Range	+500.00 mV	-500.00 mV
		2's Complement HEX	7FFF	8000
04	-1V to +1V	Input Range	+1.0000 V	-1.0000 V
		2's Complement HEX	7FFF	8000
05	-2.5V to +2.5V	Input Range	+2.5000 V	-2.5000 V
		2's Complement HEX	7FFF	8000
06	-20mA to +20mA with 125Ω resistor	Input Range	+20.000 mA	-20.000 mA
		2's Complement HEX	7FFF	8000
08 (default)	-10V to +10V	Input Range	+10.000 V	-10.000 V
		2's Complement HEX	7FFF	8000
09	-5V to +5V	Input Range	+5.0000 V	-5.0000 V
		2's Complement HEX	7FFF	8000
0A	-1V to +1V	Input Range	+1.0000 V	-1.0000 V
		2's Complement HEX	7FFF	8000

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

0B	-500mV to +500mV	Input Range	+500.00 mV	-500.00 mV
		2's Complement HEX	7FFF	8000
0C	-150mV to +150mV	Input Range	+150.00 mV	-150.00 mV
		2's Complement HEX	7FFF	8000
0D	-20mA to +20mA with 125Ω resistor	Input Range	+20.000 mA	-20.000 mA
		2's Complement HEX	7FFF	8000

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87018/ 87018R/ 87019R Thermocouple Type Definition				Back to table
Range Code (Hex)	**Thermocouple Type**	**Data Format**	**Max Value**	**Min Value**
0E	J Type	Input Range	+760.00℃	-210.00℃
		2's Complement HEX	7FFF	DCA2
0F	K Type	Input Range	+1372.0℃	-0270.0℃
		2's Complement HEX	7FFF	E6D0
10	T Type	Input Range	+400.00℃	-270.00℃
		2's Complement HEX	7FFF	A99A
11	E Type	Input Range	+1000.0℃	-0270.0℃
		2's Complement HEX	7FFF	DD71
12	R Type	Input Range	+1768.0℃	+0000.0℃
		2's Complement HEX	7FFF	0000
13	S Type	Input Range	+1768.0℃	+0000.0℃
		2's Complement HEX	7FFF	0000
14	B Type	Input Range	+1820.0℃	+0000.0℃
		2's Complement HEX	7FFF	0000
15	N Type	Input Range	+1300.0℃	-0270.0℃
		2's Complement HEX	7FFF	E56B
16	C Type	Input Range	+2320.0℃	+0000.0℃
		2's Complement HEX	7FFF	0000
17	L Type	Input Range	+800.00℃	-200.00℃
		2's Complement HEX	7FFF	E000
18	M Type	Input Range	+100.00℃	-200.00℃
		2's Complement HEX	4000	8000
19	L Type DIN43710	Input Range	+900.00℃	-200.00℃
		2's Complement HEX	7FFF	E38F

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87022 Analog Output Type Definition				Back to table
Range Code (Hex)	**Output Range**	**Data Format**	**Max Value**	**Min Value**
0	0 to 20mA	Input Range	20.000 mA	00.000 mA
		Hexadecimal	FFF	000
1	4 to 20mA	Input Range	20.000 mA	04.000 mA
		Hexadecimal	FFF	000
2 (default)	0 to 10V	Input Range	10.000 V	00.000 V
		Hexadecimal	FFF	000

I-87024 Analog Output Type Definition				Back to table
Range Code (Hex)	**Output Range**	**Data Format**	**Max Value**	**Min Value**
30	0 to 20mA	Output Range	+20.000 mA	+00.000 mA
		2's Complement HEX	0x7FFF	0
31	4 to 20mA	Output Range	+20.000 mA	+04.000 mA
		2's Complement HEX	0x7FFF	0
32	0 to 10V	Output Range	+10.000 V	+00.000 V
		2's Complement HEX	0x7FFF	0
33 (default)	-10 to 10V	Output Range	+10.000 V	-10.000 V
		2's Complement HEX	0x7FFF	0x8000
34	0 to 5V	Output Range	+05.000 V	+00.000 V
		2's Complement HEX	0x7FFF	0
35	-5 to 5V	Output Range	+05.000 V	-05.000 V
		2's Complement HEX	0x7FFF	0x8000

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

I-87026 Analog Output Type Definition				Back to table
Range Code (Hex)	**Output Range**	**Data Format**	**Max Value**	**Min Value**
0	0 to 20mA	Output Range	20.000 mA	00.000 mA
		Hexadecimal	FFFF	0000
1	4 to 20mA	Output Range	20.000 mA	04.000 mA
		Hexadecimal	FFFF	0000
2 (default)	0 to 10V	Output Range	10.000 V	00.000 V
		Hexadecimal	FFFF	0000

I-87024C/I-87028C Analog Output Type Definition				Back to table
Range Code (Hex)	**Output Range**	**Data Format**	**Max Value**	**Min Value**
0 (default)	0 to 20mA	Output Range	+20.000 mA	+00.000 mA
		Hexadecimal	FFF	000
1	4 to 20mA	Output Range	+20.000 mA	+04.000 mA
		Hexadecimal	FFF	000

I-87028UW/I-87028VW/I-87024UW Analog Output Type Definition				Back to table
Range Code (Hex)	**Output Range**	**Data Format**	**Max Value**	**Min Value**
0	0 to 20mA	Output Range	+20.000 mA	+00.000 mA
		Hexadecimal	0xFFFF	0
1	4 to 20mA	Output Range	+20.000 mA	+04.000 mA
		Hexadecimal	0xFFFF	0
2 (default)	0 to 10V	Output Range	+10.000 V	+00.000 V
		Hexadecimal	0xFFFF	0

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3	-10 to 10V	Output Range	+10.000 V	-10.000 V
		Hexadecimal	0x7FFF	0x8000
4	0 to 5V	Output Range	+05.000 V	+00.000 V
		Hexadecimal	0xFFFF	0
5	-5 to 5V	Output Range	+05.000 V	-05.000 V
		Hexadecimal	0x7FFF	0x8000

I-8024/I-8024W Analog Output Type Definition Back to table

Range Code (Hex)	Output Range	Data Format	Max Value	Min Value
0 (default)	-10 to 10V	Output Range	+10.000 V	-100.000 V
		Hexadecimal	7FFF	8000
1	0 to 20mA	Output Range	+20.000 mA	+00.000 mA
		Hexadecimal	7FFF	8000

I-8017HS/I-8017HW Analog Input Type Definition Back to table

Range Code (Hex)	Output Range	Data Format	Max Value	Min Value
0 (default)	-10 to 10V	Input Range	+10.000 V	-10.000 V
		Hexadecimal	1FFF	2000
1	-5 to 5V	Input Range	+5.000 V	-5.000 V
		Hexadecimal	1FFF	2000
2	-2.5V to +2.5V	Input Range	+2.500 V	-2.5000 V
		Hexadecimal	1FFF	2000
3	-1.25V to +1.25V	Input Range	+1.250 V	-1.250 V
		Hexadecimal	1FFF	2000
4	-20 mA to +20 mA	Input Range	+20.000 mA	-20.000 mA

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

		Hexadecimal	1FFF	2000

I-8080/ 8084W Counter Input Type Definition				Back to table
Range Code (Hex)	**Counter Type**	**Channel number**	**Max Value**	**Min Value**
0 *1	Dir/Pulse Counter	4	2147483647	-2147483648
			1FFFFFFF	80000000
1 *1	Up/Down Counter	4	2147483647	-2147483648
			1FFFFFFF	80000000
2	Frequency	8	-	-
			-	-
3 (default)	Up Counter	8	4294967295	0
			FFFFFFFF	00000000
4 *1 (for I-8084W)	AB Phase	4	2147483647	-2147483648
			1FFFFFFF	80000000

Note:

*1: The sub-index of all parameters and input channels are still 8, but the sub-index 1 is equal to sub-index 2, and sub-index 3 is equal to sub-index 4, and so on.

I-8088W PWM Output Type Definition				Back to table
Range Code (Hex)	**Counter Type**	**Channel number**	**Max Value**	**Min Value**
0	Burst Counter	8	65535	1
			FFFF	1
1 (default)	Continue Counter	8	---	---
			---	---

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Appendix B: DIO Type Define of I-8050 Modules

I-8050 is a selectable 16-channel DIO module. User can decide which channel will be DI and which channel will be DO. In CAN-8x23, users can achieve this purpose by setting the type code in the object index 0x2004~0x2007. The object index 0x2004, 0x2005, 0x2006 and 0x2007 are for the module plugged in slot 0, slot 1, slot 2 and slot 3 respectively.

For example, if the I-8050 module is plugged on the slot 0 of CAN-8423, user can set the object index 0x2004 to decide the channel type of I-8050. The object index 0x2004 with sub-index 1 controls the channel type of ch-0 to ch-7 of I-8050, and sub-index 2 controls the ch-8 to ch-15. If one bit of the object data is set to 1, it means that the corresponding channel will be set to DI. So if users set the sub-index 1 to 0x77, it means that only the ch-3 and ch-7 are DO channels and the others are DI channels. In default, all the channels of I-8050 are DI channels.

Back to table

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000