I-87123 CANopen Master Module

User's Manual

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assume no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2010 by ICP DAS. All rights are reserved.

Trademark

The names used for identification only maybe registered

trademarks of their respective companies.

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

Tables of Content

1. General Information 5

1.1. CANopen Introduction 5

1.2. CANopen Applications 6

1.3. I-87123 Library Characteristics 7

2. Hardware Specification 10

2.1. Hardware Structure 10

2.2. Wire Connection 11

2.3. Power LED 13

2.4. Tx/Rx LED 13

2.5. ERR LED 13

3. I-87123 Function Library 14

3.1. Function List 14

3.2. Function Return Code 16

3.3. CANopen Master Library Application Flowchart 18

3.4. Communication Services Introduction 20

3.5. Function Description 23

3.5.1. I87123_GetVersion 23

3.5.2. I87123_SetFunctionTimeout 24

3.5.3. I87123_GetCANStatus 25

3.5.4. I87123_Pause 26

3.5.5. I87123_Start 27

3.5.6. I87123_InitMaster 28

3.5.7. I87123_ShutdownMaster 29

3.5.8. I87123_MasterSendBootupMsg 30

3.5.9. I87123_SetMasterMode 31

3.5.10. I87123_GetMasterMode 32

3.5.11. I87123_GetFirmwareVersion 33

3.5.12. I87123_EDS_Load 34

3.5.13. I87123_AddNode 35

3.5.14. I87123_RemoveNode 37

3.5.15. I87123_RemoveAndResetNode 38

3.5.16. I87123_DelayAndResponseTimeout 39

3.5.17. I87123_ScanNode 40

3.5.18. I87123_GetNodeList 41

3.5.19. I87123_NMTChangeState 42

3.5.20. I87123_NMTGetState 43

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

3.5.21. I87123_NMTGuarding 44
3.5.22. I87123_NMTHeartbeat 45
3.5.23. I87123_SDOReadData 46
3.5.24. I87123_SDOReadFile 47
3.5.25. I87123_SDOWriteData 48
3.5.26. I87123_SDOAbortTransmit 49
3.5.27. I87123_PDOWrite 50
3.5.28. I87123_PDOWrite_Fast 51
3.5.29. I87123_PDORemote 52
3.5.30. I87123_PDORemote_Fast 53
3.5.31. I87123_SetPDORemotePolling 54
3.5.32. I87123_GetPDOLastData 55
3.5.33. I87123_GetMultiPDOData 56
3.5.34. I87123_GetRxPDOID 57
3.5.35. I87123_GetTxPDOID 58
3.5.36. I87123_InstallPDO 59
3.5.37. I87123_DynamicPDO 60
3.5.38. I87123_RemovePDO 61
3.5.39. I87123_ChangePDOID 62
3.5.40. I87123_GetPDOMapInfo 63
3.5.41. I87123_InstallPDO_List 64
3.5.42. I87123_RemovePDO_List 66
3.5.43. I87123_PDOUseEntry 67
3.5.44. I87123_PDOTxType 68
3.5.45. I87123_PDOEventTimer 69
3.5.46. I87123_PDOInhibitTime 70
3.5.47. I87123_ChangeSYNCID 71
3.5.48. I87123_SetSYNC_List 72
3.5.49. I87123_GetSYNCID 73
3.5.50. I87123_SendSYNCMsg 74
3.5.51. I87123_GetCyclicSYNCInfo 75
3.5.52. I87123_ChangeEMCYID 76
3.5.53. I87123_SetEMCY_List 77
3.5.54. I87123_GetEMCYID 78
3.5.55. I87123_ReadLastEMCY 79
3.5.56. I87123_GetBootUpNodeAfterAdd 80
3.5.57. I87123_GetEMCYData 81
3.5.58. I87123_GetNMTErr 82

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

3.5.59. I87123_InstallBootUpISR 83
3.5.60. I87123_RemoveBootUpISR 84
3.5.61. I87123_InstallEMCYISR 85
3.5.62. I87123_RemoveEMCYISR 86
3.5.63. I87123_InstallNMTErrISR 87
3.5.64. I87123_RemoveNMTErrISR 88
3.5.65. I87123_GetMasterReadSDOEvent 89
3.5.66. I87123_GetMasterWriteSDOEvent 90
3.5.67. I87123_ResponseMasterSDO 91
3.5.68. I87123_InstallReadSDOISR 92
3.5.69. I87123_RemoveReadSDOISR 93
3.5.70. I87123_InstallWriteSDOISR 94
3.5.71. I87123_RemoveWriteSDOISR 95
3.5.72. I87123_GetMasterRemotePDOEvent 96
3.5.73. I87123_GetMasterRxPDOEvent 97
3.5.74. I87123_ResponseMasterPDO 98
3.5.75. I87123_InstallRxPDOISR 99
3.5.76. I87123_RemoveRxPDOISR 100
3.5.77. I87123_InstallRemotePDOISR 101
3.5.78. I87123_RemoveRemotePDOISR 102
4. Demo Programs 103
4.1. Brief of the demo programs 103
4.1.1. Listen_Mode 104
4.1.2. NMT_Protocol 105
4.1.3. PDO_Parameter 106
4.1.4. PDO_Protocol 107
4.1.5. Scan_Node 108
4.1.6. SDO_PDO_PROC 109
4.1.7. SDO_Read 110
4.1.8. SDO_Write 111
4.1.9. SYNC_Protocol 112
4.1.10. PDO_MultiData 113
5. Update Firmware 114

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

1. General Information

1.1. CANopen Introduction

The CAN (Controller Area Network) is a serial communication protocol, which efficiently supports distributed real-time control with a very high level of security. It is an especially suited for networking "intelligent" devices as well as sensors and actuators within a system or sub-system. In CAN networks, there is no addressing of subscribers or stations in the conventional sense, but instead, prioritized messages are transmitted. CANopen is one kind of the network protocols based on the CAN bus and it is applied in a low level network that provides connections between simple industrial devices (sensors, actuators) and higher-level devices (controllers), as shown in Figure 1.1.

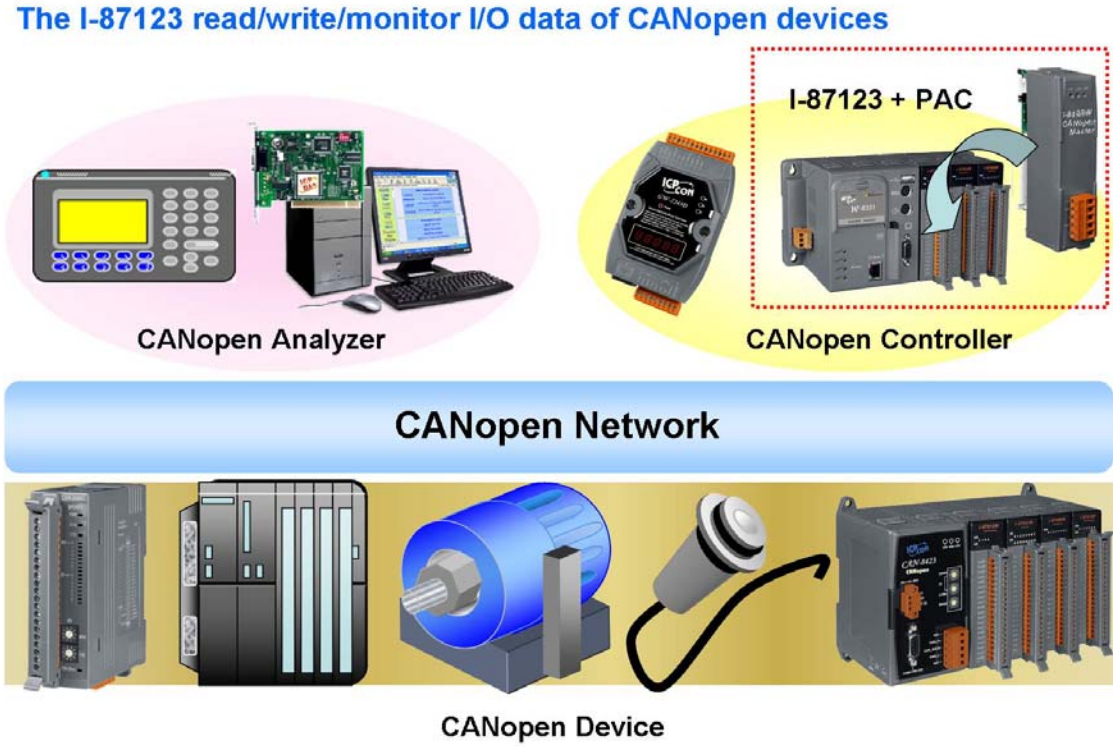


Figure 1.1 Example of the CANopen network

CANopen was developed as a standardized embedded network with highly flexible configuration capabilities. It provides standardized communication objects for real-time data (Process Data Objects, PDO), configuration data (Service Data Objects, SDO), network management data (NMT message, and Error Control), and special functions (Time Stamp, Sync message, and Emergency message). Nowadays, CANopen is used in many various application fields, such as medical equipment, off-road vehicles, maritime electronics, public transportation, building automation and so on.

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

1.2. CANopen Applications

CANopen is the standardized network application layer optimized for embedded networks. Its specifications cover the standardized application layer, frameworks for the various applications (e.g. general I/O, motion control system, maritime electronics and so forth) as well as device, interface, and application profiles.

The main CANopen protocol and products are generally applied in the low-volume and mid-volume embedded systems. The following examples show some parts of the CANopen application fields. (For more information, please refer to the web site, http://www.can-cia.org)**:**

- Truck-based superstructure control systems
- Off-highway and off-road vehicles
- Passenger and cargo trains
- Maritime electronics
- Factory automation
- Industrial machine control
- Lifts and escalators
- Building automation
- Medical equipment and devices
- Non-industrial control
- Non-industrial equipment



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

1.3. I-87123 Library Characteristics

In order to use the I-87123 module, we provide I-87123 library for ViewPAC-2xWx, WinPAC-8000, XPAC-8000 and XPAC-8000-CE6 series main control unit, and users can use it establish CANopen communication network rapidly. Most of the CANopen communication protocols, such as PDO, SDO and NMT, will be handled by the library function automatically. Therefore, it is helpful to reduce the complexity of developing a CANopen master interface, and let users ignore the detail CANopen protocol technology. This library mainly supports connection sets of master-slave architecture, which include some useful functions to control the CANopen slave device in the CANopen network. The following figure describes the general application architecture of I-87123.

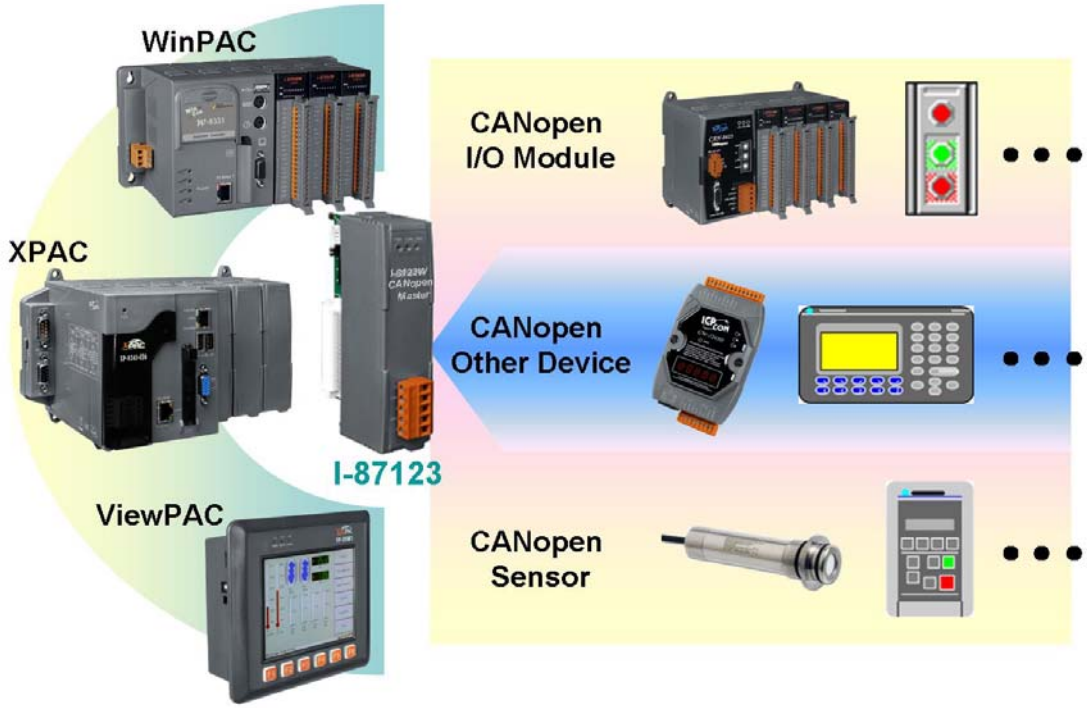


Figure 1.2 Application architecture

I-87123 follows the CiA CANopen specification DS-301 V4.01, and supports the several CANopen features. The CANopen communication general concept is shown as Figure 1.3.

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

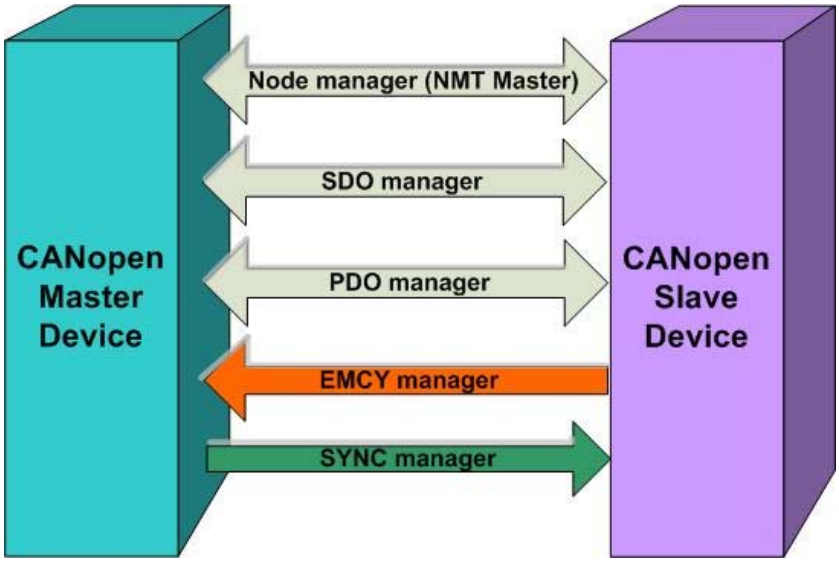


Figure 1.3 CANopen communication general concept

- **Node Manager (NMT Master)**
  - Functions for changing the slave device state
  - Node Guarding and Heartbeat Protocol for error control
  - Support Emergency (EMCY) messages
- **SDO Manager**
  - Expedited, segmented and block methods for SDO download and upload
- **PDO Manager**
  - Support all transmission types and event timer
- **SYNC Manager**
  - SYNC message production
  - SYNC cycles of 1ms resolution
- **EMCY Manager**
  - EMCY message consumer

For more information about the CANopen functions described above, please refer to the function descriptions and demo programs shown in the chapter 3 and chapter 4.

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

Specifications

- CPU:80186, 80MHz
- NXP SJA1000 CAN controller with 16MHz clock
- NXP 82C250 CAN transceiver
- Power LED, Tx/Rx LED, Error LED
- 120Ω terminal resister selected by jumper
- CAN bus interface: ISO 11898-2, 5-pin screw terminal with on-board optical isolator protection
- 2500 Vrms isolation on CAN side
- 1000 Vdc voltage protection
- Power Consumption: 2W
- Operating Temperature: -25℃ to +75℃
- Storage Temperature: -30℃ to +80℃
- Humidity: 10% ~ 90%, non-condensing

Features

- One CAN communication port.
- Support PAC series MCU (main control unit) with Windows CE5, CE6 and XPe.
- Library provides eVC++, VC++6.0, VC++2005, C#.Net2005, and VB.Net2005 developments.
- Three indication LEDs (Pwr, Tx/Rx and Err LEDs).
- Support 8 kinds of baud rates: 10 kbps, 20 kbps, 50 kbps, 125 kbps, 250 kbps, 500 kbps, 800 kbps, and 1 Mbps.
- Support the maximum nodes up to 127.
- Follow CiA DS-301 V4.02.
- Support upload and download SDO Segment protocol.
- Support Node Guarding and Heartbeat protocol.
- Provide Master Listen Mode to listen the slave status of the CANopen network.
- Block-function and non-block-function selected.
- Demos and utility are provided.

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

2. Hardware Specification

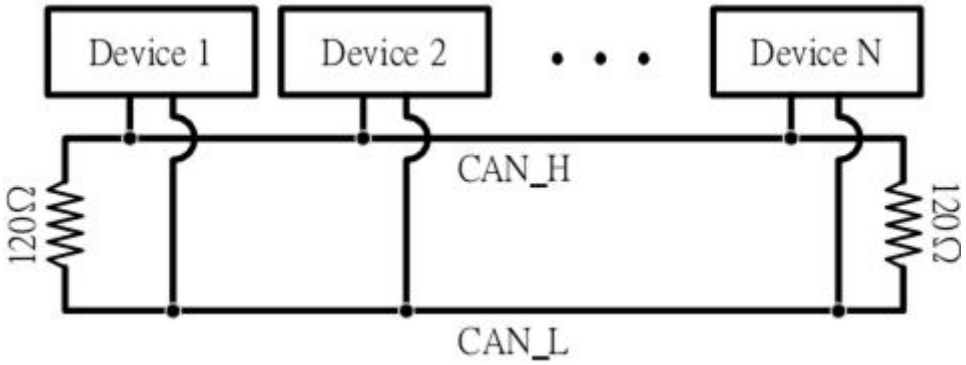
2.1. Hardware Structure



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

2.2. Wire Connection

In order to minimize the reflection effects on the CAN bus line, the CAN bus line has to be terminated at both ends by two terminal resistances as in the following figure. According to the ISO 11898-2 spec, each terminal resistance is 120Ω(or between 108Ω~132Ω). The length related resistance should have 70mΩ/m. Users should check the resistances of the CAN bus, before they install a new CAN network.

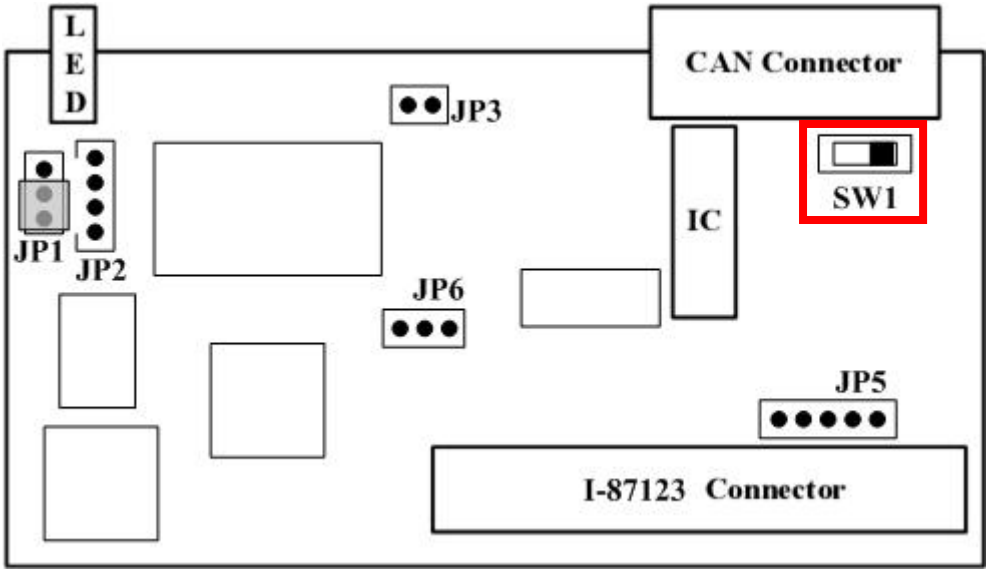


Moreover, to minimize the voltage drop over long distances, the terminal resistance should be higher than the value defined in the ISO 11898-2. The following table can be used as a good reference.

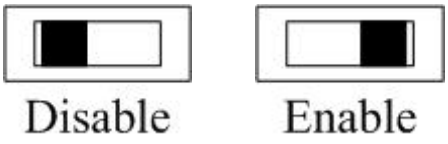
Bus Length (meter)	Bus Cable Parameters		Terminal Resistance (Ω)
	Length Related Resistance (mΩ/m)	Cross Section (Type)	
0~40	70	0.25(23AWG)~0.34 mm^2 (22AWG)	124 (0.1%)
40~300	<60	0.34(22AWG)~0.6 mm^2 (20AWG)	127 (0.1%)
300~600	<40	0.5~0.6mm^2 (20AWG)	150~300
600~1K	<20	0.75~0.8mm^2 (18AWG)	150~300

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

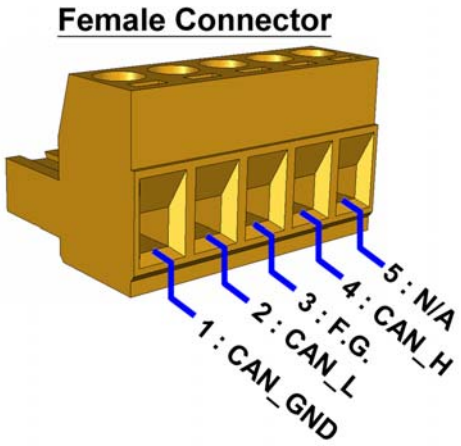
In the I-87123, the 120Ω terminal resistance is supplied. The SW1 of the I-87123 is for the terminal resistance. Its location is shown in the following figure.



The following connection statuses are presented for the condition if the terminal resister is enabled or disabled.



SW1 Switch selection.



Pin assignment of I-87123 connector

Pin No.	Signal	Description
1	CAN_GND	CAN Ground
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	No use

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

2.3. Power LED

I-87123 slot module needs 2W power consumption. If the electric power is supplied normally, the Power LED will be turn on always. If any other situation, please check the power supply or contact to your distributor.

2.4. Tx/Rx LED

Each I-87123 slot module provides Tx/Rx LED to check the CAN messages transmission and reception situation. If the I-87123 is transmitting or receiving a CAN message, the Tx/Rx LED will blink. If I-87123's loading is heavy, the Tx/Rx LED will always turn on.

2.5. ERR LED

The ERR LED indicates the error status of the CAN physical layer and indicates the errors due to missing CAN message.

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

3. I-87123 Function Library

3.1. Function List

In order to use the I-87123 more easily, we provide some useful and easy-to-use functions in I-87123 library. There are several kinds of library versions for the ViewPAC-2xWx, WinPAC-8000, XPAC-8000 and XPAC-8000-CE6 MCUs. When using the library, the library version must be confirmed. The following table shows the all functions provided by the I-87123 library.

Listen Mode	Function Name	Description
O	I87123_GetVersion	Get the version of the I-87123 library
O	I87123_SetFunctionTimeout	Set the max. timeout value of all functions
O	I87123_InitMaster	Activate the I-87123 module
O	I87123_ShutdownMaster	Remove all nodes and stop the master
O	I87123_SetMasterMode	Set the master to normal mode or listen mode
O	I87123_GetMasterMode	Get operation mode of the master
O	I87123_GetFirmwareVersion	Get the version of the I-87123 firmware
X	I87123_AddNode	Add a node into the I-87123 master manager
O	I87123_RemoveNode	Remove a node from the I-87123 master manager
X	I87123_ScanNode	Scan all nodes on the CANopen network
O	I87123_GetNodeList	Get the node list of the I-87123 master manager
X	I87123_NMTChangeState	Change the CANopen node state
O	I87123_NMTGetState	Get the CANopen node state
O	I87123_NMTGuarding	Start to the node guarding function
O	I87123_NMTHeartbeat	Start to the node heartbeat function
X	I87123_SDOReadData	Read data by using the upload SDO protocol
X	I87123_SDOWriteData	Write data by using the download SDO protocol
X	I87123_SDOAbortTransmit	Send the SDO abort message
X	I87123_PDOWrite	Use the PDO to write data to the CANopen node
X	I87123_PDORemote	Use the PDO to get data from the CANopen node
O	I87123_GetPDOLastData	Get the last Tx or Rx PDO data
O	I87123_GetRxPDOID	Get all COB-ID of the RxPDO of the specific slave
O	I87123_GetTxPDOID	Get all COB-ID of the TxPDO of the specific slave
X	I87123_InstallPDO	Install and enable a specific PDO
X	I87123_DynamicPDO	New or change a PDO mapping to the slave

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

X	I87123_RemovePDO	Remove a specific PDO mapping entry or object
X	I87123_ChangePDOID	Change the PDO COB-ID of a specific slave
O	I87123_GetPDOMapInfo	Obtain all the PDO-related information
O	I87123_InstallPDO_List	Install a PDO manually without check if it exists
O	I87123_RemovePDO_List	Remove a PDO object without checking real states
X	I87123_PDOUseEntry	Change the valid entry number of the PDO objects
X	I87123_PDOTxType	Set transmission type of the specific TxPDO
X	I87123_PDOEventTimer	Set event timer of the specific TxPDO
X	I87123_ChangeSYNCID	Change the SYNC COB-ID
O	I87123_SetSYNC_List	Set the SYNC COB-ID without check if it exists
O	I87123_GetSYNCID	Get the SYNC COB-ID
X	I87123_SendSYNCMsg	Send the SYNC message
X	I87123_GetCyclicSYNCInfo	Get all the cyclic sending SYNC information
X	I87123_ChangeEMCYID	Change the EMCY COB-ID
O	I87123_SetEMCY_List	Set the EMCY COB-ID and don’t check if it exists
O	I87123_GetEMCYID	Get the EMCY COB-ID
O	I87123_ReadEMCY	Get the last EMCY message of the slave
O	I87123_CheckNetworkEvent	Check if any events come from other devices
O	I87123_GetNMTErrorStatus	Get guarding/heartbeat fail event if it happens
O	I87123_GetNewEMCYData	Get EMCY event if it happens
O	I87123_GetMasterReadSDOEvent	Get “Read SDO event” sent from other masters
O	I87123_GetMasterWriteSDOEvent	Get “Write SDO event” sent from other masters
X	I87123_ResponseMasterSDOEvent	Reply Read/Write SDO demand from other masters
O	I87123_GetMasterRemotePDOEvent	Get “Remote PDO event” sent from other masters
O	I87123_GetMasterRxPDOEvent	Get “Output PDO event” sent from other masters
X	I87123_ResponseMasterPDOEvent	Reply Remote PDO message to other masters

Table 3.1 Description of functions

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

3.2. Function Return Code

The following table interprets all the return code returned by the I-87123.

Return Code	Error ID	Description
0	CPM_NoError	OK
2	CPM_OpenComErr	Open PAC backplane com port fail.
3	CPM_SlotNumberErr	There is no I-87123 on the specific slot No.
5	CPM_ConfigErr	The I-87123 hasn't been configured successfully
6	CPM_MasterInitErr	The I-87123 initialization error.
7	CPM_MasterNotInit	The I-87123 hasn't been initialized
8	CPM_ListenMode	The I-87123 is in listen mode now
9	CPM_NodeErr	Set node number error
10	CPM_NodeExist	The node had been added to the Master
11	CPM_AddModeErr	The mode parameter of AddNode function is error
12	CPM_TxBusy	Tx buffer is busy, please wait a minute to send again
13	CPM_UnknowCmd	This version of firmware doesn't support the function
14	CPM_CmdReceErr	I-87123 receive command of wrong length
15	CPM_DataEmpty	There is no data to receive
16	CPM_MemAllocErr	I-87123 has not enough memory
17	CPM_SendCycMsgErr	Cyclic message send error
18	CPM_StatusErr	NMT state of CANopen slave is error
20	CPM_SetGuardErr	Set Guarding and LifeTime parameter error
21	CPM_SetHbeatErr	Set Heartbeat parameter error
22	CPM_SegLenErr	SDO Segment receive error length
23	CPM_SegToggleErr	SDO Segment receive error toggle
24	CPM_SegWriteErr	SDO write segment error
25	CPM_Abort	The return message is an Abort message
26	CPM_PDOLenErr	PDO output data error
27	CPM_COBIDErr	The COB-ID isn't exist or isn't correct one
28	CPM_PDOInstErr	Install the PDO object error
29	CPM_PDODynaErr	The PDO mapping data is setting error
30	CPM_PDONumErr	The PDO number and COB-ID is not match
31	CPM_PDOSetErr	PDO parameter is setting error
32	CPM_PDOEntryErr	The PDO entry parameter is more then useful entry
33	CPM_SetCobIdErr	The EMCY or SYNC COB-ID is setting error
34	CPM_CycFullErr	There are already 5 cyclic message running

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

35	CPM_Timeout	Message response timeout
36	CPM_DataLenErr	Data length setting error
39	CPM_SendCmdErr	Send command to I-87123 error
40	CPM_Wait	Command is uncompleted (only for non-block mode)
41	CPM_Processing	Command is running (only for non-block mode)
42	CPM_ComPortPause	The I-87123 is pausing and can’t be accessed
50	CPM_LoadEDSErr	Loading the EDS file fails
51	CPM_EDSFormatErr	The format of the EDS file is incorrect

Table 3.2 Description of return code

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

3.3. CANopen Master Library Application Flowchart

In this section, it describes that the operation procedure about how to use the CANopen Master Library to build users applications. This information is helpful for users to apply the CANopen Master Library easily. Besides, the CANopen operation principles must be obeyed when build a CANopen master application. For example, if the CANopen node is in the pre-operational status, the PDO communication object is not allowed to use. For more detail information, please refer to the demo programs in the section 4.

When users programs apply the CANopen Master Library functions, the function I87123_InitMaster must be called first. The function is used to initialize I-87123 and configure the CAN port.

After initializing the CAN interface successfully, users need to use the function I87123_AddNode to install at least one CANopen device into the node list. The function can install slave node automatically or manually. When the user applies the function to install node manually, the PDO ID, SYNC ID, and EMCY ID of the node must be added manually by the functions I87123_InstallPDO_List, I87123_SetSYNC_List, and I87123_SetEMCY_List.

If the function I87123_InitMaster and I87123_AddNode has been executed, the communication services (NMT, SYNC, EMCY, SDO, and PDO services) can be used at any time before calling the function I87123_ShutdownMaster, because the function I87123_ShutdownMaster will stop all process created by the function I87123_InitMaster.

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

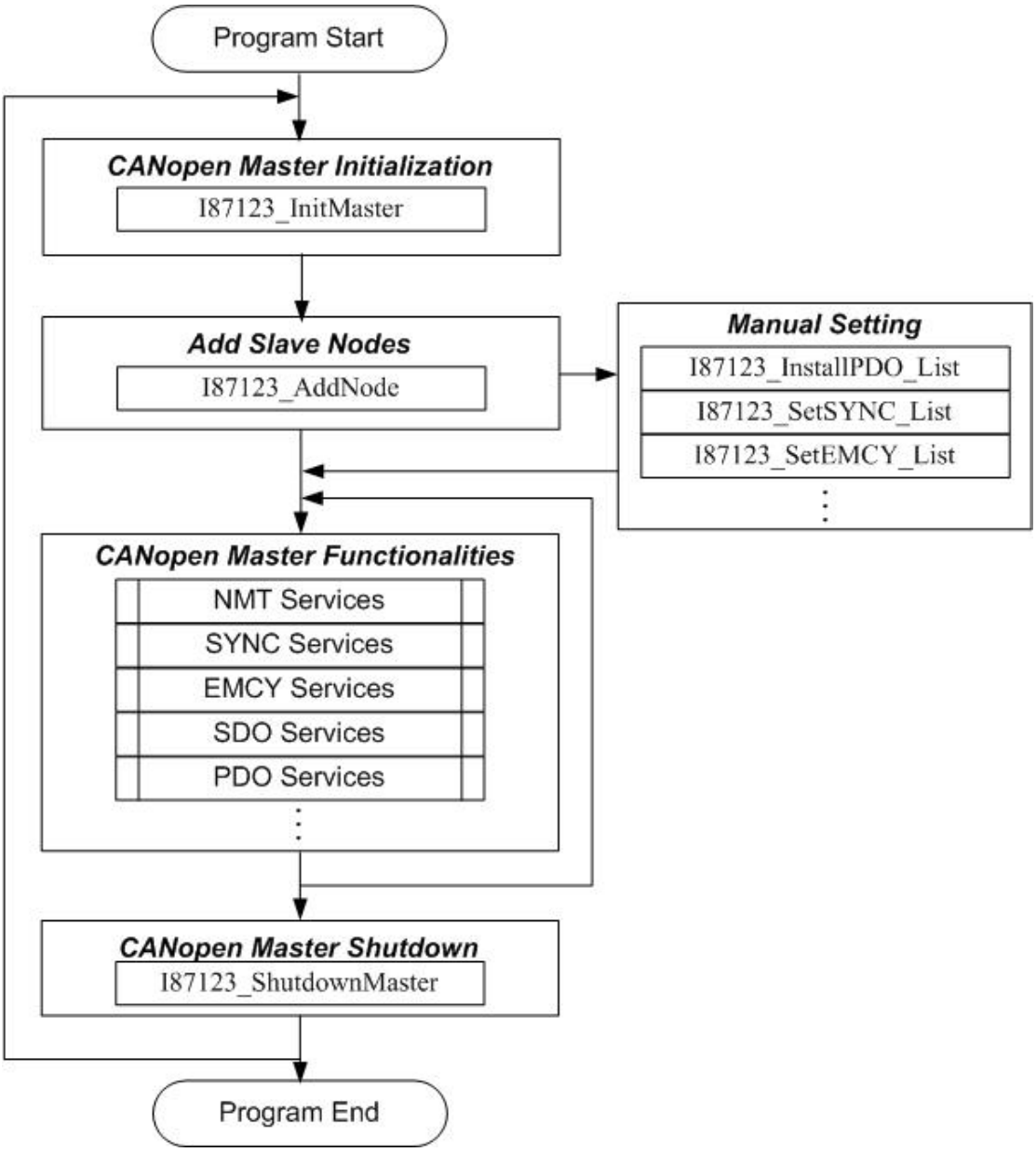


Figure 3.1 Main programming flow chart

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

3.4. Communication Services Introduction

NMT Services

The CANopen Master Library provides several NMT services functions, such as the functions I87123_AddNode, I87123_RemoveNode, I87123_NMTChangeState, I87123_NMTGetState, I87123_NMTHeartbeat and I87123_NMTGuarding. As the prerequisite for the master, the slave nodes have to be registered by the function I87123_AddNode with providing its Node-ID. The registered slave nodes can be individually removed from the node list by the function I87123_RemoveNode. Through NMT services, the NMT Master controls the state of the slaves. Table 3.3 is the command value and corresponding NMT command for the input parameters of the function I87123_NMTChangeState. When using the function I87123_NMTGetState, the slave status value and their descriptions are shown in the table 3.4. The Node Guarding and Heartbeat protocol is implemented via the function I87123_NMTGuarding and the function I87123_NMTHeartbeat. If the slave nodes are in the node list, users can change the node guarding parameters defined in the slave nodes by calling the function I87123_NMTGuarding and I87123_NMTHeartbeat.

Command Value	Description
1 (0x01)	Enter Operational
2 (0x02)	Stop
128 (0x80)	Enter Pre-Operational
129 (0x81)	Reset_Node
130 (0x82)	Reset_Communication

Table 3.3 NMT Command Specifier

State of Slave	Description
4 (0x04)	STOPPED
5 (0x05)	OPERATIONAL
127 (0x7F)	PRE-OPERATIONAL

Table 3.4 The State of The Slave

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

SDO Services

"Initiate SDO download protocol" or "Initiate SDO upload protocol" is used when the object data length $\leqq$ 4 bytes. If the object data length $>$ 4 bytes, "Download SDO segment protocol" or "Upload SDO segment protocol" will be used. Calling these two functions, I87123_SDOReadData and I87123_SDOWriteData, the initiate protocol and segment protocol will be selected automatically according to the object data length.

I87123_SDOAbortTransmit function can abort a pending SDO transfer at any time. Applying the abort service will have no confirmation from the salve device.

PDO Services

After using the I87123_AddNode to add a slave, the default TxPDOs and RxPDOs (TxPDO 1 ~ 10, RxPDO 1 ~ 10) will also be added to the I-87123's control list. If there are PDOs other than the default setting, the function I87123_InstallPDO is used for enabling these TxPDOs or RxPDOs object. After installing the PDOs, the function I87123_DynamicPDO can add or change the PDOs' mapping objects. Each PDO object supports 0~8 application objects. These application objects defined in the CANopen specification DS401, and they are mapped to the relative parameters of the DI/DO/AI/AO channels. After calling the function I87123_InstallPDO and I87123_DynamicPDO, the PDO communication object will be mapped and activated. If the PDO communication object is not needed no more, use the function I87123_RemovePDO to remove it.

When you want to output data via PDO, the function I87123_PDOWriteData is useful. This function can write all PDO 8-byte data or write some parts of PDO 8-byte data. Calling this function will send the specific data to the corresponding node via PDO protocol, and put the output data into PDO buffer at the same time. Therefore, you can check the output history of the PDO. But, if the connection between the I-87123W and the slave is lost, the history output information may be not the same with the real status on the slave.

In CANopen specification, you can get the TxPDOs data by applying the remote transmit request CAN frame. The function I87123_PDORemote is needed in this case. Or you can use I87123_GetPDOLastData to get the last RxPDO and TxPDO data from the PDO buffer.

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

SYNC Services

Calling the function I87123_SendSYNC starts the SYNC object transmission. This function supports single SYNC message transmission and cyclic SYNC message transmission. The timer parameter of the function I87123_SendSYNCMsg can adjust the cyclic period of the SYNC COB-ID sent by master. And the times parameter can set the sending times of the SYNC message. If the timer parameter is set to 0, the SYNC object transmission will be stopped. When the times parameter is set to non-zero value, the function will send the SYNC message until set the timer to 0 to stop the function.

EMCY Services

The Emergency messages are triggered by the occurrence of a device internal error situation. Users can use the function I87123_GetNewEMCYData to check if any EMCY messages of the slaves are received by the I-87123. If yes, call the function I87123_ReadEMCY to get the EMCY message from the specific slave.

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

3.5. Function Description

3.5.1. I87123_GetVersion

- **Description:**

  This function is used to obtain the version information of I87123.lib library.

- **Syntax:**

  float I87123_GetVersion(**void**)

- **Parameter:**

  None

- **Return:**

  LIB library version information.

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

3.5.2. I87123_SetFunctionTimeout

- **Description:**

  Sometimes, some function cost more time to complete its task, such as I87123_ScanNode. If some API of the I-87123 library never gets the feedback from the I-87123 until timeout value goes by, the error code “CPM_Timeout” will be returned. Through this function, the user can adjust the suitable maximum timeout value of all functions for application. Default timeout value is 1 second.

- **Syntax:**

  void I87123_SetFunctionTimeout(**BYTE** SlotNo, **WORD** Timeout)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Timeout: [input] Maximum timeout value of all functions (ms).

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

3.5.3. I87123_GetCANStatus

- **Description:**

  The function I87123_GetCANStatus can obtain the status of the CAN controller for the specific I-87123.

- **Syntax:**

  WORD I87123_GetCANStatus(**BYTE** SlotNo, **BYTE** *Status)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  ***Status:** [output] The pointer for obtaining the CAN status. It indicates the value of status register of SJA1000. Its meanings are described below.

Bit NO.	Description
7 (MSB)	Bus status. 1 for bus off, 0 for bus on.
6	Error status. 1 for at least one error, 0 for OK.
5	SJA1000 Transmit status. 1 for transmitting, 0 for idle
4	SJA1000Receive status. 1 for receiving, 0 for idles.
3	SJA1000 Transmit completes status. 1 for complete, 0 for incomplete.
2	SJA1000 Transmit buffer status. 1 for released, 0 for locked.
1	Data overrun status. 1 for SJA1000 reception buffer overrun, 0 for OK.
0 (LSB)	Receive buffer status. 1 for at least one message stored in the SJA1000 reception buffer, 0 for empty.

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

3.5.4. I87123_Pause

- **Description:**

  After using function I87123_InitMaster the backplane com port of the PAC will be occupied by the I-87123. And it will cause the other I-87k modules access fail. To avoid this situation, users must use the function I87123_Pause to release the backplane resources from the I-87123 before access other I-87k modules.

  Note: If there are all I-8k modules on users' PAC, I87123_Pause and I87123_Start are not needed to use.

- **Syntax:**

  void I87123_Pause(**WORD** Timeout_ms)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  Timeout_ms: [input] Maximum pause time value (ms). If set the parameter to 0, the I-87123 will be paused unlimitedly until I87123_Start be used.

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

3.5.5. I87123_Start

- **Description:**

  If users want to access the I-87123 after used function I87123_Pause, users can wait for pause timeout or use function I87123_Start to get the backplane resources immediately.

  Note: If there are all I-8k modules on users' PAC, I87123_Pause and I87123_Start are not needed to use.

- **Syntax:**

  void I87123_Start(**void**)

- **Parameter:**

  None

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

3.5.6. I87123_InitMaster

- **Description:**

  The function must be applied when configuring the CAN controller and initialize the I-87123. It must be called once before using other functions of the I87123.lib.

- **Syntax:**

  WORD I87123_InitMaster(**BYTE** SlotNo**, HANDLE** Com0, **BYTE** Node**,**
  BYTE BaudRate, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  Com0: [input] A handle to the open COM port 0. The parameter can obtain from uart_Open function of the WinPacSDK library.

  Node: [input] Master's node ID. If the parameter is 0, the master will be a normal master, and no other master can control it. If the parameter is 1 ~ 127 (different from other slave), other master can do some control to it through some ISR function.

  BaudRate: [input] The baudrate of the I-87123

Value	Baud rate	Value	Baud rate
0	10Kbps	4	250Kbps
1	20Kbps	5	500Kbps
2	50Kbps	6	800Kbps
3	125Kbps	7	1Mbps

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.7. I87123_ShutdownMaster

- **Description:**

  The function I87123_ShutdownMaster removes all the slaves that had added to master and stop all the functions of the I-87123. The function must be called before exit the users' application programs.

- **Syntax:**

  WORD I87123_ ShutdownMaster(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.8. I87123_MasterSendBootupMsg

- **Description:**

  To use the function I87123_MasterSendBootupMsg can let I-87123 sends a boot up message after I87123_InitMaster is called.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_MasterSendBootupMsg(**BYTE** SlotNo,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users' application will be held in the function until return. If set to 0, this function will return "CPM_ Processing" directly. This function will return its process status while users apply it with the same "SlotNo" again. If the procedure is still not complete, it will return "CPM_Wait".

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

3.5.9. I87123_SetMasterMode

- **Description:**

  This function can configure if the master is into listen mode or normal mode (default mode). In listen mode, the I-87123 can't send any CANopen message, and some functions will be useless in this mode. User can select normal mode or listen mode at any time after calling function I87123_InitMaster.

- **Syntax:**

  WORD I87123_SetMasterMode(**BYTE** SlotNo, **BYTE** Mode,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  Mode: [input] 1 is listen mode, and others are normal mode (default mode).

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.10. I87123_GetMasterMode

- **Description:**

  If user want to know what operation mode of the master is, call the function to get it.

- **Syntax:**

  WORD I87123_GetMasterMode(**BYTE** SlotNo, **BYTE** *Mode,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  ***Mode:** [output] 0 is normal mode (default mode), and 1 is listen mode.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.11. I87123_GetFirmwareVersion

- **Description:**

  The function can let users know what is the version number of the I-87123's firmware.

- **Syntax:**

  WORD I87123_GetFirmwareVersion(**BYTE** SlotNo, **WORD** *Fir_Ver, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  ***Fir_Ver:** [output] I-87123 firmware version information.

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.12. I87123_EDS_Load

- **Description:**

  The function I87123_EDS_Load can let users load EDS file for adding a CANopen slave with specified Node ID into the master node list. Using this function will not send any message to check if the slave is existent or not.

- **Syntax:**

  WORD I87123_EDS_Load(**BYTE** SlotNo, **BYTE** Node, **char** *FilePath,
  WORD DelayTime, **WORD** ResTimeout, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123W slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***FilePath:** [input] Absolute or relative file path of the EDS file.
  DelayTime: [input] The parameter defines the time interval between sending two messages from the I-87123W. It can avoid the master frequently sending messages that may overrun the buffer of the slave. Too large value of this parameter reduces the performance of the I-8123W. The unit of the parameter is ms. This parameter will be applied to the specified slave after finishing the ESD loading.
  ResTimeout: [input] The timeout value of the response of the CANopen slaves. When the master sends a CANopen message to the slave, it will wait the response until timeout if there is a response. The unit of this parameter is millisecond. This parameter will be applied to the specified slave after finishing the ESD loading.
  BlockMode: [input] 0 means this function is non-block-function, and 1 is block-function. If set this parameter to 1, the running procedure of the users' application is held until finishing this function. If 0, this function returns "CPM_ Processing" directly. While users apply it with the same "SlotNo" again, it returns the process status. If the procedure is still not complete, it returns "CPM_Wait".

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

3.5.13. I87123_AddNode

- **Description:**

The function I87123_AddNode can add a CANopen slave with specified Node ID into the master node list. There are two mode of the function. One is adding node automatically and another is adding node manually, and mode 3 is allowing the automatic adding the node while it boots up. In the automatic mode, after calling this function to add a slave, the slave will be into the operational state directly. And master will scan the TxPDO1 ~ TxPDO10 and RxPDO1 ~ RxPDO10 and install them into the firmware of the I-87123 if the slave supports these PDO objects. In the manual mode, this function will add a CANopen slave into the master node list only, and will not send any message to check if the slave is existent or not. Besides, the manual mode doesn't install the SYNC ID, EMCY ID, and any PDO object into the firmware of the I-87123. Users must call I87123_SetSYNC_List, I87123_SetEMCY_List, and I87123_InstallPDO_List to complete the object installation to finish the adding node process.

The added node can be removed from the master node list by the function I87123_RemoveNode.

- **Syntax:**

WORD I87123_AddNode(**BYTE** SlotNo, **BYTE** Node, **BYTE** AddMode,
WORD DelayTime, **WORD** ResTimeout,
BYTE BlockMode)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
Node: [input] Slave device Node-ID (1~127).
AddMode: [input] 1: Automatic mode. 2: Manual mode. 3: Wait Boot-up mode.
DelayTime: [input] The parameter is the shortest time interval between sending two messages from the I-87123. It can avoid the master to send too much CANopen messages in a short time that may let some slaves occur the errors. But if the

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

delay time is too long, the performance of the I-87123 is down. The unit of the parameter is ms.

ResTimeout: [input] The timeout value of the responded messages from the CANopen slaves. When the master sends a CANopen message to the slave, it will wait the feedback until timeout if there is a feedback. The unit of this parameter is millisecond.

BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.14. I87123_RemoveNode

- **Description:**

  The function I87123_RemoveNode removes the slave with the specified Node-ID from node list of the master. It requires a valid Node-ID, which has installed by the function I87123_AddNode before.

- **Syntax:**

  WORD I87123_RemoveNode(**BYTE** SlotNo, **BYTE** Node,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.15. I87123_RemoveAndResetNode

- **Description:**

  The function I87123_RemoveAndResetNode removes the slave with the specified Node-ID from node list of the master and reset the slave. It requires a valid Node-ID, which has installed by the function I87123_AddNode before.

- **Syntax:**

  WORD I87123_RemoveAndResetNode(**BYTE** SlotNo, **BYTE** Node,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  Node: [input] Slave device Node-ID (1~127).

  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users' application will be held in the function until return. If set to 0, this function will return "CPM_ Processing" directly. This function will return its process status while users apply it with the same "ComPort" and "Node" again. If the procedure is still not complete, it will return "CPM_Wait".

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

3.5.16. I87123_DelayAndResponseTimeout

- **Description:**

  Call the function to change CAN message minimum interval time and response timeout value of the slave node at any time.

- **Syntax:**

  WORD I87123_DelayAndResponseTimeout(**BYTE** SlotNo,
  BYTE Node, **WORD** DelayTime,
  WORD ResTimeout, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  DelayTime: [input] The parameter is the same as “DelayTime” of the function I87123_AddNode.
  ResTimeout: [input] The parameter is the same as “ResTimeout” of the function I87123_AddNode.
  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users’ application will be held in the function until return. If set to 0, this function will return “CPM_ Processing” directly. This function will return its process status while users apply it with the same “SlotNo” and “Node” again. If the procedure is still not complete, it will return “CPM_Wait”.

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

3.5.17. I87123_ScanNode

- **Description:**

  User can use the function I87123_ScanNode to know how many slave nodes are on the CANopen network.

- **Syntax:**

  WORD I87123_ScanNode(**BYTE** SlotNo, **BYTE** S_Node,
  BYTE E_Node, **BYTE** *NodeList, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  S_Node: [input] Start node ID.

  E_Node: [input] End node ID. This function will scan node ID from S_Node to E_Node. If S_Node >= E_Node, it will scan all slave node ID (1 ~ 127) on the CANopen network.

  ***NodeList:** [output] This is a 16-byte array parameter. Each bit of the parameter means a node ID, the bit is 0 means that the node ID doesn't exist and 1 means the node ID is on the CANopen network. For example, if the NodeList[0] is 0x16 (0001 0110), it means that the nodes with ID 1, 2, and 4 exist, and the nodes with ID 0, 3, 5, 6, and 7 don't exist. If the NodeList[1] is 0x41 (0100 0001), it means that the nodes with ID 8 and 14 exist, and the nodes with ID 9, 10, 11, 12, 13, and 15 don't exist.

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.18. I87123_GetNodeList

- **Description:**

  User can use the function I87123_GetNodeList to know how many slave nodes are added to the node list of the I-87123.

- **Syntax:**

  WORD I87123_GetNodeList(**BYTE** SlotNo, **BYTE** *NodeList,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  ***NodeList:** [output] This is a 16-byte array parameter. Each bit of the parameter means a node ID, the bit is 0 means the node ID not exist and 1 means the node ID now on the CANopen bus. For example, if the NodeList[0] is 0x16 (0001 0110), it means that the nodes with ID 1, 2, and 4 have been added into node list, and the nodes with ID 0, 3, 5, 6, and 7 don't. If the NodeList[1] is 0x41 (0100 0001), it means that the nodes with ID 8 and 14 have been added into node list, and the nodes with ID 9, 10, 11, 12, 13, and 15 don't.

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.19. I87123_NMTChangeState

- **Description:**

  The function I87123_NMTChangeState is used to change the NMT state of a slave. If the node parameter of this function is set to 0, the state of all nodes on the same CANopen network will be changed.

- **Syntax:**

  WORD I87123_NMTChangeState(**BYTE** SlotNo, **BYTE** Node,
  BYTE State, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127). Set this parameter to 0 to indicate all slave devices.
  State: [input] NMT command specifier.
  1: start
  2: stop
  128: enter PRE-OPERATIONAL
  129: Reset_Node
  130: Reset_Communication
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.20. I87123_NMTGetState

- **Description:**

  The function I87123_NMTGetState can get the NMT state from slaves.

- **Syntax:**

  WORD I87123_NMTGetState(**BYTE** SlotNo, **BYTE** Node, **BYTE** *State, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***State:** [output] The NMT state of the slave.

  4: STOPPED
  5: OPERATIONAL
  127: PRE-OPERATIONAL

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.21. I87123_NMTGuarding

- **Description:**

  Use the function I87123_NMTGuarding to set guard time and life time factor of the slave with the specified node ID. Then, the master will automatically send the guarding message to slave device according to the “GuardTime” parameters of this function. If the master doesn’t receive the confirmation of guarding message from the salve, the I-87123 will produce a Node_Guarding_Event event to users. Users may use the function I87123_InstallNMTErrISR to install a user-defined process to get the guarding fail event and process the guarding fail procedure. However, if the slave doesn’t receive the guarding message during the Node Life time period (Node Life time = “GuardTime” * “LifeTime”), it will be triggered to send the EMCY message. It is recommended that “LifeTime” value is set to more than 1.

  Take a note that following the CANopen specification, it is not allowed for one slave device to use both error control mechanisms Guarding Protocol and Heartbeat Protocol at the same time.

- **Syntax:**

  int I87123_NMGuarding(**BYTE** SlotNo**, BYTE** Node,
  WORD GuardTime, **BYTE** LifeTime, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  GuardTime: [input] Guard Time (1 ~ 65535 ms).
  LifeTime: [input] Life Time Factor (1 ~ 255).
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.22. I87123_NMTHeartbeat

- **Description:**

  Use the function I87123_NMTHeartbeat to set heartbeat time of the slave with the specified node ID and consume time with the I-87123. Then, the slave will automatically send the heartbeat message to master according to the “ProduceTime” parameters of this function. If the master doesn’t receive the heartbeat message from the slave until the “ConsumeTime” time (unit is millisecond) is up, the I-87123 will produce a “Heartbeat_Event” event to users. Users may use the function I87123_InstallNMTErrISR to install a user-defined process to get the heartbeat fail event and process the heartbeat fail procedure. It is recommended that “ConsumeTime” value is set to bigger than the “ProduceTime” value.

  Take a note that following the CANopen specification, it is not allowed for one slave device to use both error control mechanisms Guarding Protocol and Heartbeat Protocol at the same time.

- **Syntax:**

  int I87123_NMHartbeat(**BYTE** SlotNo, **BYTE** Node,
  WORD ProduceTime, **WORD** ConsumeTime,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ProduceTime: [input] Produce Time of slave device (1 ~ 65535 ms).
  ConsumeTime: [input] Consume Time of master (1 ~ 65535 ms).
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.23. I87123_SDOReadData

- **Description:**

  The function I87123_SDOReadData is useful to the SDO upload from a specified slave. When users use this function, pass the slave device node ID, object index and object subindex into this function. This function supports both expedition mode (less then 4-byte data length) and segment mode (more then 4-byte data length).

- **Syntax:**

  WORD I87123_SDOReadData(**BYTE** SlotNo, **BYTE** Node,
  WORD Index, **BYTE** SubIndex
  WORD *RDLen, **BYTE** *RData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Index: [input] Object index of object dictionary of slave devices.
  SubIndex: [input] Object subindex of object dictionary of slave devices.
  ***RDLen:** [output] Total data length.
  ***RData:** [output] SDO data respond from the specified slave device.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.24. I87123_SDOReadFile

- **Description:**

  The function I87123_SDOReadFile is useful as uploading the SDO data more then 1024 bytes. While users use the I87123_ReadData to read the SDO data and the return data length is more then 1024 byte. The SDO data are stored in a file. Users can use the function I87123_SDOReadFile for reading the SDO data from the file.

- **Syntax:**

  WORD I87123_SDOReadFile(**BYTE** SlotNo, **BYTE** Node,
  WORD Index, **BYTE** SubIndex,
  DWORD Start, **DWORD** Len,
  DWORD *RLen, **BYTE** *RData)

- **Parameter:**

  SlotNo: [input] I-81723 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Index: [input] Object index of object dictionary of slave devices.
  SubIndex: [input] Object subindex of object dictionary of slave devices.
  Start: [input] Start position for reading the SDO data from file. The range is from 0 to maximum length.
  Len: [input] Data length for reading the SDO data.
  ***RDLen:** [output] Returned data length in reality.
  ***RData:** [output] The SDO data respond from the specified slave device.

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

3.5.25. I87123_SDOWriteData

- **Description:**

The function I87123_SDOWriteData can send out a SDO message to the specified salve device. This procedure is also called download SDO protocol. The parameter node of the function I87123_SDOWriteData is used to point which slave device will receive this SDO message. Because the data length of each object stored in object dictionary is different, users need to know the data length when writing the object of the object dictionary of the specified slave devices. This function supports both expedition mode (less then 4-byte data length) and segment mode (more then 4-byte data length)

- **Syntax:**

WORD I87123_SDOWriteData(**BYTE** SlotNo, **BYTE** Node,
WORD Index, **BYTE** SubIndex, **WORD** TDLen,
BYTE *TData, **WORD** *RDLen, **BYTE** *RData,
BYTE BlockMode)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
Node: [input] Slave device Node-ID (1~127).
Index: [input] The index value of object of the object dictionary.
SubIndex: [input] The subindex value of object of the object dictionary.
TDLen: [input] Total data size to be written.
***TData:** [input] The SDO data written to slave device.
***RLen:** [output] Total data size of responded data.
***RData:** [output] SDO data responded from the specified slave device.
BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.26. I87123_SDOAbortTransmit

- **Description:**

  Call the function I87123_SDOAbortTransmit to cancel the SDO transmission. The parameter node of this function is used to specify which SDO communication will be terminated between the master and the specified salve device.

- **Syntax:**

  WORD I87123_SDOAbortTransmit(**BYTE** SlotNo, **BYTE** Node,
  WORD Index, **BYTE** SubIndex, **DWORD** *AData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Index: [input] The object index value of the object dictionary.
  SubIndex: [input] The object subindex value of the object dictionary.
  ***AData:** [input] Abort data to be send to slave.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.27. I87123_PDOWrite

- **Description:**

Call the function I87123_PDOWrite to send out a PDO message to the specified slave device. Before using this function, users need to use the function I87123_InstallPDO to install the PDO object into I-87123 if users want to use non-default PDO. Then, change the NMT state of the target slave device to operational mode by using the function I87123_NMTChangeState if the slave is not in the operational mode. Use the parameter offset to set the start position of the PDO data, and use the parameters "*TData" and "DLen" to point the data and data length. Then, this function will follow the data length to cover the slave PDO buffer of the I-87123 with the data from the specified start position, and send the data to the specified slave via PDO protocol at the same time.

- **Syntax:**

WORD I87123_PDOWrite(**BYTE** SlotNo, **WORD** Cobid, **BYTE** Offset,
BYTE DLlen, **BYTE** *TData,
BYTE BlockMode)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
Cobid: [input] COB-ID used by the PDO object.
Offset: [input] The start byte position of PDO data (0 ~ 7).
DLen: [input] data size (dlen + offset ≦ 8 (total length of the PDO).
***TData:** [output] The data pointer point to the PDO data.
BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.28. I87123_PDOWrite_Fast

- **Description:**

The function is like I87123_PDOWrite but does not check whether the PDO message really sends to CAN bus or not. So I87123_PDOWrite_Fast is about twice as faster than I87123_PDOWrite at high speed baud rate (greater than or equal to 250kbps)..

- **Syntax:**

WORD I87123_PDOWrite_Fast(**BYTE** SlotNo, **WORD** Cobid,
BYTE Offset, **BYTE** DLlen, **BYTE** *TData)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
Cobid: [input] COB-ID used by the PDO object.
Offset: [input] The start byte position of PDO data (0 ~ 7).
DLen: [input] data size (dlen + offset $\leqq$ 8 (total length of the PDO).
***TData:** [output] The data pointer point to the PDO data.

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

3.5.29. I87123_PDORemote

- **Description:**

  Use the function I87123_PDORemote to send a RTR (remote-transmit-request) PDO message to the slave device.

- **Syntax:**

  WORD I87123_PDORemote(**BYTE** SlotNo, **WORD** Cobid,
  BYTE *DLen, **BYTE** *TData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  ***DLen:** [output] The data length of the RTR PDO message.
  ***TData:** [output] The PDO message received from the slave device.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.30. I87123_PDORemote_Fast

- **Description:**

  Use the function I87123_PDORemote_Fast only to send a RTR (remote-transmit-request) PDO message to the slave device but not wait for the response message.

- **Syntax:**

  WORD I87123_PDORemote_Fast(**BYTE** SlotNo, **WORD** Cobid)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.

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

3.5.31. I87123_SetPDORemotePolling

- **Description:**

  If the CANopen slaves do not support the event timer function of the TxPDOs, using the function I87123_SetPDORemotePolling can config the most 125 TxPDO objects into the remote polling list. Then, the I-87123 will poll the configured TxPDOs and update the data into buffer automatically. Users can use I87123_GetMultiPDOData to get these TxPDOs data from the buffer faster and easily.

- **Syntax:**

  WORD I87123_SetPDORemotePoling(**BYTE** SlotNo, **BYTE** PDOCnt, **WORD** *Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  PDOCnt: [input] The number of the *Cobid array.
  ***Cobid:** [input] COB-ID array used by the TxPDO objects.
  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users' application will be held in the function until return. If set to 0, this function will return "CPM_ Processing" directly. This function will return its process status while users apply it with the same "SlotNo" and "Cobid" again. If the procedure is still not complete, it will return "CPM_Wait".

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

3.5.32. I87123_GetPDOLastData

- **Description:**

  Using the function I87123_GetPDOLastData can get the last data of the RxPDO and TxPDO from the PDO data buffer. The last PDO data is saved in PDO buffer, so it may not the same with the real situation.

- **Syntax:**

  WORD I87123_GetPDOLastData(**BYTE** SlotNo, **WORD** Cobid,
  BYTE *IsNew, **BYTE** *DLen, **BYTE** *RData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  ***IsNew:** [output] Check the data if had been read before. 0 is been read before, and 1 is new one.
  ***DLen:** [output] The data length of the PDO message.
  ***RData:** [output] The PDO message gets from the PDO buffer.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait . Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.33. I87123_GetMultiPDOData

- **Description:**

  This can get the last data of the RxPDO and TxPDO from the PDO data buffer such as the function I87123_GetPDOLastData. But the difference between these two functions is that user can use the function I87123_GetMultiPDOData to get maximum 50 PDO data at the same time.

- **Syntax:**

  WORD I87123_GetMuliPDOData(**BYTE** SlotNo, **BYTE** PDOCnt,
  WORD *Cobid, **BYTE** *IsNew,
  BYTE *DLen, **BYTE** *RData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  PDOCnt: [input] Total PDO number that want to get.
  ***Cobid:** [input] Maximum 50 COB-ID used by the PDO objects.
  ***IsNew:** [output] Check these PDO data if they have been read before. 0 is to be read before, and 1 is new one.
  ***DLen:** [output] The total data length obtained from the PDO buffer.
  ***RData:** [output] The PDO messages get from the PDO buffer.
  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users' application will be held in the function until return. If set to 0, this function will return "CPM_ Processing" directly. This function will return its process status while users apply it with the same "SlotNo" and "Cobid" again. If the procedure is still not complete, it will return "CPM_Wait".

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

3.5.34. I87123_GetRxPDOID

- **Description:**

  Use the function I87123_GetRxPDOID to get all the RxPDO COB-IDs of the specified slave, which have been installed to the master.

- **Syntax:**

  WORD I87123_GetRxPDOID(**BYTE** SlotNo, **BYTE** Node,
  BYTE *PDO_Cnt, **WORD** *ID_List,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***PDO_Cnt:** [output] The number of installed RxPDO.
  ***ID_List:** [output] The RxPDO COB-ID list.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.35. I87123_GetTxPDOID

- **Description:**

  Use the function I87123_GetTxPDOID to get all the TxPDO COB-IDs of the specified slave, which have been installed to the master.

- **Syntax:**

  WORD I87123_GetTxPDOID(**BYTE** SlotNo, **BYTE** Node,
  BYTE *PDO_Cnt, **WORD** *ID_List,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***PDO_Cnt:** [output] The number of installed TxPDO.
  ***ID_List:** [output] The TxPDO COB-ID list.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.36. I87123_InstallPDO

- **Description:**

  After calling the I87123_InstallPDO function, a PDO COB-ID will be installed in the PDO object list of the CANopen Master Library stack. If the slave device has defined the default PDO object in RxPDO1 ~ RxPDO10 and TxPDO1 ~ TxPDO10, in this case, these default PDO will be installed automatically while using the function I87123_AddNode with automatic mode. Otherwise, the TxPDOs or RxPDOs need to be installed manually by calling the function I87123_InstallPDO.

- **Syntax:**

  WORD I87123_InstallPDO(**BYTE** SlotNo, **BYTE** Node, **WORD** Cobid,
  BYTE RxTx, **WORD** PDO_No,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the PDO object.
  RxTx: [input] PDO type (0: RxPDO, 1: TxPDO).
  PDO_No: [input] PDO mapping object No (1 ~ 512).
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.37. I87123_DynamicPDO

- **Description:**

  This function can modify the mapping data of PDO object in the PDO object list of the CANopen Master Library stack. Take a note that before calling this function user must check if the PDO had been installed in the I-87123.

- **Syntax:**

  WORD I87123_DynamicPDO(**BYTE** Slot, **BYTE** Node, **WORD** Cobid, **BYTE** RxTx, **BYTE** Entry, **DWORD** EntryData, **BYTE** BlockMode)

- **Parameter:**

  Slot: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the PDO object.
  RxTx: [input] PDO type (0: RxPDO, 1: TxPDO).
  Entry: [input] PDO mapping object subindex value (1 ~ 8).
  EntryData: [input] A Double Word (4-byte) information of mapped application object. Users need to look up the user manual of the CANopen slave device to find the information of the application object, and obey the following example format to fill this parameter. **For Example, EntryData = 0x64110310:** Mapping to index 0x6411 and subindex 0x03 with data length 0x10 bit (2-byte).

  If the function parameters are as following, **Cobid = 0x333, RxTx = 0, Entry = 2, EntryData = 0x64110310**. This example will map the 16-bit data of index 0x6411 and subindex 0x03 object to the 2nd entry of COB-ID 0x333 RxPDO.

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.38. I87123_RemovePDO

- **Description:**

The function I87123_RemovePDO can remove a TxPDO or RxPDO object had installed by the I87123_InstallPDO or I87123_AddNode. This function also can remove single object mapped in TxPDO or RxPDO.

- **Syntax:**

WORD I87123_RemovePDO(**BYTE** SlotNo, **BYTE** Node,
WORD Cobid, **BYTE** Entry,
BYTE BlockMode)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
Node: [input] Slave device Node-ID (1~127).
Cobid: [input] COB-ID used by the PDO object.
Entry: [input] PDO mapping object subindex value (0 ~ 8). If this value is set to 0, the specified PDO object will be removed. If others (1 ~ 8), the specified subindex content of the PDO will be removed.
BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.39. I87123_ChangePDOID

- **Description:**

  Use the function I87123_ChangePDOID to change the PDO COB-ID from old “Old_Cobid” to new “New_Cobid” of a slave device.

- **Syntax:**

  WORD I87123_ChangePDOID(**BYTE** SlotNo, **WORD** Old_Cobid,
  WORD New_Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Old_Cobid: [input] Old COB-ID used by the PDO object.
  New_Cobid: [input] New COB-ID used by the PDO object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.40. I87123_GetPDOMapInfo

- **Description:**

  The function I87123_GetPDOMapInfo can get the mapping data information of the PDO object.

- **Syntax:**

  WORD I87123_GetPDOMapInfo(**BYTE** SlotNo, **WORD** Cobid,
  BYTE *RxTx, **BYTE** *Tx_Type,
  WORD *Event_Timer, **BYTE** *Entry_Cnt,
  DWORD *Map_Data, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  ***RxTx:** [output] PDO type (0: RxPDO, 1: TxPDO).
  ***Tx_Type:** [output] Transmission type.
  ***Event_Timer:** [output] PDO event timer.
  ***Entry_Cnt:** [output] Useful PDO entry number of the PDO object.
  ***Map_Data:** [output] Double Word array parameter. Response the mapping data of the PDO object's every useful entry.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.41. I87123_InstallPDO_List

- **Description:**

  This function is similar with the I87123_InstallPDO function. It can install the old or new PDO object in the PDO object list of the I-87123. It is the same as I87123_InstallPDO. But the I87123_InstallPDO_List doesn't send any message to check if the PDO object exists in the real slave. It just changes the list in the memory of the I-87123. It means that user can use this function to install PDO and change PDO mapping data arbitrarily without disturbing the CANopen network. After using this function, the I-87123 will process the slave PDOs which have the same IDs configured by the function I87123_InstallPDO_List. It is very useful when the I-87123 is running in listen mode. User can use the function I87123_RemovePDO_List to remove the PDO object which is installed by I87123_InstallPDO_List.

- **Syntax:**

  WORD I87123_InstallPDO_List(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** RxTx, **WORD** PDO_No,
  BYTE EntryUse, **DWORD** *EntryData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127). If the "Node" parameter is the I-87123 Node-ID, the parameters, EntryUse & EntryData, will be useless. In this case, users can use the function I87123_InstallRxPDOISR or I87123_InstallRemotePDOISR to install the users' callback function to process the received PDOs of the I-87123.
  Cobid: [input] COB-ID used by the PDO object.
  RxTx: [input] PDO type (0: RxPDO, 1: TxPDO).
  PDO_No: [input] PDO mapping object No (1 ~ 512).
  EntryUse: [input] Total entry of mapping object that will be installed.
  ***EntryData:** [input] Double Word array information of mapped application

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

object. For example:

If the configuration is “**Cobid = 0x333, RxTx = 0, PDO_No = 10, Entry = 2, EntryData[0] = 0x64110310, EntryData[1] = 0x62000108**”, it will map the RxPDO10 with COB-ID 0x333. The 1st entry is 16-bit data of index 0x6411 and subindex 0x03 object and the 2nd entry is 8-bit data of index 0x6200 and subindex 0x01 object.

BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.42. I87123_RemovePDO_List

- **Description:**

  The function I87123_RemovePDO_List can remove a TxPDO or RxPDO object had installed by the I87123_InstallPDO_List. This function also can remove single object mapped in the TxPDO or RxPDO.

- **Syntax:**

  WORD I87123_RemovePDO_List(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** Entry,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  Node: [input] Slave device Node-ID (1~127). If the “Node” parameter is the I-87123 Node-ID, users need to use the function I87123_RemoveRxPDOISR or I87123_RemoveRemotePDOISR to uninstall the callback functions which are used to process the received PDOs of the I-87123.

  Cobid: [input] COB-ID used by the PDO object.

  Entry: [input] PDO mapping object subindex value (0 ~ 8). If this parameter is set to 0, the specified PDO object will be removed. If others (1 ~ 8), the specified subindex content of the PDO will be removed.

  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.43. I87123_PDOUseEntry

- **Description:**

  Use this function to change the useful object mapping entry of PDO object. The useful entry starts from 1 to the parameter Entry. Therefore, if the parameter Entry is 0, it means that the PDO have no useful object mapping entry.

- **Syntax:**

  WORD I87123_PDOUseEntry(**BYTE** SlotNo, **WORD** Cobid,
  BYTE Entry, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  Entry: [input] Useful entry number of PDO mapping object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.44. I87123_PDOTxType

- **Description:**

  Use this function to change transmission type of TxPDO. The default transmission type is 255.

- **Syntax:**

  WORD I87123_PDOTxType(**BYTE** SlotNo, **WORD** Cobid,
  BYTE Tx_Type, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  Tx_Type: [input] Transmission type of TxPDO (0 ~ 255).

Description of transmission type

transmission type	PDO transmission				
	cyclic	acyclic	synchronous	asynchronous	RTR only
0		**X**	**X**		
1-240	**X**		**X**		
241-251	**- reserved -**				
252			**X**		**X**
253				**X**	**X**
254				**X**	
255				**X**	

BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.45. I87123_PDOEventTimer

- **Description:**

  Use this function to change the event timer of the TxPDO. The default event timer is 0. When the event timer of the PDO object of the slave is more than 0, the PDO will be sent to master due to the parameter “Timer” automatically.

- **Syntax:**

  WORD I87123_PDOEventTimer(**BYTE** SlotNo, **WORD** Cobid,
  WORD Timer, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  Timer: [input] Event timer of TxPDO. The unit is millisecond.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.46. I87123_PDOInhibitTime

- **Description:**

  Use this function to set the inhibit time to the TxPDO. This time is a minimum interval for PDO transmission.

- **Syntax:**

  WORD I87123_PDOInhibitTime(**BYTE** SlotNo, **WORD** Cobid,
  WORD Time, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the PDO object.
  Time: [input] Inhibit time of TxPDO.
  BlockMode: [input] 0 means this function is non-block-function, and 1 means this function is block-function. If set this parameter to 1, the running procedure of the users' application will be held in the function until return. If set to 0, this function will return "CPM_ Processing" directly. This function will return its process status while users apply it with the same "SlotNo" and "Cobid" again. If the procedure is still not complete, it will return "CPM_Wait".

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

3.5.47. I87123_ChangeSYNCID

- **Description:**

  Use the function I87123_ChangeSYNCID to change the SYNC COB-ID of a slave device.

- **Syntax:**

  WORD I87123_ChangeSYNCID(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the SYNC object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.48. I87123_SetSYNC_List

- **Description:**

  If the user uses I87123_AddNode function to add the slave with manual mode, the function I87123_SetSYNC_List must be called while the SYNC ID of the slave needs to be changed or be set. The function I87123_SetSYNC_List can only change the SYNC COB-ID in the COB-ID list of the I-87123, the real value stored in the slave device may be different from the configuration which is set by the function I87123_SetSYNC_List. The users need to confirm that by themselves.

- **Syntax:**

  WORD I87123_SetSYNC_List(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the SYNC object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.49. I87123_GetSYNCID

- **Description:**

  This function can get the SYNC ID from the COB-ID list of the I-87123.

- **Syntax:**

  WORD I87123_GetSYNCID(**BYTE** SlotNo, **BYTE** Node,
  WORD *Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***Cobid:** [output] Return the COB-ID used by the SYNC object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.50. I87123_SendSYNCMsg

- **Description:**

  Use the function I87123_SendSYNCMsg to send a SYNC message with specified COB-ID cyclically. If the parameter “Timer” is 0, the SYNC message will be stopped. If the parameter “Timer” is more than 0, the function will send SYNC message per “Timer” millisecond until finish the parameter “Times”. When the “Times” is set to 0, the function will send SYNC message continuously until set “Timer” to 0. Users can set at most 5 SYNC messages with different ID to be sent cyclically.

- **Syntax:**

  WORD I87123_SendSYNCMsg(**BYTE** SlotNo, **WORD** Cobid,
  WORD Timer, **DWORD** Times,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Cobid: [input] COB-ID used by the SYNC object.
  Timer: [input] SYNC message transmission period. If the timer is 0, the SYNC message will be stopped.
  Times: [input] SYNC message transmission times. If the time is 0, the SYNC message will be sending until “Timer” is set to 0.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn’t finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.51. I87123_GetCyclicSYNCInfo

- **Description:**

  This function can get at most 5 SYNC messages information which have been configured by the function I87123_SendSYNCMsg. User can know what SYNC ID had been set.

- **Syntax:**

  WORD I87123_GetCyclicSYNCInfo(**BYTE** SlotNo, **WORD *Cobid,**
  WORD *Timer, **DWORD** *Times,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  ***Cobid:** [input] COB-ID array. Return most 5 SYNC ID.
  ***Timer:** [input] 5 WORD array. Each value is the cyclic period of the SYNC message.
  ***Times:** [input] 5 Double WORD array. Each one is the SYNC message sending times that set before.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.52. I87123_ChangeEMCYID

- **Description:**

  Use the function I87123_ChangeEMCYID to change the EMCY COB-ID of a specific slave device.

- **Syntax:**

  WORD I87123_ChangeEMCYID(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the EMCY object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.53. I87123_SetEMCY_List

- **Description:**

  If the user uses I87123_AddNode function to add the slave with manual mode, the function I87123_SetEMCY_List must be called while the EMCY ID of the slave needs to be changed or be set. I87123_SetEMCY_List only can change the EMCY COB-ID in the COB-ID list of the I-87123. Afterwards, the I-87123 process the EMCY messages with the specific EMCY COB-ID which is configured by the function I87123_SetEMCY_List.

- **Syntax:**

  WORD I87123_SetEMCY_List(**BYTE** SlotNo, **BYTE** Node,
  WORD Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  Cobid: [input] COB-ID used by the EMCY object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.54. I87123_GetEMCYID

- **Description:**

  This function can get the EMCY ID from the COB-ID list of the I-87123.

- **Syntax:**

  WORD I87123_GetEMCYID(**BYTE** SlotNo, **BYTE** Node,
  WORD *Cobid, **BYTE** BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***Cobid:** [output] Return the COB-ID used by the EMCY object.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.55. I87123_ReadLastEMCY

- **Description:**

  This function can check if one slave had produced EMCY. If yes, this function will return the last EMCY message of the specific slave.

- **Syntax:**

  WORD I87123_ReadLastEMCY(**BYTE** SlotNo, **BYTE** Node,
  BYTE *IsNew, **BYTE** *RData,
  BYTE BlockMode)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  Node: [input] Slave device Node-ID (1~127).
  ***IsNew:** [output] Check the data if had been read before. 0 is been read before, and 1 is new one.
  ***RData:** [output] 8-byte EMCY message gets from the EMCY buffer.
  BlockMode: [input] The value 0 and 1 indicate the non-block mode and block mode for the function respectively. In the non-block mode, the function starts its procedure after calling it. Then, return a return code immediately. If the procedure doesn't finish, the return code is CPM_Wait. Users can recall the function with same parameters to check if it finishes. In the block mode, calling the function will hang up the main program until the procedure of the function finishes.

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

3.5.56. I87123_GetBootUpNodeAfterAdd

- **Description:**

  If users don't know that which slave node occurred the boot-up message and which I-87123 received it. Users can use the function I87123_GetBootUpNodeAfterAdd. This function can get not only the slave node ID but also the slot number of the I-87123. The parameter, SlotNo, indicates the number of the I-87123 which receives the boot-up message. The parameter, Node, is the ID of the slave node which produces the boot-up message. The I87123_GetBootUpNodeAfterAdd function is usually applied with the function I87123_InstallBootUpISR. But note that, this function can only get the slave node ID that has already been added (I87123_AddNode) to the I-87123.

- **Syntax:**

  WORD I87123_GetBootUpNodeAfterAdd(**BYTE** *SlotNo, **BYTE** *Node)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the boot-up message.

  ***Node:** [output] Get the slave node ID of the received boot-up message.

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

3.5.57. I87123_GetEMCYData

- **Description:**

  If users don't know that which slave node occurred the EMCY message and which I-87123 received it. Users can use the function I87123_GetEMCYData. This function can get not only the EMCY message with the slave node ID but also the slot number of the I-87123. The parameter, SlotNo, indicates the number of the I-87123 which receives the EMCY message. The parameter, Node, is the ID of the slave node which produces the EMCY. The parameters, *RData, is the 8-bytes EMCY message data. The function I87123_GetEMCYData is usually applied with the function I87123_InstallEMCYISR. About the demo please refer to the section 4.1.2 NMT_Protocol.

- **Syntax:**

  WORD I87123_GetEMCYData(**BYTE** *SlotNo, **BYTE** *Node,
  BYTE *RData)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the EMCY message.

  ***Node:** [output] Get the slave node ID of the received EMCY message.

  ***RData:** [output] 8-byte EMCY message obtained from the EMCY buffer.

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

3.5.58. I87123_GetNMTErr

- **Description:**

  User can use the function I87123_GetNMTError to check if the I-87123 gets NMT Error Event for any slave node. The parameters of the function indicate that which I-87123 gets the NMT Error Event, which node produces this event, and what kind of event it is. The parameter, SlotNo, indicates the number of the I-87123 which indicates the Heartbeat_Event or Node_Guarding_Event. The parameter, Node, is the ID of the slave node which responds the heartbeat protocol or guarding protocol. The parameter, NMTErr, is the NMTErr event mode. If the NMTErr event is Node_Guarding_Event, the NMTErr parameter is CPM_Node_Guarding_Event or else the CPM_Heartbeat_Event is obtained. The function I87123_GetNMTError is usually applied with the function I87123_InstallNMTErrISR. About the demo please refer to the section 4.1.2 NMT_Protocol.

- **Syntax:**

  WORD I87123_GetNMTErr(**BYTE** *SlotNo, **BYTE** *Node,
  BYTE *NMTErr)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the NMT Error Event.

  ***Node:** [output] Get the slave node ID of the NMT Error Event.

  ***NMTErr:** [output] The value CPM_Node_Guarding_Event indicates the Node Guarding Event, and the CPM_Heartbeat_Event is the Heartbeat Event.

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

3.5.59. I87123_InstallBootUpISR

- **Description:**

  This function allows the user to apply the slave boot-up IST (interrupt service thread). When the user puts his boot-up process into this function, all the boot-up triggered by the slaves will go to the boot-up IST. If the boot-up message of a slave which has been added to the I-87123 is happen, the I-87123 will go into the boot-up process to do some specified mechanism which follows the user's boot-up process.

- **Syntax:**

  WORD I87123_InstallBooUpISR(**BYTE** SlotNo, **void** (*BOOTISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*BOOTISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of the user's boot-up process. This process is usually applied with the function I87123_GetBootUpNodeAfterAdd.

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

3.5.60. I87123_RemoveBootUpISR

- **Description:**

  When the user doesn't need the boot-up IST function, call this function to remove the user's IST.

- **Syntax:**

  WORD I87123_RemoveBootUpISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.61. I87123_InstallEMCYISR

- **Description:**

  This function allows the user to apply the EMCY IST (interrupt service thread). When the user puts his EMCY process into this function, all the EMCY triggered by the slaves will go to the EMCY IST. If the EMCY of a slave is happen, the I-87123 will go into the EMCY process to do some security mechanism which follows the user's EMCY process.

- **Syntax:**

  WORD I87123_InstallEMCYISR(**BYTE** SlotNo, **void** (*EMCYISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*EMCYISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of the user's EMCY process. This process is usually applied with the function I87123_GetEMCYData.

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

3.5.62. I87123_RemoveEMCYISR

- **Description:**

  When the user doesn't need the EMCY IST function, call this function to remove the user's IST.

- **Syntax:**

  WORD I87123_RemoveEMCYISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.63. I87123_InstallNMTErrISR

- **Description:**

  This function allows the user to apply NMTErr IST (interrupt service thread). When the user puts his NMTErr process into this function, all the Heartbeat_Event and Node_Guarding_Event triggered by the slaves will go to the IST. If the user had used the I87123_NMTGuarding to enable the guarding protocol or had used the I87123_Heartbeat to enable the heartbeat protocol, the I-87123 will go into the NMTErr IST to do the user's NMTErr process while the guarding confirms or heartbeat indicator doesn't be received.

- **Syntax:**

  WORD I87123_InstallNMTErrISR(**BYTE** SlotNo,
  void (*NMTERRISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*NMTERRISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of the user's process. This process is usually applied with the function I87123_GetNMTError.

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

3.5.64. I87123_RemoveNMTErrISR

- **Description:**

  When the user doesn't need the NMTErr IST function, call this function to remove the user's IST.

- **Syntax:**

  WORD I87123_RemoveNMTErrISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.65. I87123_GetMasterReadSDOEvent

- **Description:**

  Using this function can get all the read SDO messages sent to the specific node ID of the I-87123. For example, the I-87123 is initialized with node ID 2. If someone sends an SDO message with the COB-ID 0x602 to the I-87123 for reading an object, users can use the function I87123_GetMasterReadSDOEvent for obtaining this SDO message, and respond some information to the SDO sender. The parameter, SlotNo, indicates the number of the I-87123 which receives the read SDO message. The parameters, Index and SubIndex, are the object indicator. The function I87123_GetMasterReadSDOEvent is usually applied with the function I87123_InstallReadSDOISR. About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_GetMasterReadSDOEvent(**BYTE** *SlotNo,
  WORD *Index, **BYTE** *SubIndex)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the read SDO message.

  ***Index:** [output] Get the object index of the SDO message.

  ***SubIndex:** [output] Get the object subindex of the SDO message.

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

3.5.66. I87123_GetMasterWriteSDOEvent

- **Description:**

  Using this function can get all the write SDO messages sent to the specific node ID of the I-87123. For example, the I-87123 is initialized with node ID 2. If someone sends an SDO message with the COB-ID 0x602 to the I-87123 for writing an object, users can use the function I87123_GetMasterWriteSDOEvent for obtaining this SDO message. The parameter, SlotNo, indicates the number of the I-87123 which receives the write SDO message. The parameters, Index and SubIndex, are the object indicator. The parameter WLen is the data length of the parameter *WData. The function I87123_GetMasterWriteSDOEvent is usually applied with the function I87123_InstallWriteSDOISR. About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_GetMasterWriteSDOEvent(**BYTE** *SlotNo,
  WORD *Index, **BYTE** *SubIndex,
  BYTE *WLen, **BYTE** *WData)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the write SDO message.

  ***Index:** [output] Get the object index of the SDO message.

  ***SubIndex:** [output] Get the object subindex of the SDO message.

  ***WLen:** [output] The data length of the write data.

  ***WData:** [output] Return 0~4 bytes of the SDO write data.

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

3.5.67. I87123_ResponseMasterSDO

- **Description:**

Using this function can reply the SDO messages to the SDO sender. For example, the I-87123 is initialized with node ID 2. If someone sends a SDO message with the COB-ID 0x602 for reading or writing the object of the I-87123, the I-87123 need to reply the corresponding SDO message, use the function I87123_ResponseMasterSDO to do it. When users implement the function I87123_ResponseMasterSDO, the I-87123 will send a SDO message with COB-ID 0x582 to the CANopen network. This function is usually applied with the SDO ISR series function .About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

Note1: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

Note2: If the I-87123 want to reply a SDO Abort message, please use the function I87123_SDOAbortTransmit (section 3.5.20) to do it.

- **Syntax:**

WORD I87123_ResponseMasterSDO(**BYTE** SlotNo, **BYTE** ResType,
WORD Index, **BYTE** SubIndex,
BYTE Len, **BYTE** *Data)

- **Parameter:**

SlotNo: [input] I-87123 slot number (0~7).
ResType: [input] Response type of SDO message, 0 for replying the read SDO message and 1 for the write SDO message.
Index: [input] Object index of object dictionary of slave devices.
SubIndex: [input] Object subindex of object dictionary of slave devices.
Len: [input] The data length of the response data. If the ResType is 1 (write type), the Len and *Data parameter is useless.
***Data:** [input] Return 0~4 bytes of the SDO response data.

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

3.5.68. I87123_InstallReadSDOISR

- **Description:**

  This function allows the user to apply the ReadSDO IST (interrupt service thread) of I-87123. When the user puts his read SDO process into this function, all the read SDO messages sent to the specified I-87123 will trigger the IST. For example, the I-87123 is initialized with node ID 2. If someone sends an SDO message with the COB-ID 0x602 for reading the object of the I-87123, the I-87123 will go into the IST if the user has installed it.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_InstallReadSDOISR(**BYTE** SlotNo,
  void (*RSDOISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*RSDOISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of the user's process. This process is usually used with the function I87123_GetMasterReadSDOEvent.

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

3.5.69. I87123_RemoveReadSDOISR

- **Description:**

  When the user doesn't need the ReadSDO IST function, call this function to remove the user IST.

- **Syntax:**

  WORD I87123_RemoveReadSDOISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.70. I87123_InstallWriteSDOISR

- **Description:**

  This function allows the user to apply the WriteSDO IST (interrupt service routine) of the I-87123. When the user puts the process into this function, all the written SDO messages sent to the specified I-87123 will trigger the IST. For example, the I-87123 is initialized with node ID 2. If someone sends an SDO message with the COB-ID 0x602 to the I-87123 for writing an object, the I-87123 will go into the IST if the user has installed it.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_InstallWriteSDOISR(**BYTE** SlotNo,
  void (*WSDOISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*WSDOISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of user's process. This process is usually used with the function I87123_GetMasterWriteSDOEvent.

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

3.5.71. I87123_RemoveWriteSDOISR

- **Description:**

  When the user doesn't need the WriteSDO IST function, call this function to remove the user IST.

- **Syntax:**

  WORD I87123_RemoveWriteSDOISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.72. I87123_GetMasterRemotePDOEvent

- **Description:**

  Using this function can get all the Remote PDO messages sent to the I-87123. For example, the I-87123 has used the function I87123_InstallPDO_List to install an I-87123's TxPDO object with the COB-ID 0x444. If someone sends a Remote PDO message with the COB-ID 0x444 to the I-87123, users can use the function I87123_GetMasterRemotePDOEvent to get this PDO message. The parameter, SlotNo, indicates the number of the I-87123 which receives the Remote PDO message. The parameter, Cobid, is the PDO COB-ID sent to the I-87123. This function is usually used with the function I87123_InstallRemotePDOISR. About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_GetMasterRemotePDOEvent(**BYTE** *SlotNo,
  WORD *CobId)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the Remote PDO message.

  ***CobId:** [output] Return the COB-ID of the Remote PDO message.

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

3.5.73. I87123_GetMasterRxPDOEvent

- **Description:**

  Using this function can get all the RxPDO messages sent to the I-87123. For example, the I-87123 has used the function I87123_InstallPDO_List to install an I-87123‘s RxPDO object with the COB-ID 0x333. If someone sends an RxPDO message with the COB-ID 0x333 to the I-87123, users can use this function to get this RxPDO message. The parameter, SlotNo, indicates the number of the I-87123 which receives the RxPDO message. The parameter, Cobid, is the RxPDO COB-ID ID. The two parameters, *WLen and *WData, are the data length and contents of the RxPDO message. This function is usually applied with the function I87123_InstallRxPDOISR. About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_GetMasterRxPDOEvent(**BYTE** *SlotNo, **WORD** *CobId, **BYTE** *WLen, **BYTE** *WData)

- **Parameter:**

  ***SlotNo:** [output] Get the slot number of the I-87123 which receives the RxPDO message.

  ***CobId:** [output] Return the RxPDO COB-ID.

  ***WLen:** [output] The data length of the RxPDO data.

  ***WData:** [output] Return 0~4 bytes of the RxPDO data.

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

3.5.74. I87123_ResponseMasterPDO

- **Description:**

  Using this function can reply the Remote PDO messages to the sender. For example, the I-87123 has used I87123_InstallPDO_List to install a TxPDO object with the COB-ID 0x444. If someone sends a Remote PDO message with the COB-ID 0x444 to the I-87123, and the I-87123 needs to reply a TxPDO message, users can use this function to do it. When users implement the function I87123_ResponseMasterPDO, the I-87123 will send a TxPDO message to the CANopen network. This function is usually used with the I87123_InstallRemotePDOISR function .About the demo please refer to the section 4.1.6 SDO_PDO_ISR.

  Note: The function is valid while the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_ResponseMasterPDO(**BYTE** SlotNo, **WORD** CobId,
  BYTE Len, **BYTE** *Data)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).
  CobId: [input] TxPDO COB-ID for replying the PDO message.
  Len: [input] The data length of the response data.
  ***Data:** [input] Return the COB-ID of the TxPDO PDO message.

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

3.5.75. I87123_InstallRxPDOISR

- **Description:**

  This function allows the user to apply the RxPDO IST (interrupt service routine) of the I-87123. When the user puts his process into this function, all the RxPDO messages with the I-87123's PDO objects will trigger the IST. For example, the I-87123 has used I87123_InstallPDO_List to install a PDO object with the COB-ID 0x333 of the I-87123. If some one sends a PDO message with the COB-ID 0x333 to the I-87123, the I-87123W will go into the IST if the user had installed it.

  Note: The function will usefully when the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_InstallRxPDOISR(**BYTE** SlotNo,
  void (*RXPDOISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*RXPDOISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of user's process. This process is usually used with the function I87123_GetMasterRxPDOEvent.

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

3.5.76. I87123_RemoveRxPDOISR

- **Description:**

  When the user doesn't need the RxPDO IST function, call this function to remove the user IST.

- **Syntax:**

  WORD I87123_RemoveRxPDOISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

3.5.77. I87123_InstallRemotePDOISR

- **Description:**

  This function allows the user to apply the RemotePDO IST (interrupt service routine) of the I-87123. When the user puts his process into this function, all the Remote PDO messages of the I-87123's PDO objects will trigger the IST. For example, the I-87123 has used I87123_InstallPDO_List to install a TxPDO object with the COB-ID 0x444 of the I-87123. If some one sends a Remote PDO message with the COB-ID 0x444 to the I-87123, the I-87123 will go into the IST if the user had installed it.

  Note: The function will usefully when the Node parameter of the function I87123_InitMaster is > 0.

- **Syntax:**

  WORD I87123_InstallRemotePDOISR(**BYTE** SlotNo,
  void (*REMOTEPDOISR)( ))

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

  (*REMOTEPDOISR)( ): [input] The pointer which points a function with format "void XXX( )". The XXX is the function name of user's process. This process is usually used with the function I87123_GetMasterRemotePDOEvent.

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

3.5.78. I87123_RemoveRemotePDOISR

- **Description:**

  When the user doesn't need the RemotePDO IST function, call this function to remove the user IST.

- **Syntax:**

  WORD I87123_RemoveRemotePDOISR(**BYTE** SlotNo)

- **Parameter:**

  SlotNo: [input] I-87123 slot number (0~7).

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

4. Demo Programs

The I-87123 provides 9 demos of the various applications for NMT protocol, SDO protocol, PDO protocol, NMT Error event…etc. All these demos support eVC++ 4.0, VB.net 2005, and C# 2005. Users can find these demos in the fieldbus CD or on the web site.

The path of field bus CD

fieldbus_cd://canopen/master/I_87123

The address of the web site

http://www.icpdas.com/products/Remote_IO/can_bus/I-87123.htm

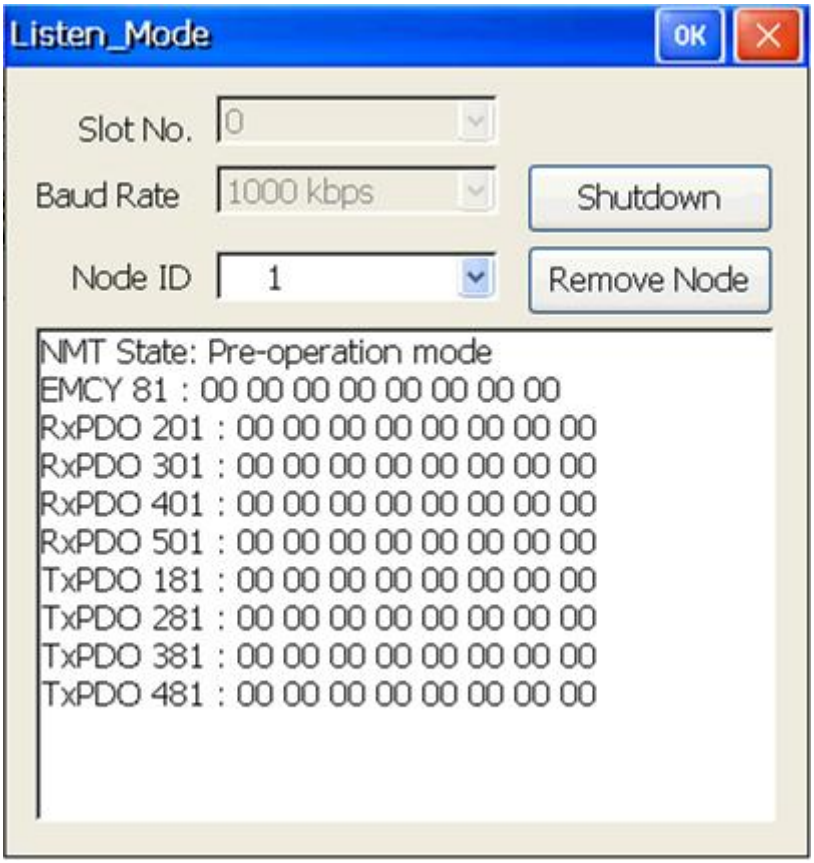
4.1. Brief of the demo programs

These demo programs are developed for demonstrating how to use the CANopen master library to apply the general CANopen communication protocol. These demo programs provide the SDO, PDO, NMT, SYNC communication applications. Each demo program includes some functions of the CANopen master library. The relationship between CANopen master library functions and demo programs are displayed in the following description.

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

4.1.1. Listen_Mode

Initialize the I-87123 with the “listen mode” and add slave nodes with the “manual mode”, then the I-87123 listens CANopen messages only and does not send any message to the CANopen network. In this demo, the I-87123 will listen NMT state, 4 TxPDO messages (with the COB ID 0x180+Node ID, 0x280+Node ID, 0x380+Node ID, and 0x480+Node ID), 4 RxPDO messages (with the COB ID 0x200+Node ID, 0x300+Node ID, 0x400+Node ID, and 0x500+Node ID), and the EMCY messages.



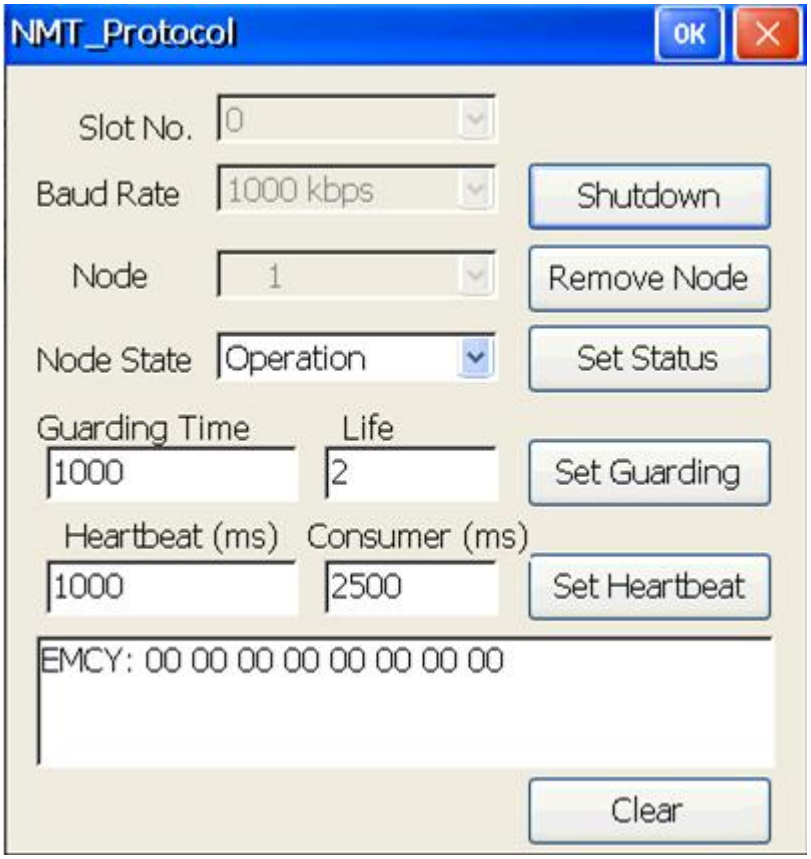
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_EDS_Load, I87123_SetMasterMode, I87123_NMTGetState, I87123_InstallPDO_List, I87123_GetPDOLastData, I87123_GetRxPDOID, I87123_GetTxPDOID, I87123_SetEMCY_List, I87123_GetEMCYID, I87123_ReadEMCY.

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

4.1.2. NMT_Protocol

This is a NMT network control demo. The demo not only tells users how to control the NMT status of a specific slave node, but also how to protect the slave through the “Guarding” and “Heartbeat” functions.



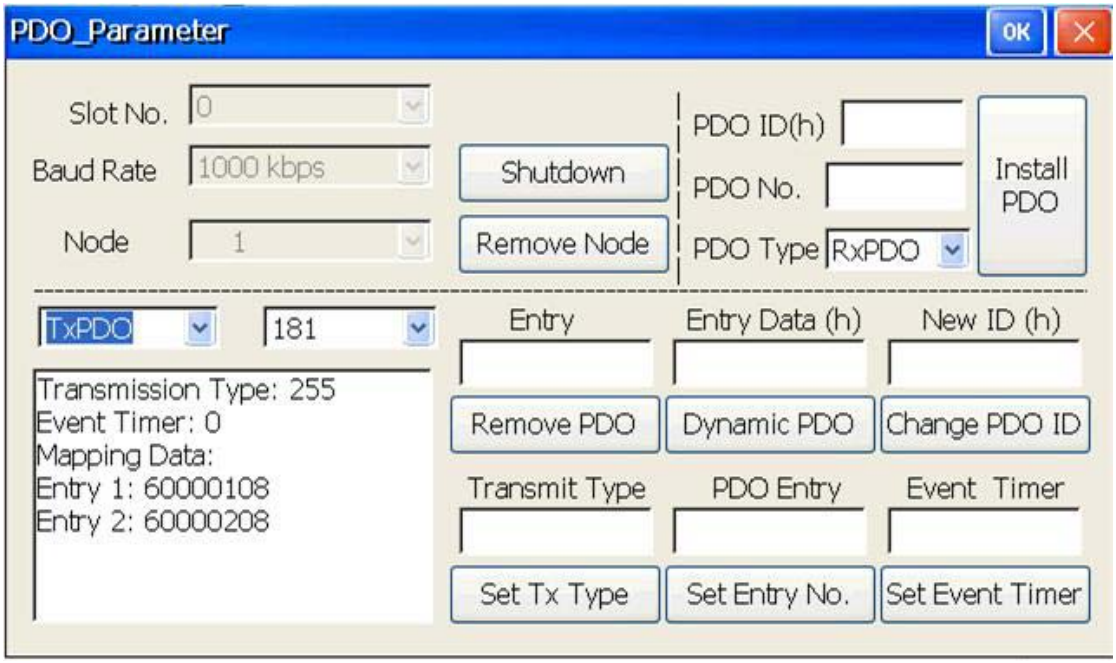
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_NMTChangeState, I87123_NMTGuarding, I87123_NMTHeartbeat, I87123_CheckNetworkEvent, I87123_GetNMTError, I87123_GetEMCYData

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

4.1.3. PDO_Parameter

Sometimes, the default PDO configuration can't satisfy users. Users need to change the configuration of the PDO related parameters such as transmission type, PDO ID, event timer, dynamic PDO, and so forth. This demo will demonstrate how to change settings of these PDO parameters and show the configuration result.



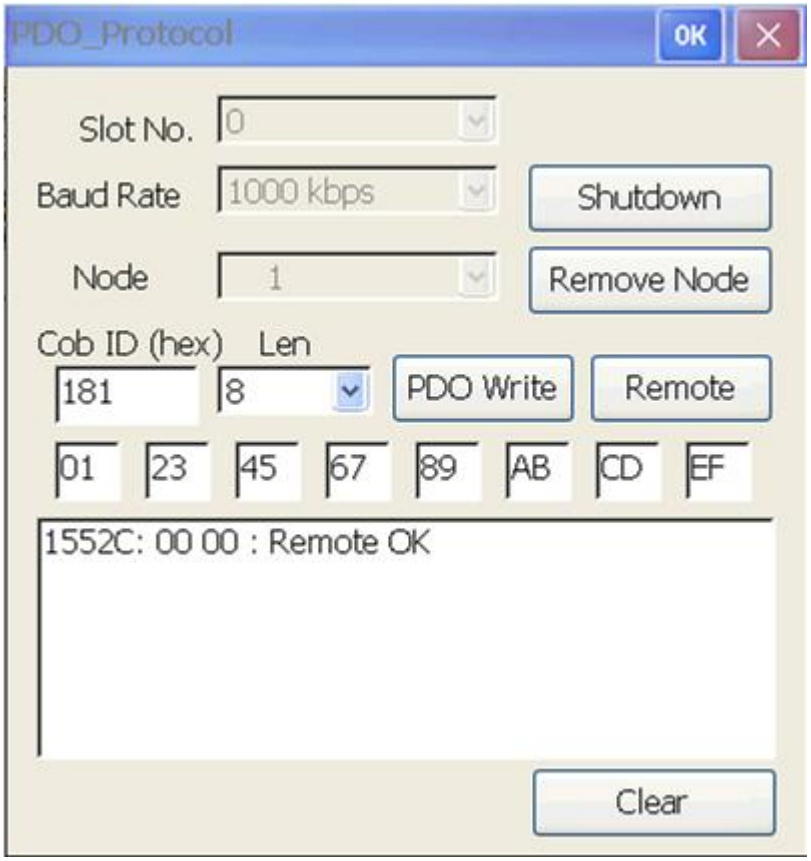
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_InstallPDO, I87123_RemovePDO, I87123_DynamicPDO, I87123_ChangePDOID, I87123_PDOTxType, I87123_PDOUseEntry, I87123_PDOEventTimer, I87123_GetTxPDOID, I87123_GetRxPDOID, I87123_GetPDOMapInfo.

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

4.1.4. PDO_Protocol

The PDO protocol is the main protocol to control the I/O of the specific slave device in the CANopen network. This demo shows how to read and write data to the slave device with the PDO functions.



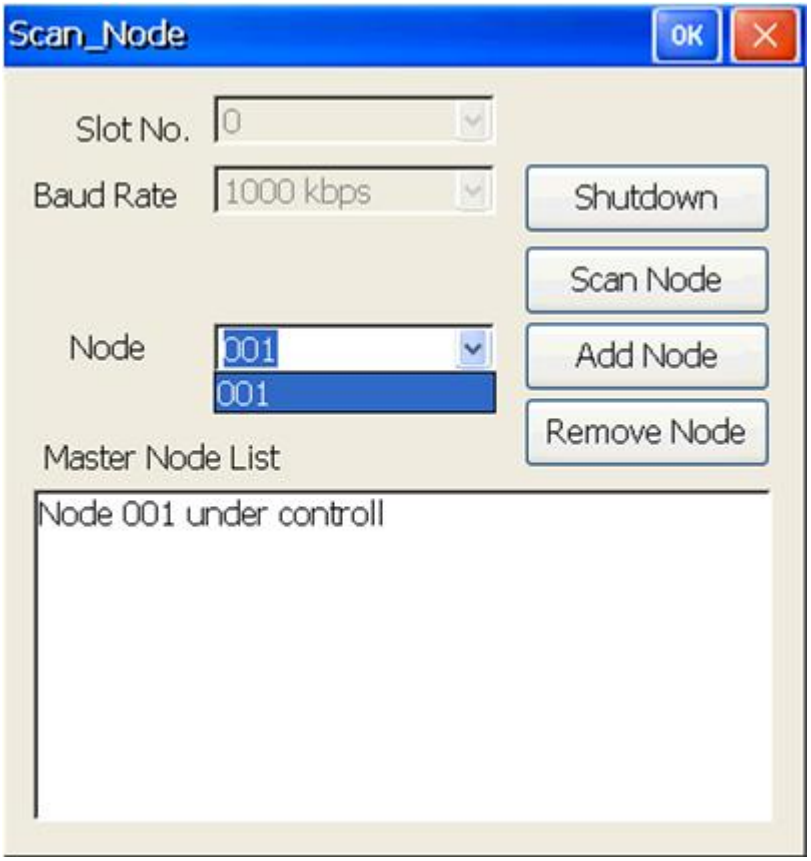
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_PDOWrite, I87123_PDORemote, I87123_GetPDOLastData

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

4.1.5. Scan_Node

When users want to know which slave nodes exist on the CANopen network or which slave nodes are under the control of the I-87123, this demo will is useful.



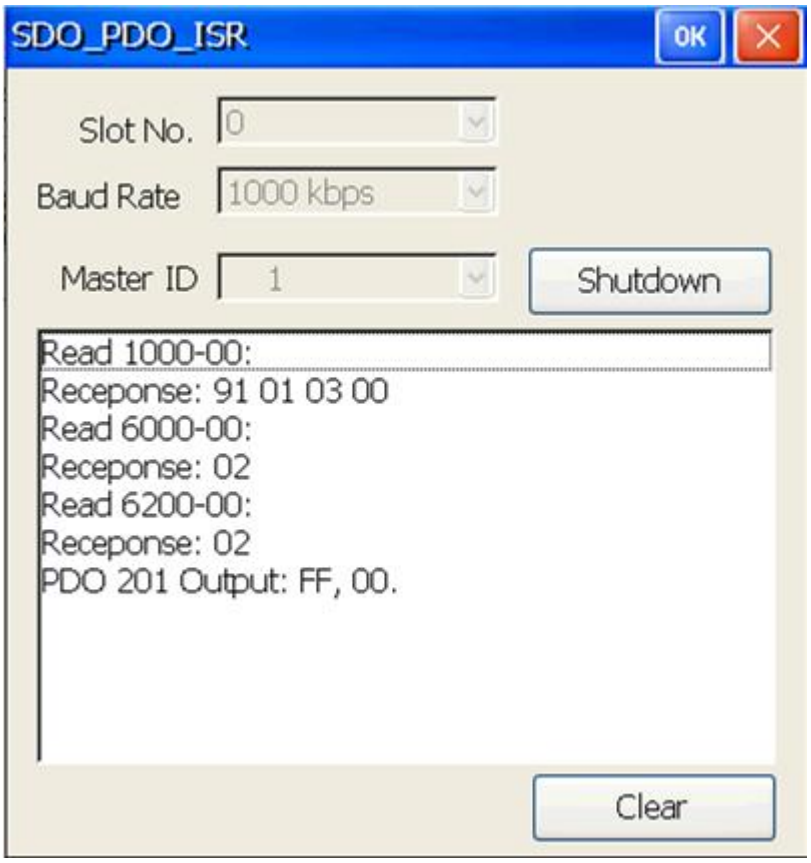
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_SetFunctionTimeout, I87123_ScanNode, I87123_GetNodeList.

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

4.1.6. SDO_PDO_PROC

In this demo, it is allowed to configure the I-87123 as a CANopen slave. Users can use another CANopen master to read/write the users' defined object dictionary of the I-87123 by SDO protocol or to get/set the DIO status by PDO protocol when the I-87123, the I-8053W DI module, and the I-8057W DO module are plugged in the same MCU. If the user has an application like this, this demo may be a good reference.



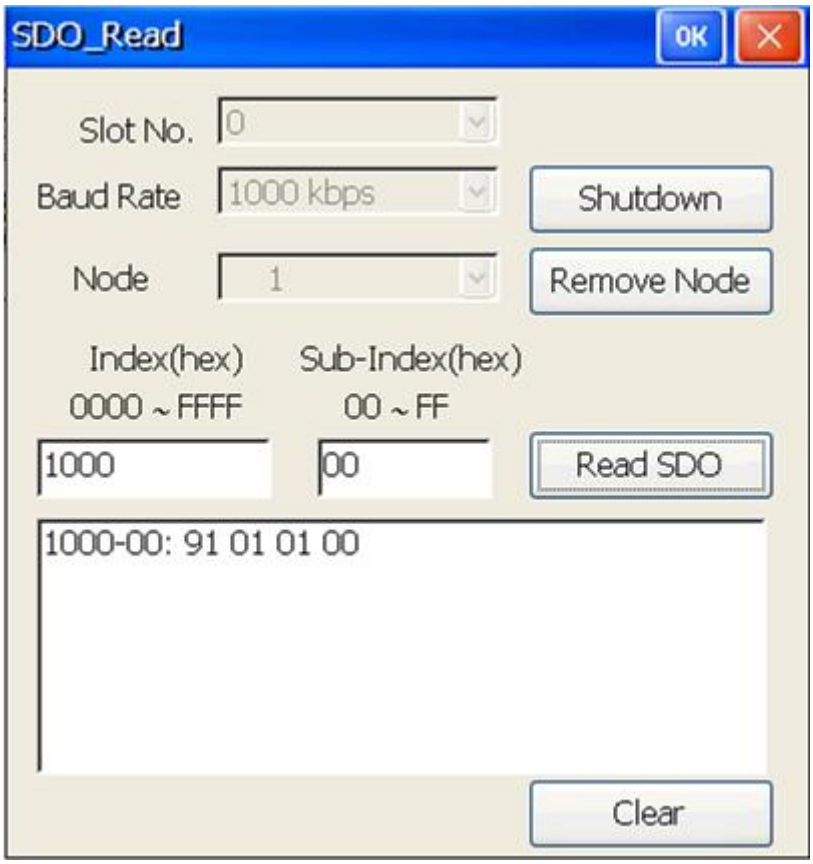
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_InstallPDO_List, I87123_CheckNetworkEvent, I87123_GetMasterRxPDOEvent, I87123_GetMasterRemotePDOEvent, I87123_ResponseMasterPDO, I87123_GetMasterWriteSDOEvent, I87123_GetMasterReadSDOEvent, I87123_ResponseMasterSDO.

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

4.1.7. SDO_Read

SDO protocol is a kind of the communication functions used to read/write CANopen object dictionary. You can read any object data of the object dictionary through the object address (index and sub-index) by SDO protocol. This demo is a good model to do that.



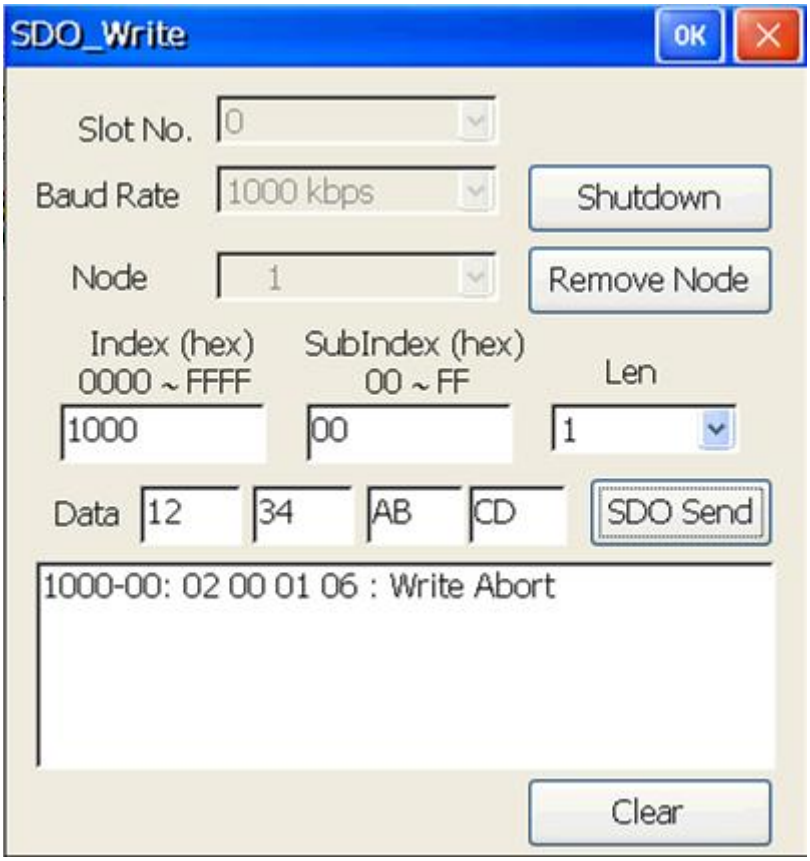
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_SDOReadData.

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

4.1.8. SDO_Write

SDO protocol is a kind of the communication functions used to read/write CANopen object dictionary. You can write any data to the specific object of the object dictionary through the object address (index and sub-index) by SDO protocol. This demo is a good model to do that.



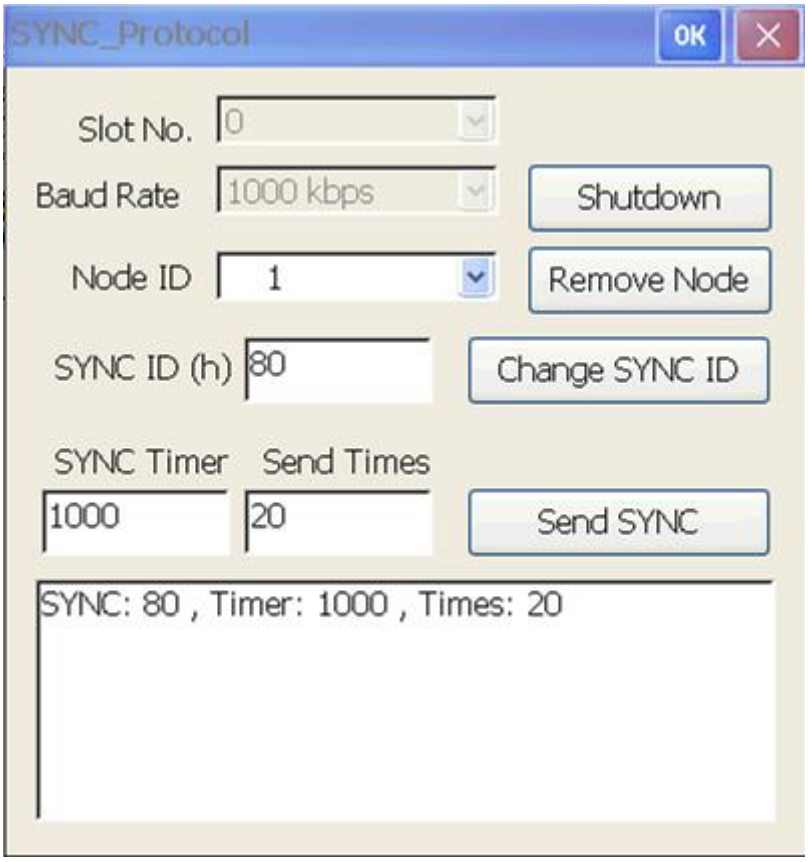
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_SDOWriteData.

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

4.1.9. SYNC_Protocol

SYNC protocol is a synchronous function of the PDO communication. It is always used with the transmission type of the PDO communication. In this demo, users can know how to use the SYNC related functions.



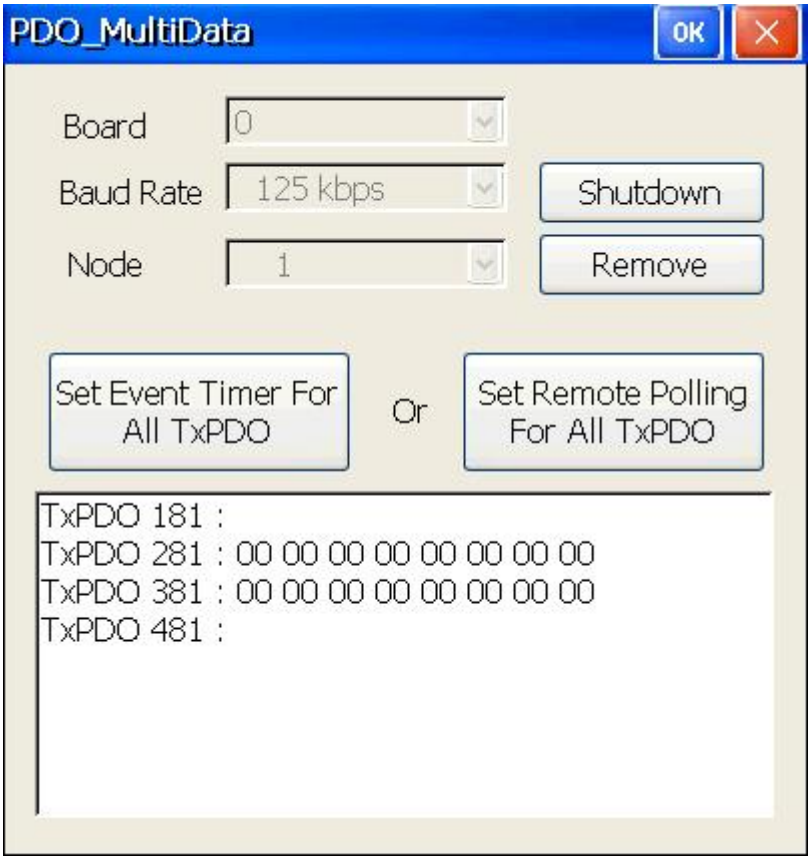
Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_ChangeSYNCID, I87123_GetSYNCID, I87123_SendSYNCMsg, I87123_GetCyclicSYNCInfo.

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

4.1.10. PDO_MultiData

Sometimes, users want to poll several PDO objects data at the same time for increasing the performance. But it is slower that sending the Remote PDO to poll each PDO data one by one. So users can set event timer or remote list for these PDO. When the PDO data are polled by the I-87123 or are replied from slave automatically, then use the I87123_GetMultiPDOData function to obtain these PDO data from the buffer at the same time.



Applied function list:

I87123_InitMaster, I87123_ShutdownMaster, I87123_AddNode, I87123_RemoveNode, I87123_GetTxPDOID, I87123_SetPDORemotePolling, I87123_PDOEventTimer, I87123_GetMultiPDOData

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

5. Update Firmware

Sometimes the firmware of I-87123 has some problem and users need to update it. This section will introduce how to update the firmware of I-87123.

Firmware

User can download the newest firmware from the path:

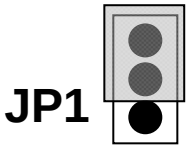
http://ftp.icpdas.com.tw/pub/cd/fieldbus_cd/canopen/master/i_87123/firmware/

It includes 5 files, 7188xw.exe, 7188xw.ini, 7188xw.f4, 87123xxx.exe and autoexec.bat, and these files must be in the same folder after download in your PC (xxx of 87123xxx.exe file means firmware version, such as 87123202.exe).

Connection and Jumper

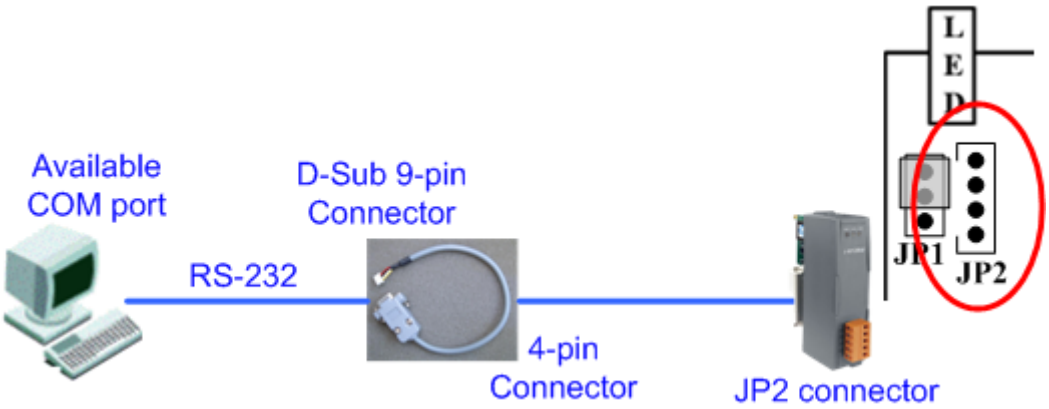
Before update firmware, users need to connect the download cable and adjust some jumper:

Step1: Adjust JP1 (Flash Protection) of I-87123 to pin1 and pin2 (disable).



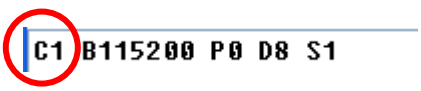
Step2: Short JP3 of I-87123.

Step3: Connection as below.



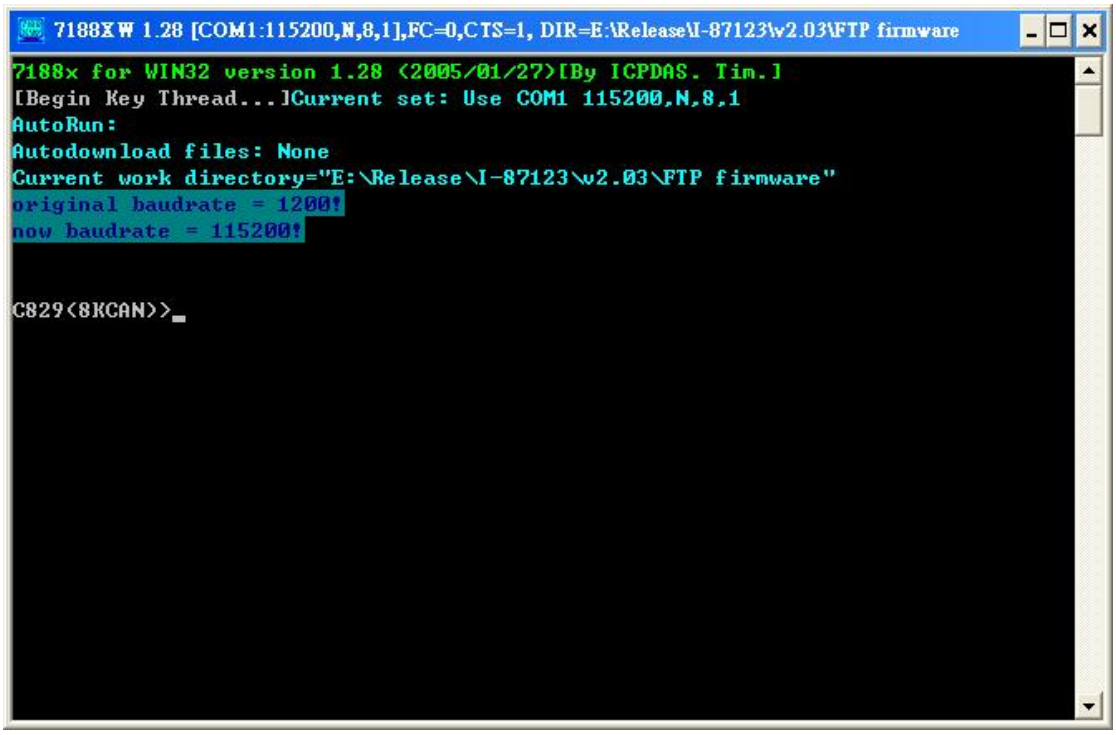
Start to Update

Step1: Execute the 7188xw.exe. Note that, the COM port setting in the 7188xw.ini is COM1 default. **If your COM port is not COM1, please edit the 7188xw.ini yourself (Ex: C2 means COM2)**.

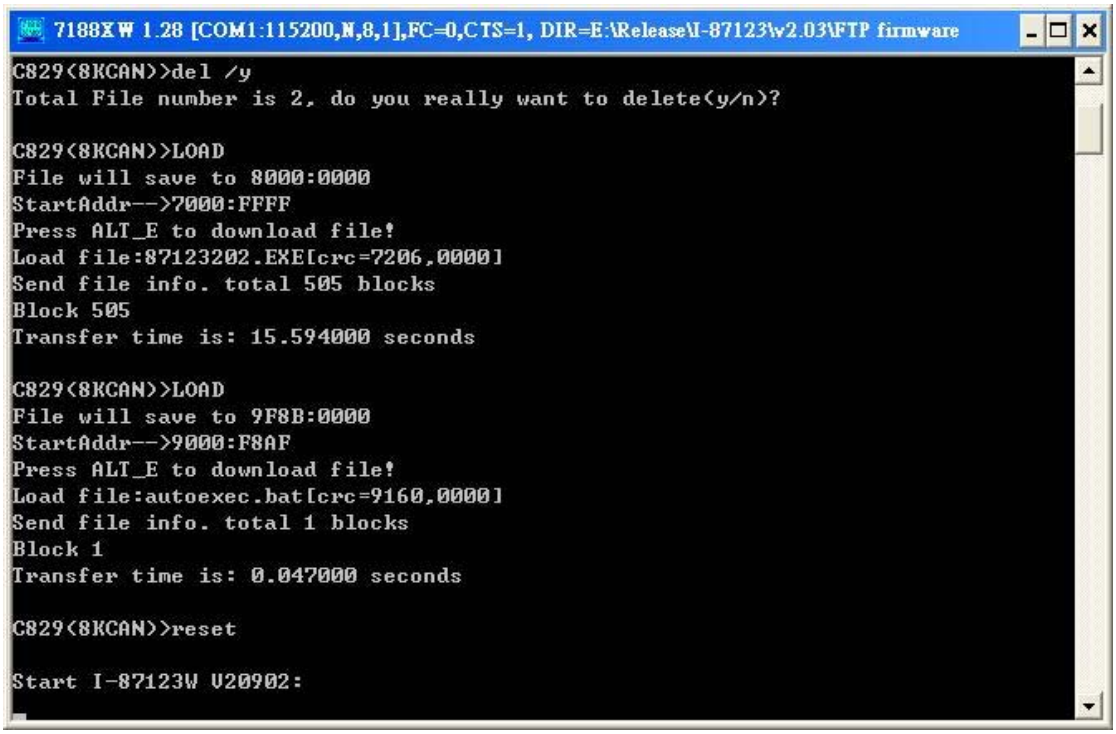


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step2: Power on the I-87123 and the 7188xw.exe will show the command line as ”C829(8KCAN)>”.



Step3: Click “F4” key on your keyboard and than the I-87123 will start to update the new firmware. When completely update firmware, the I-87123 will reset and start working with new firmware.



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