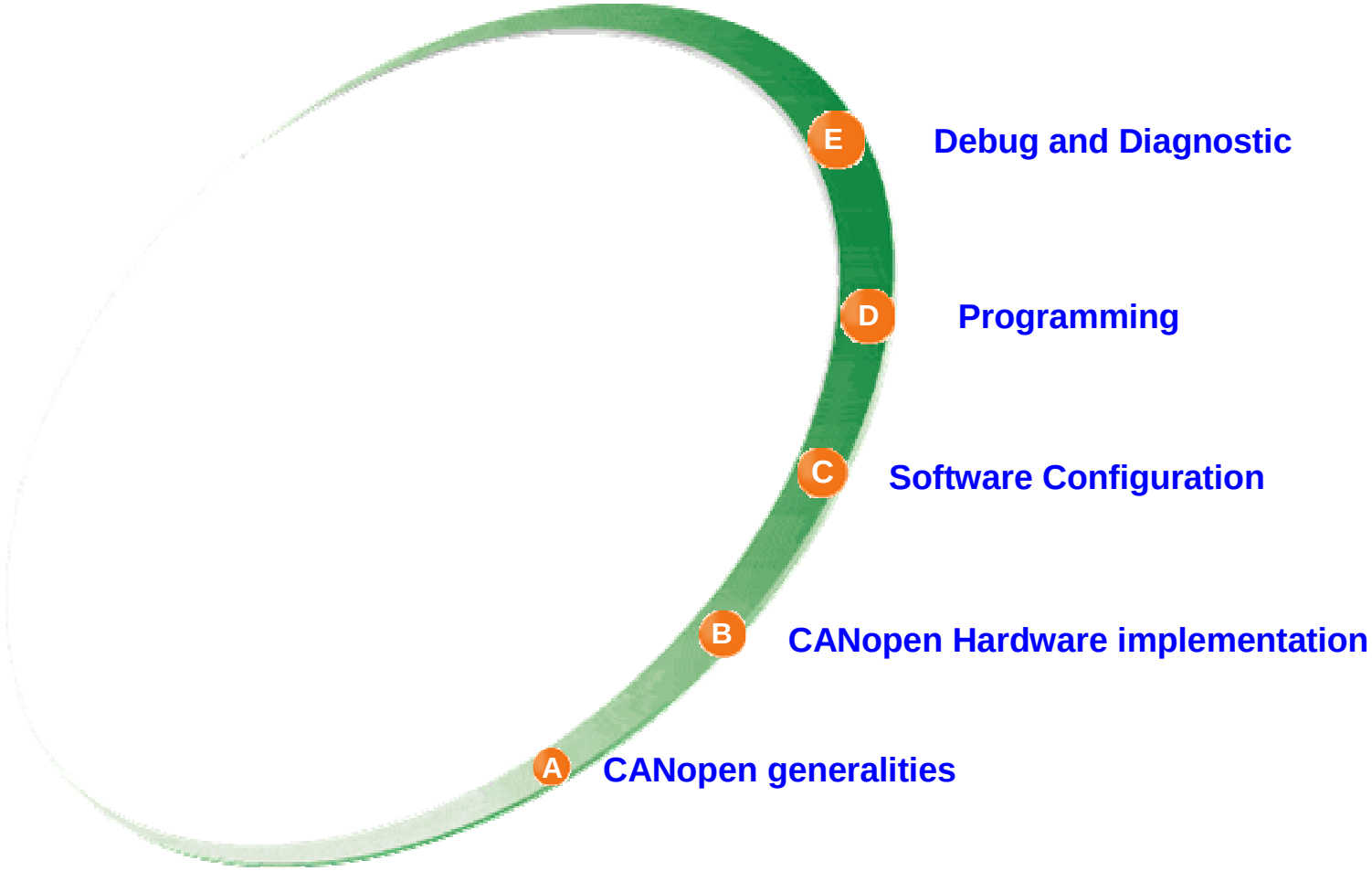
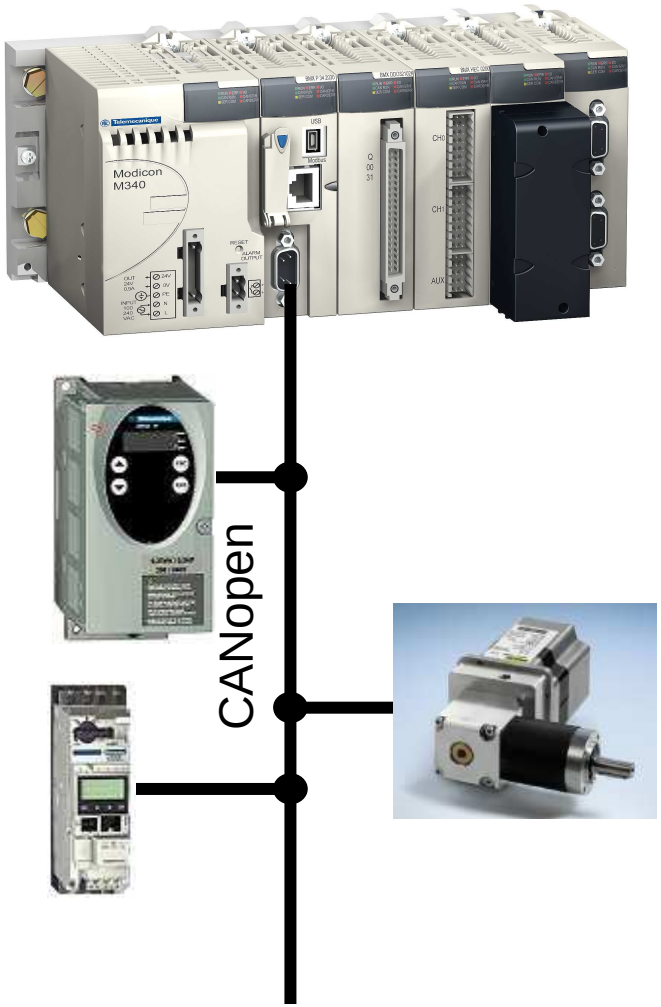
M7 - Modicon M340 : CANopen field bus



A – CANOpen generalities

CANopen architecture overview



- Canopen field bus is used to exchange data with equipement connected to the bus.
  - Implicit exchanges (through **%QW, %IW**)
  - Explicit exchanges (through **write/read var**)

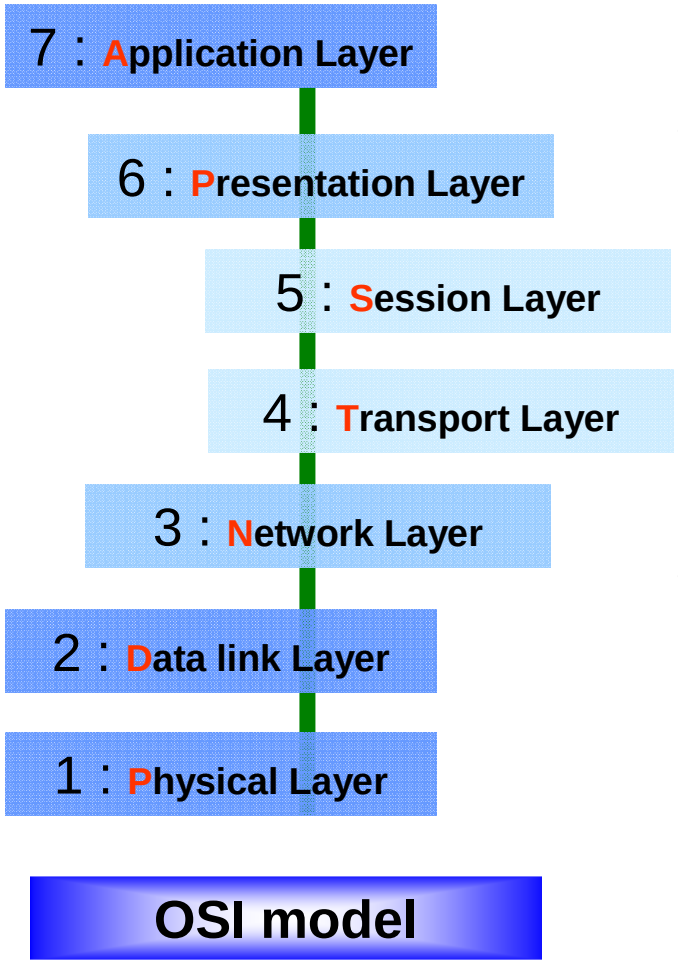
- CANopen field bus is available on some processors equipped with an integrated CANopen port.
  - BMX P34 2010 / 2030 processors
  - SubD 9 points connector as physical interface

Characteristic of integrated channel

	Character mode
Baud rate	**20, 50, 125, 250, 500, 1000** kbds (according with the lenght)
Equipments	**63** Equipments max
Bus length	**0** to **2500** meters
Services	**NMT, PDO, SDO, SYNC, EMCY**
Speed / Lenght **(the speed is directly linked with the bus's length)**	

1 Mbits	**20 m**
500 Kbits	**100 m**
250 Kbits	**250 m**
125 Kbits	**500 m**
50 Kbits	**1000 m**
20 Kbits	**2500 m**

OSI model dedicated for the CAN bus



CANopen defines the application layer

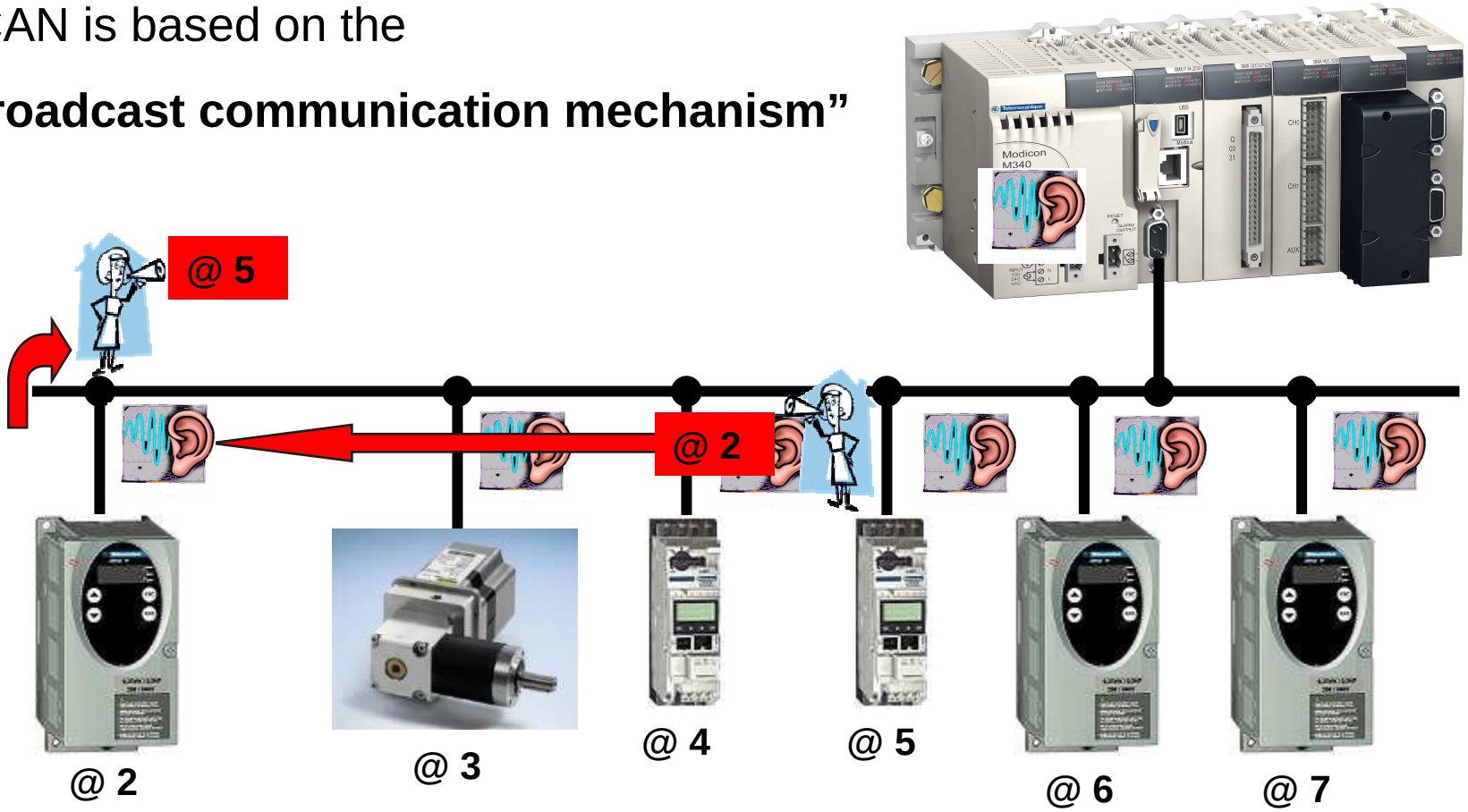
Not used

The Controller Area Network (CAN) protocol defines the data link layer and part of the Physical layer in the OSI model.

Communication protocol principle

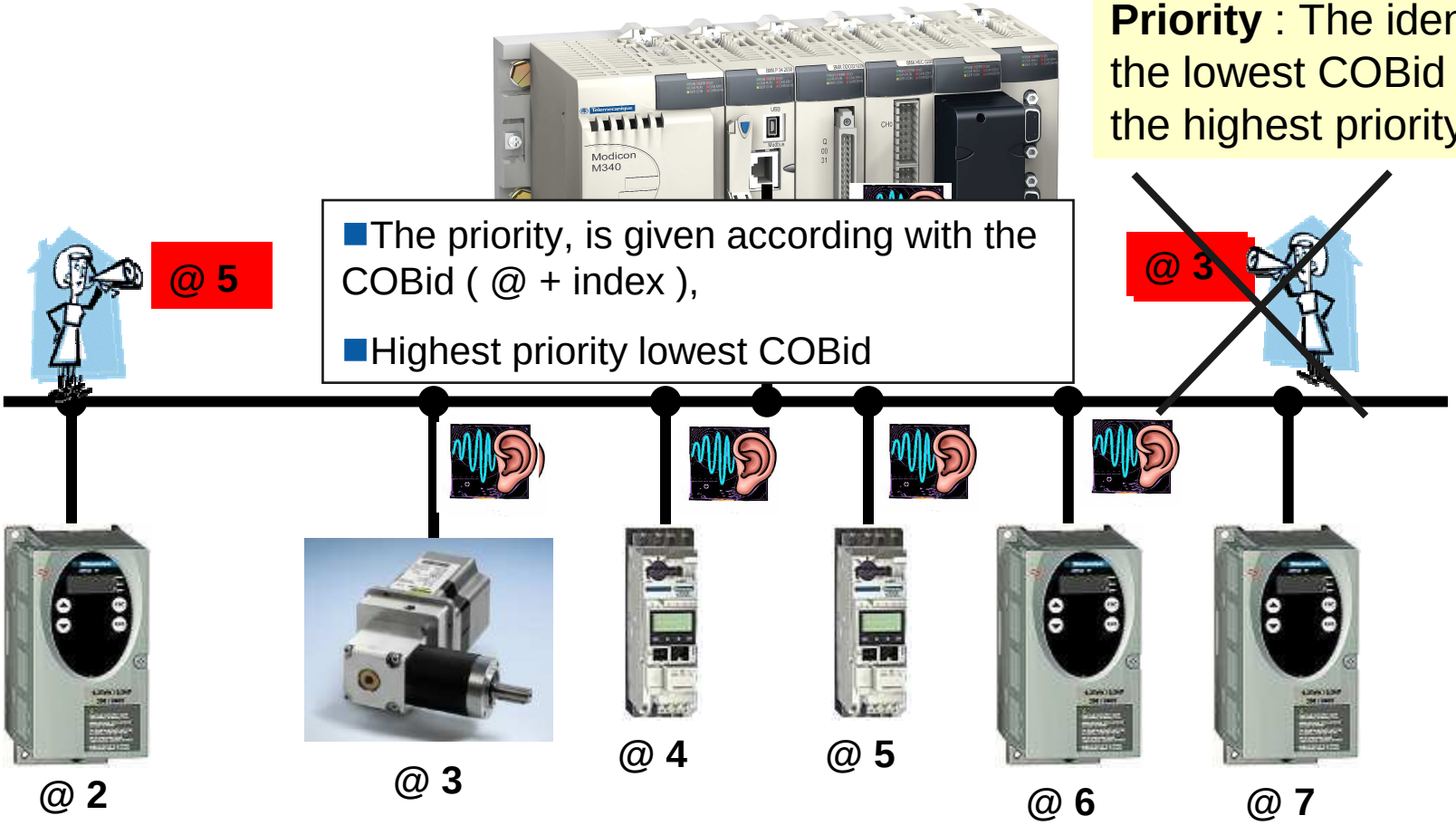
- CAN is based on the

“broadcast communication mechanism”



Communication protocol principle

Priority : The identifier with the lowest COBid number has the highest priority.



Communication protocol principle

Priority : The identifier with the lowest COBid number has the highest priority.

@ 5

@ 2

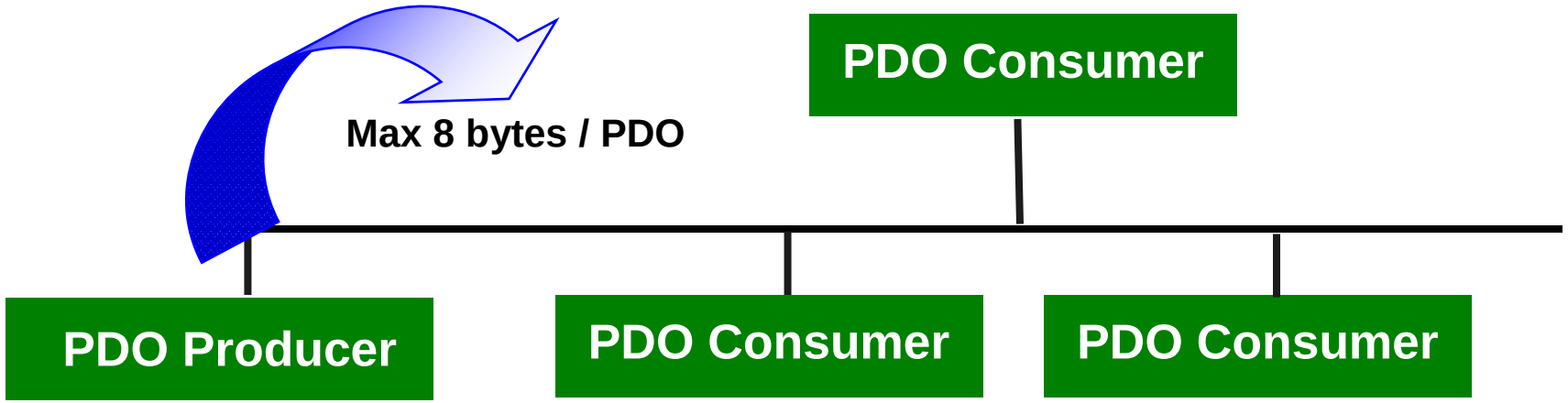
■The priority, is given according with the

Communications object	Function code	Node address-node-Id [1...127]	COB-IDdecimal (hexadecimal)
NMT Start/Stop Service	0 0 0 0	0 0 0 0 0 0 0	0 (0_h)
SYNC object	0 0 0 1	0 0 0 0 0 0 0	128 (80_h)
EMCY object	0 0 0 1	x x x x x x x	128 (80_h) + node-Id
T_PDO1	0 0 1 1	x x x x x x x	384 (180_h) + node-Id
R_PDO1	0 1 0 0	x x x x x x x	512 (200_h) + node-Id
T_PDO2	0 1 0 1	x x x x x x x	640 (280_h) + node-Id
R_PDO2	0 1 1 0	x x x x x x x	768 (300_h) + node-Id
T_PDO3	0 1 1 1	x x x x x x x	896 (380_h) + node-Id
R_PDO3	1 0 0 0	x x x x x x x	1024 (400_h) + node-Id
T_PDO4	1 0 0 1	x x x x x x x	1152 (480_h) + node-Id
R_PDO4	1 0 1 0	x x x x x x x	1280 (500_h) + node-Id
T_SDO	1 0 1 1	x x x x x x x	1408 (580_h) + node-Id
NMT error control	1 1 1 0	x x x x x x x	1792 (700_h) + node-Id
R_SDO	1 1 0 0	x x x x x x x	1536 (600_h) + node-Id

PDO (Process Data Objects)

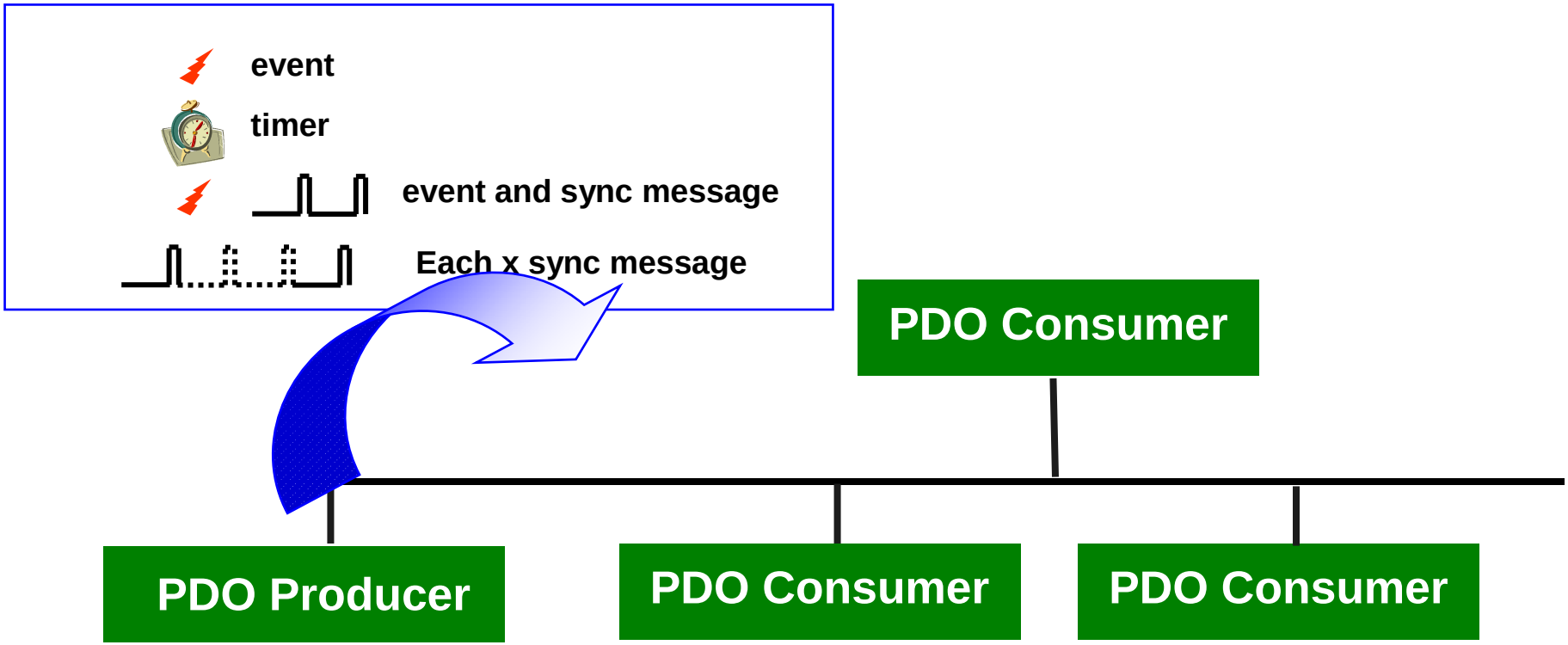
- PDOs are implicits exchanges between a Producer and one or more Consumer (For this exchange M340 uses the %QW and %IW objets)

- PDOs (Process Data Objects) are mapped to a single CAN frame using data 8 bytes max.



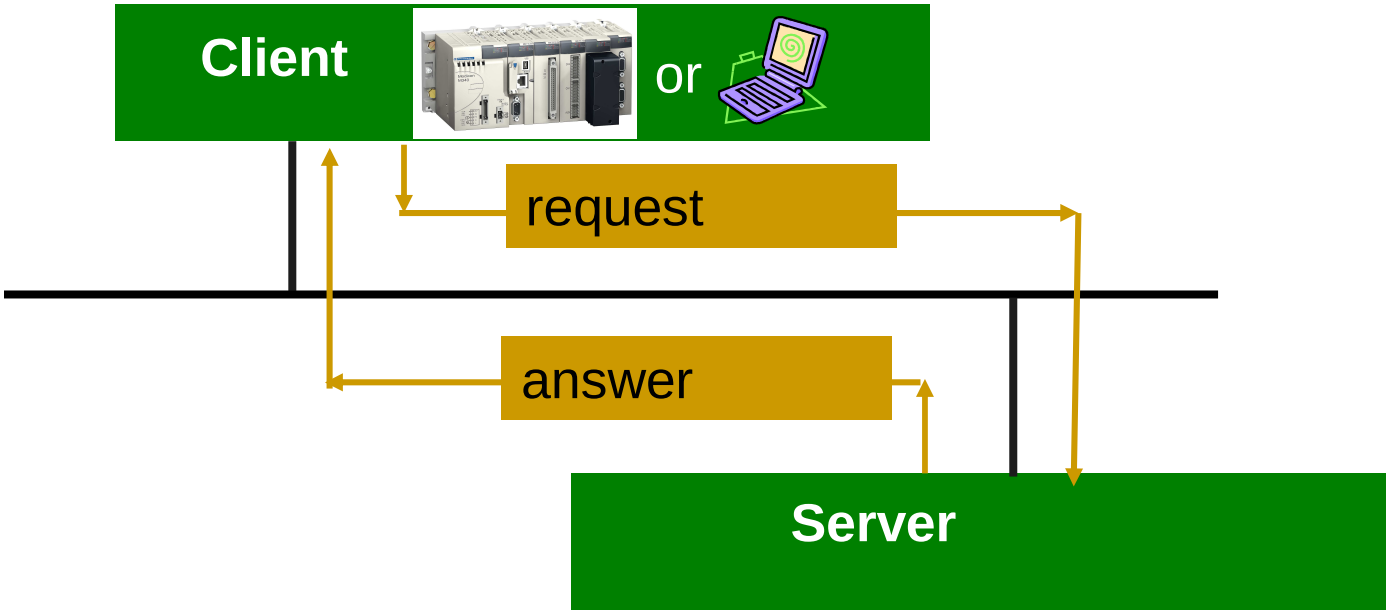
PDO (Process Data Objects)

- PDOs exchange may be driven by an internal event, by an internal timer, by remote request or by the Sync message .



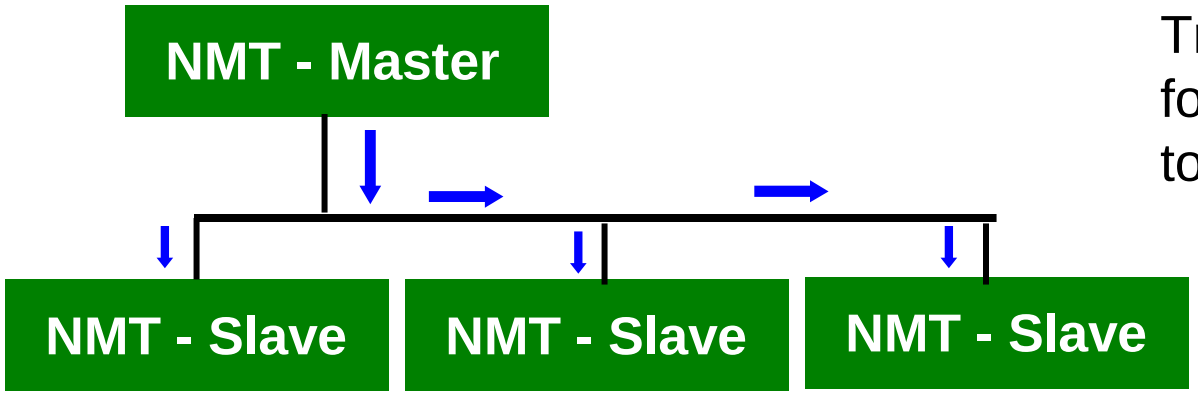
SDO (Service Data Object)

- SDOs are explicit exchanges and use the READ_VAR and WRITE_VAR commands
- SDO (Service Data Object) reads or writes the Objects predifined in the Dictionary (EDS file).
- The SDO transport protocol allows transmitting objects of any size.



NMT (Network ManagemenT)

- The NMT (Network management) include three services:
  1. NMT message
  2. Boot up message
  3. The Error Control service

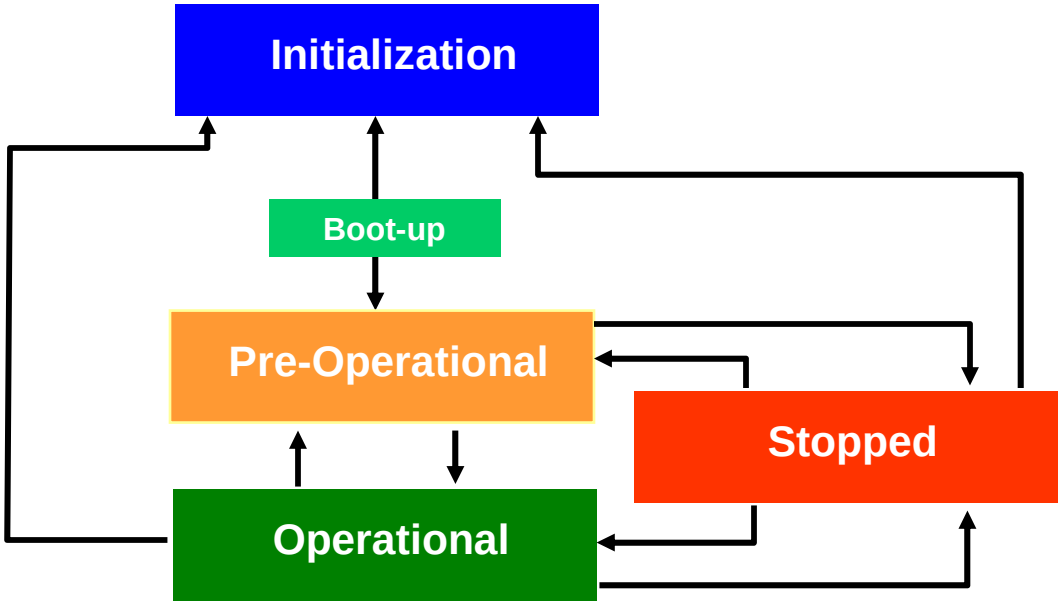


1. **The NMT message :**

   Transmitted by the NMT master forces one node or all the nodes to transit to another NMT state.

NMT (Network ManagemenT)

- The NMT (Network management) include three services :
  1. NMT message
  2. Boot up message
  3. The Error Control service



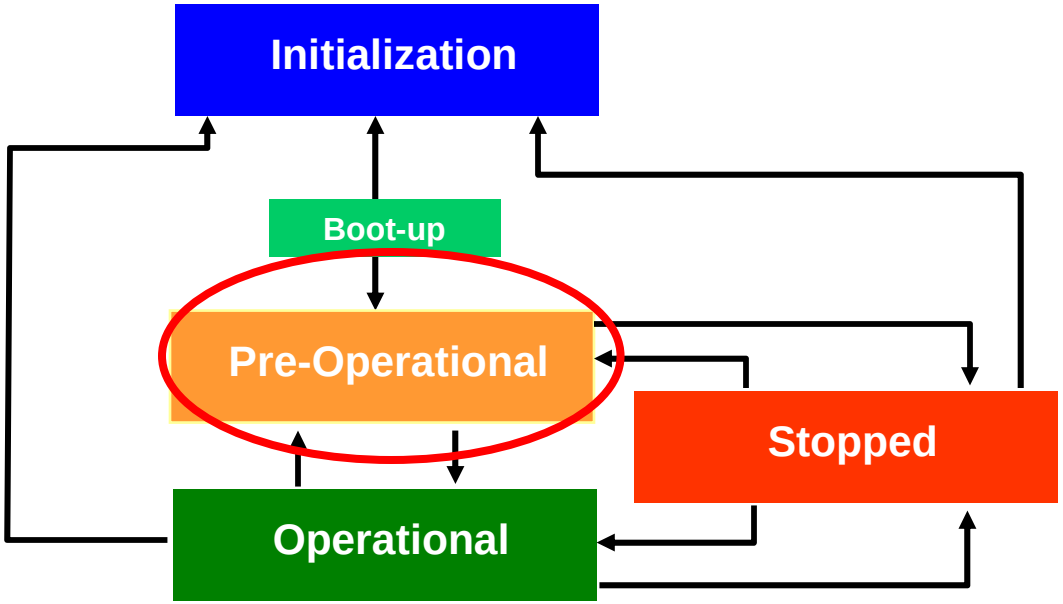
1. The NMT message :

The CANopen state machine (NMT state) specifies the states :

- Initialization
- Pre-Operational SDO allowed
- Operational SDO and PDO allowed
- Stopped. SDO and PDO are not allowed

NMT (Network ManagemenT)

- The NMT (Network management) include three services :
  1. NMT message
  2. Boot up message
  3. The Error Control service

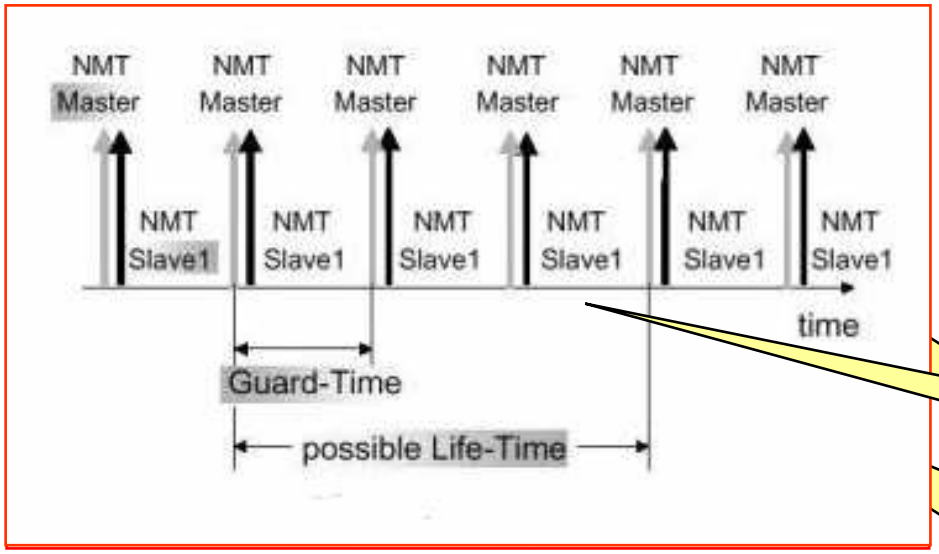


2. The BOOT UP message

A device sends the **Boot-up message** to indicate to the NMT master that it has reached the state Pre-operational.

NMT (Network ManagemenT)

- The NMT (Network management) include three services :
  1. NMT message
  2. Boot up message
  3. The Error Control service



3. The Error Control service

Through this service the NMT detect the error on the field bus The Error control service can manage :

The Node guarding protocol

Periodically the master send a request to the slave in order to know its state

EDS file (Electronical Data Sheets)

- **Goal :** The EDS file describes the capabilities of CANopen nodes:
  - General information about the node.
  - Informations of all dictionary object of the node.
  - Configuration of the PDO communication.

  The EDS file is an ASCII file, and can be acquired from the manufacturer of the device.

- Ex : Part of the MFBLXM15.eds

```
[FileInfo]
FileName=MFBLEX15L1.eds
FileVersion=1
FileRevision=5
EDSVersion=4.0
Description=LEXIUM15LP1 servodrive on MFB library
CreationTime=9:00AM
CreationDate=06-21-2005
CreatedBy=VAC
ModificationTime=00:40AM
ModificationDate=02-14-2006
ModifiedBy=PhB
; révision 14/02/2006 default values for MFB
;*************************************************
*****************************

[DeviceInfo]
VendorName=Telemecanique
VendorNumber=0x02000005A
ProductName=MFB_LEXIUM15LP
ProductNumber=0
RevisionNumber=1
```

```
[1A00]
SubNumber=0x3
ParameterName=1st transmit PDO-Mapping
ObjectType=0x9

[1A00sub0]
ParameterName=Number of Entries
ObjectType=0x7
DataType=0x0005
AccessType=ro
DefaultValue=0x2
PDOMapping=0

[1A00sub1]
ParameterName=Position actual value in user units
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0x60640020
PDOMapping=0
```

Mapping of the Transmit PDO1

B – Hardware implementation

Processors with an integrated CANopen

- Modules **BMX P34 2010 / 2030**

Value	Designation
1	Display
2	USB port
3	Eth port
4	Serial port
5	CANopen port

Wiring on the processor

- The connection is done on a SUBD 9 pts pin male connector of the processor

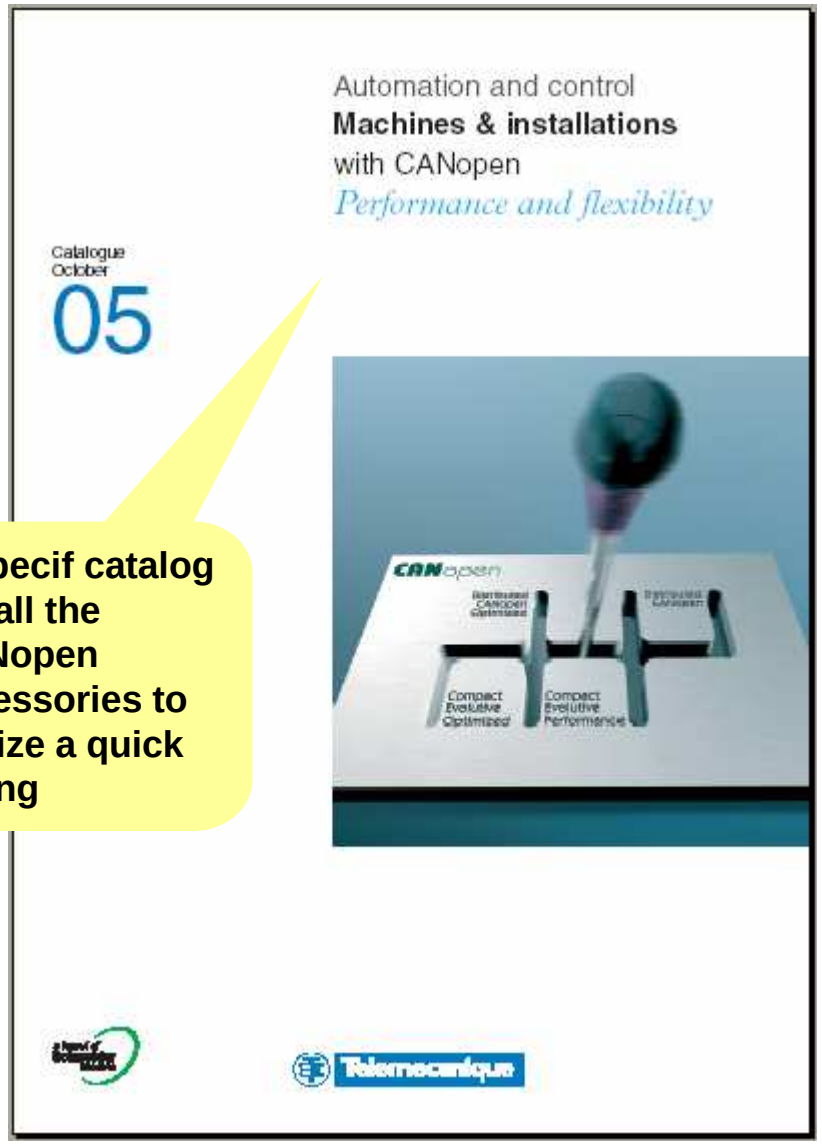
PIN	SIGNAL	
1		
2	**CAN-L**	
3	**CAN-GND**	
4		
5		
6	**GND**	
7	**CAN-H**	
8		
9		

Wiring (a global offer)

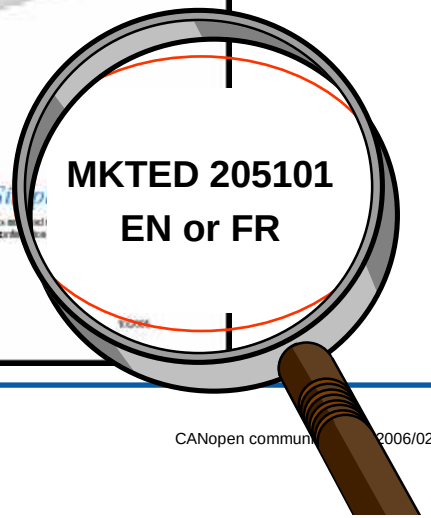
Example :

(1)	TSX CAN TD M4	(2)	TSX CAN KCDF xxxT
(4)	TSX CAN KCDF 90TP	(5)	TSX CAN Cx xx
(6)	TSX CAN CxDx	(7)	FTX CN 31 xx
(9)	VW3 CAN TAP2	(10)	VW3 CANN CARRxx
(12)	FTX CN 32 xx	(13)	FTX CNTL 12
(14)	AMO 2CA 001V000		

Wiring (a global offer)

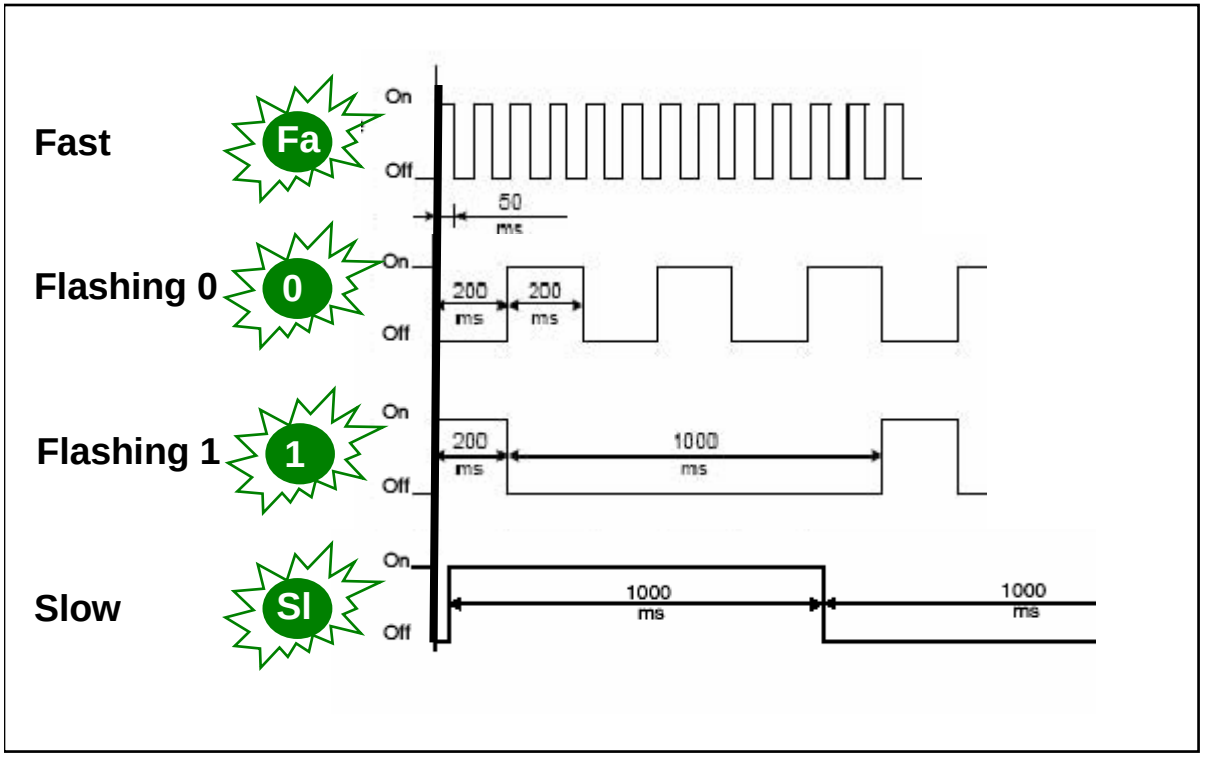


A specif catalog list all the CANopen accessories to realize a quick wiring



Visual diagnostic of CANopen

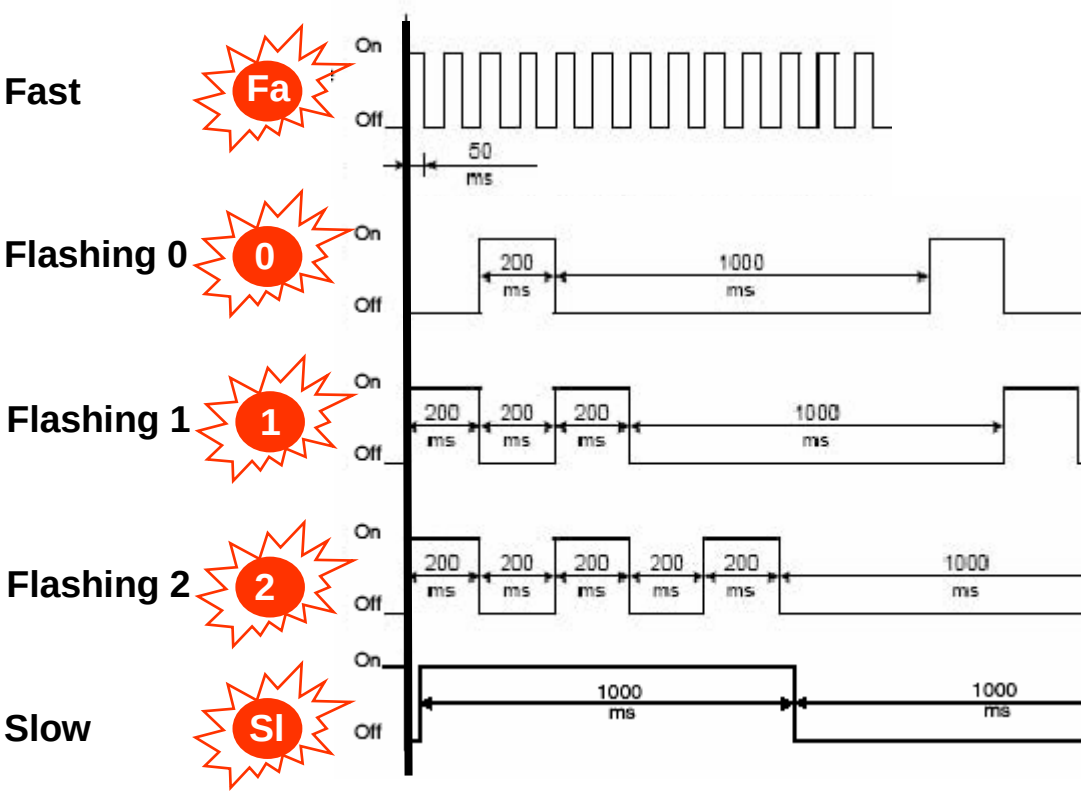
The information provided by the the **CAN RUN** lamp



State	Comment
	Operationnal state
Fa	Speed automatic detection
0	Pre Operationnal state
1	Stopped state
Sl	Check & init the slave
	-

Visual diagnostic of CANopen

The information provide by the The **CAN ERR** lamp



State	Comment
	Bus stopped
Fa	Speed automatic detection
0	Error counter overflow
1	Heartbeat fault
2	No Sync message
Sl	Canopen can't start
	No ERROR

C – Software configuration

Software Configuration

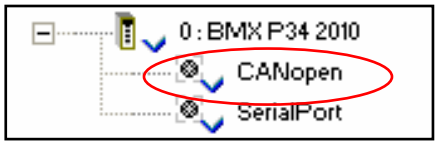
- The configuration for CANopen is fully integreted in UNITY (no external configurator need)
  - Here is the main steps to respect in order to configure the CANopen bus.

Configuration du Bus CANopen :

step 1 : Master Configuration (Speed, @, PDO Offset)
step 2 : Slave Configuration (PDO and exchanges)
- Standard
- Expert

step 1 : Master Configuration

Processor configuration



Using the symbol for programming

yes

Creating an IODDT structure for CANopen

no

Don't use the IODDT

1 Define IODDT

I know the Nb words and bits

yes

Modify the datas reserved for PDO

no

I will modify later

2 Define the OFFSET

I adjust the speed according with the device

yes

Select the Bus speed

no

I will modify later

3 Define the speed

I want exchange the PDO values in the FAST

yes

Select the CPU TASK

no

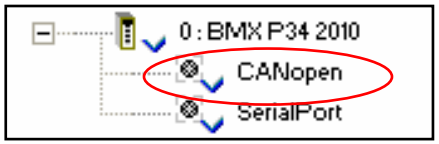
The MAST is Ok (given by default)

4 Define the task

END

step 1 : Master Configuration

Processor configuration



Using the symbol for programming

yes

Creating an IODDT structure for CANopen

no

Don't use the IODDT

1 Define IODDT

I know the Nb words and bits

yes

Modify the datas reserved for PDO

no

I will modify later

2 Define the OFFSET

I adjust the speed according with the device

yes

Select the Bus speed

no

I will modify later

3 Define the speed

I want exchange the PDO values in the FAST

yes

Select the CPU TASK

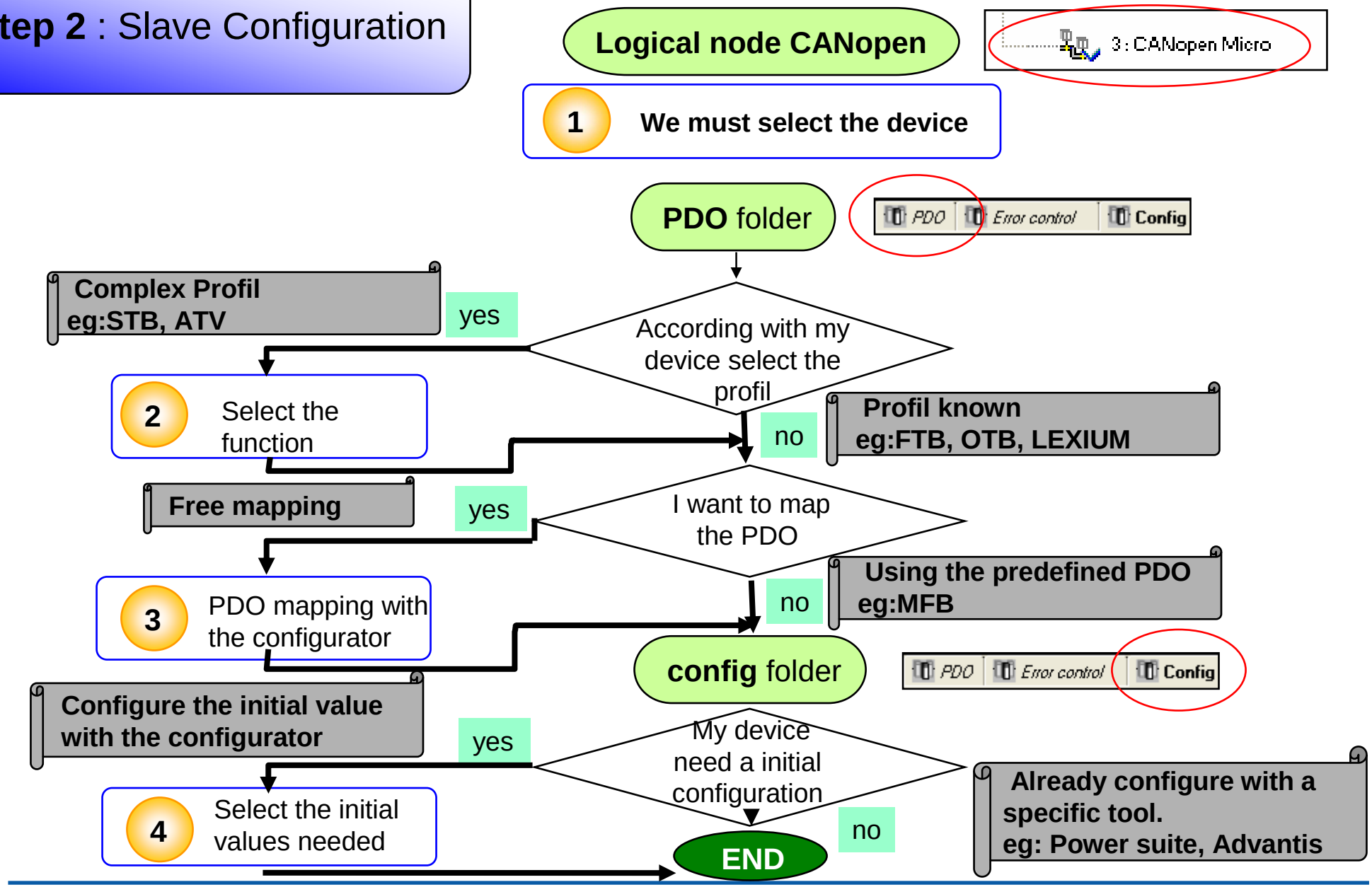
no

The MAST is Ok (given by default)

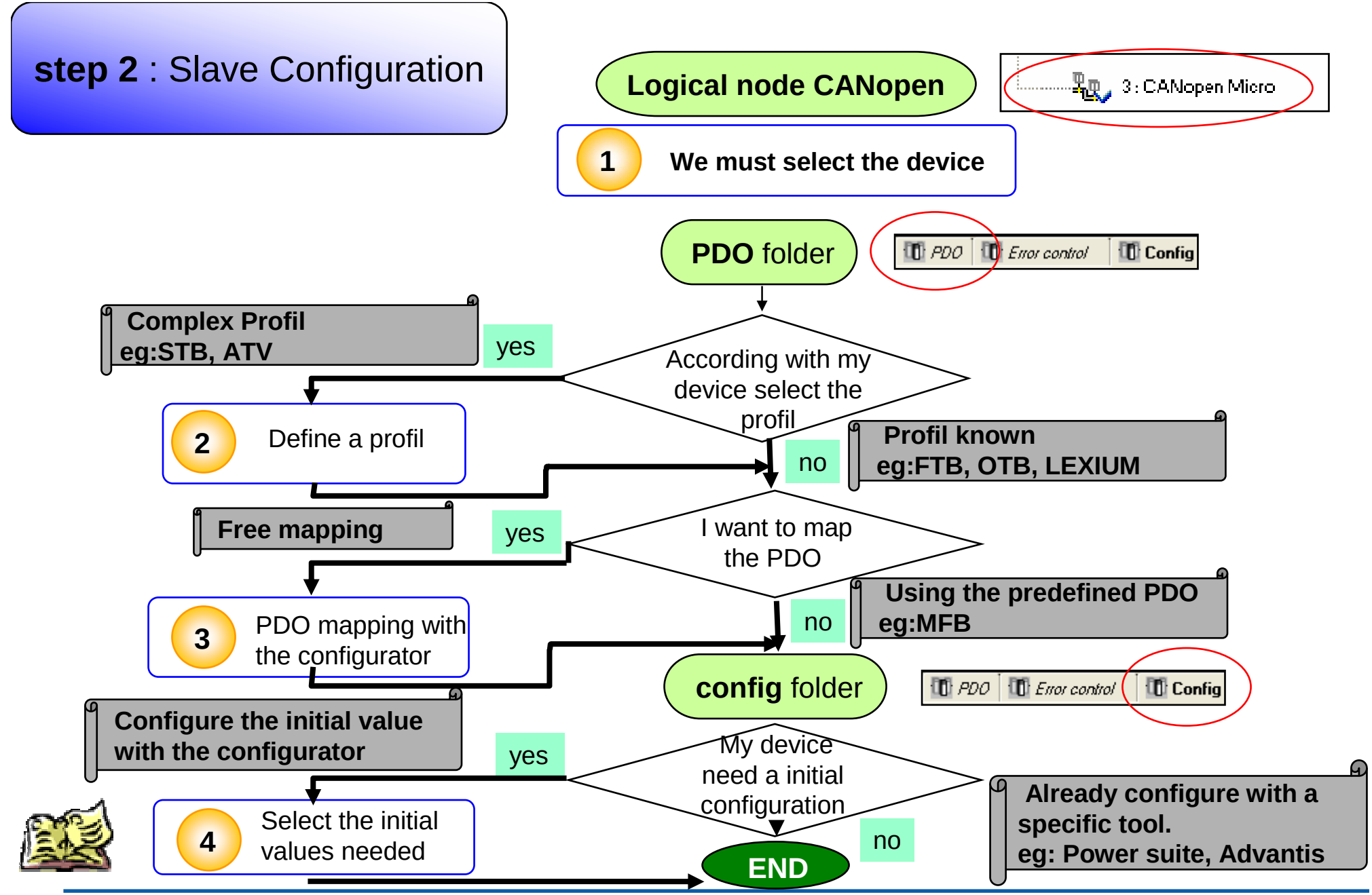
4 Define the task

END

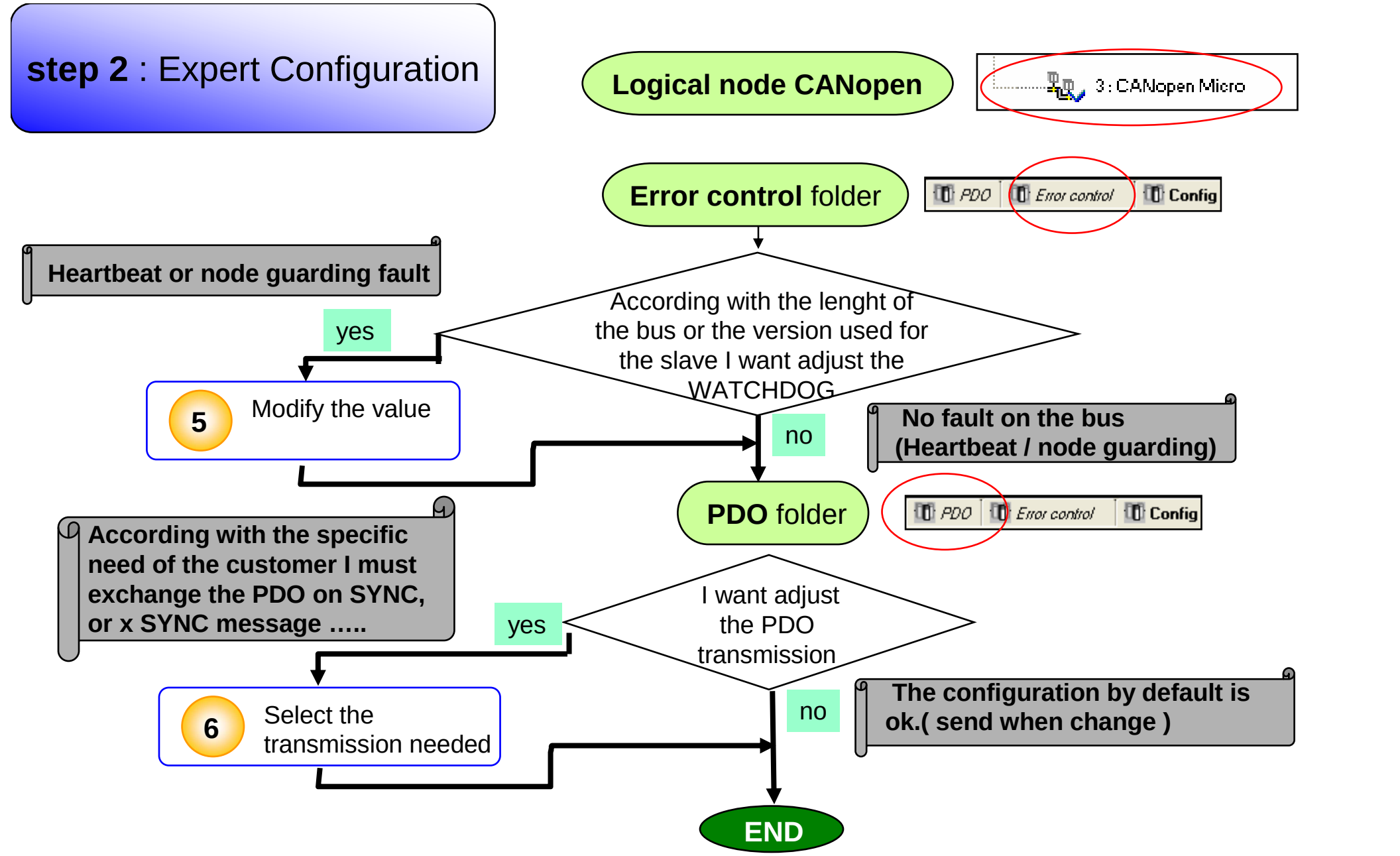
step 2 : Slave Configuration



step 2 : Slave Configuration

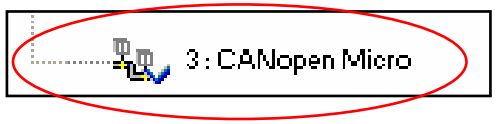


step 2 : Expert Configuration



step 2 : Expert Configuration

Logical node CANopen



Error control folder

Heartbeat or node guarding fault

According with the lenght of the bus or the version used for the slave I want adjust the WATCHDOG

yes

5 Modify the value

no

No fault on the bus (Heartbeat / node guarding)

PDO folder

According with the specific need of the customer I must exchange the PDO on SYNC, or x SYNC message …..

I want adjust the PDO transmission

yes

6 Select the transmission needed

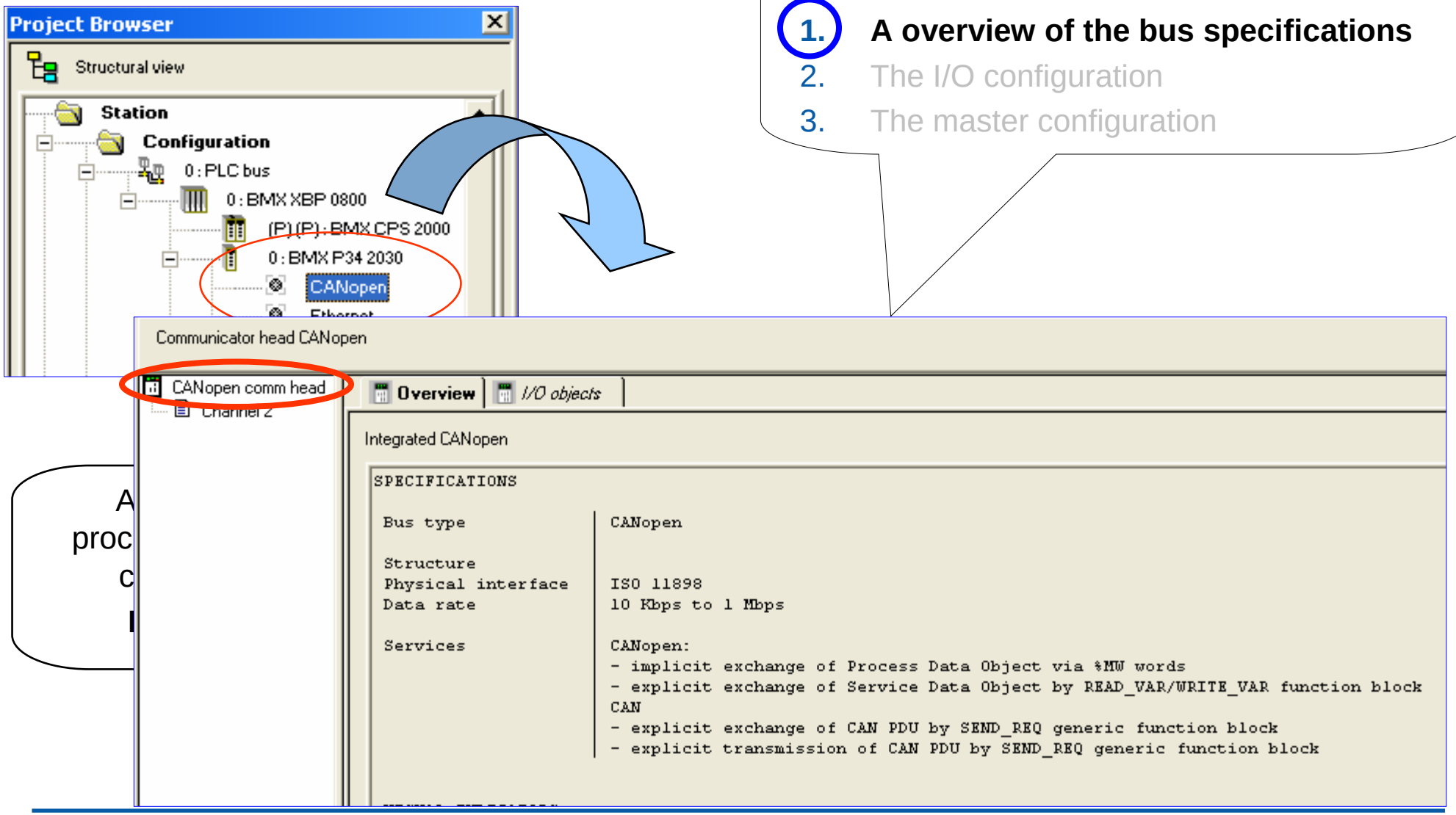
no

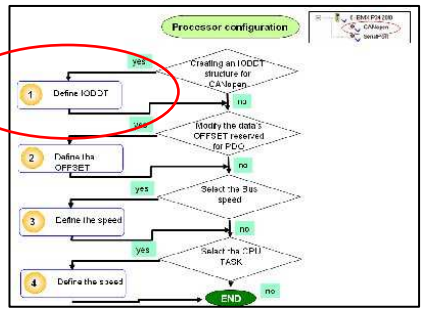
The configuration by default is ok.(send when change)

END

Step 1: Master configuration

Step 1 : Master configuration

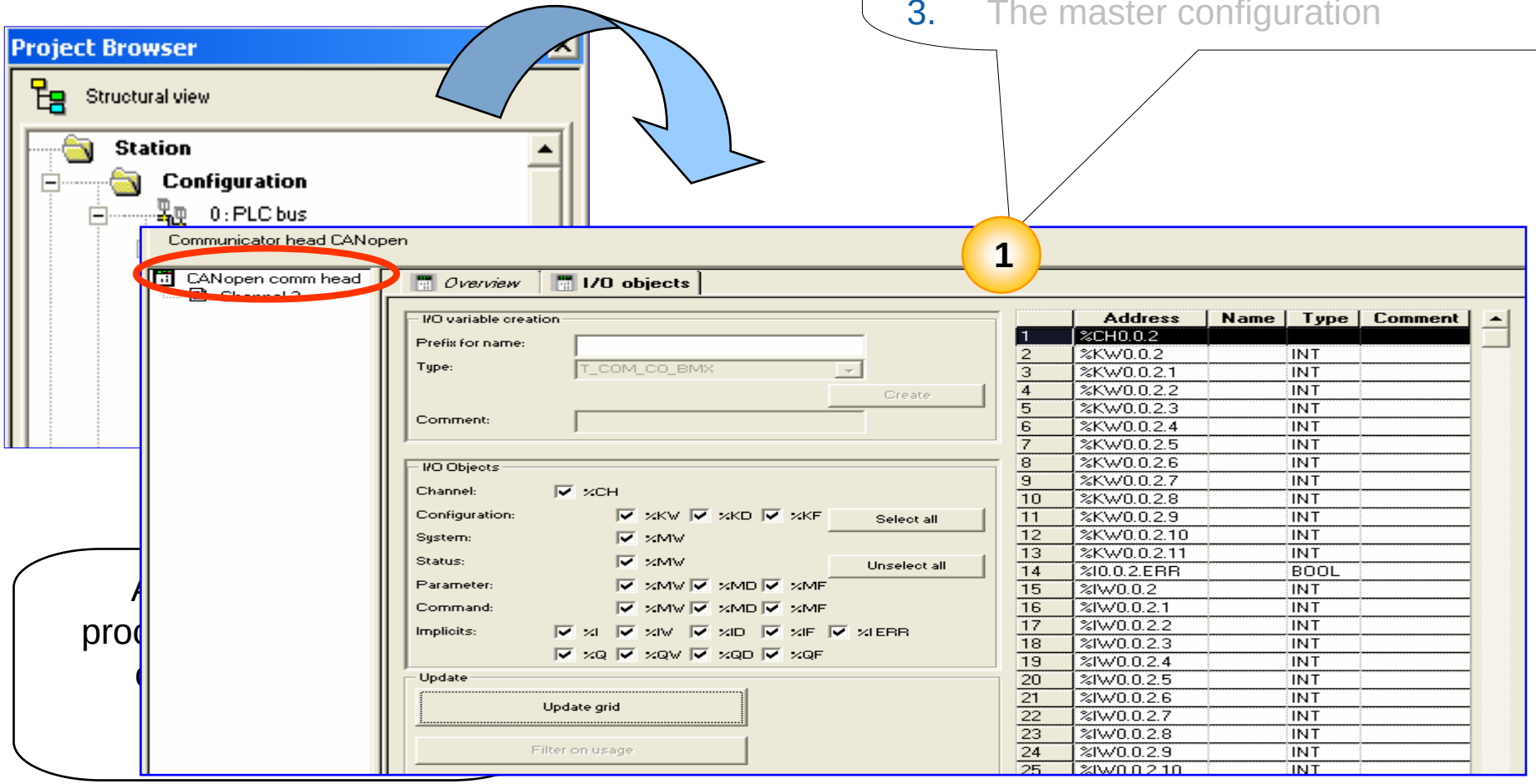


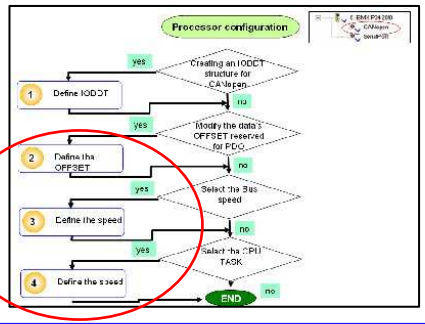


Step 1 : Master configuration 'IODDT'

In the CAN screen we have :

1. A overview of the bus specifications
2. **The I/O configuration**
3. The master configuration

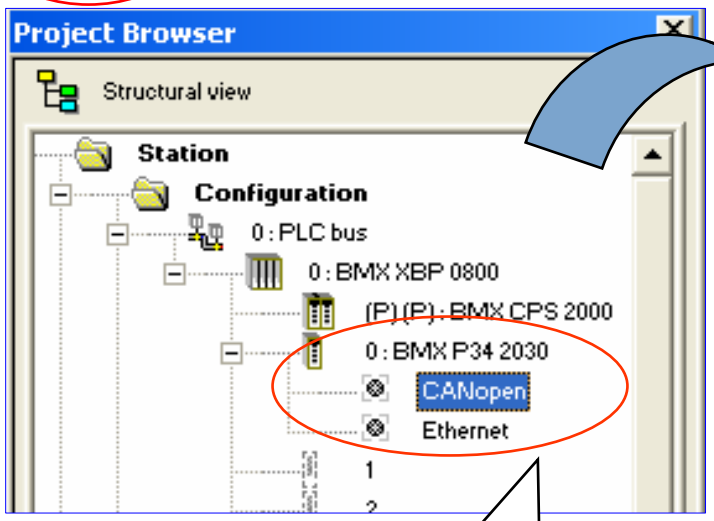




Step 1 : Master configuration

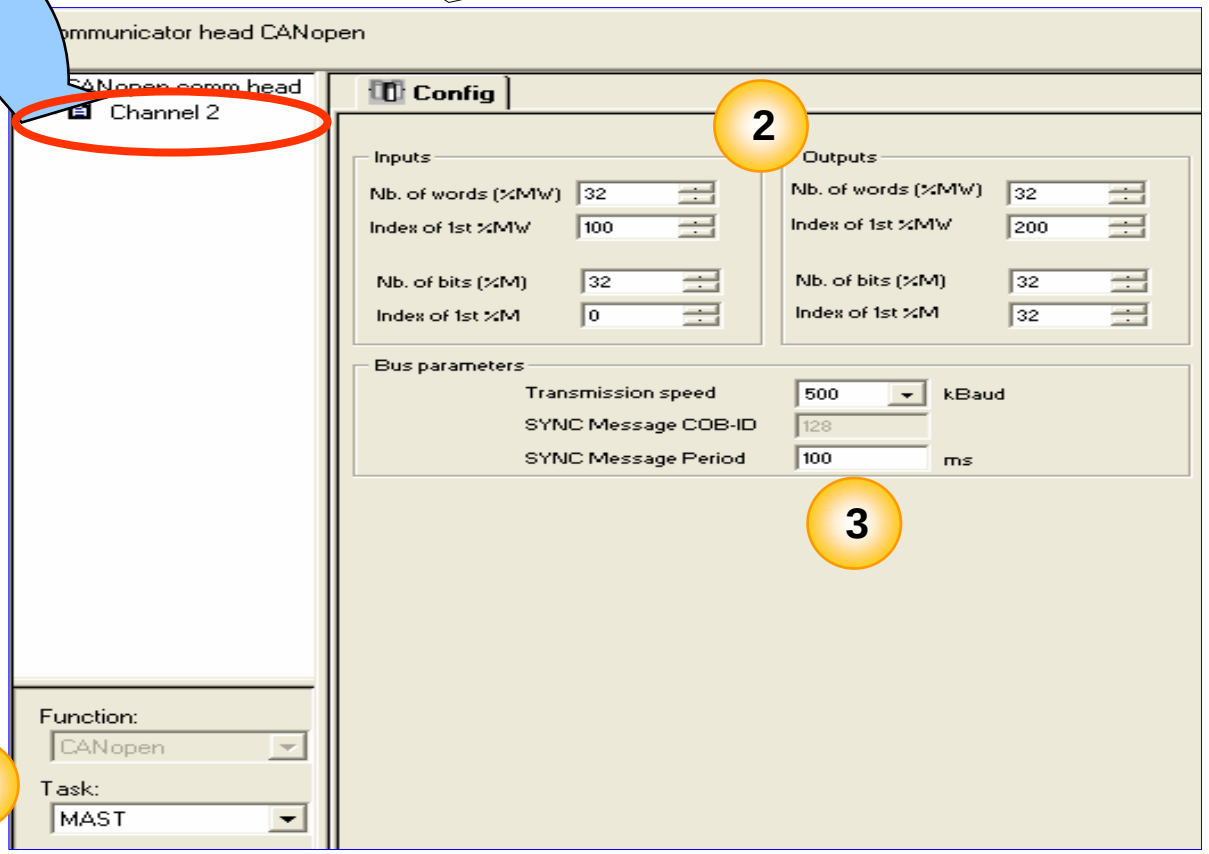
In the CAN screen we have :

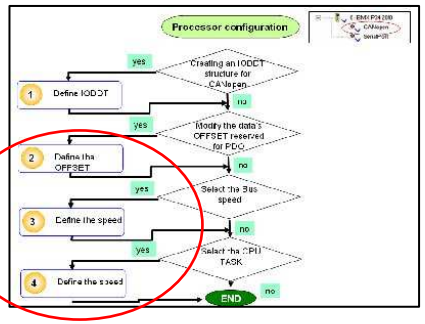
1. A overview of the bus specifications
2. The I/O configuration
3. **The Master configuration**



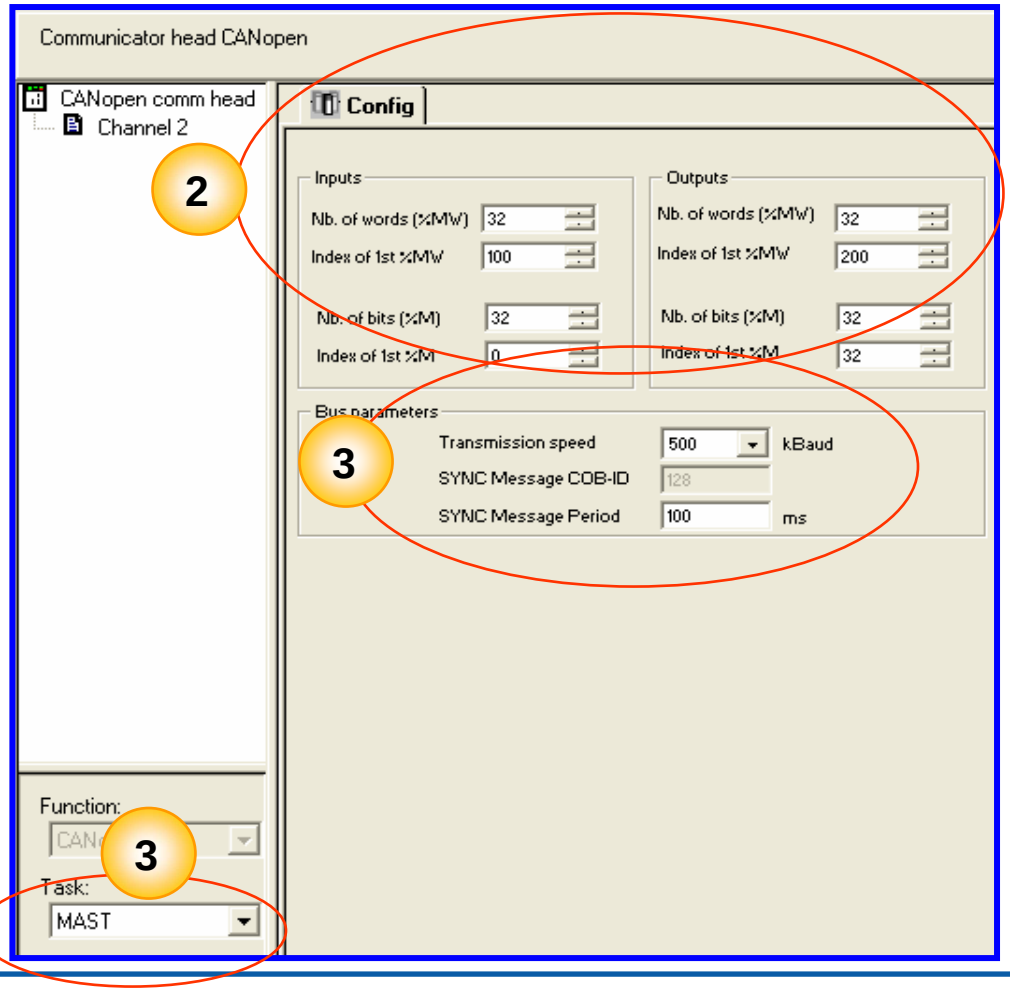
According with the processor it's possible to configure the bus.

Here : CANopen



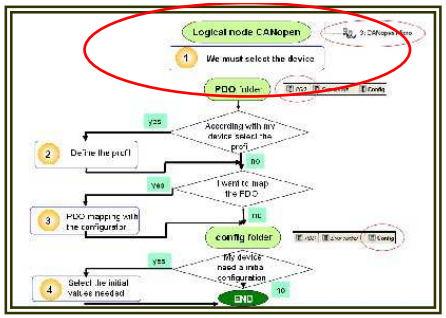


Step 1 : Master configuration

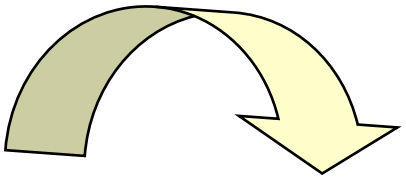


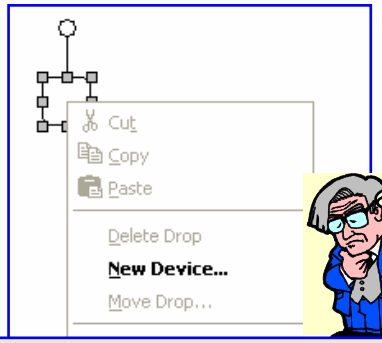
- Map the The PDO's to memory word (%mw) and bit (%m) location (mainly dedicated for the MODBUS communication).
  - Bus Parameters :
    1. Set speed according to the length of the bus.
    2. Period of the Sync Message
  - Task assigned to the PDOs

Step 2: Slave configuration

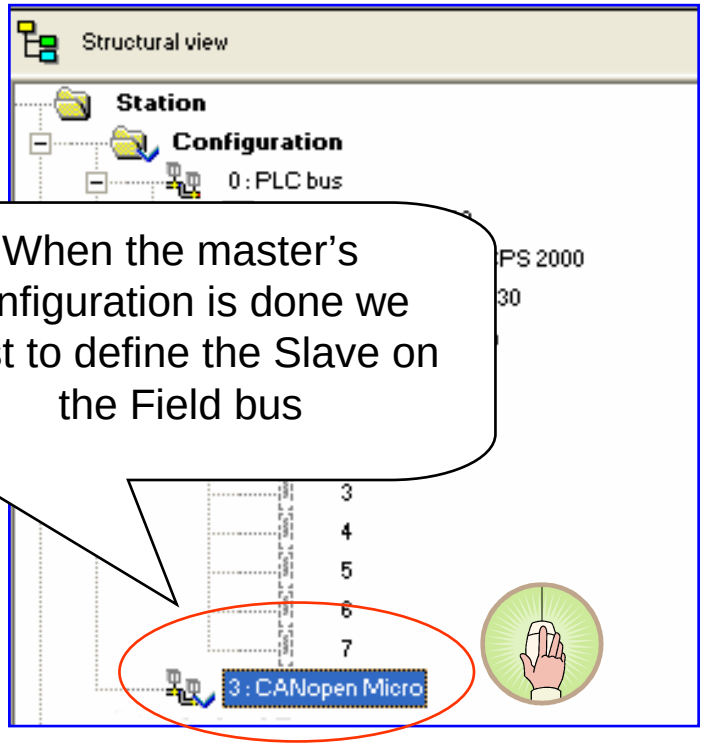


Step 2 : Slave Configuration

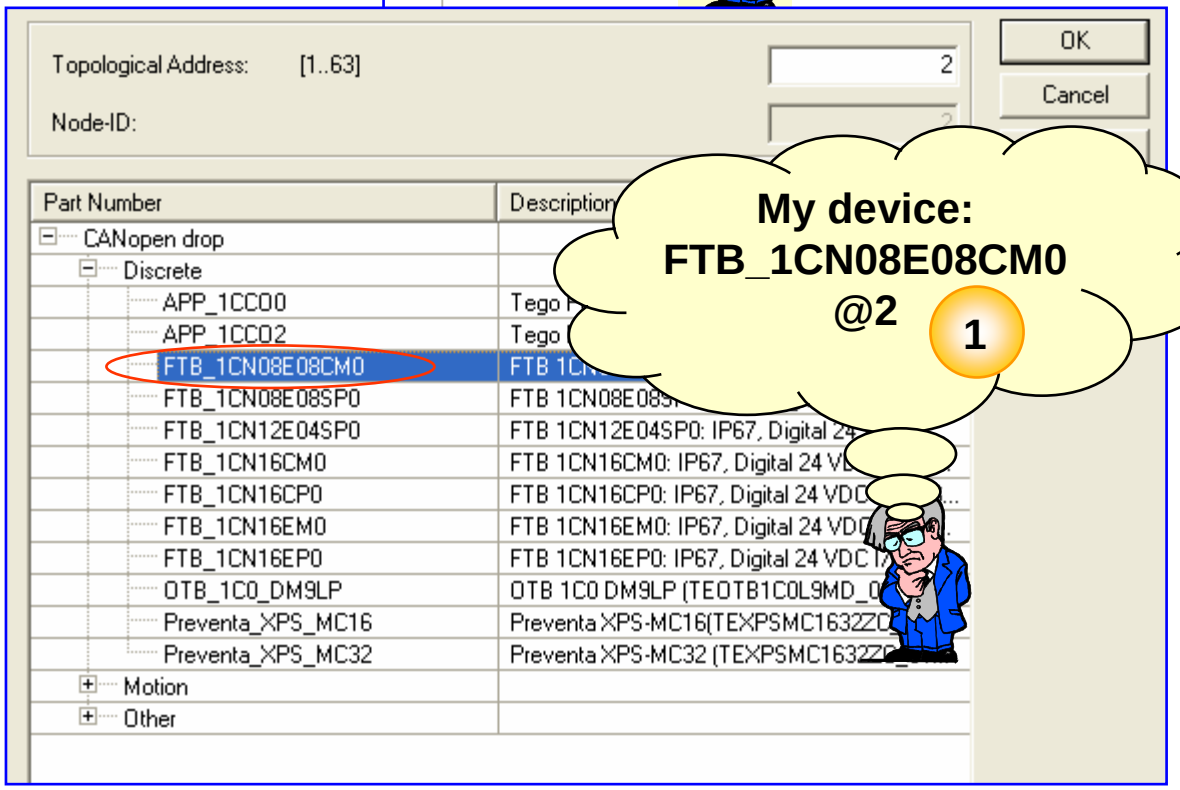




New device: yes

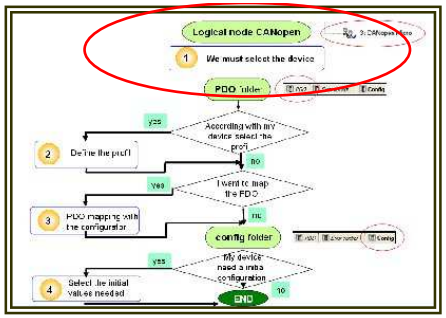


When the master's configuration is done we must to define the Slave on the Field bus

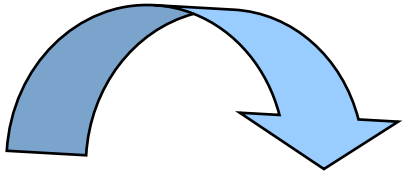


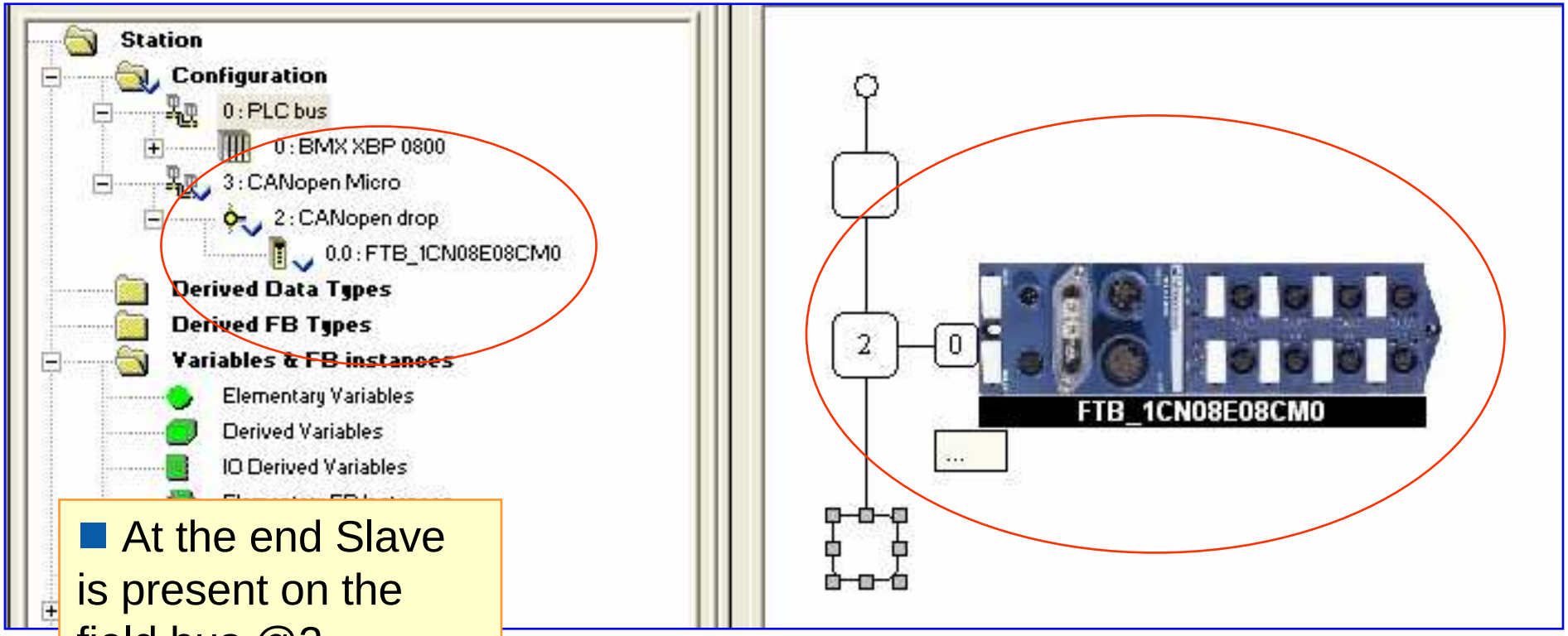
My device: FTB_1CN08E08CM0 @2

1

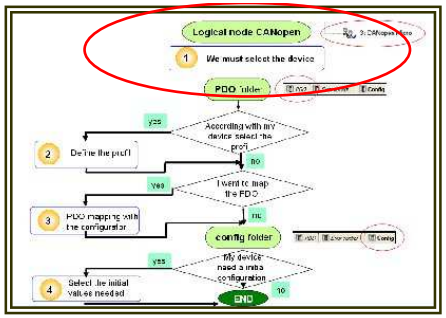


Step 2 : Slave Configuration





- At the end Slave is present on the field bus @2

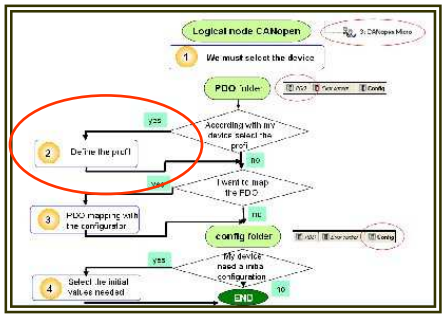


Step 2 : Slave Configuration

Part Number	Description
CANopen drop	
Discrete	
APP_1CC00	Tego Power CANopen (APP1CC00.eds)
APP_1CC02	Tego Power CANopen (APP1CC02.eds)
FTB_1CN08E08CM0	FTB 1CN08E08CM0: IP67, Digital 24 VDC I/O, 8 In...
FTB_1CN08E08SP0	FTB 1CN08E08SP0: IP67, Digital 24 VDC I/O, 8 Inp...
FTB_1CN12E04SP0	FTB 1CN12E04SP0: IP67, Digital 24 VDC I/O, 12 In...
FTB_1CN16CM0	FTB 1CN16CM0: IP67, Digital 24 VDC I/O, 16 Input ...
FTB_1CN16CP0	FTB 1CN16CP0: IP67, Digital 24 VDC I/O, 16 Input ...
FTB_1CN16EM0	FTB 1CN16EM0: IP67, Digital 24 VDC I/O, 16 Input ...
FTB_1CN16EP0	FTB 1CN16EP0: IP67, Digital 24 VDC I/O, 16 Input ...
OTB_1C0_DM9LP	OTB 1C0 DM9LP (TEOTB1C0L9MD_0100E.eds)
Preventa_XPS_MC16	Preventa XPS-MC16(TEXPSMC1632ZC_0105E.eds)
Preventa_XPS_MC32	Preventa XPS-MC32 (TEXPSMC1632ZC_0105E.eds)
Motion	
ATV31_V1_1	Altivar 31 CANopen Slave DSP402 (TEATV3111E.e...
ATV31_V1_2	Altivar 31 CANopen Slave DSP402 (TEATV3112E.e...
ATV31T_V1_3	Altivar 31 CANopen Slave DSP402 (TEATV31T13E....
ATV61_V1_1	ATV61 (TEATV6111E.eds)
ATV71_V1_1	ATV71 (TEATV7111E.eds)
IclA_IFA	IclA-IFA CANopen (IclA-IFA.eds)
IclA_IFE	IclA-IFE CANopen (IclA-IFE.eds)
IclA_IFS	IclA-IFS CANopen (IclA-IFS.eds)
IclA_N065	IclAN065 based on profiles DS301V4.01 and DSP4...
Lexium05	DCX170 CANopen (TEDCX170_0100E.eds)
Lexium05_MFB	LXM05A PLCopen (LEXIUM05_MFB.EDS)
Lexium15_HP	LEXIUM 15 HP servodrive (Lexium 15 MP HP.eds)
Lexium15_MP	LEXIUM 15 MP servodrive (Lexium 15 MP HP.eds)
Osicoder	Osicoder - absolute rotary multi-turn encoders based ...
Other	
STB_NCO_1010	STB NCO 1010 CANopen Network Interface Modul...
STB_NCO_2212	STB NCO 2212 CANopen Network Interface Modul...

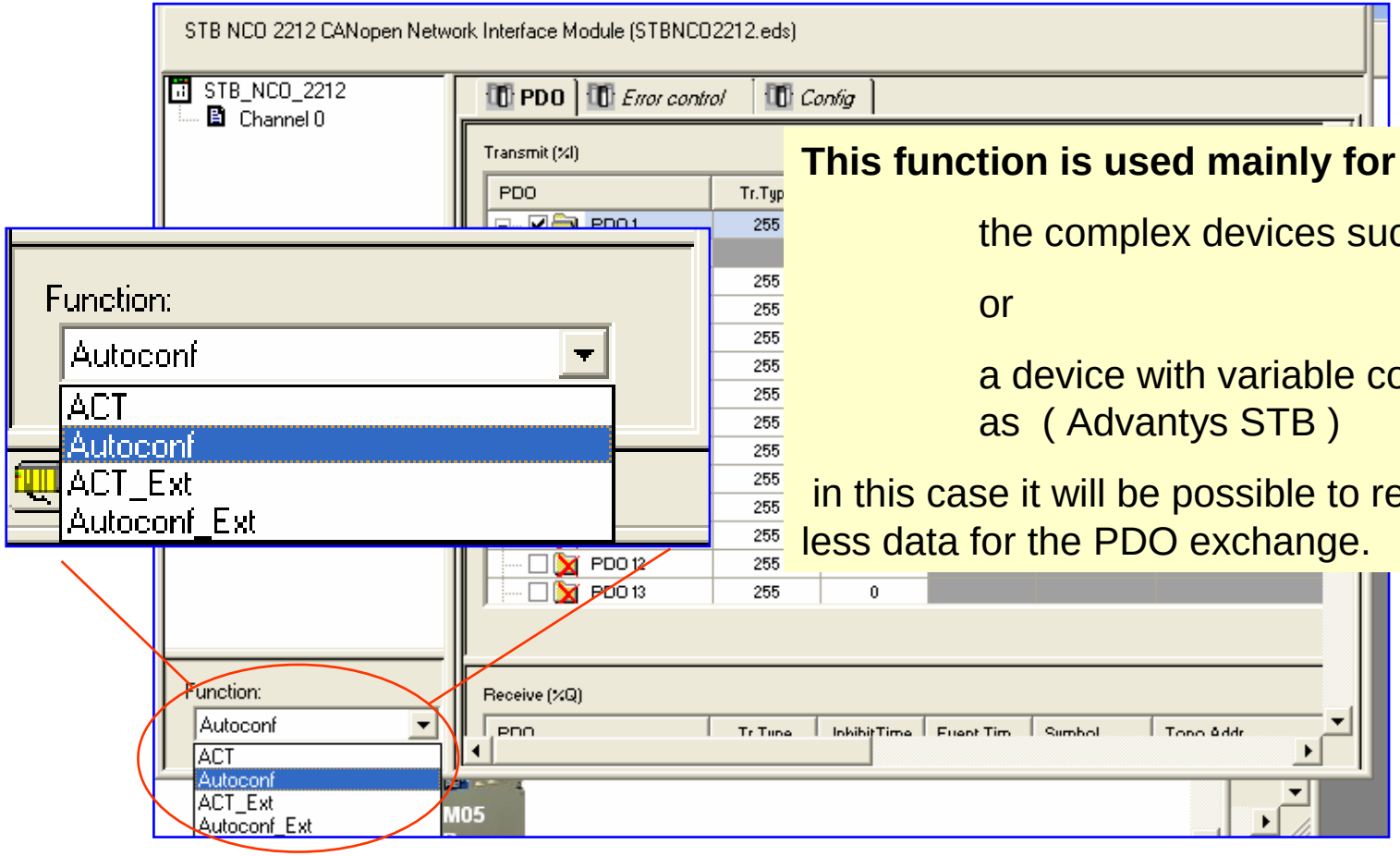
In L11 Unity includes following Schneider devices.

The possibility to add a third party product will be added in L12.



Step 2 : Slave Configuration

- In the configuration screen according with the device used we can select a specific function.



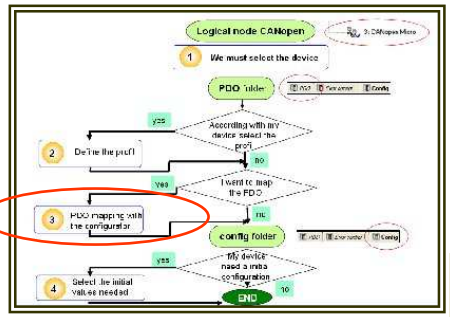
This function is used mainly for :

the complex devices such as (ATV 71)

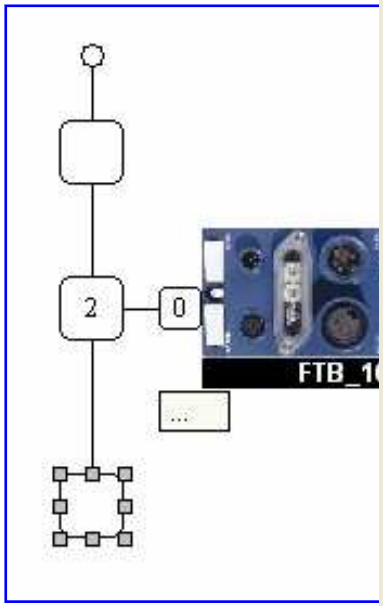
or

a device with variable configuration such as (Advantys STB)

in this case it will be possible to reserve more or less data for the PDO exchange.



Step 2 : Slave Configuration



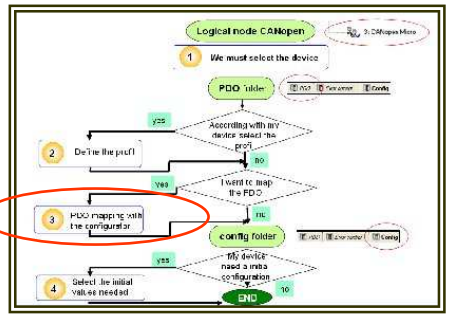
FTB 1CN08E08CM0: IP67, Digital 24 VDC I/O, 8 Input Points, 8 Input Points or Output Points (Configurable), 1.6 A (TEFTB05M01E.eds)

PDO | Error control | Config

PDO	Tr.Type	InhibitTime	Event Ti...	Symbol	Topo.Addr.	%M..	COBID	Index
PDO 1	255	0	0				-	
Digital Input 8 Bit...					%I...			6000:01
Digital Input 8 Bit...					%I...			6000:02
PDO 6	255	0	0				-	
Actuator warning...					%I...			3000:05
Actuator shutdo...					%I...			3000:03
Common diagno...					%IW3.2\0.0.0.0			3000:01
Sensor short cir...					%I...			3000:02

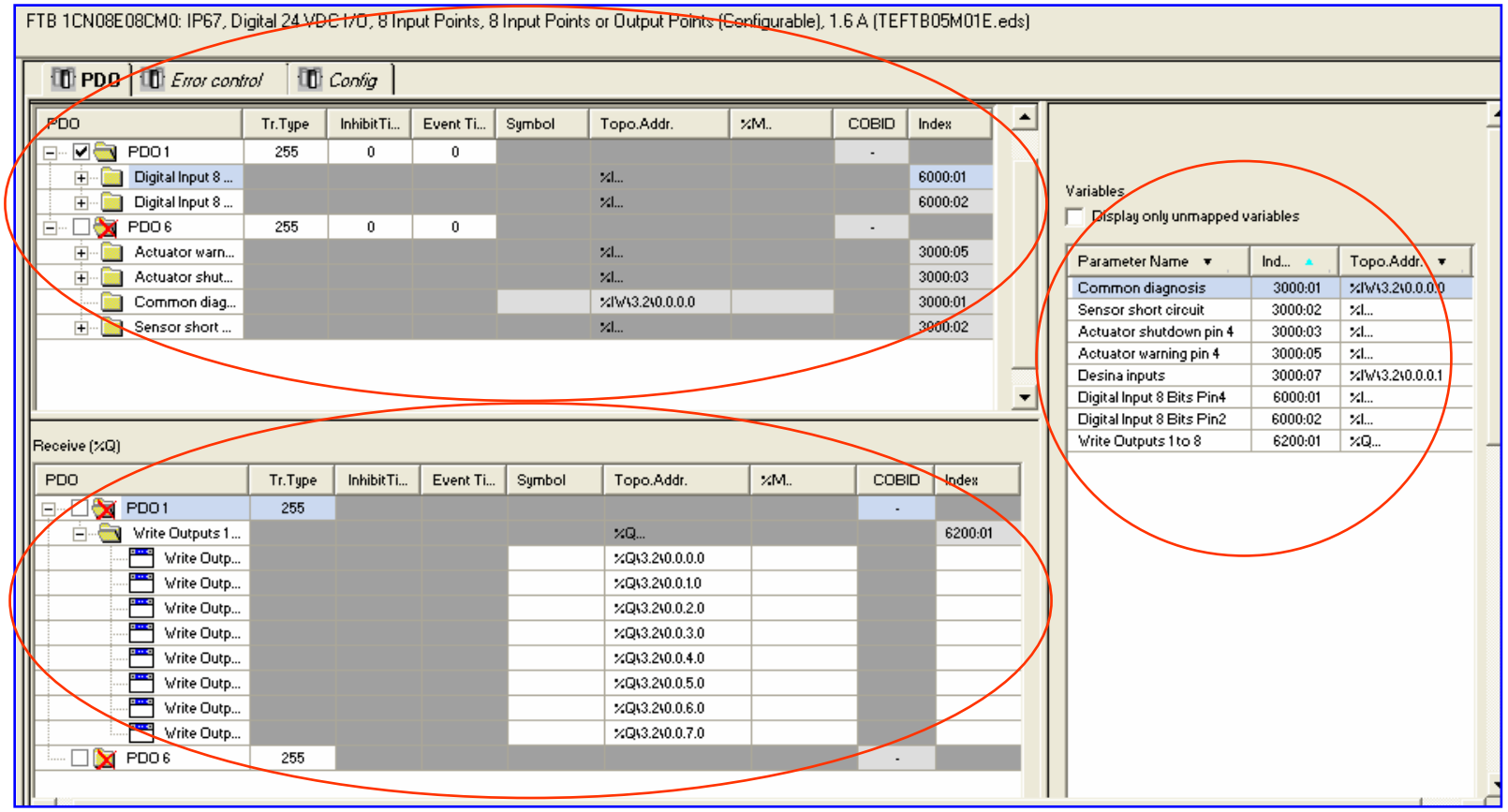
Receive (%Q)

PDO	Tr.Type	InhibitTime	Event Tim...	Symbol	Topo.Addr.	%M..	COBID
PDO 1	255						-
Write Outputs 1 to 8					%Q...		
Write Outputs 1 to 8 Ch:0					%Q3.2\0.0.0.0		
Write Outputs 1 to 8 Ch:1					%Q3.2\0.0.1.0		
Write Outputs 1 to 8 Ch:2					%Q3.2\0.0.2.0		
Write Outputs 1 to 8 Ch:3					%Q3.2\0.0.3.0		
Write Outputs 1 to 8 Ch:4					%Q3.2\0.0.4.0		
Write Outputs 1 to 8 Ch:5					%Q3.2\0.0.5.0		
Write Outputs 1 to 8 Ch:6					%Q3.2\0.0.6.0		
Write Outputs 1 to 8 Ch:7					%Q3.2\0.0.7.0		
PDO 6	255						-



Step 2 : Slave Configuration

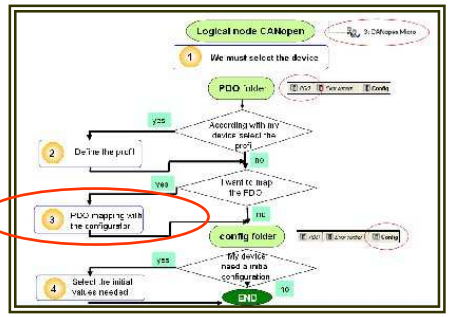
- **PDO** : In this screen it is possible to map the PDO communication.
- This screen is divided in 3 parts :



Transmits PDOs (slave transmits an information to the master)

Receives PDOs (Slave receives an information from the Master)

A list of the variables that can be mapped in the PDOs



Step 2 : Slave Configuration

- **PDO** : In this screen it is possible to map the PDO communication.
- This screen is divided in 3 parts :

FTB 1CN08E08CM0: IP67, Digital 24 VDC I/O, 8 Input Points, 8 Input Points or Output Points (Configurable), 1.6 A (TEFTB05M01E.eds)

PDO | Error control | Config

PDO	Tr.Type	InhibitTi...	Event Ti...	Symbol	Topo.Addr.	%M..	COBID	Index
PDO 1	255	0	0				-	
Digital Input 8 ...					%I...			6000:01
Digital Input 8 ...					%I...			6000:02
PDO 6	255	0	0				-	
Actuator warn...					%I...			3000:05
Actuator shut...					%I...			3000:03
Common diag...					%IW3.2\0.0.0.0			3000:01
Sensor short ...					%I...			3000:02

Variables

Display only unmapped variables

Parameter Name	Ind...	Topo.Addr.
Common diagnosis	3000:01	%IW3.2\0.0.0.0
Sensor short circuit	3000:02	%I...
Actuator shutdown pin 4	3000:03	%I...
Actuator warning pin 4	3000:05	%I...
Desina inputs	3000:07	%IW3.2\0.0.0.1
Digital Input 8 Bits Pin4	6000:01	%I...
Digital Input 8 Bits Pin2	6000:02	%I...
Write Outputs 1 to 8	6200:01	%Q...

Receive (%Q)

PDO	Tr.Type	InhibitTi...	Event Ti...	Symbol	Topo.Addr.	%M..	COBID	Index
PDO 1	255						-	
Write Outputs 1 ...					%Q...			6200:01
Write Outp...					%Q\3.2\0.0.0.0			
Write Outp...					%Q\3.2\0.0.1.0			
Write Outp...					%Q\3.2\0.0.2.0			
Write Outp...					%Q\3.2\0.0.3.0			

With the CANopen configurator if I add a PDO or if I modify a PDO, the mapping for the %Q/I and %MW will never change, so it's not necessary to adjust your application.

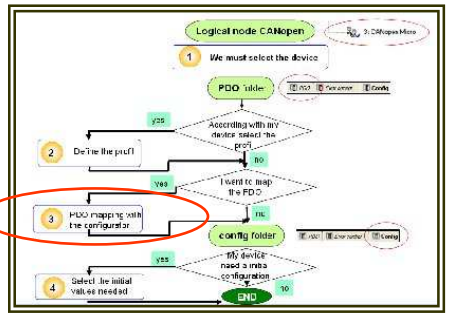
- According to the EDS file somes PDOs are already mapped
- The PDOs used Topologics addresses %Ix or %Qx.

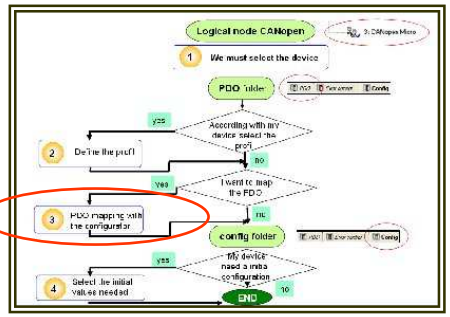
and

Internals variables %MW (these variables are always present in the product for the MODBUS communication)

Step 2: Slave configuration

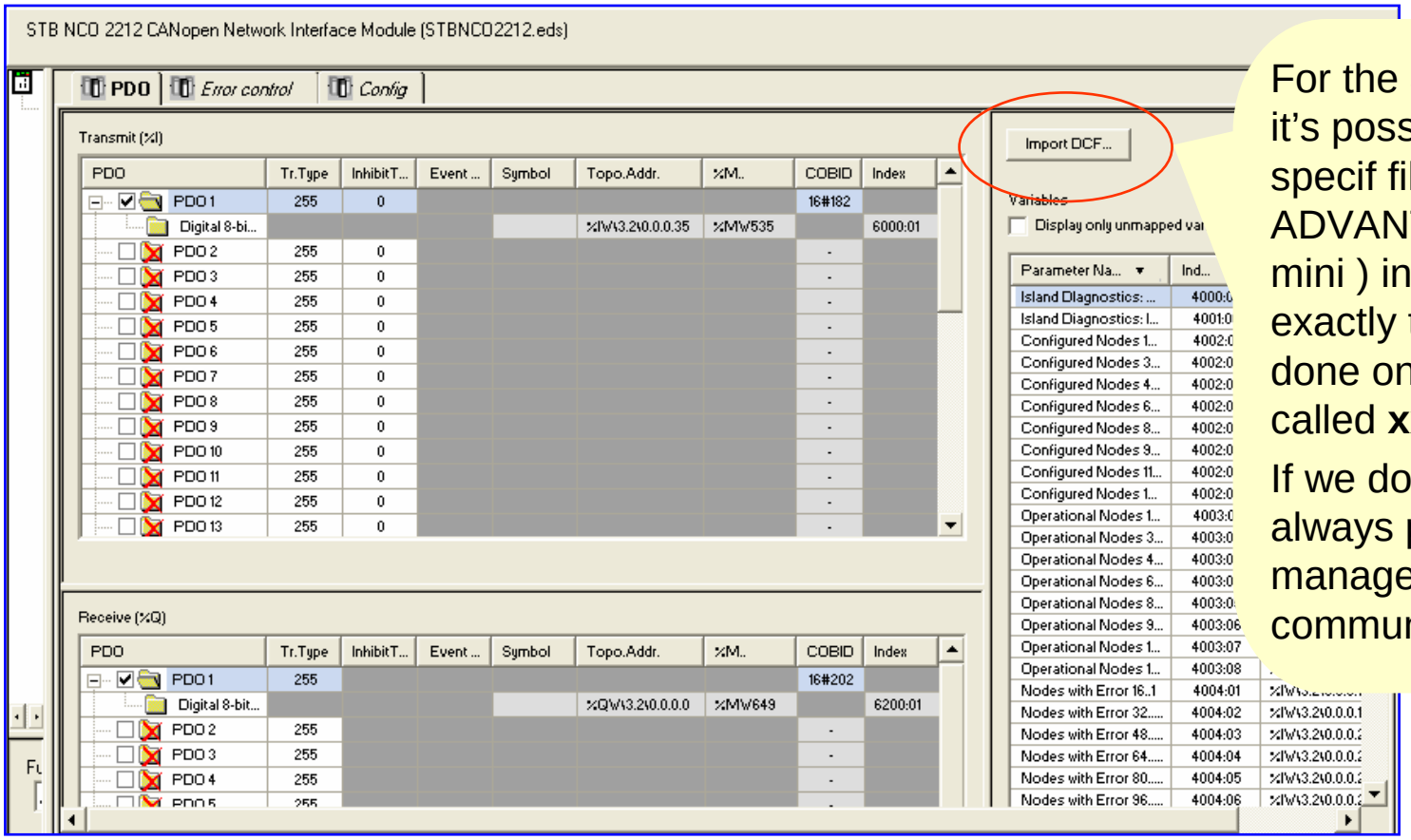
How to do the PDO configuration





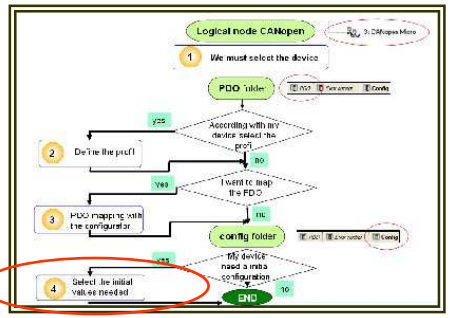
Step 2 : Slave Configuration

- Particularity : For the STB devices.

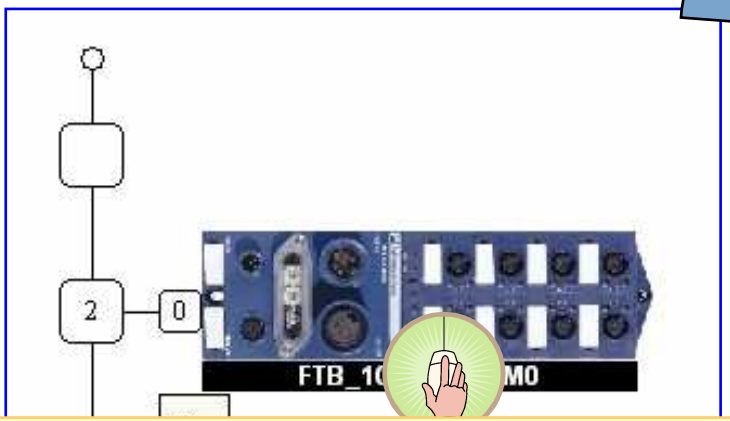


For the **STB** configuration it's possible to associate a specif file created with ADVANTYS (version 2.3 mini) in order to design exactly the configuration done on STB, this file is called **xxx.DCF.**

If we don't used this file it's always possible to manage the communication manually



Step 2 : Slave Configuration



FTB 1CN08E08CM0: IP67, Digital 24 VDC I/O, 8 Input Points, 8 Input Points or Output Points (Configurable), 1.6 A (TEFTB05M01E.eds)

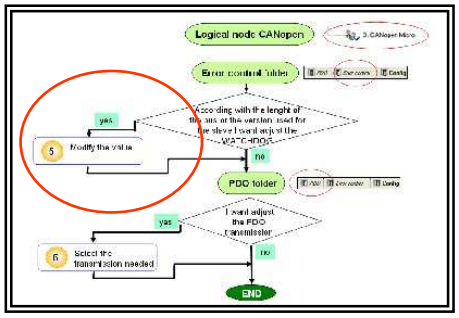
FTB_1CN08E08CM0
Channel 0

PDO | Error contro | Config

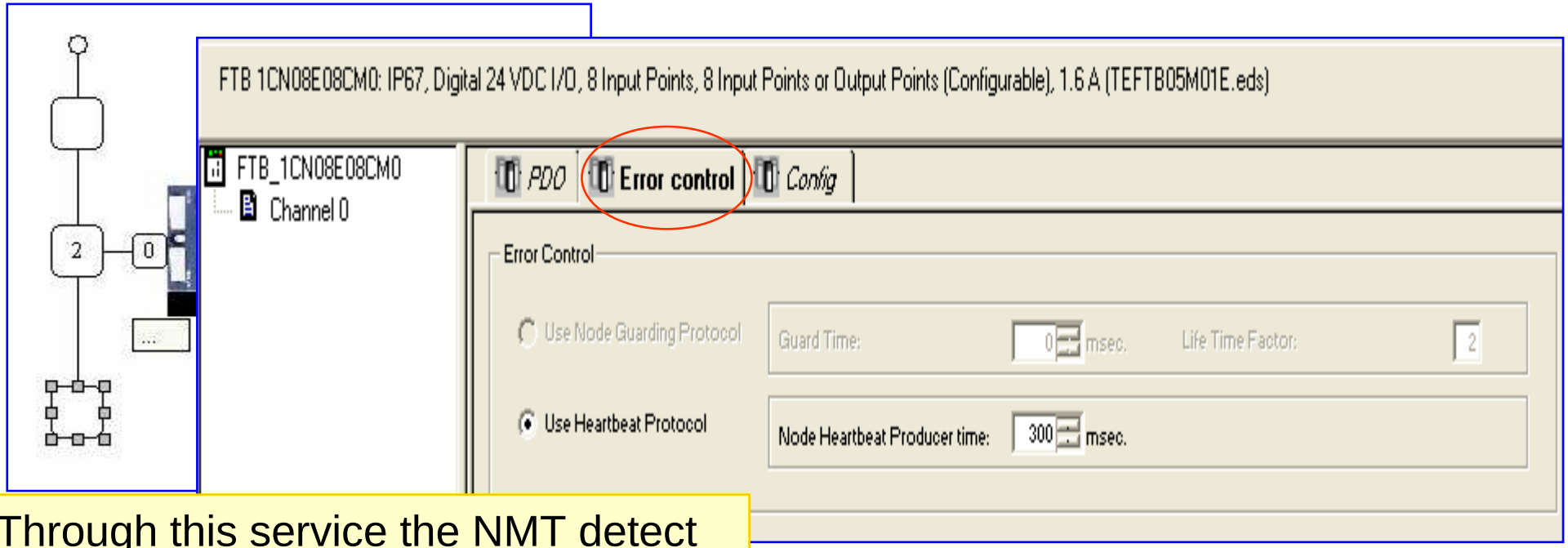
	Index	Label	Value
0	2001:01	Input/Output Ch:0	0
1	2001:01	Input/Output Ch:1	0
2	2001:01	Input/Output Ch:2	0
3	2001:01	Input/Output Ch:3	0
4	2001:01	Input/Output Ch:4	0
5	2001:01	Input/Output Ch:5	0
6	2001:01	Input/Output Ch:6	0
7	2001:01	Input/Output Ch:7	0
8	6103:01	Filter Input Ch:0	0
9	6103:01	Filter Input Ch:1	0
10	6103:01	Filter Input Ch:2	0
11	6103:01	Filter Input Ch:3	0
12	6103:01	Filter Input Ch:4	0
13	6103:01	Filter Input Ch:5	0

- The goal here is to provide a method of configuring the node without an external configuration software (here ADVANTIS)
- if the value is different to 0 the value will be send during the initialisation phase

Step 2: Slave configuration ‘EXPERT’

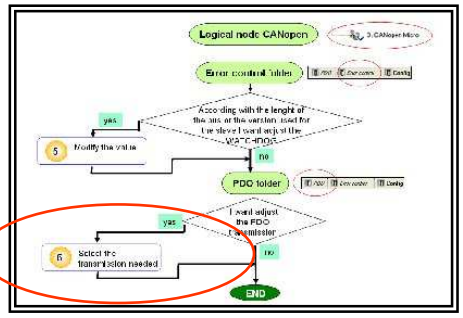


Step 2 : Slave Configuration ‘EXPERT’



Through this service the NMT detect the error on the field bus, we can select Heartbeat or node guarding protocol, and adjust the Timeout

5



Step 2 : Slave Configuration 'EXPERT'

- In this screen it is possible to map the PDO communication.
- This screen is divided in 3 parts :

DCX170 CANopen (TEDCX170_0100E.eds)

PDO | Error control | Config

PDO	Tr.Type	InhibitTi...	Event Ti...	Symbol	Topo.Addr.	%M..	COBID	Index
PDO 1 (Static)	255	0	0				16#182	
Statusword					%IW\3.2\0.0.0.16			6041:00
PDO 2 (Static)	255		100				-	
Statusword					%IW\3.2\0.0.0.16			6041:00
Position actu...					%ID\3.2\0.0.0.8			6064:00
PDO 3 (Static)	255		100				-	
Statusword					%IW\3.2\0.0.0.16			6041:00
Velocity actua...								
PDO 4	254	0						

Bi left

Select **Type, Inhibit or event** on the PDO line in order to define the PDO communication :

Synchronous

Asynchronous

6

Receive (%Q)

PDO	Tr.Type	InhibitTi...	Event Ti...	Symbol	Topo.Addr.	%M..	COBID	Index
PDO 1 (Static)	255						16#202	
Controlword					%QW\3.2\0.0.0.25			6040:00
PDO 2 (Static)	255						-	
Controlword					%QW\3.2\0.0.0.25			6040:00
Target position					%QD\3.2\0.0.0.14			607A:00
PDO 3 (Static)	255						-	
Controlword					%QW\3.2\0.0.0.25			6040:00

Bi left

Variables

Display only unmapped va

Parameter Name		
RAMPsym		
_IO_act		
ANA1_act		
PTPp_relpact		
GEARdenom		
GEARnum		
Controlword		
Statusword		
position actual value int		
Position actual value	6064:00	%ID\3.2\0.0.0.8
Velocity actual value	606C:00	%ID\3.2\0.0.0.10
Target position	607A:00	%QD\3.2\0.0.0.14
Profile velocity	6081:00	%QD\3.2\0.0.0.16

PDO 1 (Static)

Transmission Type

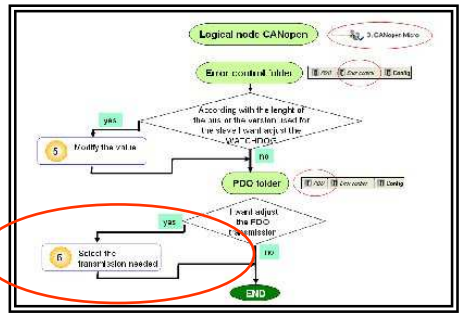
- Synchronous acyclic (0)
- Synchronous cyclic (1-240) — 1 — x Sync period
- Asynchronous (Manuf. Event) (254)
- Asynchronous (Profile Event) (255)

Properties

Inhibit time (0-65535) : 2 x 100 us

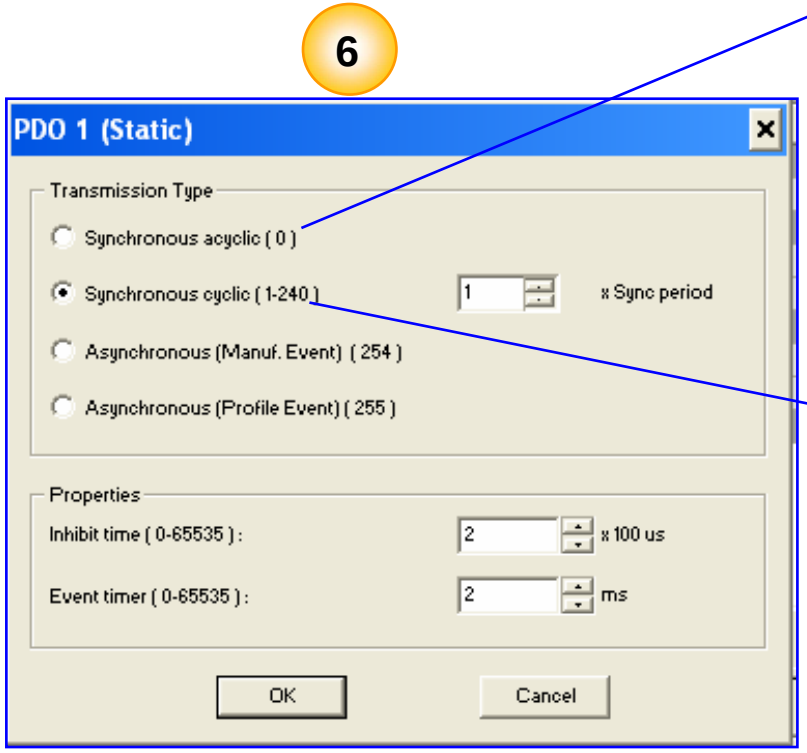
Event timer (0-65535) : 2 ms

OK | Cancel



Step 2 : Slave Configuration 'EXPERT'

- The PDO are exchanges according with the configuration done .(example transmission type)

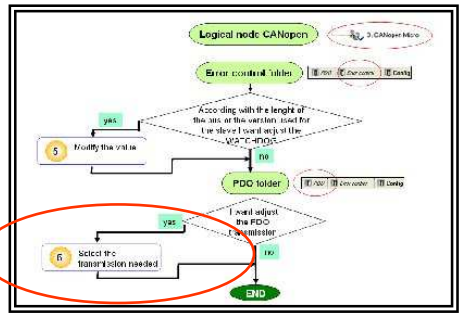


Synchronous acyclic PDO :

A transmission type of 0 means that the message shall be transmitted synchronously with the SYNC message but not periodically according to the value. Mainly used for drives.

Synchronous cyclic PDO :

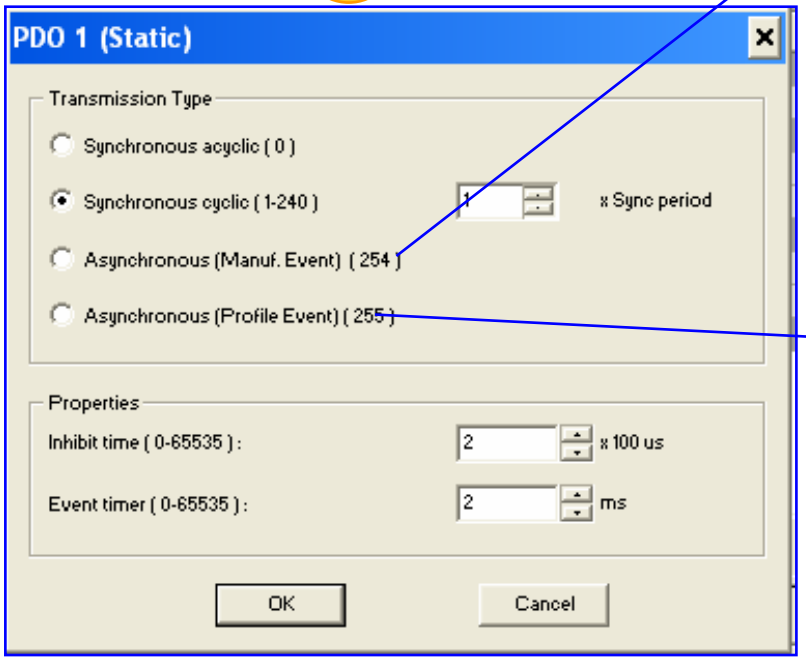
A value between 1 and 240 means that the PDO is transmitted synchronously and cyclically, the transmission type value indicating the number of SYNC messages between two PDO transmissions.



Step 2 : Slave Configuration 'EXPERT'

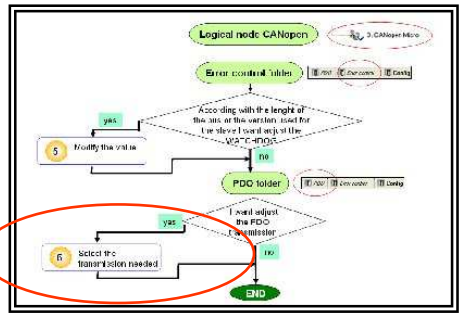
- The PDO are exchanges according with the configuration done .(example transmission type)

6



Asynchronous PDO:
The transmission type 254 means that the PDO is transmitted asynchronous. It is fully depending on the implementation in the device. Mainly used for digital I/O.

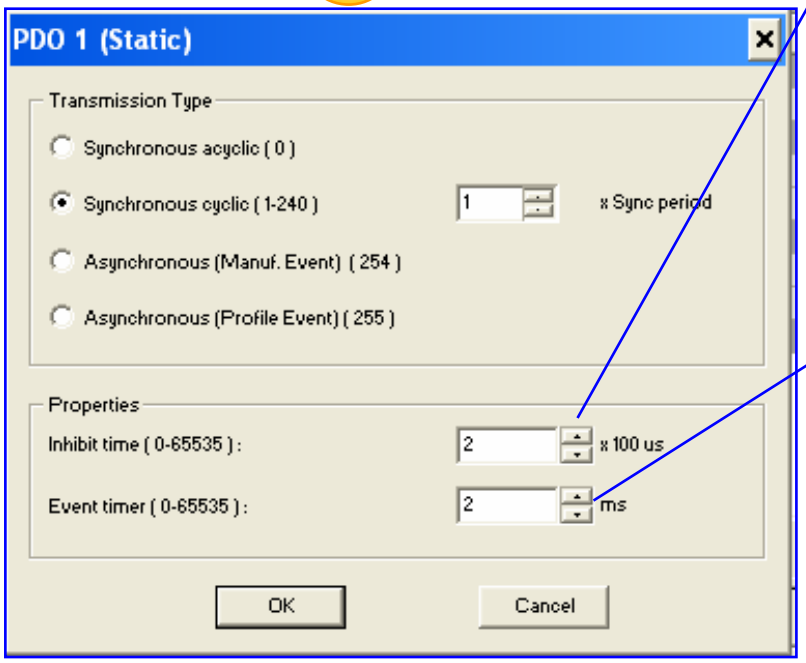
Asynchronous PDO:
The transmission type 255 means that the PDO is transmitted asynchronous when the value change.



Step 2 : Slave Configuration 'EXPERT'

- The PDO are exchanges according to the configuration. (example transmission type)

6



Inhibit time :
masks the communication during this time.

Nota: With the MFB we can only adjust this value according with the traffic and the length of the bus

Event timer :
Time to manage an event in order to start a PDO

Reminder : How to realize a complete configuration

- In this example we will configure one device :

LEXIUM 05

@2

Speed 500Kb

With

PDO4 In receive:

JOG-Activate

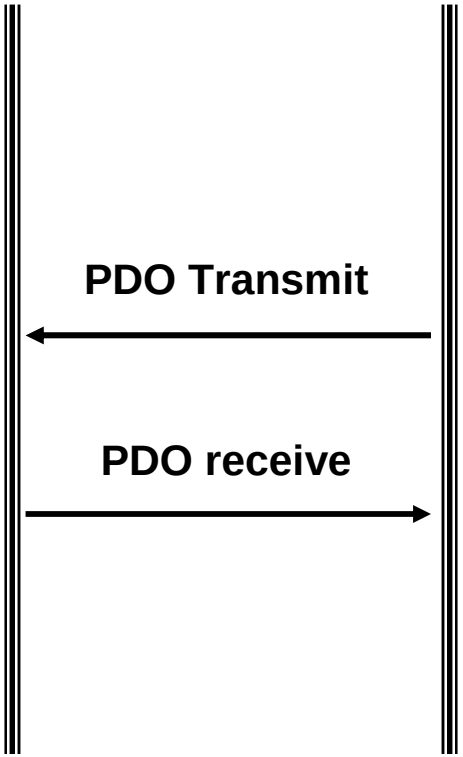
PDO4 in transmit

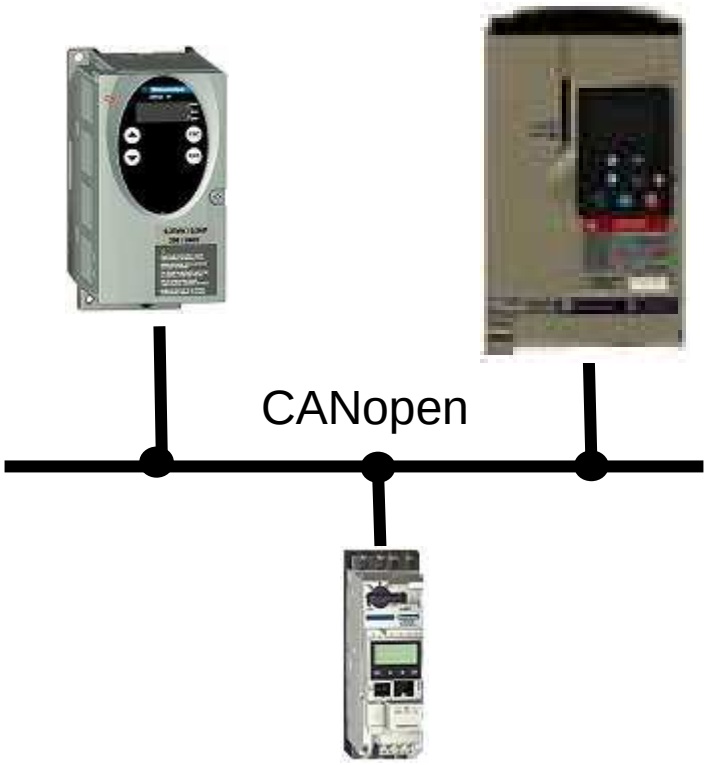
IO-act

D – Programming

Management Implicit exchanges PDO

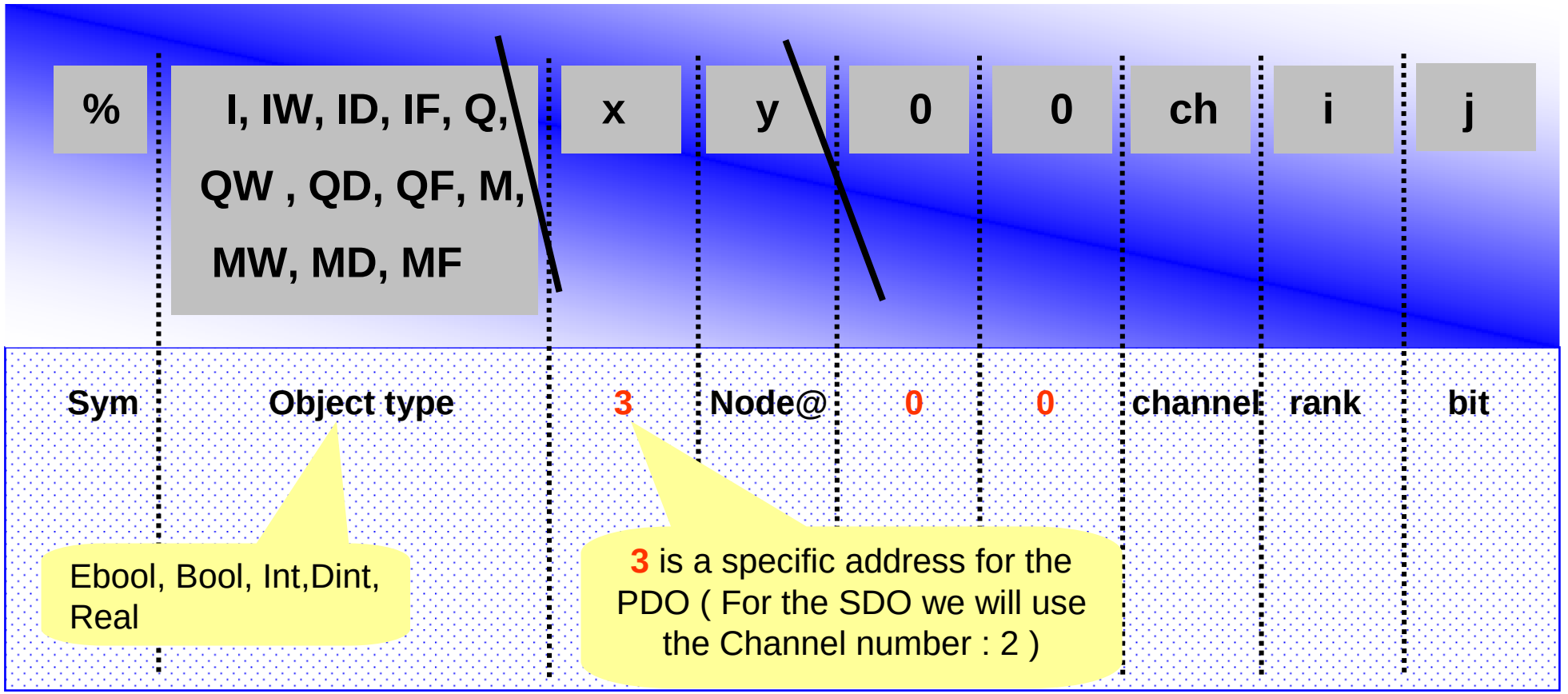
PDOs allow the implicit data management between the Master, in our case the M340 CPU and the slave by read/write words.





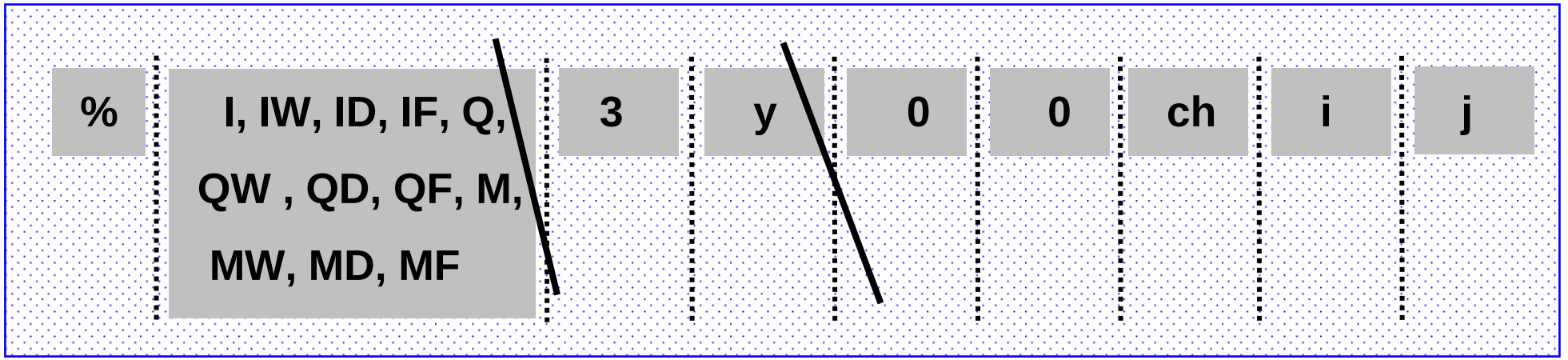
Addressing

- On Modicon M340, PDO variable addresses can be defined as word, double word and floating point types.



Addressing

- Example.



1. Read an input on an ADVANTYS @ 4, channel 2, Word 3, bit2

   %IW \ 3. 4 \ 0 . 0 . 2 . 3 . 2

2. Write control word on an LEXIUM @ 9, channel 4, Word 3

   %QW \ 3. 9 \ 0 . 0 . 4 . 3

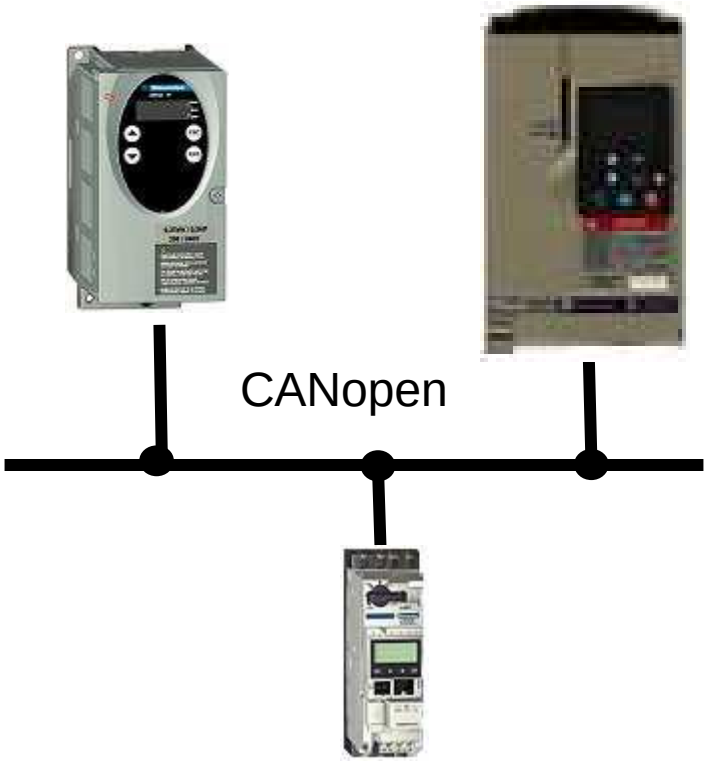
Management explicit exchanges SDO

SDOs allow the management of the explicit data between the Master, in our case the M340 and the slave using of request READ_var / WRITE_var

PLC Management of the request

READ_VAR request

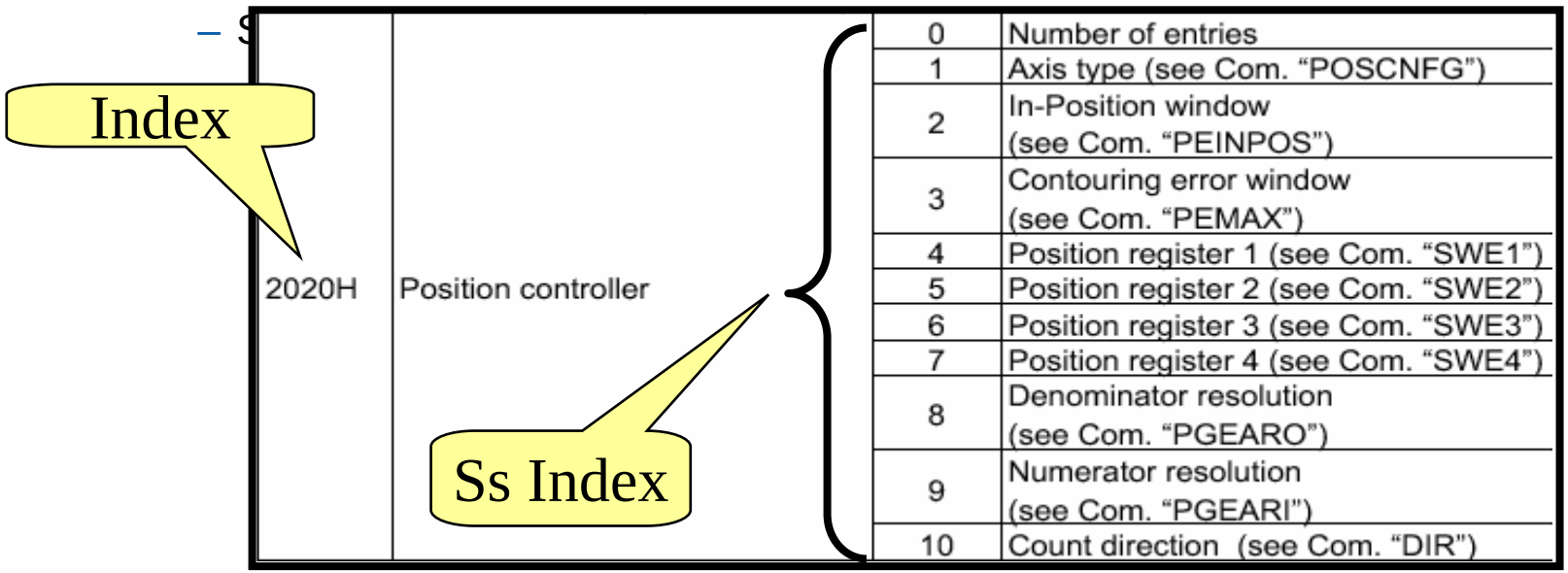
WRITE_VAR request



Management explicit exchanges SDO

INDEX / Ss INDEX Concept .

- All requests on CANopen are defined with an INDEX / Ss INDEX
  .
- Here is an example on a CANopen device ‘ LEXIUM 15x ‘
  - INDEX : Define the function wished .
    - Ex : 2022 => Position data for the ‘ position ’ mode .
      2020 => Position Controller
  - S



Management explicit exchanges SDO

- Syntax to program a SDO (Write var) :

```
If write_SDO then
        buffer_gest_write[2] := 50;     (*Time-out*)
        buffer_gest_write[3] := 4;      (*Longueur*)
        (*buffer_sent[0] := 200;*)

        WRITE_VAR (ADR := ADDM('0.0.2.2'),
                OBJ := 'SDO',
                NUM := 16#00001005,
                NB := 2,
                EMIS := buffer_sent,
                GEST := buffer_gest_write) ;

        write_SDO := false;
end_if;
```

Parameters	descriptions
ADDM('0.0.2.node')	Address of the exchange destination **0.0 :** processor slot in the rack (0) **2** : channel (always 2 for CANopen) 2: Device adress.
'SDO'	SDO object type (always **SDO** in capitals)
Subindex / index:	Double word or immediate value identifying the CANopen SDO index or subindex:
NodeID	Word or value identifying the destination device on the CANopen bus
%MWi:L	Table of words containing the data to be sent (minimum length = 1)
%MWk:4	Exchange management parameters

Management explicit exchanges SDO

- Syntax to program a SDO (Read var) :

```
IF read_SDO      then
        buffer_gest_read[2] := 50;(*Time-out*)
        buffer_rec[0]:=0;

        READ_VAR
                (ADR := ADDM('0.0.2.2'),
                OBJ := 'SDO',
                NUM := 16#00001005,
                NB := 2,
                GEST := buffer_gest_read,
                RECP => buffer_rec) ;



                read_SDO := false;
                END_IF;
```

Parameters	descriptions
ADDM('0.0.2.node')	Address of the exchange destination **0.0 :** processor slot in the rack (0) **2** : channel (always 2 for CANopen) 2: Device adress.
'SDO'	SDO object type (always **SDO** in capitals)
Subindex / index:	Double word or immediate value identifying the CANopen SDO index or subindex:
NodeID	Word or value identifying the destination device on the CANopen bus
%MWk:4	Exchange management parameters
%MWi:L	Table of words containing the data to be received (minimum length = 1)

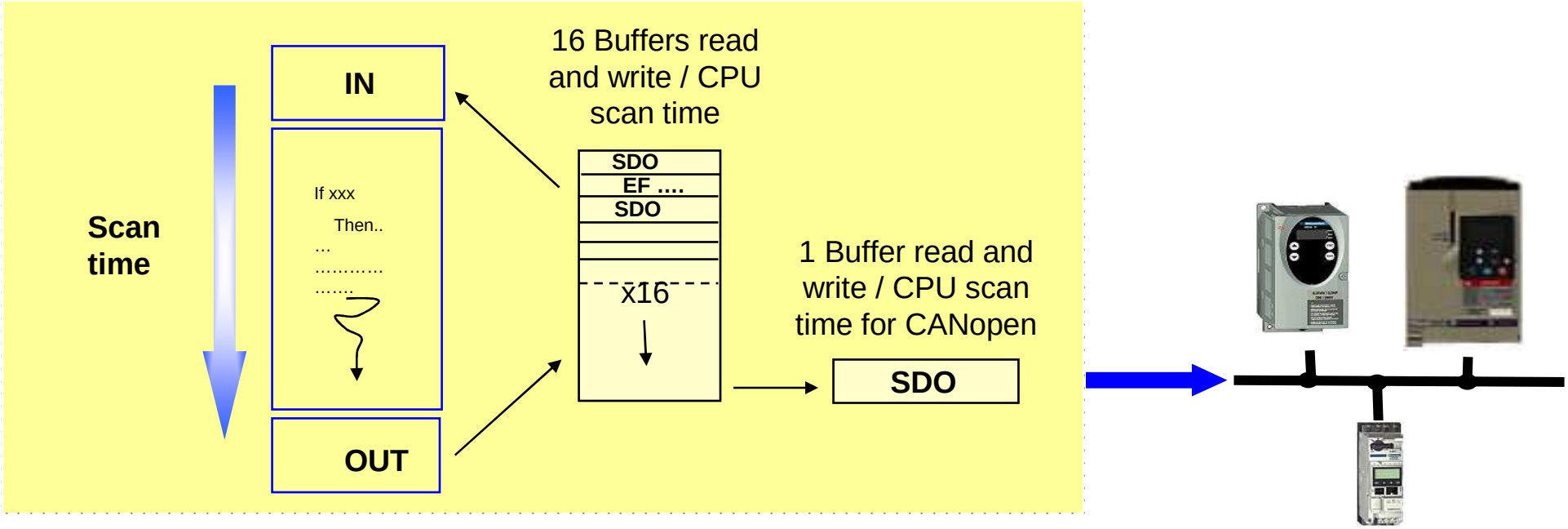
Management explicit exchanges SDO

- Performances :
  - For all the processors we can manage :

    16 messages in Input / Output by scan time of the MAST task**.**

  - For CANopen message channel (SDO) we are limited to :

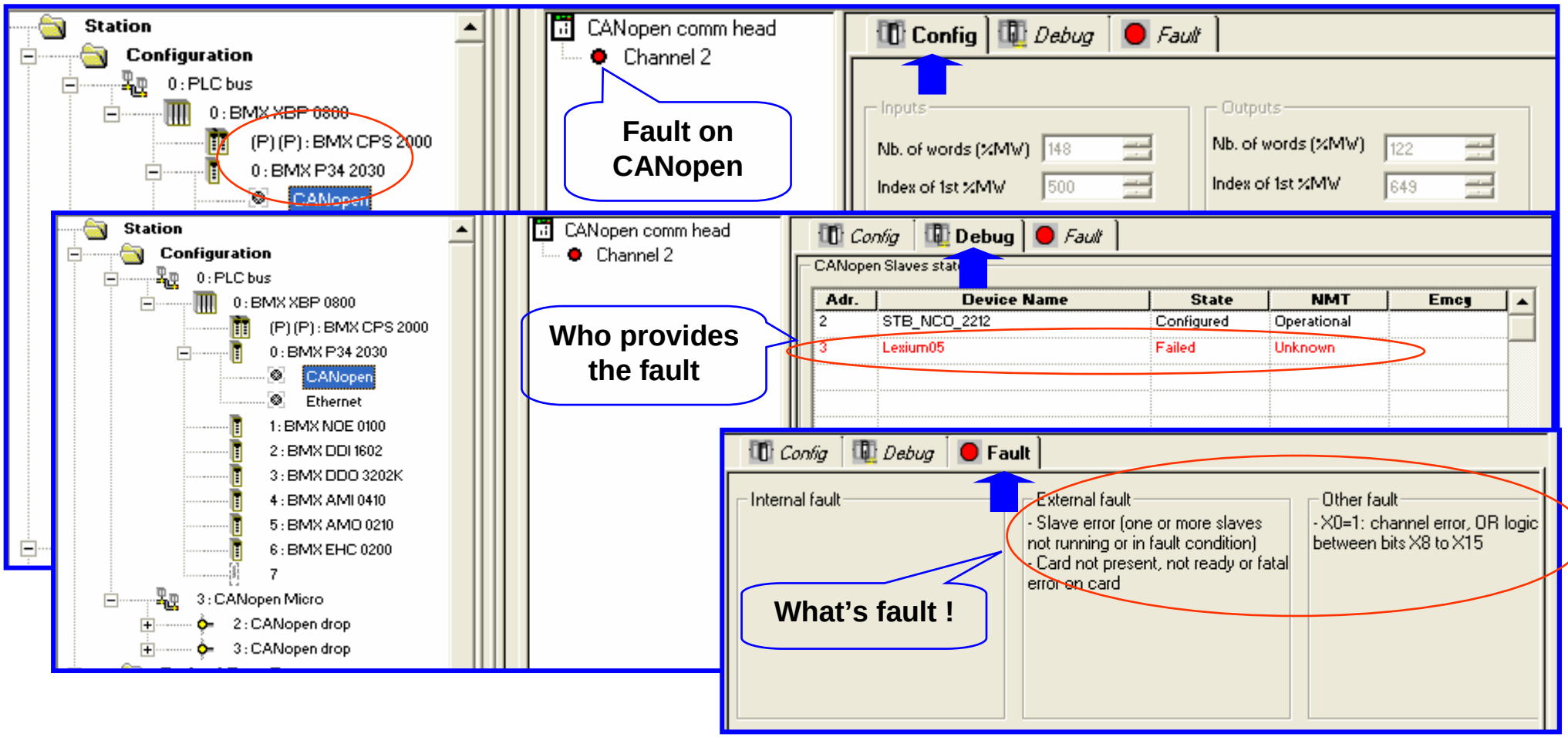
    1 message in Input / Output by scan time of the MAST task.



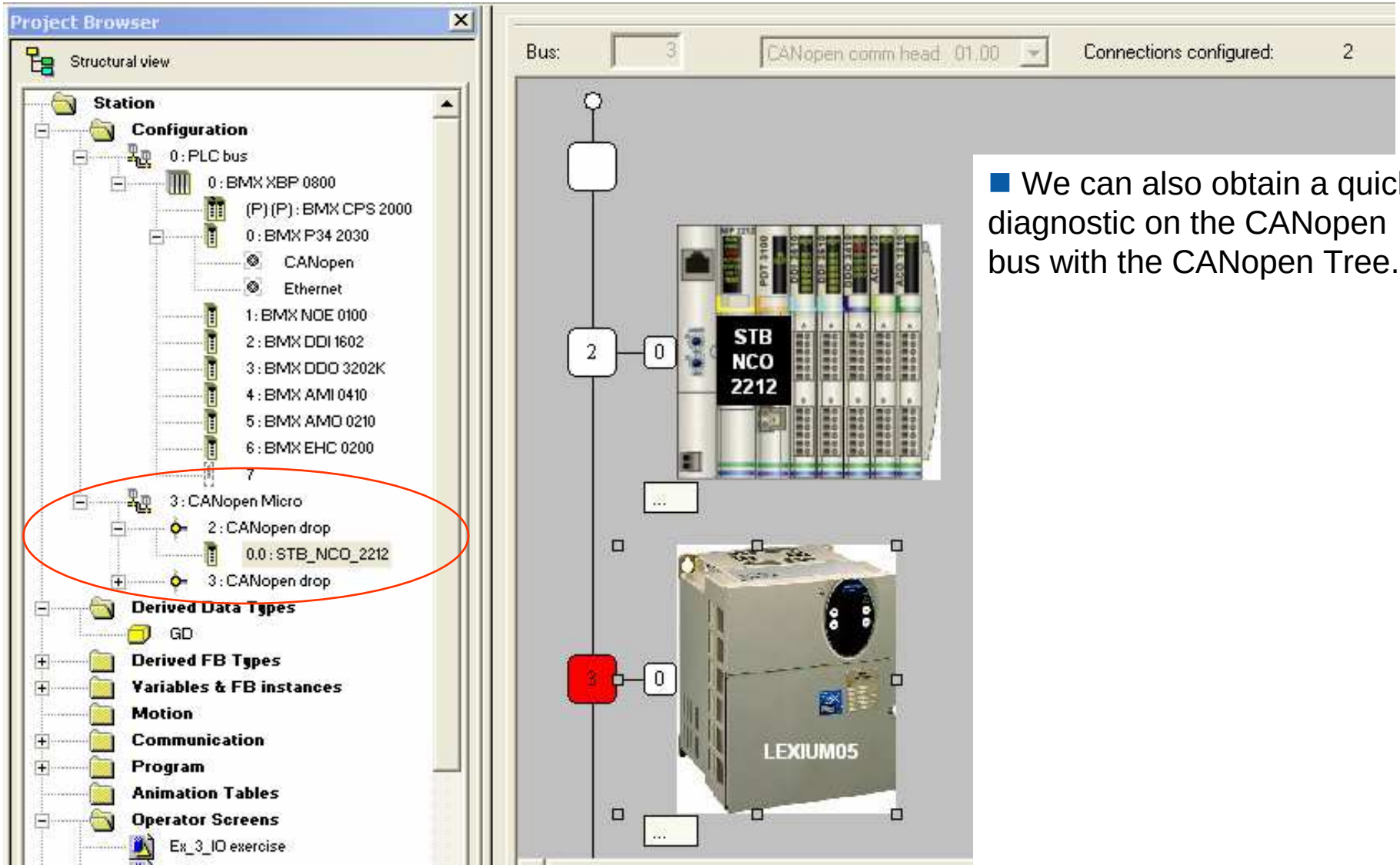
E – Debug & Diagnostic

Debug and Diagnostic Screen

- On the processor it's possible to realize a quick diagnostic on the CANopen bus



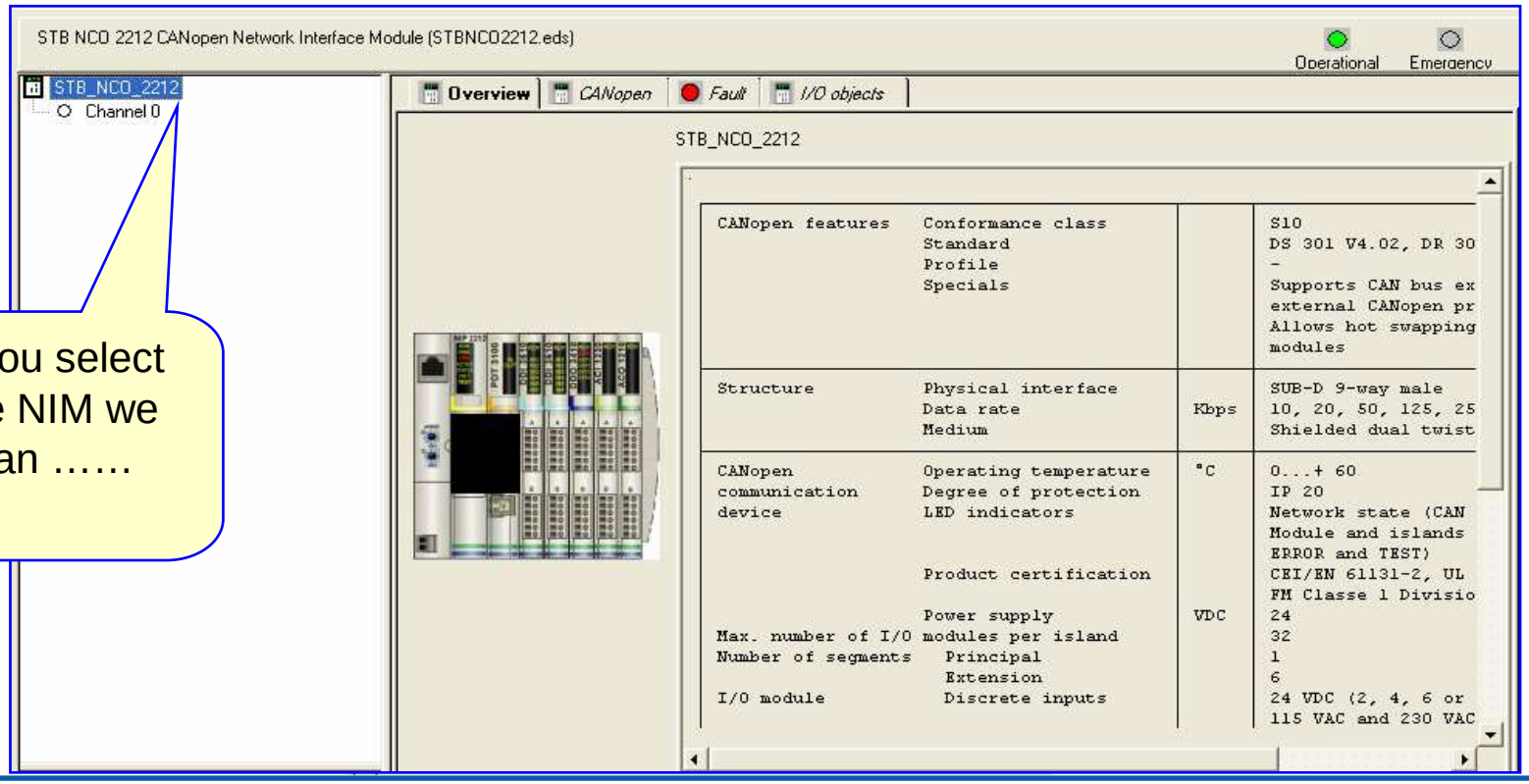
Debug and Diagnostic Screen



- We can also obtain a quick diagnostic on the CANopen bus with the CANopen Tree.

Debug and Diagnostic Screen

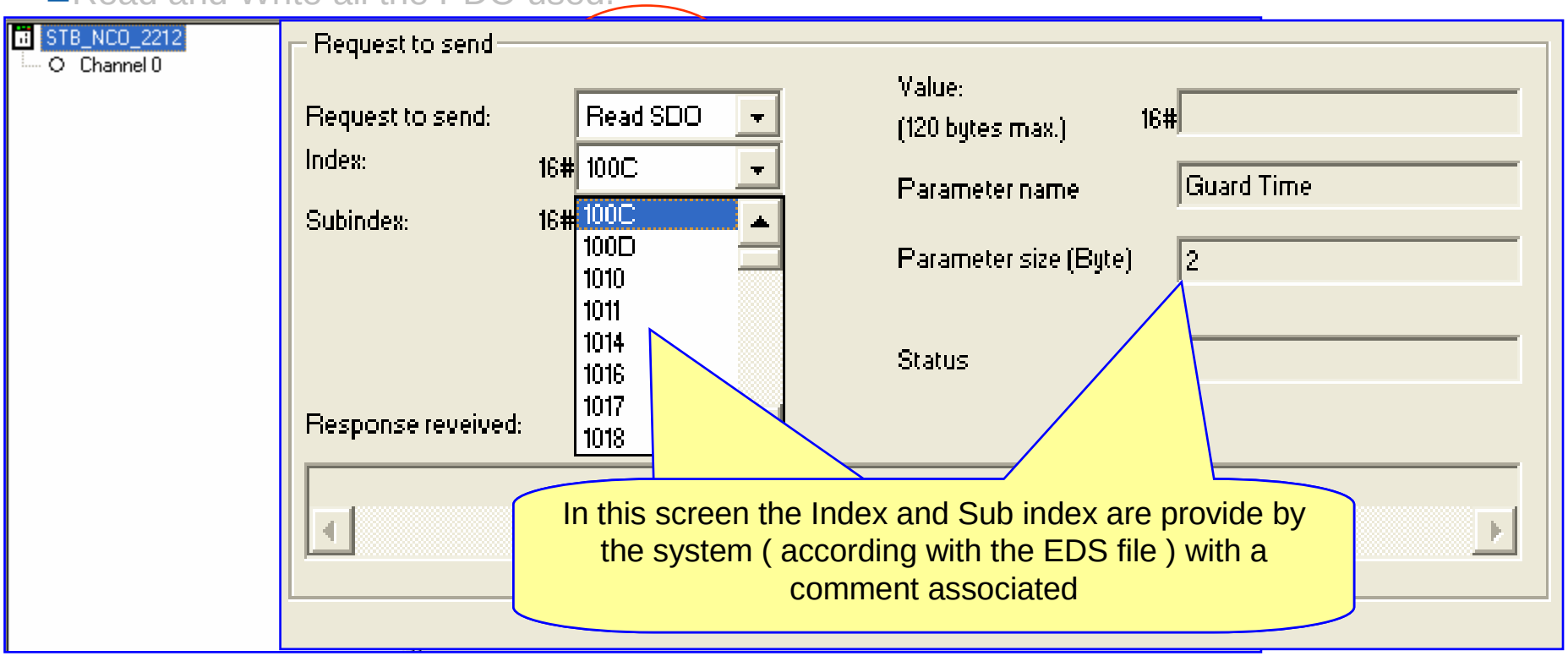
- On BUS editor It is possible to :
  - Read and Write all the SDO through the INDEX / SUB INDEX of the device.
  - Read the Emergency message (Information provides by the device)
  - Read the I/O object Available on the device.
  - Read and Write all the PDO used.



If you select the NIM we can ……

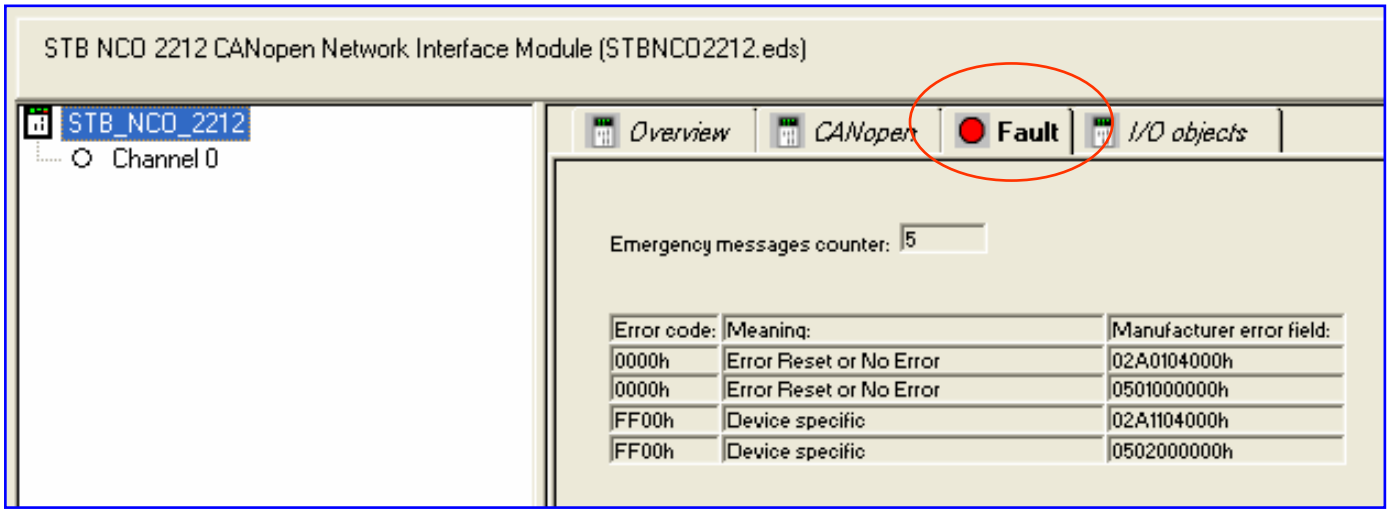
Debug and Diagnostic Screen

- On BUS editor It is possible to :
  - **Read and Write all the SDO through the INDEX / SUB INDEX of the device.**
  - Read the Emergency message (Information provides by the device)
  - Read the I/O object Available on the device.
  - Read and Write all the PDO used.



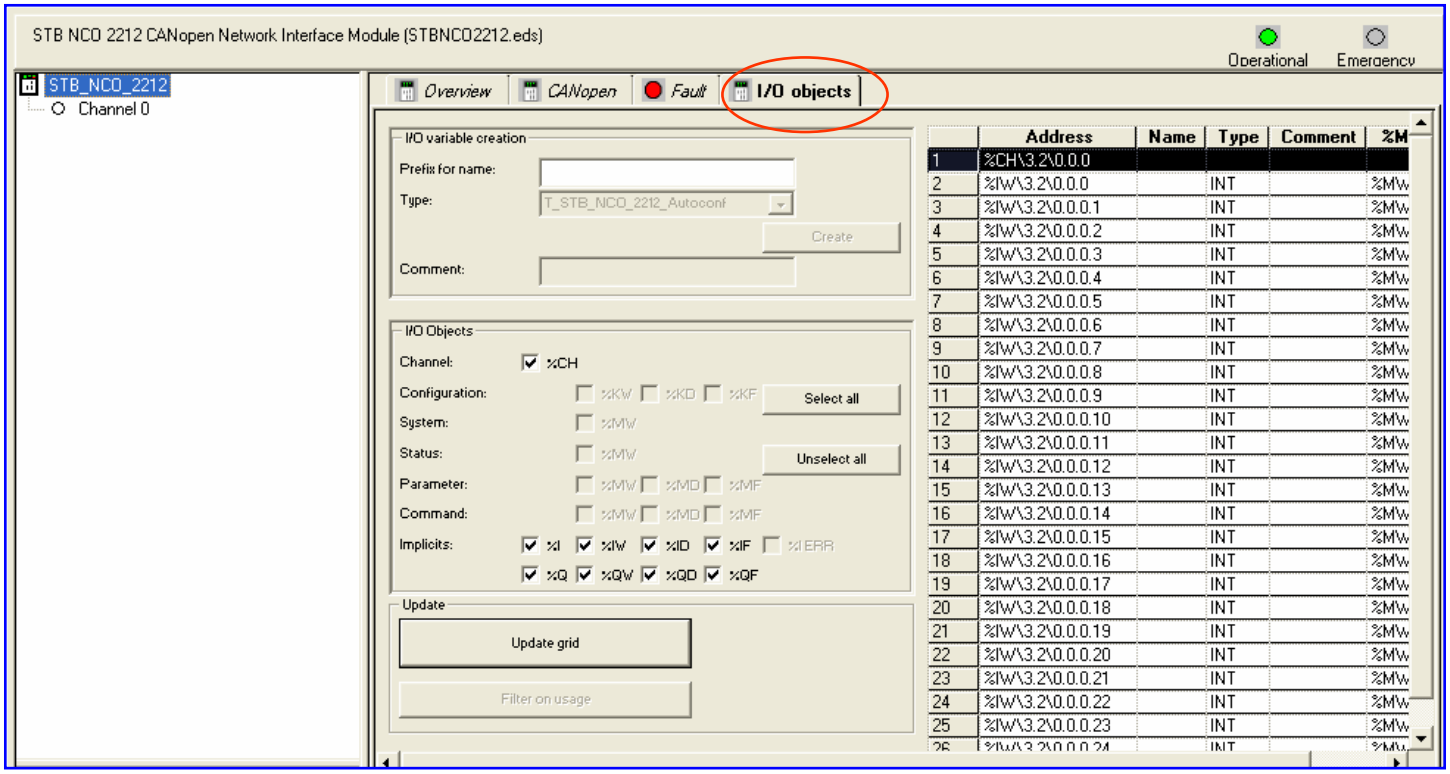
Debug and Diagnostic Screen

- On BUS editor Tree It is possible to :
  - Read and Write all the SDO through the INDEX / SUB INDEX of the device.
  - **Read the Emergency message (Information provides by the device)**
  - Read the I/O object Available on the device.
  - Read and Write all the PDO used.



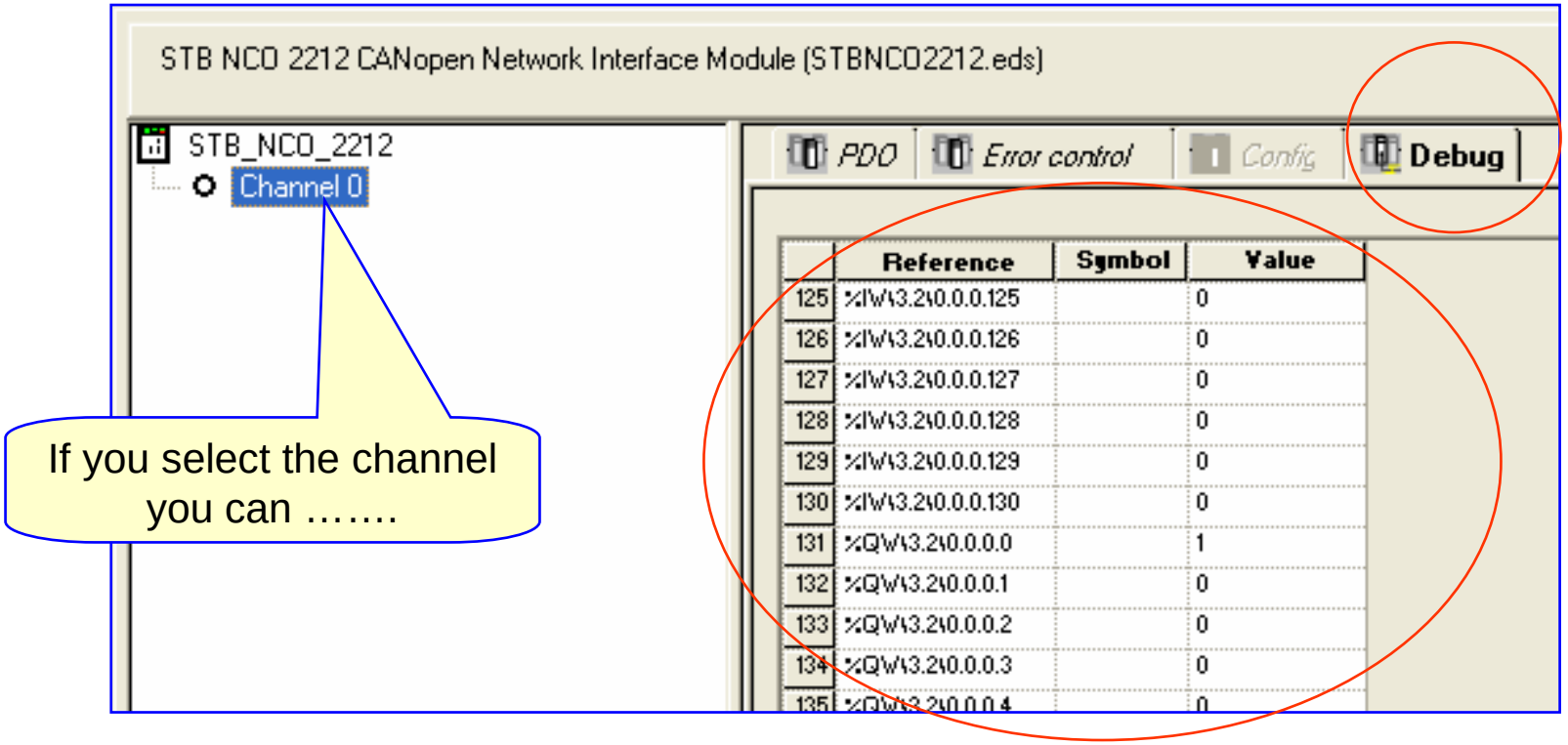
Debug and Diagnostic Screen

- On BUS editor It is possible to :
  - Read and Write all the SDO through the INDEX / SUB INDEX of the device.
  - Read the Emergency message (Information provides by the device)
  - **Read the I/O object Available on the device.**
  - Read and Write all the PDO used.



Debug and Diagnostic Screen

- On BUS editor Tree It is possible to :
  - Read and Write all the SDO through the INDEX / SUB INDEX of the device.
  - Read the Emergency message (Information provides by the device)
  - Read the I/O object Available on the device.
  - **Read and Write all the PDO used.**



CANOpen


Telemecanique
Modicon
M340
BMX P 34 2030
BMX DDO32102K
BMX EHC 0200
USB
Modbus
RESET
ALARM
OUTPUT
OUT
24V
0.9A
24V
0V
PE
INPUT
100
240
VAC
N
L
Q
00
31
CH0
CH1
AUX