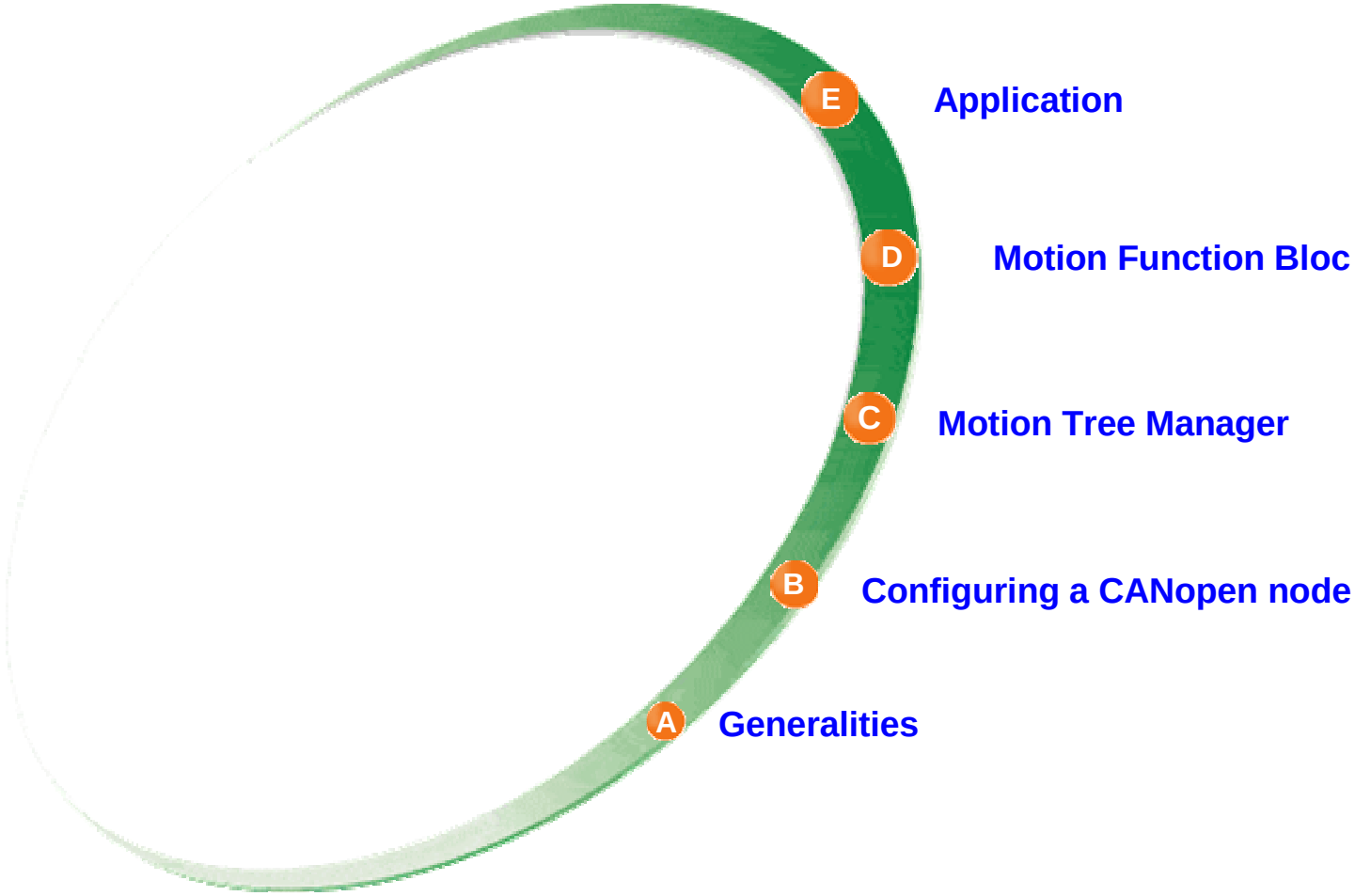
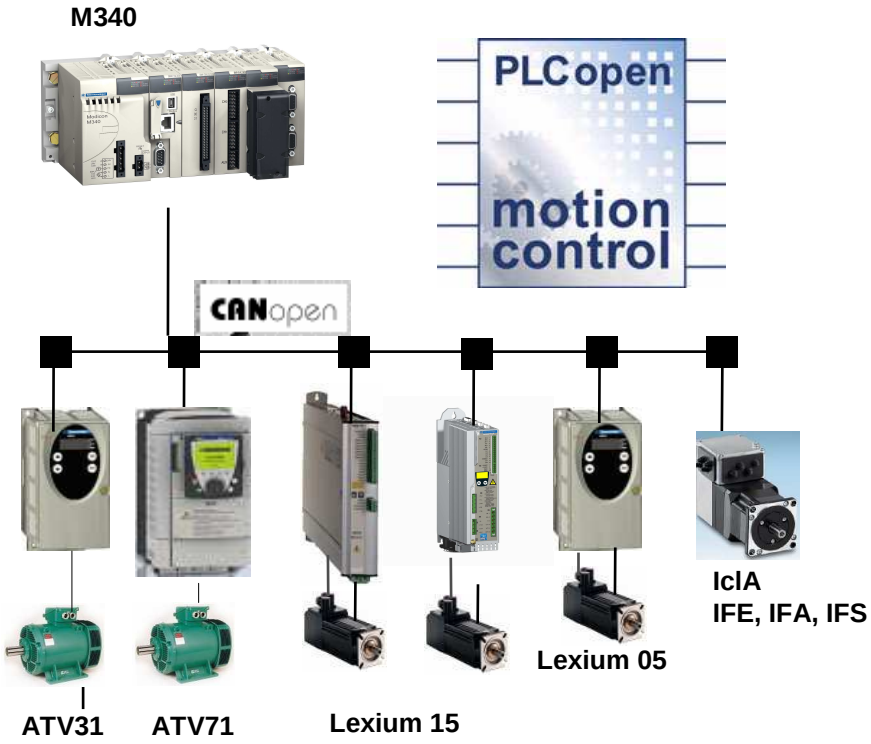
M8 - Modicon M340 : Motion Function Bloc



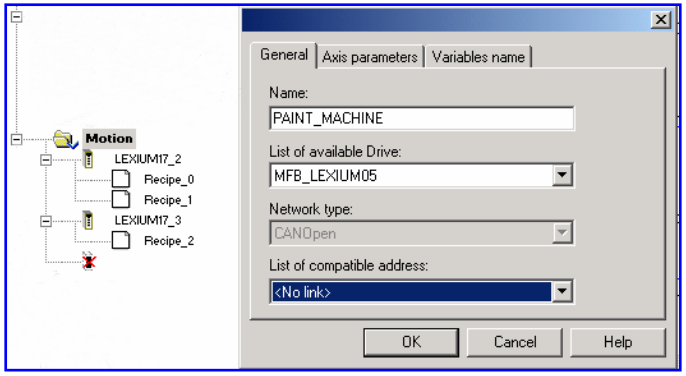
A – Motion control generalities

MFB/MTM Generalities

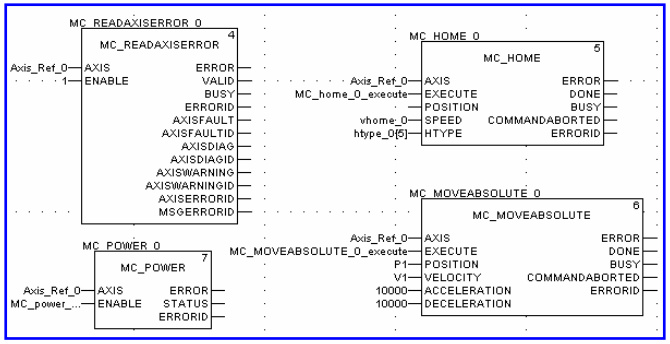
- Unity version 3.0 includes a new functionality allowing simplified access via the CANopen bus to the elementary motion functions of variable speed drives and servodrives. (examples)



– **MOTION TREE MANAGER** function



– **MOTION FUNCTION BLOCK** functions



Minimum versions & tools necessary

- A minimum version is necessary for using MFBs

‘3-phase’ implementation methodology

CANopen bus configuration:

Step 1: Configuring the CANopen communication port on the processor

Step 2: Configuring slaves using Unity configurator

- Configuring PDO exchanges

Step 3: Checking the configuration using the Unity structural view

unity Configurator

Configuration: MTM

Allows to create a logical node

- Define the variables and structures of the drives used by the MFBs.

MTM

Motion Tree Manager

Programming: MFB

Using the MFB library during programming

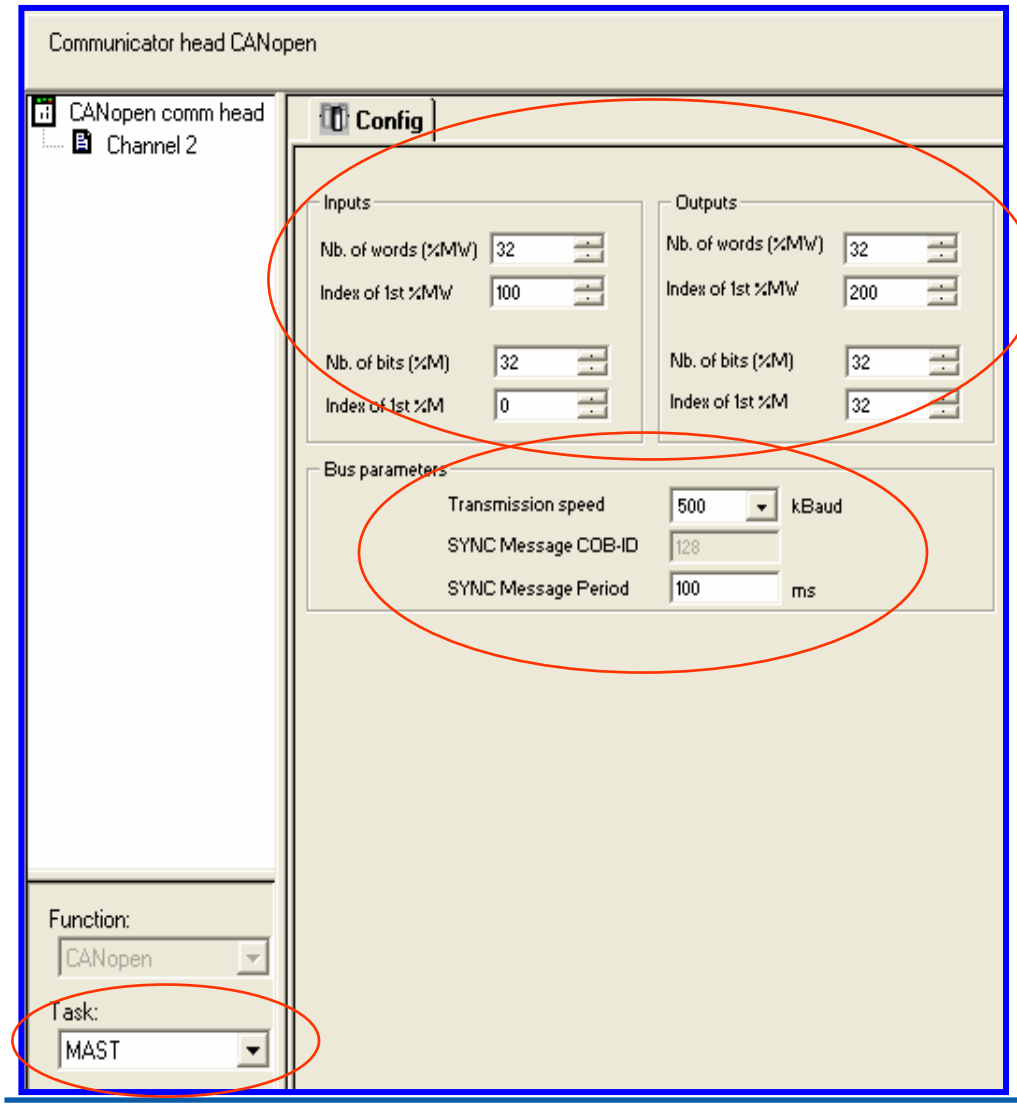
MFB

Motion Function Blocks

B – Configure CANOpen nodes

Configuring the nodes is an essential step in implementing the MOTION FUNCTION BLOCKS.

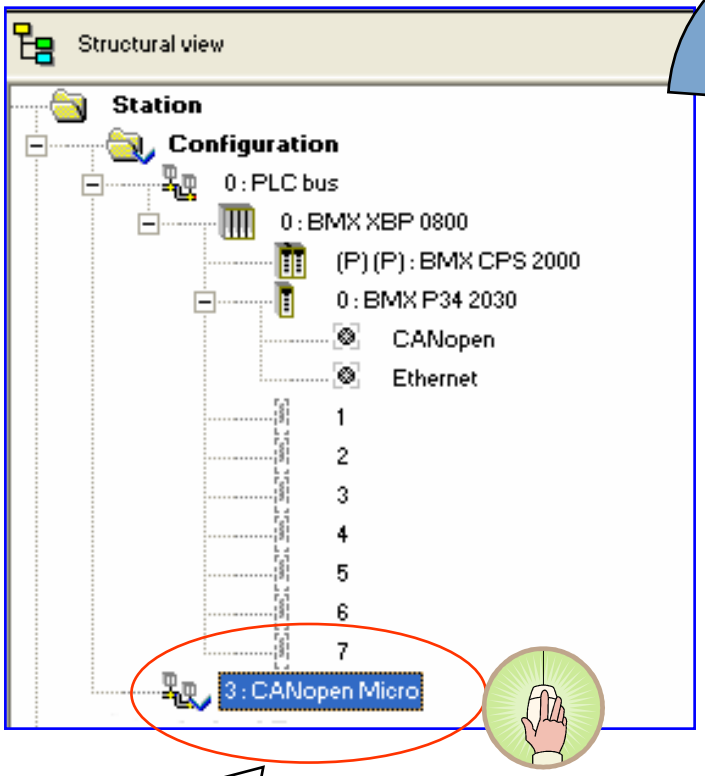
Step 1 : Master configuration

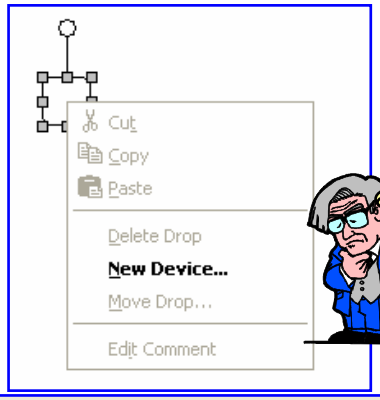


- The PDO's Offset to read and write the PDOs in the internals variables, mainly dedicated for the HMI communication.
- Bus Parameters :
  1. speed according with the length of the bus.
  2. Periodicity for the Sync Message
- Task assigned to the PDOs

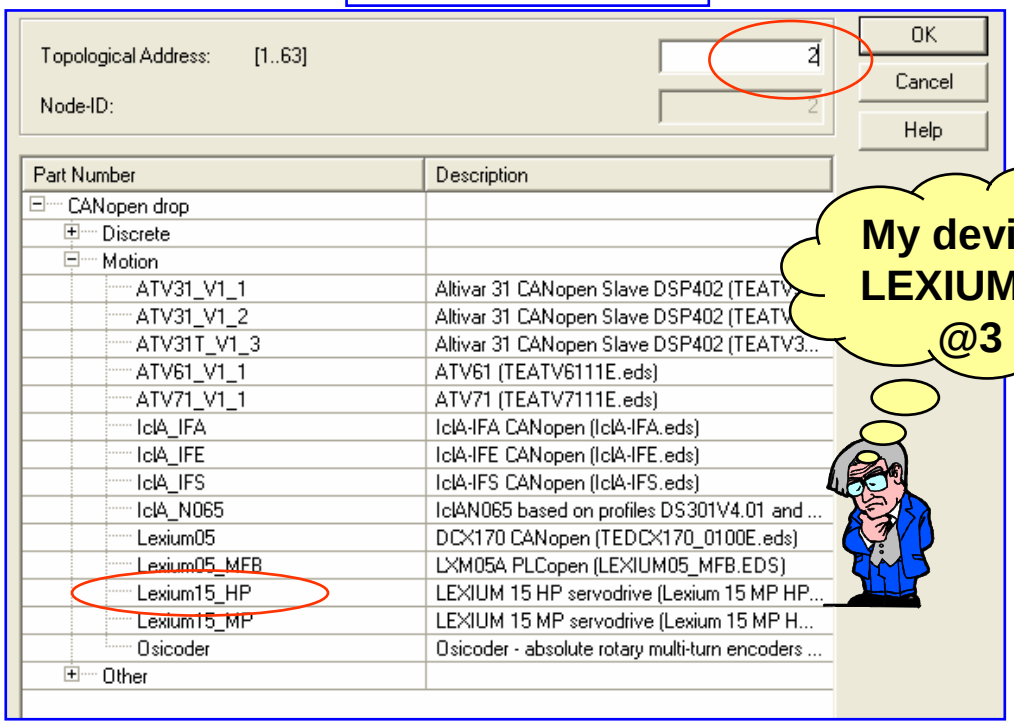
Configure a drive
(except Lexium 05)

Step 2 : Lexium 15 HP Configuration





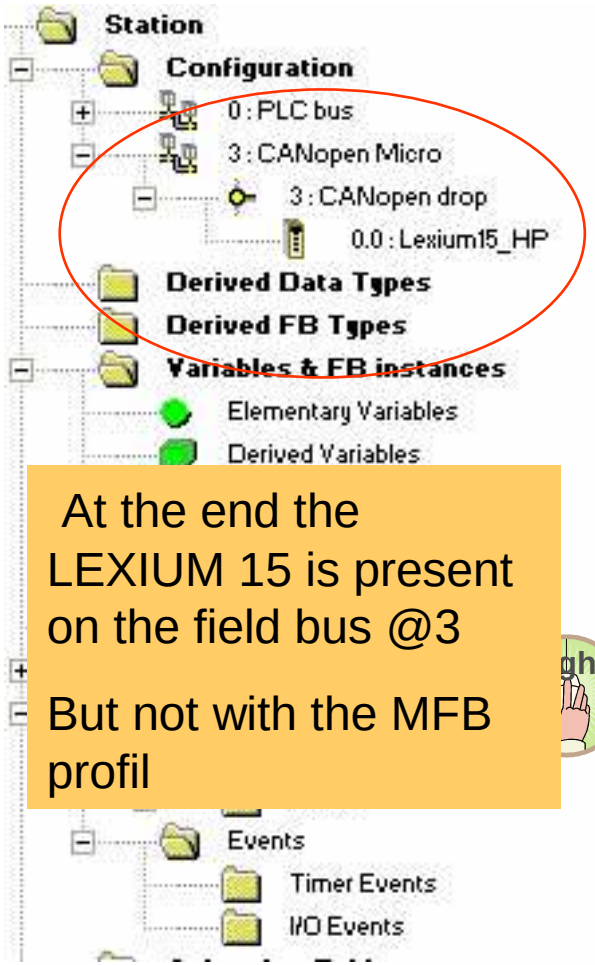
New device: yes



My device: LEXIUM 15 @3

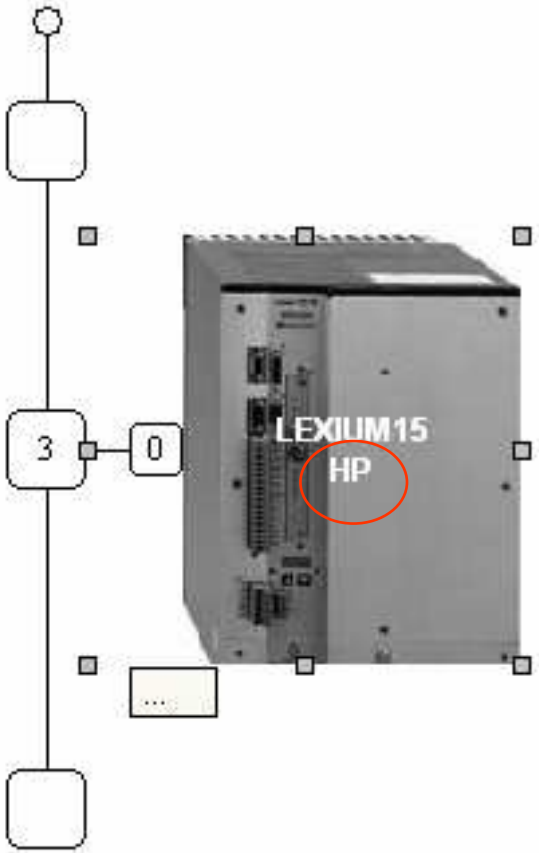
When the master's configuration is done we must to define the Slave present on the Field bus

Step 2 : **lexium 15HP** Configuration



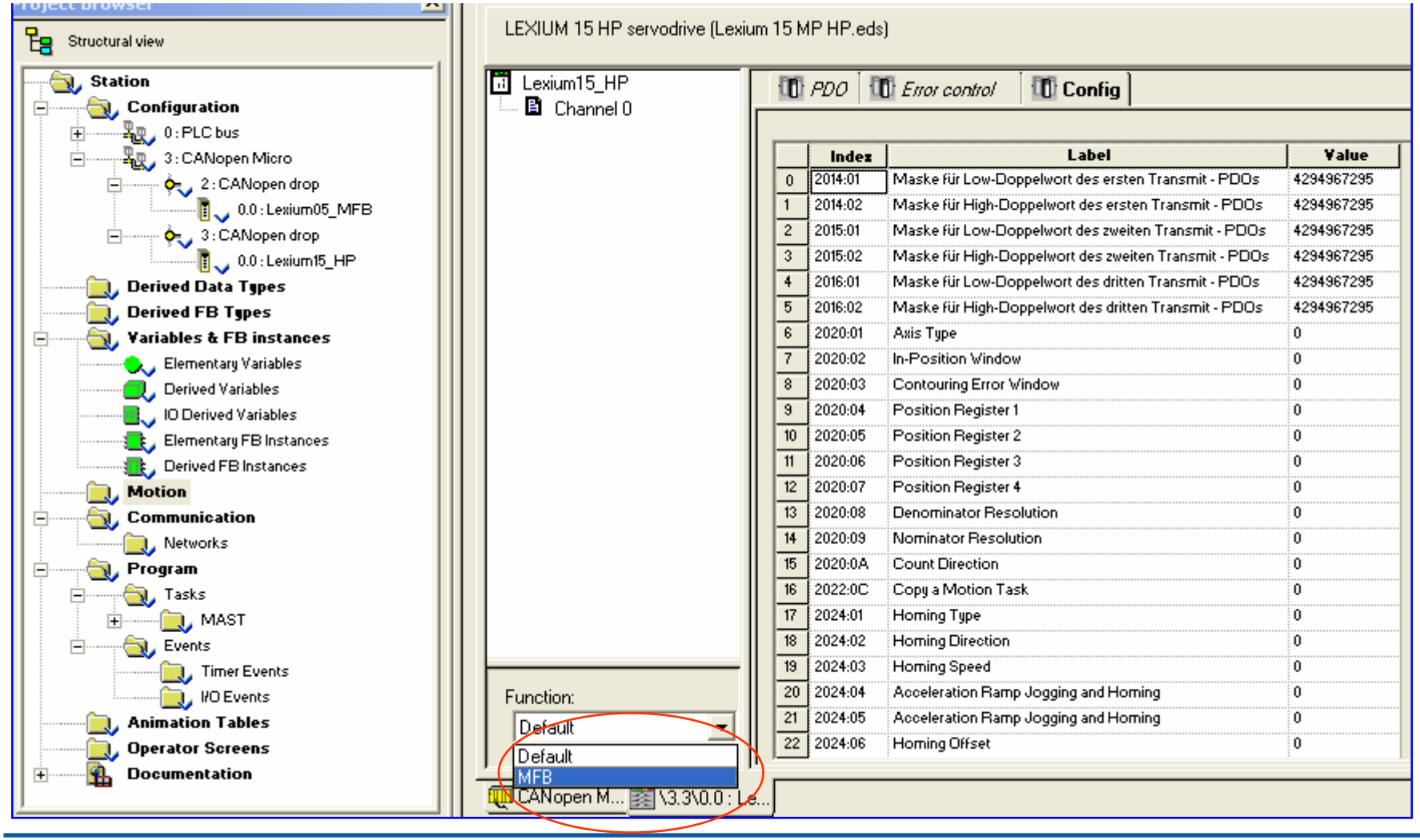
At the end the LEXIUM 15 is present on the field bus @3

But not with the MFB profil



Step 2 : lexium 15HP Configuration

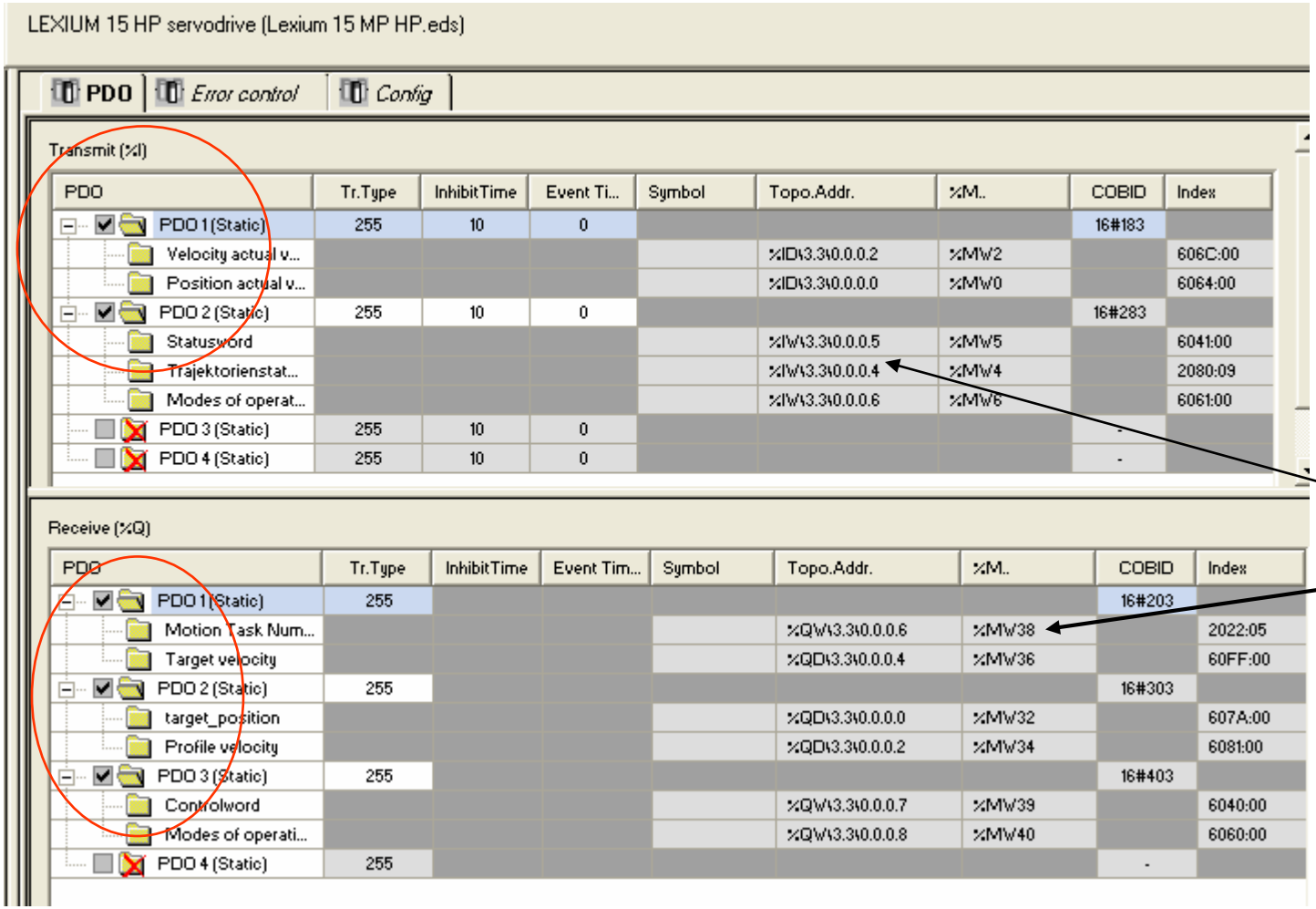
- In this screen we must select MFB in order to select Motion function.



	Index	Label	Value
0	2014:01	Maske für Low-Doppelwort des ersten Transmit - PDOs	4294967295
1	2014:02	Maske für High-Doppelwort des ersten Transmit - PDOs	4294967295
2	2015:01	Maske für Low-Doppelwort des zweiten Transmit - PDOs	4294967295
3	2015:02	Maske für High-Doppelwort des zweiten Transmit - PDOs	4294967295
4	2016:01	Maske für Low-Doppelwort des dritten Transmit - PDOs	4294967295
5	2016:02	Maske für High-Doppelwort des dritten Transmit - PDOs	4294967295
6	2020:01	Axis Type	0
7	2020:02	In-Position Window	0
8	2020:03	Contouring Error Window	0
9	2020:04	Position Register 1	0
10	2020:05	Position Register 2	0
11	2020:06	Position Register 3	0
12	2020:07	Position Register 4	0
13	2020:08	Denominator Resolution	0
14	2020:09	Nominator Resolution	0
15	2020:0A	Count Direction	0
16	2022:0C	Copy a Motion Task	0
17	2024:01	Homing Type	0
18	2024:02	Homing Direction	0
19	2024:03	Homing Speed	0
20	2024:04	Acceleration Ramp Jogging and Homing	0
21	2024:05	Acceleration Ramp Jogging and Homing	0
22	2024:06	Homing Offset	0

Step 2 : lexium 15HP Configuration

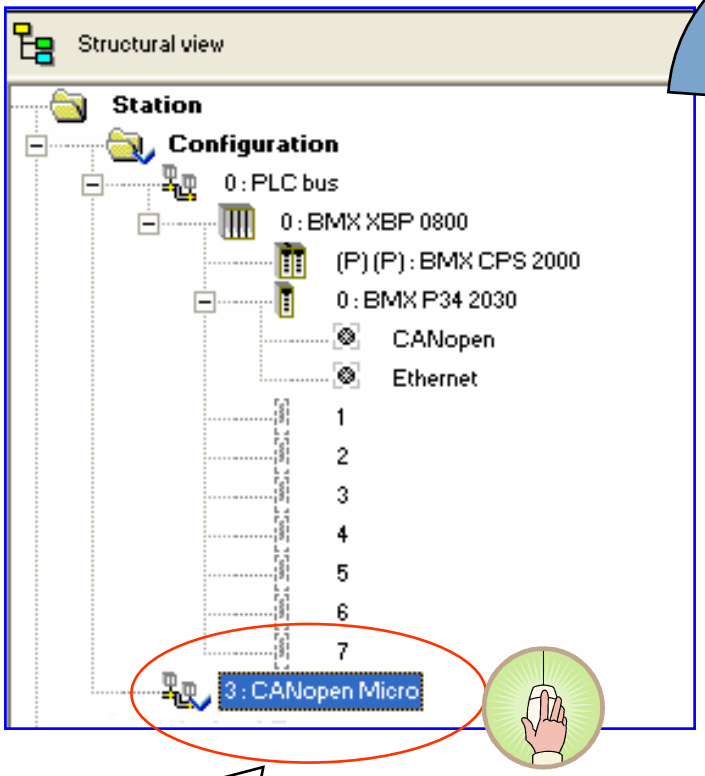
- In this screen it is possible to check the PDO communication.

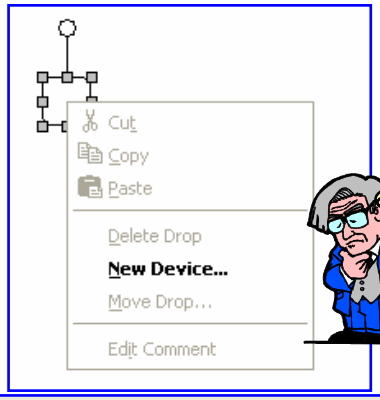


- According with the EDS file somes PDOs are already mapped, and locked
- The PDOs used Topologics addresses %ID/QD and internals variables %MW….
- **when we use a MFB profil Nothing is ajustable excepted the INHIBIT TIME**

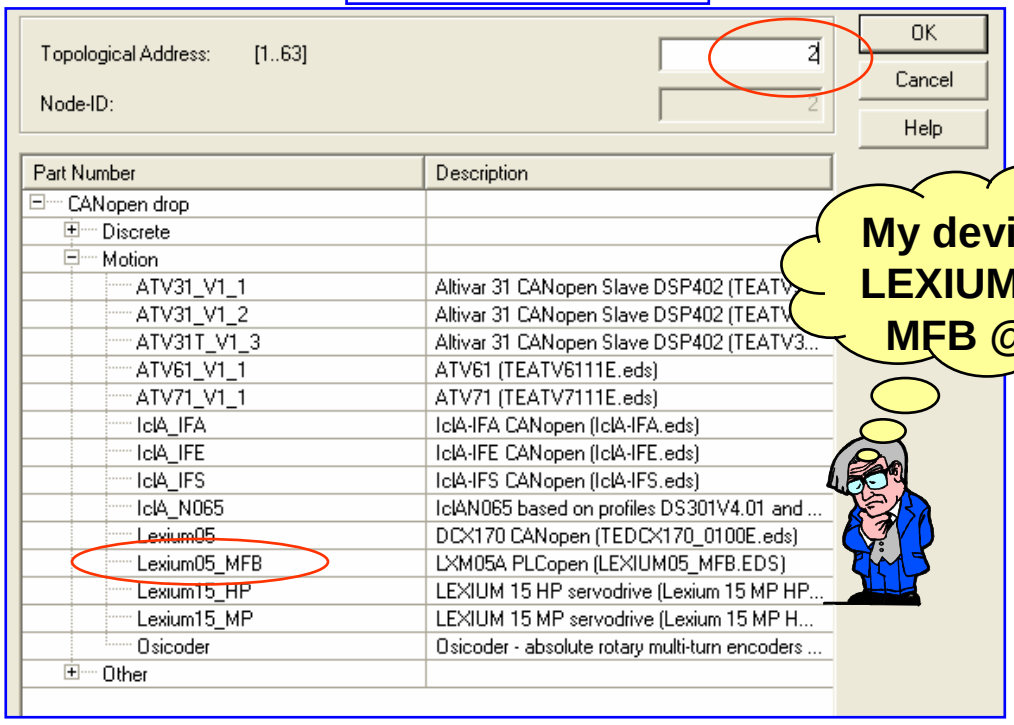
Lexium 05 specific configuration

Step 2 : lexium 05 Configuration





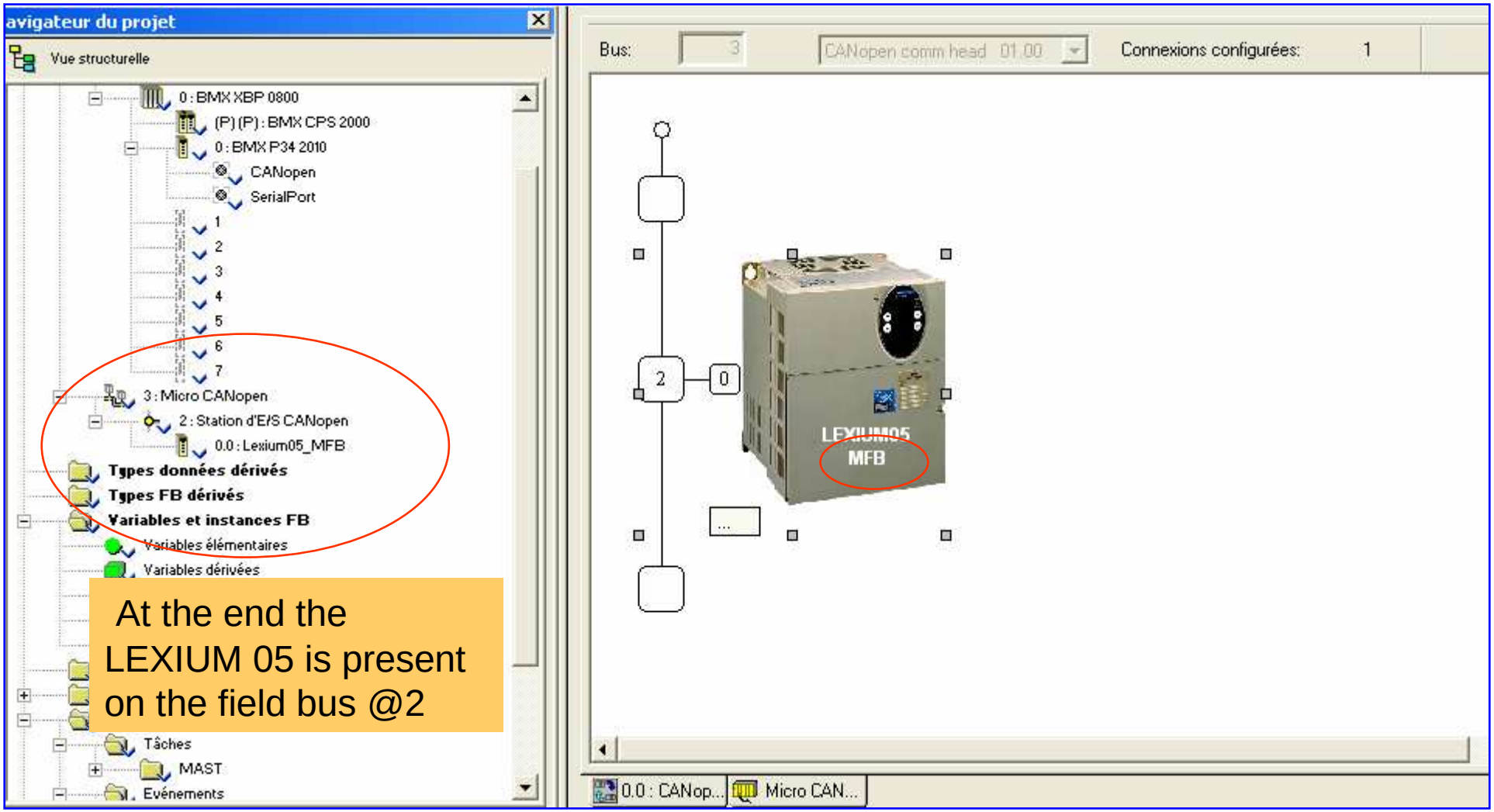
New device: yes



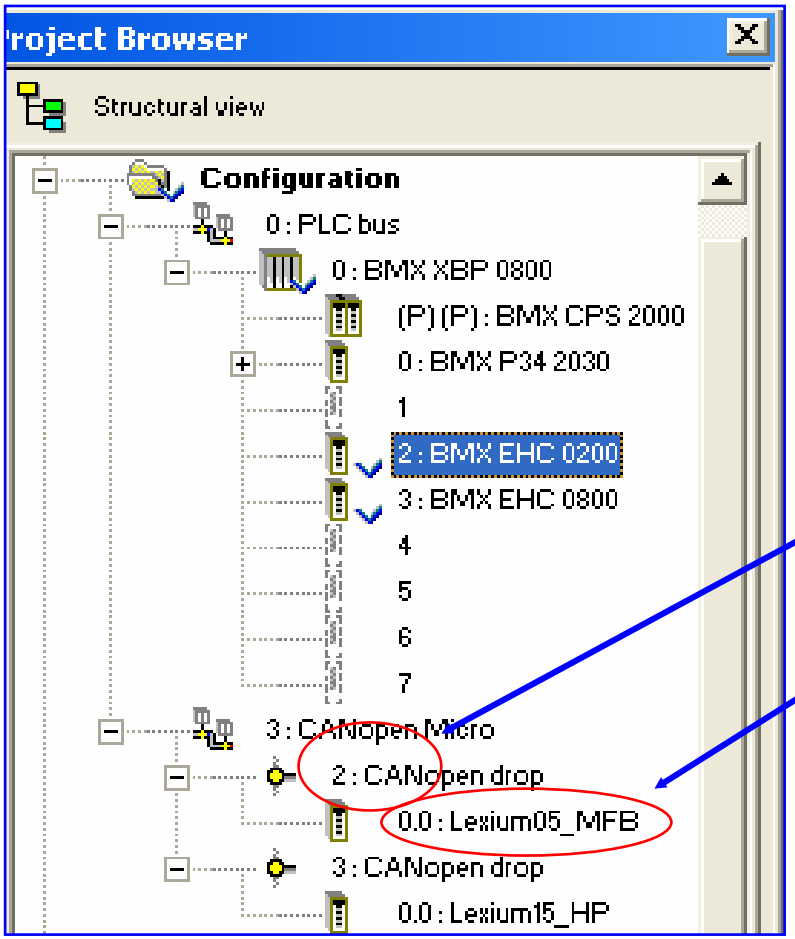
My device: LEXIUM 05 MFB @2

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

Step 2 : lexium 05 Configuration



Step 3: Checking the configuration using the Unity structural view



- The structural view can be used to see quickly which nodes are present on the CANopen bus
  - Node address (@2)
  - Drive type used

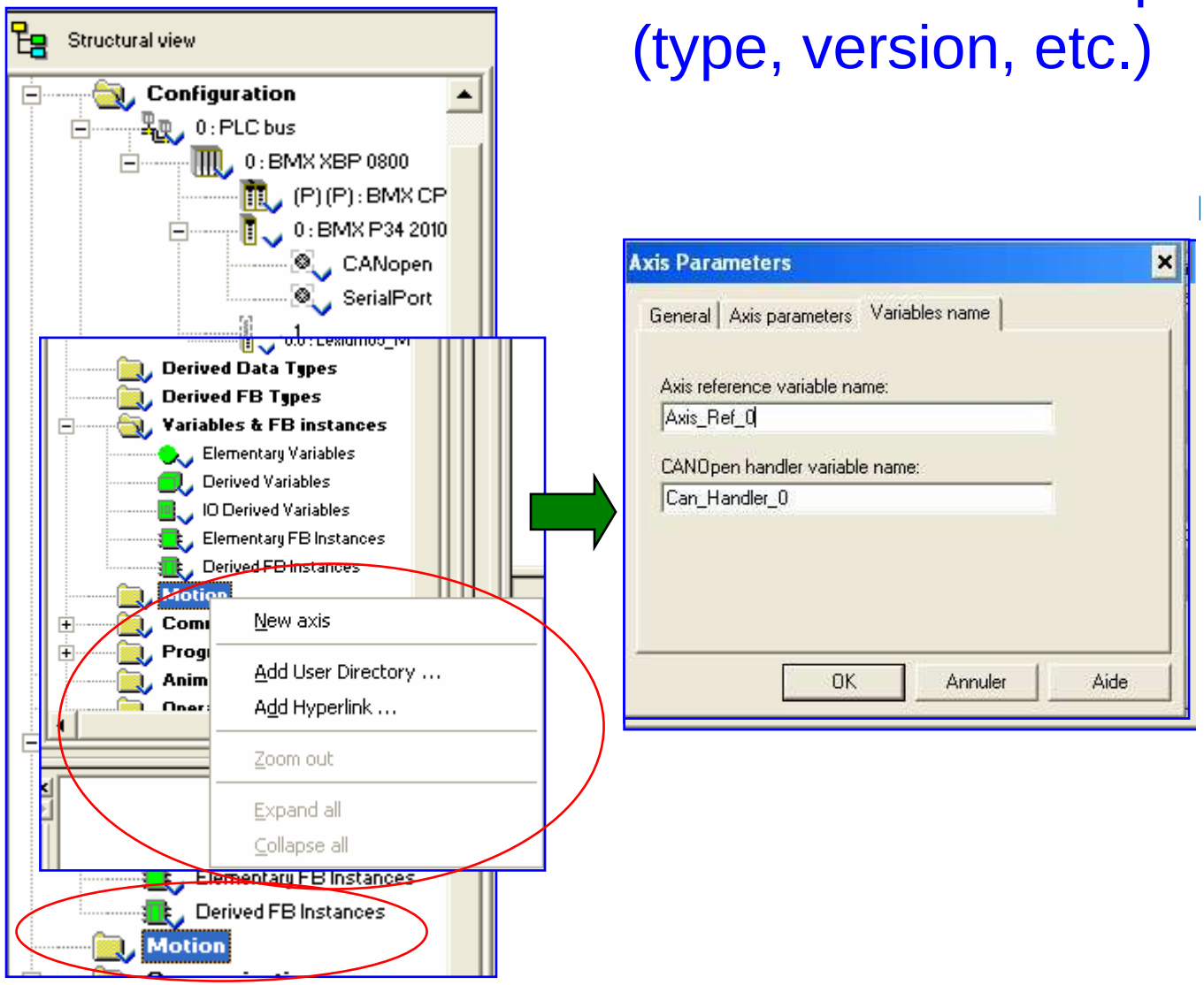
C – Motion tree manager

MOTION TREE MANAGER are used to:

Define the parameters of drives used on the CANopen bus (type, version, etc.)

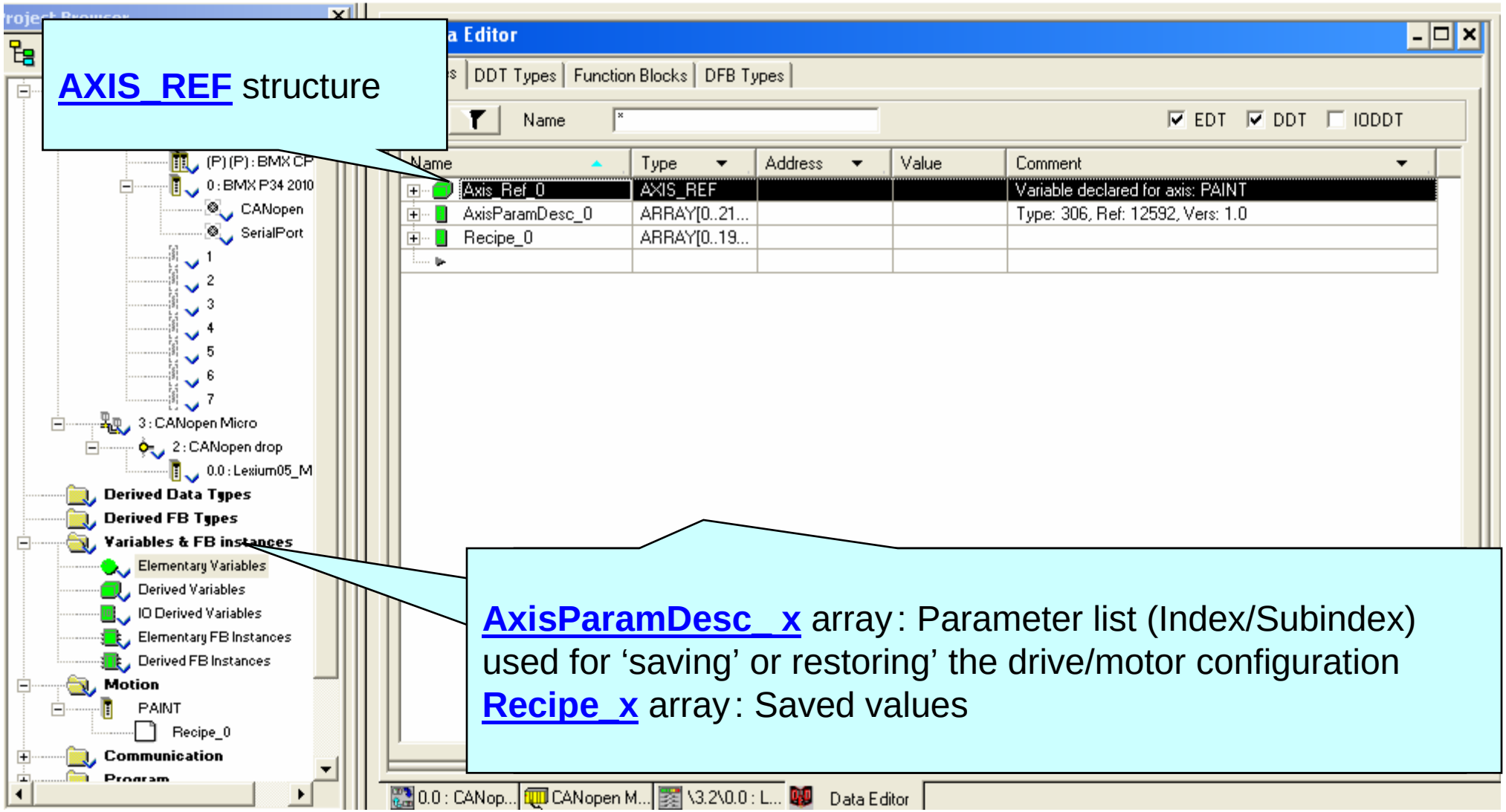
Configure the variables and structures used by the MFBs

Step 1: Defining the parameters of the drives used on the CANopen bus (type, version, etc.)



- **Variables name** view
  - Name the AXIS_Reference structure that will be used in the MFBs
  - Name the control structure variable used by the HANDLER block

Step 1: Defining the parameters of the drives used on the CANopen bus (type, version, etc.)



Example : LEXIUM05 Configuration

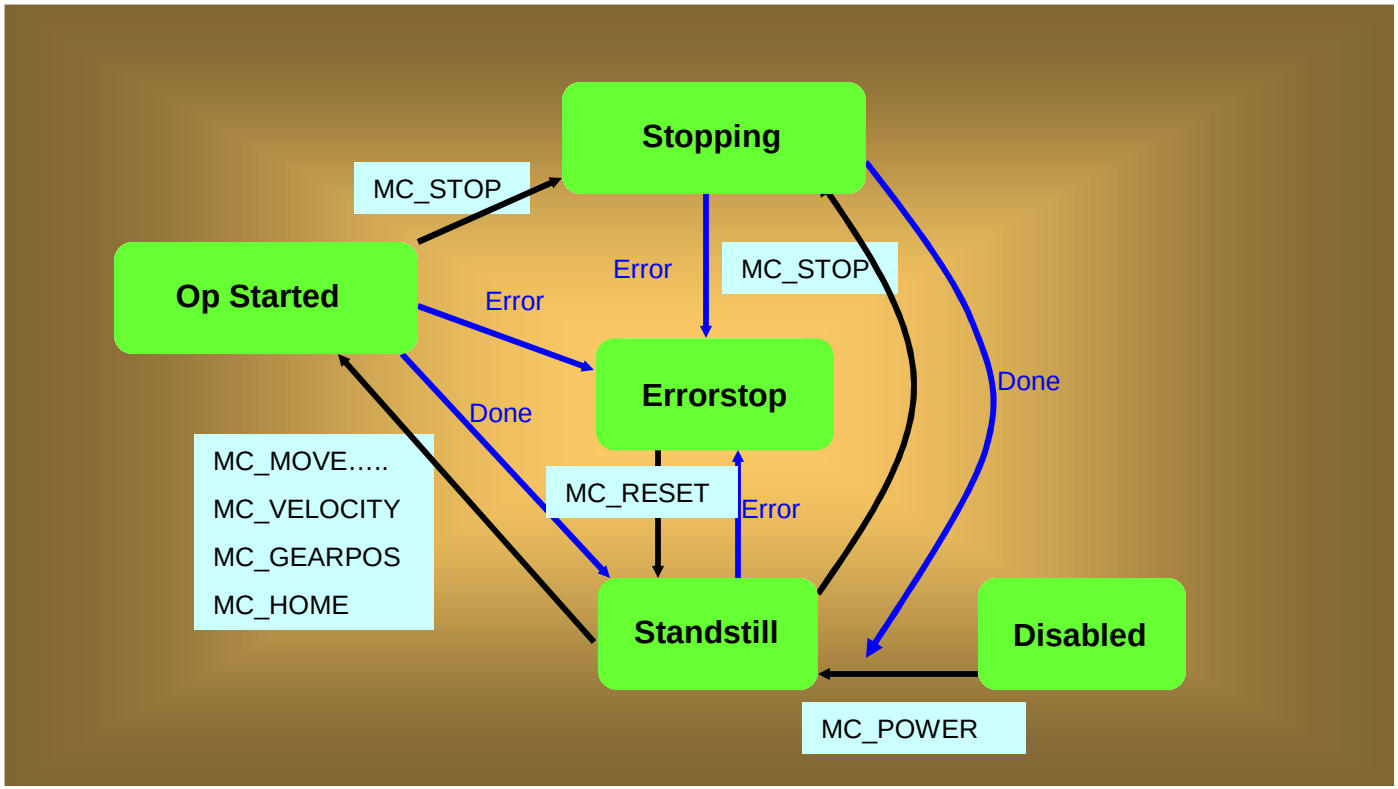
D – Motion Function Blocks

Actions are programmed through dedicated Blocks called ‘MFBs’

Motion Function Blocks provide simplified access to the main drive functions.

state diagram

- Each drive complies with its own DRIVECOM profile but the MFBs comply with the PLCopen standard (in accordance with PLCopen v1.1 specification).
  - There are 5 main states:

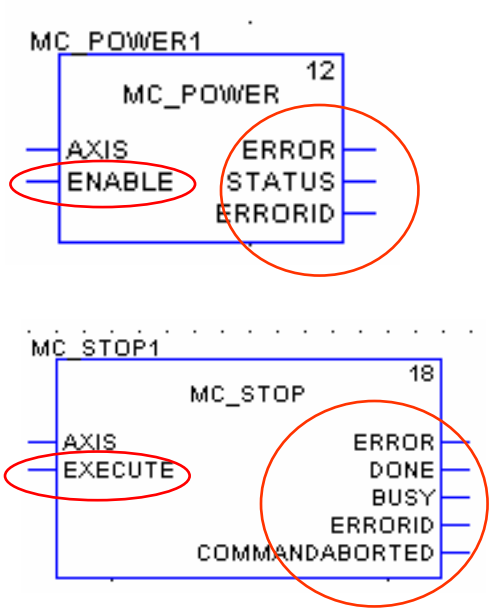


Description of the basic I/O parameters

- Each Motion Function Block has so-called basic input and output parameters.
  - Input parameters:
    - **Axis** AXIS_REF
    - **Enable** Enable on State

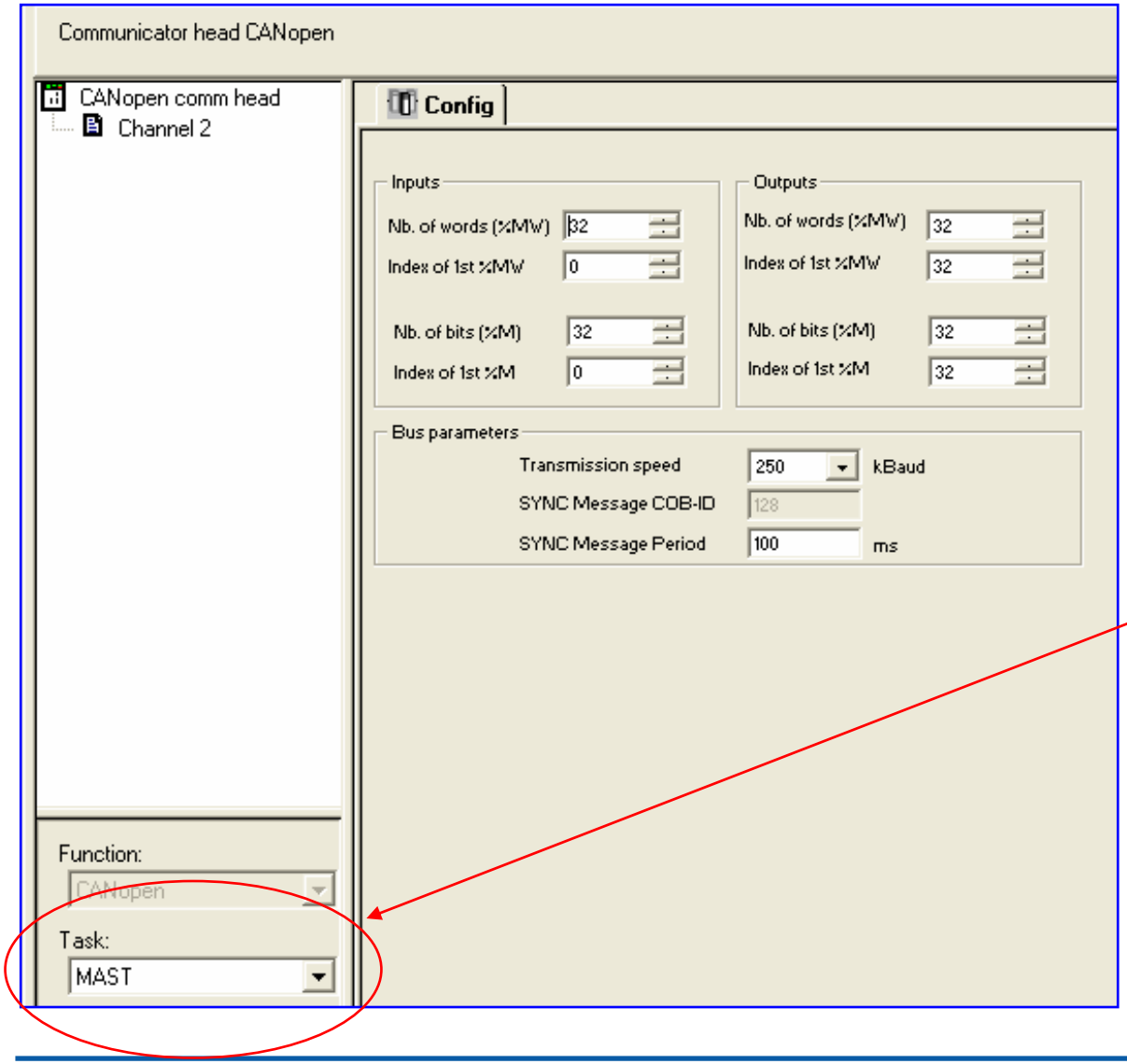
    or

    - **Execute** Enable on Edge



  - Output parameters:
    - **Error** Execution error detected (Error = 1)
    - **Done** Signals the end of the action
    - **Busy** Reflects the action in progress (Busy = 1)
    - **ErrorId** Error identifier
    - **CommandAborted** Command interrupted by execution of another Motion Block

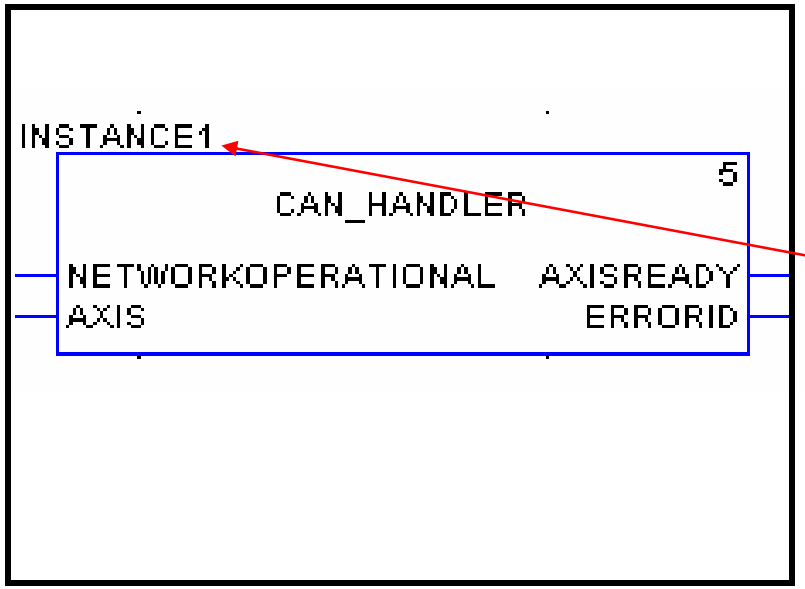
Generalities on the MFB

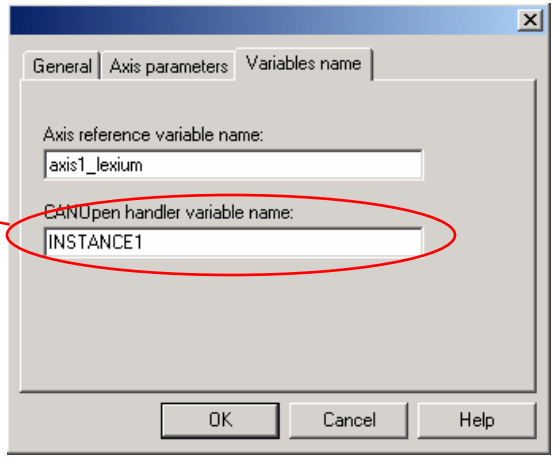


- The MFB function must be use in the task dedicated for the CANopen communication.
  - Example : Here the MAST

TE_CAN_Handler

- **Purpose**: To check the CANopen communication and the consistency between the software and physical configurations
  - The TE_CAN_Handler function is **MANDATORY** for ensuring operation of the MFBs. It must be scanned each mast or fast CPU scan cycle.

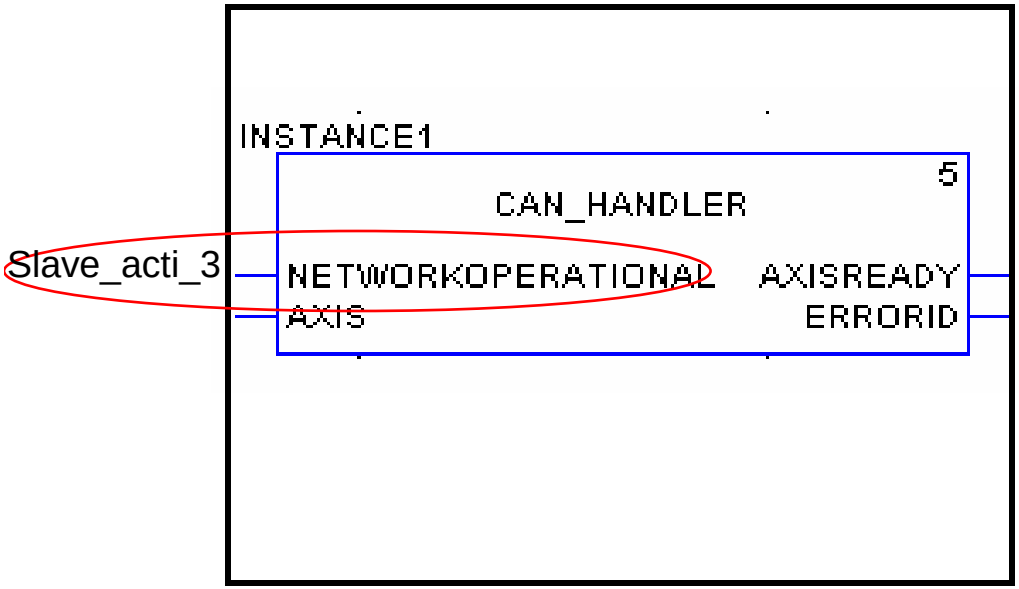




It is **MADATORY** for this block to be instantiated with the control structure variable.

TE_CAN_Handler

- **Purpose**: To check the CANopen communication and the consistency between the software and physical configurations.



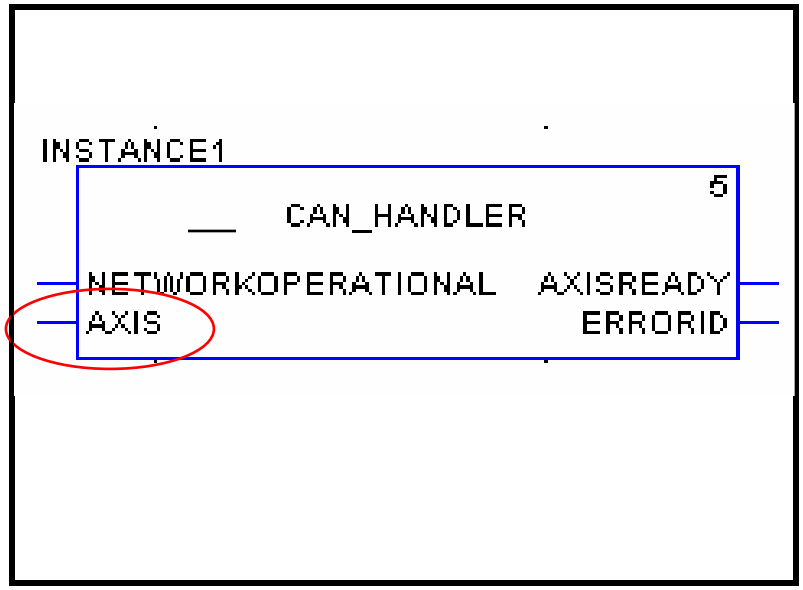
- Input parameters
  - NetworkOperational
    - System information that the slave is working correctly on the CANopen bus
  - Slave_acti_x => %IW0.m.1.8.1

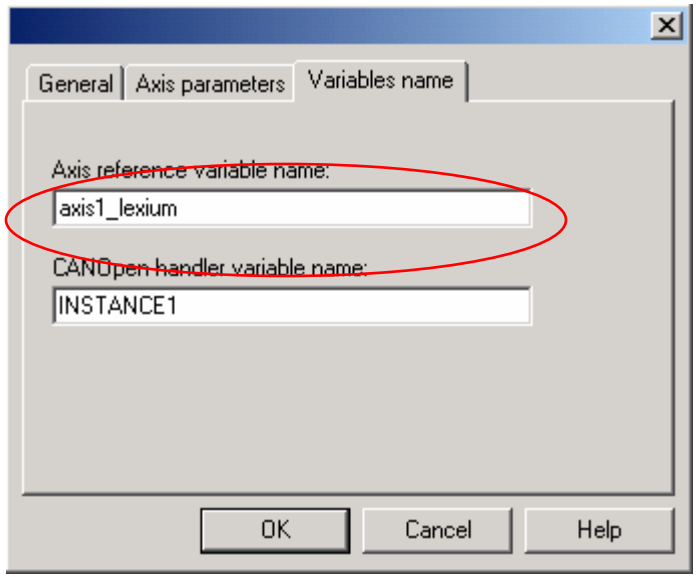
SLAVE_ACTIV_1	BOOL			Slave active on the bus: device 1
SLAVE_ACTIV_2	BOOL			Slave active on the bus: device 2
SLAVE_ACTIV_3	BOOL			Slave active on the bus: device 3
SLAVE_ACTIV_4	BOOL			Slave active on the bus: device 4
SLAVE_ACTIV_5	BOOL			Slave active on the bus: device 5

TE_CAN_Handler

- **Purpose**: To check the CANopen communication and the consistency between the software and physical configurations.

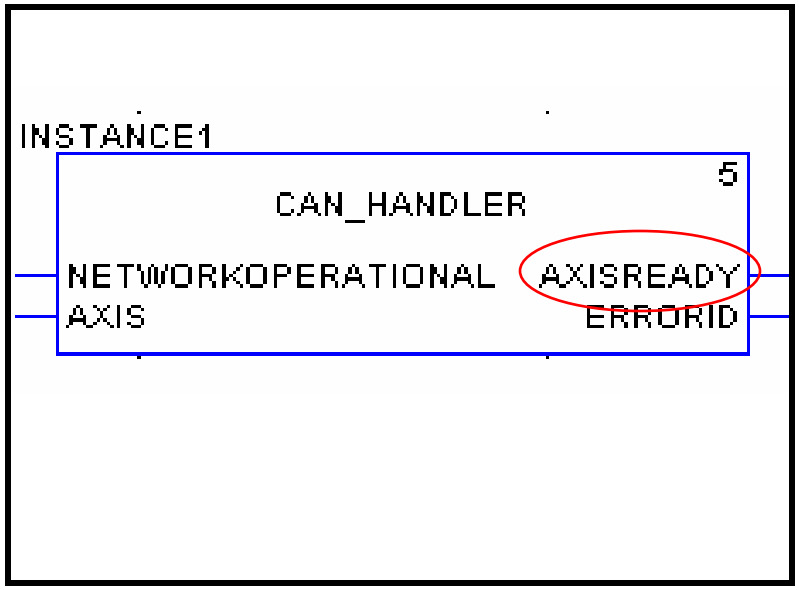
- Input parameters





TE_CAN_Handler

- **Purpose**: To check the CANopen communication and the consistency between the software and physical configurations.



- Output parameter
  - AXISREADY = 1
    - There is no fault on the CANopen bus
    - The software configuration is consistent with the hardware configuration present

Motion Blocks and associated targets

	ATV31	ATV71	Lexium 05	Lexium 15	IclA
MC_ReadParameter	☒	☒	☒	☒	☒
MC_WriteParameter	☒	☒	☒	☒	☒
MC_ReadActualPosition			☒	☒	☒
MC_ReadActualVelocity	☒	☒	☒	☒	☒
MC_Reset	☒	☒	☒	☒	☒
MC_Stop	☒	☒	☒	☒	☒
MC_Power	☒	☒	☒	☒	☒
MC_MoveAbsolute			☒	☒	☒
MC_MoveRelative			☒	☒	
MC_MoveAdditive			☒		☒
MC_Home			☒	☒	☒
MC_MoveVelocity	☒	☒	☒	☒	☒
MC_ReadStatus	☒	☒	☒	☒	☒
MC_ReadAxisError	☒	☒	☒	☒	☒
TE_UploadDriveParam	☒	☒	☒	☒	☒
TE_DownloadDriveParam	☒	☒	☒	☒	☒
Lxm_GearPos				☒	
Lxm_UploadMTask				☒	
Lxm_DownloadMTask				☒	
Lxm_StartMTask				☒	

Unity Pro

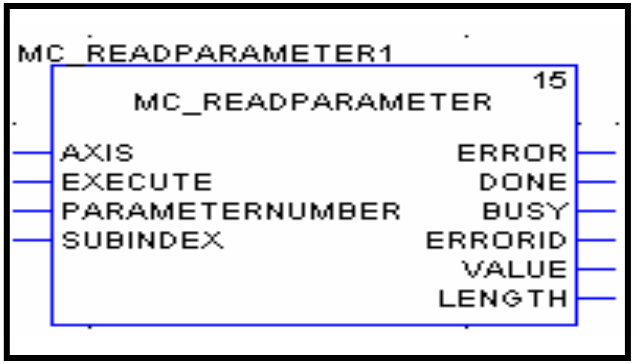
PLCopen motion control

Easy faulty device replacement

Lexium advanced functions

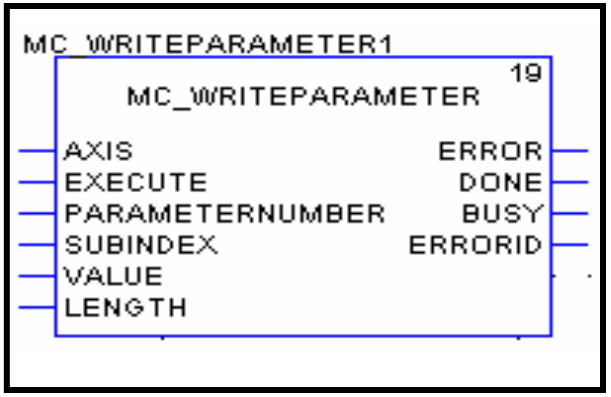
1- Motion Function Blocks

MC_ReadParameter



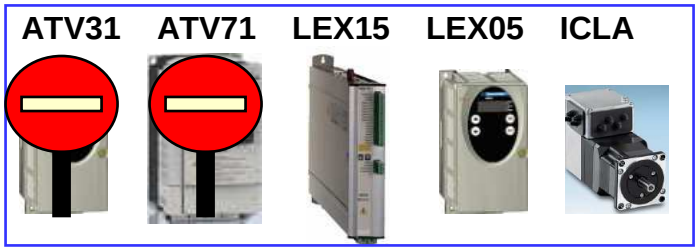
- **Purpose**:
  - Reading a variable by SDO messaging

MC_WriteParameter

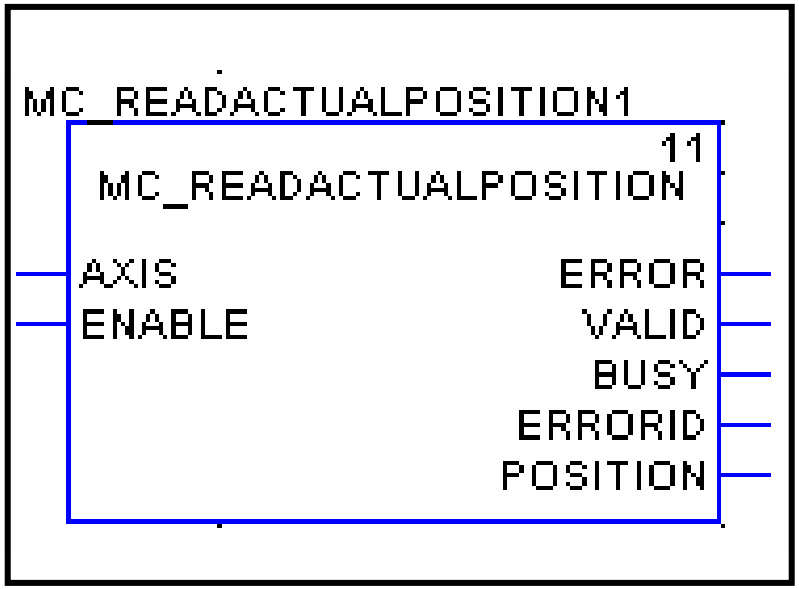


- **Purpose**:
  - Writing a variable by SDO messaging

1- Motion Function Blocks



MC_ReadActualPosition

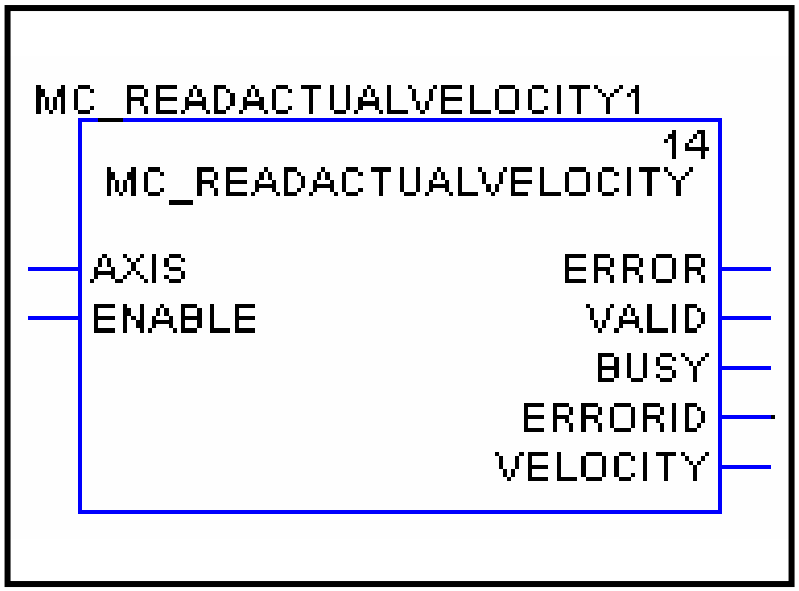


- **Purpose**:
  - Reading the actual position

- The unit is linked to the drive used

1- Motion Function Blocks

MC_ReadActualVelocity

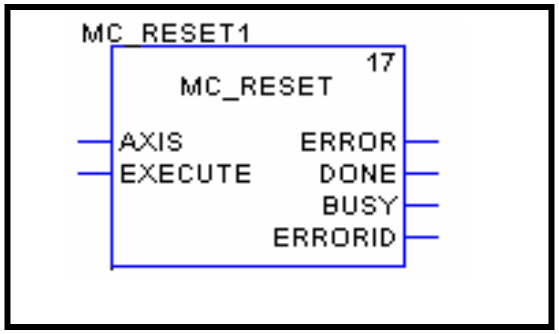


- **Purpose**:
  - Reading the actual velocity

- The unit is linked to the drive used

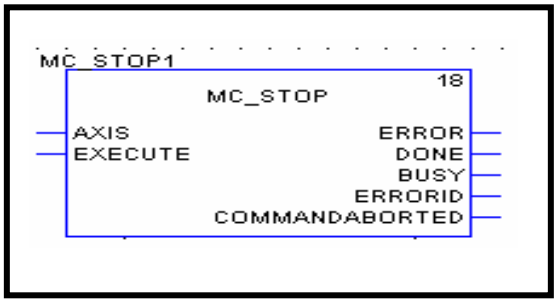
1- Motion Function Blocks

MC_Reset



- **Purpose**:
  - Acknowledging internal errors and internal warnings

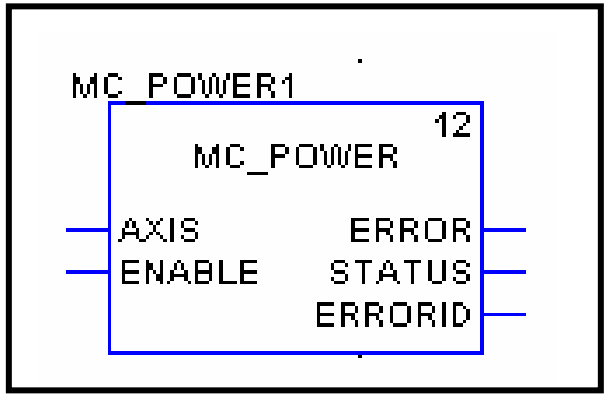
MC_Stop



- **Purpose**:
  - Used to stop any movement in progress and sets the drive to Stopping status

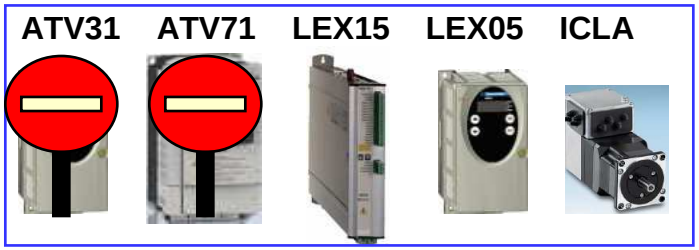
1- Motion Function Blocks

MC_Power

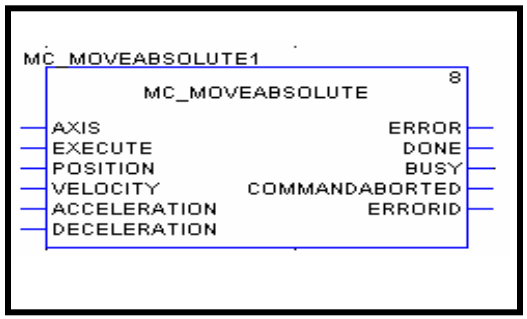


- **Purpose**:
  - Setting the drive to ‘Standstill’ status

1- Motion Function Blocks



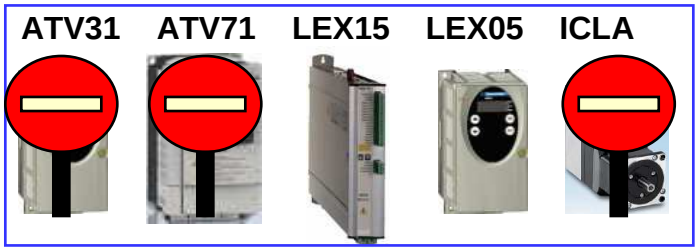
MC_MoveAbsolute



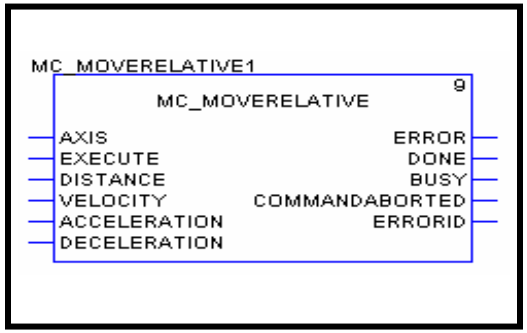
- **Purpose**:
  - Used to execute a command to move to an absolute position

- The unit is linked to the drive used

1- Motion Function Blocks



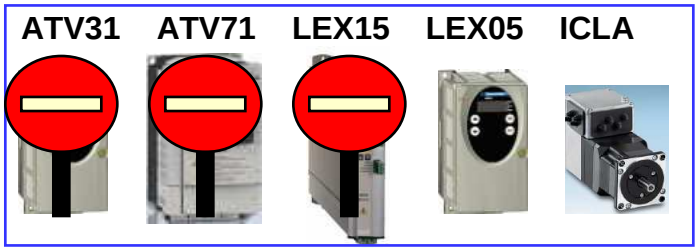
MC_MoveRelative



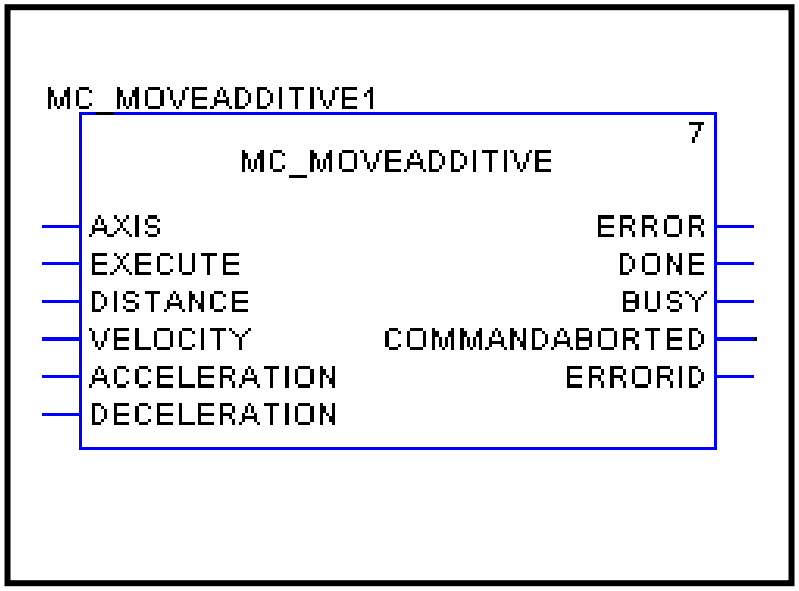
- **Purpose**:
  - Used to execute a command to move to a relative position with respect to the current position of the drive

- The unit is linked to the drive used

1- Motion Function Blocks



MC_MoveAdditive

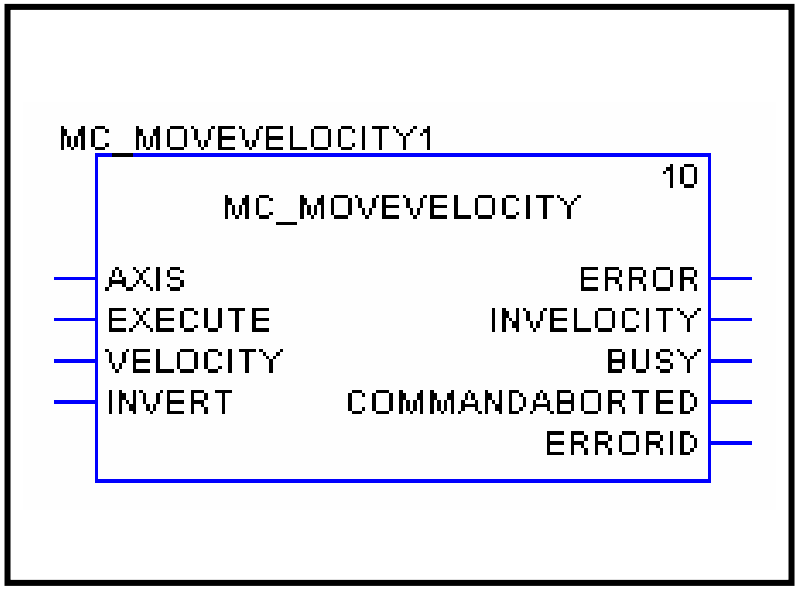


- **Purpose**:
  - Used to apply an additional movement command to the drive in motion

- The unit is linked to the drive used

1- Motion Function Blocks

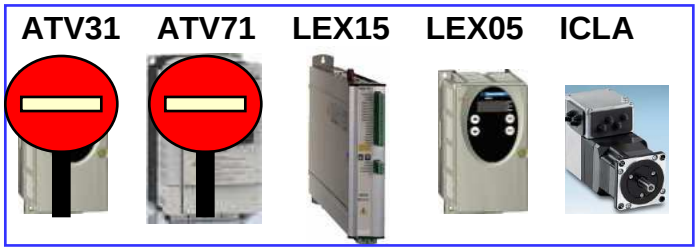
MC_MoveVelocity



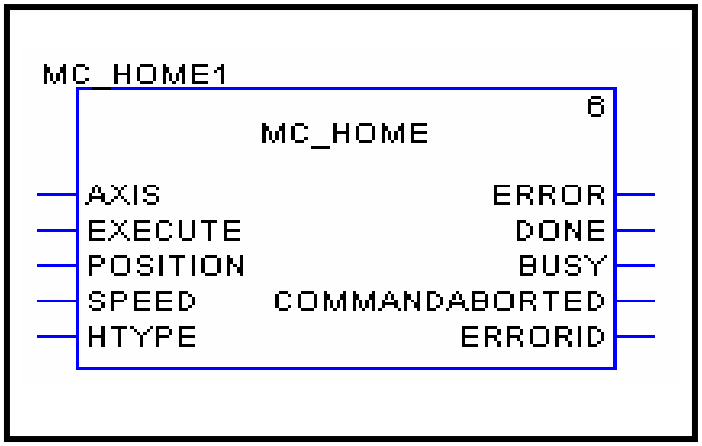
- **Purpose**:
  - Used to execute a velocity movement command

- The unit is linked to the drive used

1- Motion Function Blocks



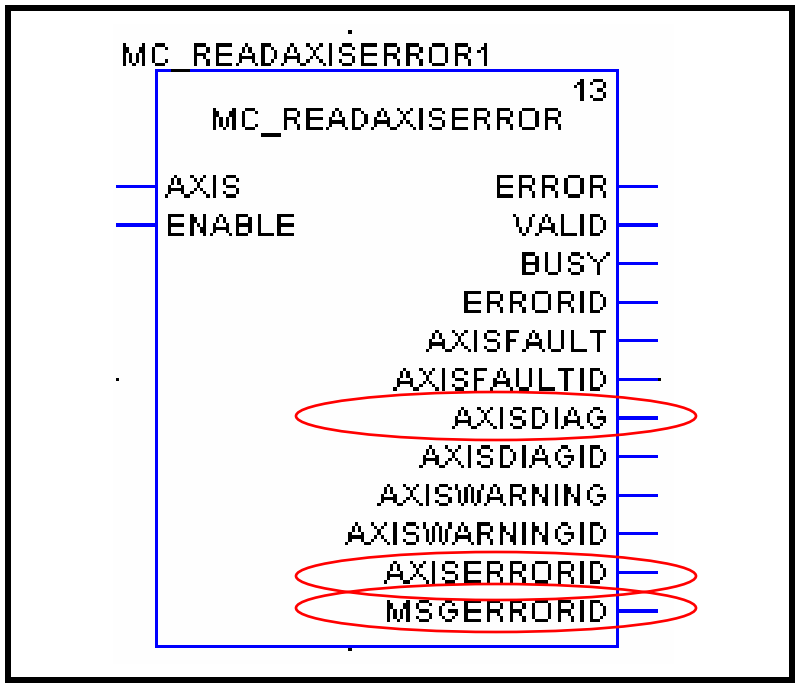
MC_Home



- **Purpose**:
  - Used to execute a Homing command

1- Motion Function Blocks

MC_ReadAxisError



- **Purpose**: To retrieve errors from the system
  - Output parameters
    - AxisDiag: TSX CPP110 card diagnostics
    - AxisErrorID: MFB error No. storage
    - MSGErrorID: Messaging system error storage

1- Motion Function Blocks

MC_ReadAxisError

- **Purpose**: To retrieve errors from the

	ATV 31	ATV71	LXM 17	LXM 05	ICLA
Axisfault id	1ft 2029:16	1ft 2029:16	Errcode 2029:16	Signlatched 301C:8	Faulsig_sr 301C:12
Axiswarn id		LRS6 2002:38	1002:0	Warnlatched 301C:0C	Warnsig 301C:A
Axisdiag id	Errd 603F:0	Errd 603F:0	1003:0	Stopfault 603F:0	Stopfault 3020:7

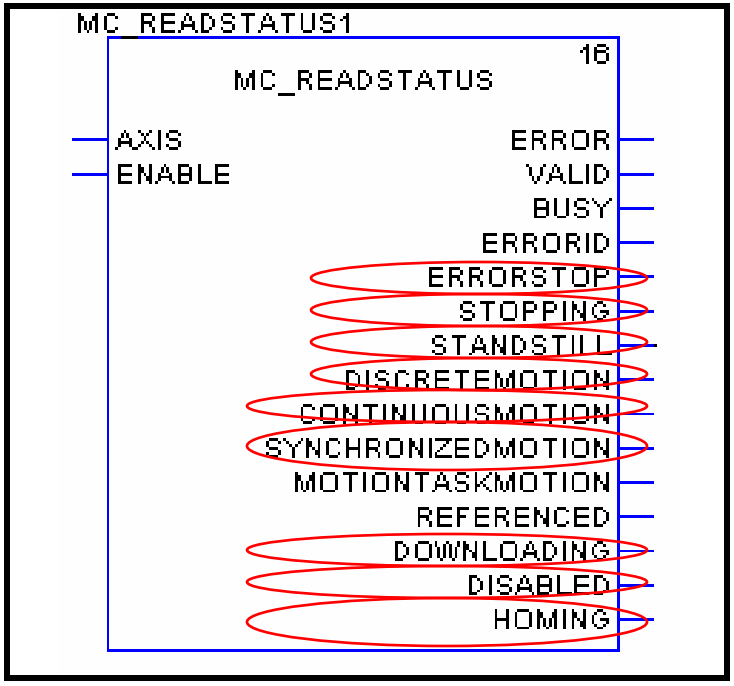
ameters
ve fault
Drive fault value

- The error values are specific to the drives
  - Here is a correspondence table

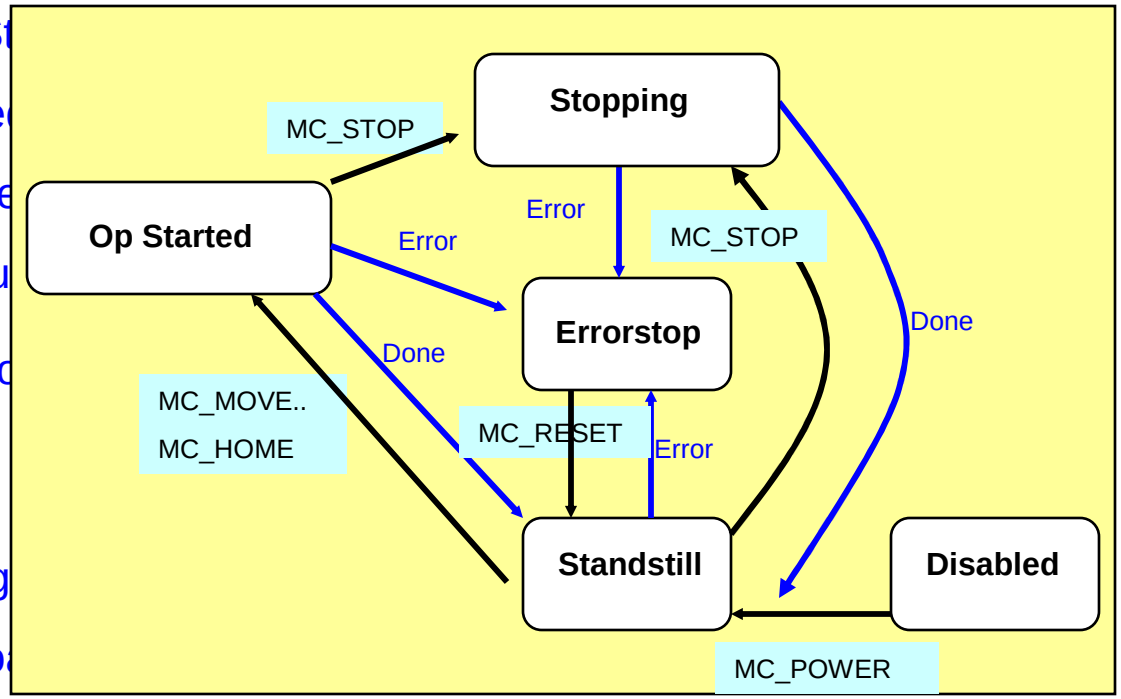
1- Motion Function Blocks

MC_ReadStatus

- **Purpose**: To ascertain the drive status

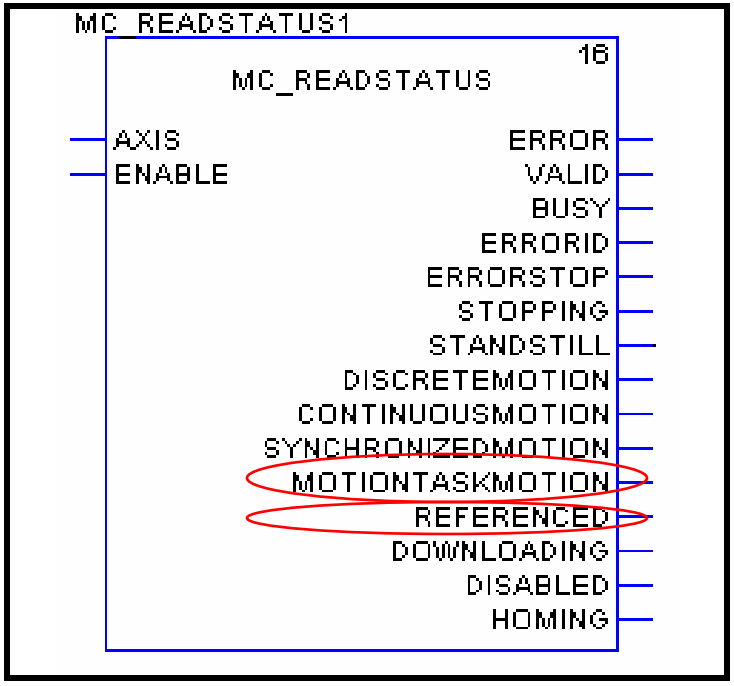


- Output parameters
  - Errorstop: Stop with error
  - Stopping: Drive stopping
  - StandSt
  - Disable
  - Discrete
  - Continu
  - Synchro
  - Homing
  - Downlo



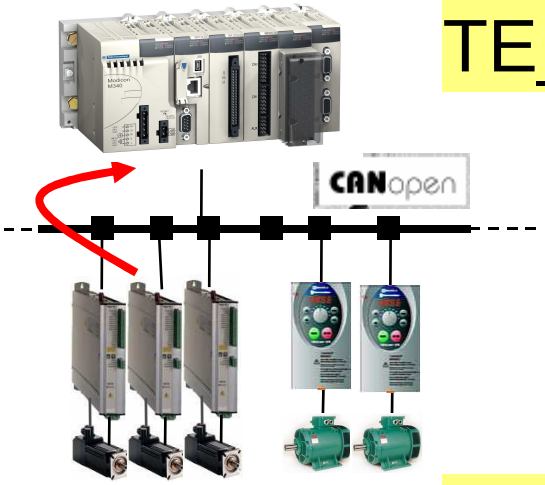
1- Motion Function Blocks

MC_ReadStatus

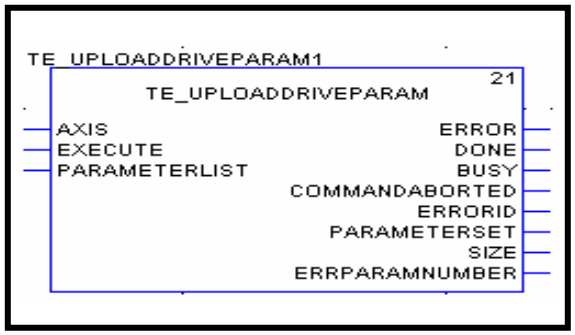


- **Purpose**: To ascertain the drive status
- Output parameters
  - MotionTaskMotion: Current step number
  - Referenced: State 1 => Drive referenced

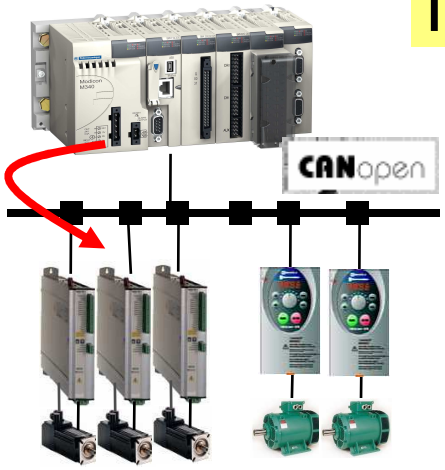
1- Motion Function Blocks



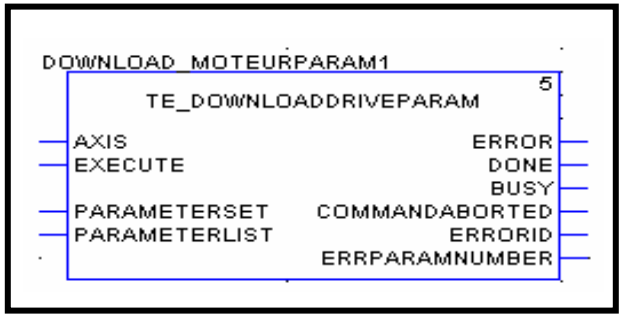
TE_UploadDriveParam



- **Purpose**:
  - Used to read drive motor parameters and store them in a PLC memory zone

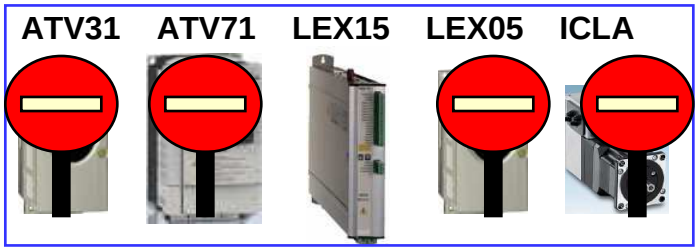


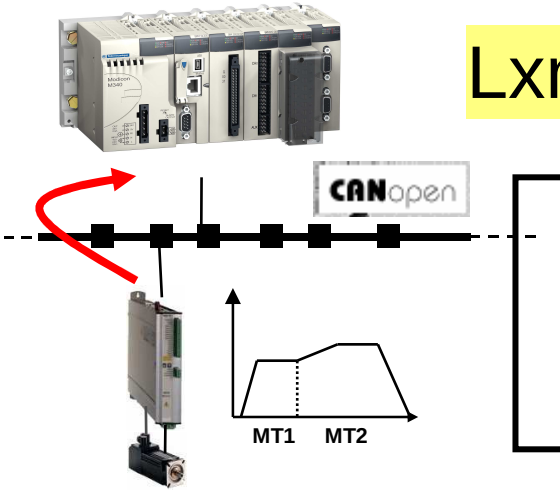
TE_DownloadDriveParam



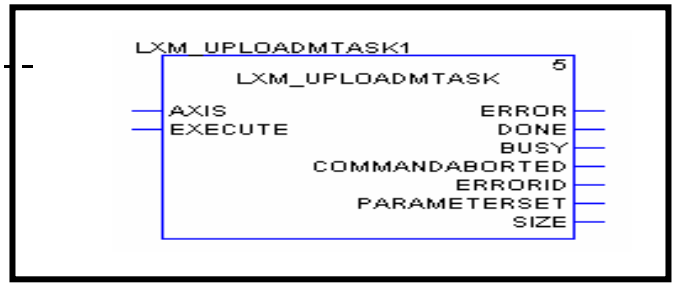
- **Purpose**:
  - Used to write drive motor parameters previously saved in a PLC memory zone

1- Motion Function Blocks

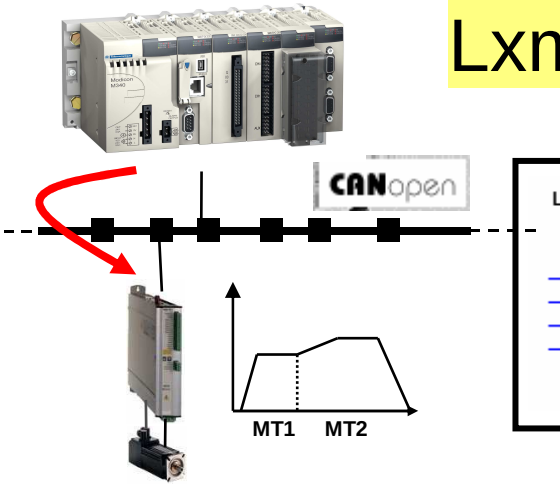




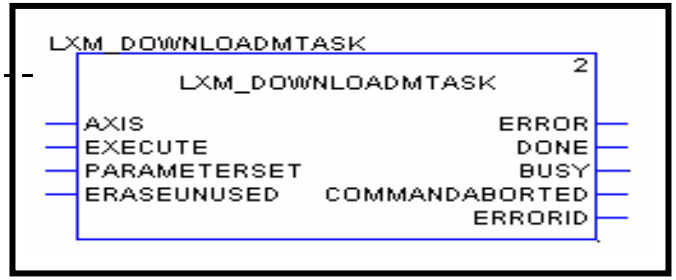
Lxm_UploadMTask



- **Purpose**:
  - Used to read Motion Tasks and save them in a PLC memory zone

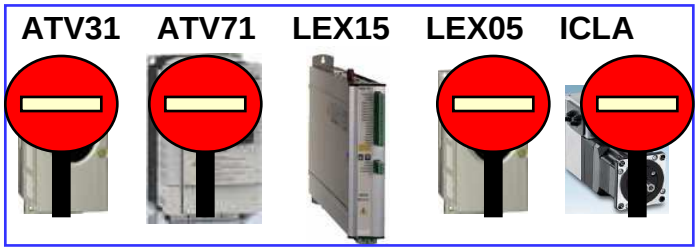


Lxm_DownloadMTask

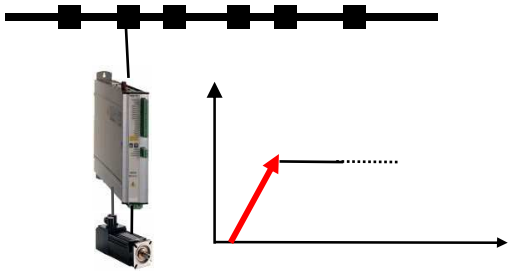


- **Purpose**:
  - Used to write Motion Tasks previously saved in a PLC memory zone

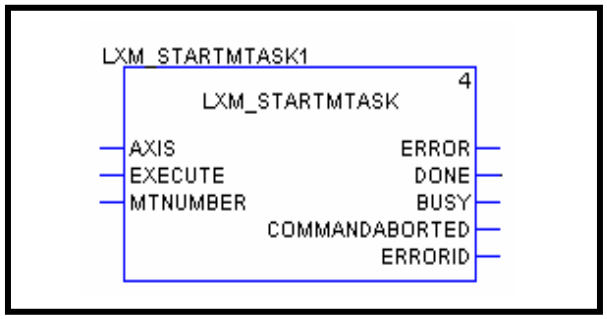
1- Motion Function Blocks



Lxm_StartMTask

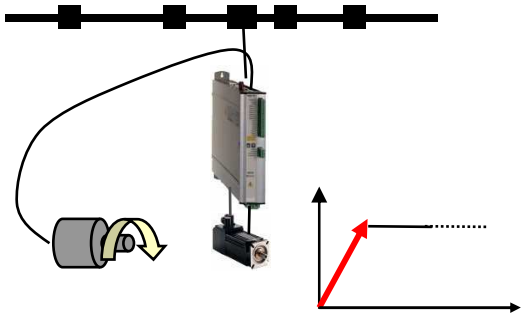


Starts an 'MT' movement

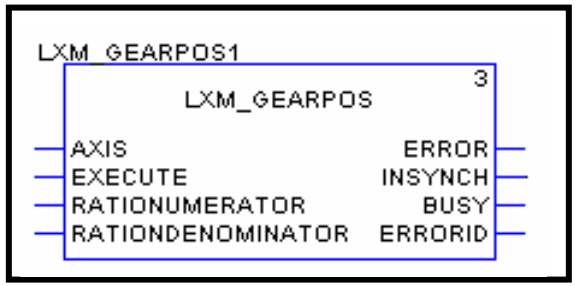


- **Purpose**:
  - Used to start a MOTION TASK step

Lxm_GearPos



Follow an incremental or SSI signal

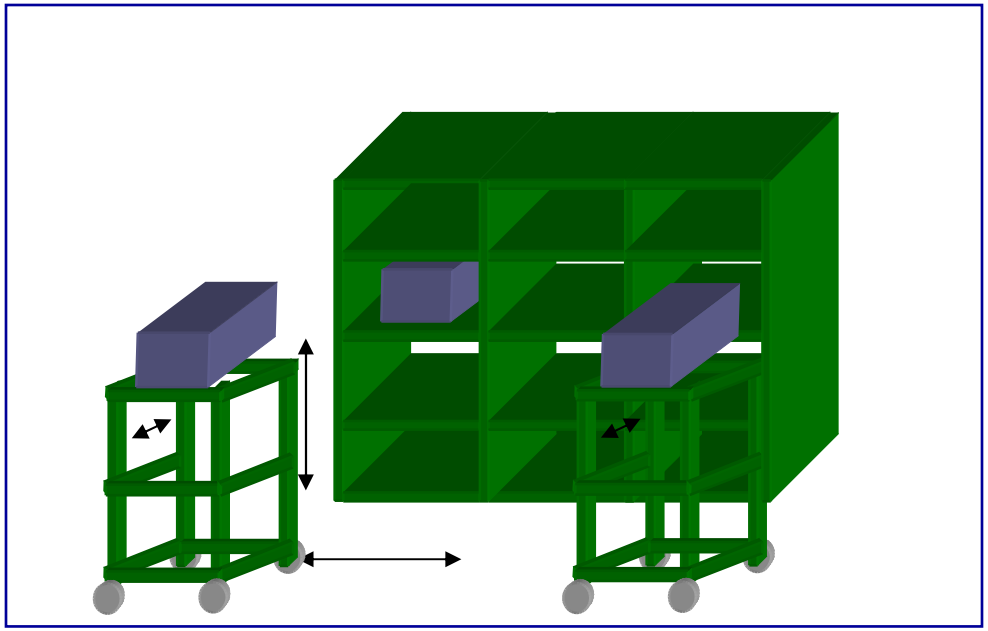


- **Purpose**:
  - Used to synchronize the position of a slave axis with respect to a master axis according to a coefficient (ratio)

Example : How to configure a MFB

E – Application examples

Automated storage

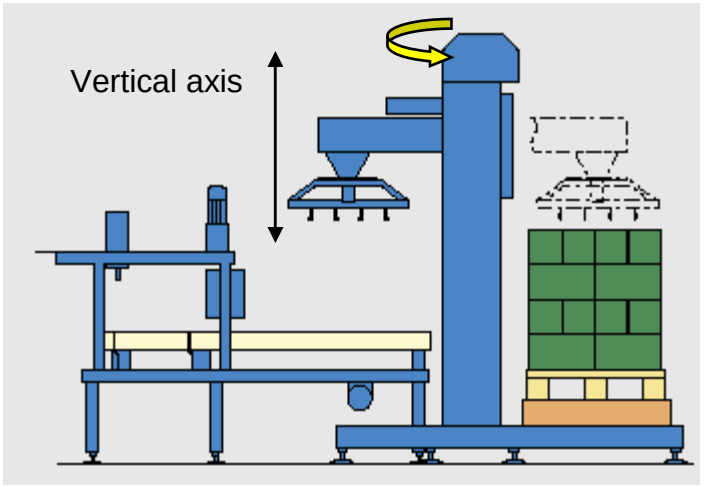


Goal of this application :

Automated storage system having three axes mechanism for putting product into the store at the right place.

- **Solution**
  - The MFB on CANopen provide the coordinated movement required with Lexium05 drives range and BDH motors.
  - Unity Pro and MFB allow to get a flexible application.
- **Key functions**
  - MC_MoveAbsolute / MC_ MoveAdditive
- **Key values**
  - Simple motion sequences in Unity
  - Flexible solution => same program for different size of products

Pick and Place

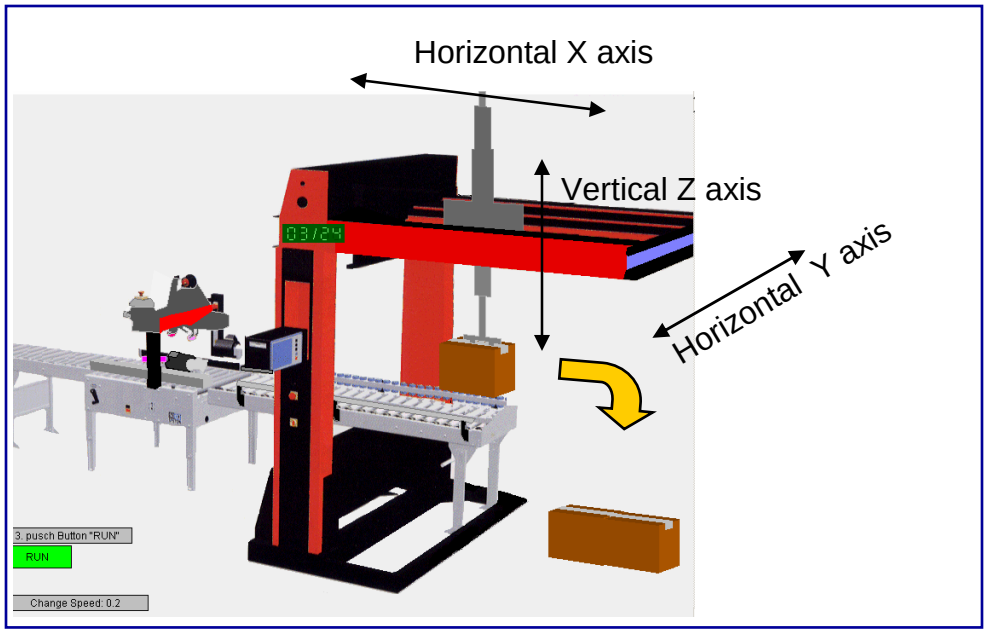


Goal of this application :

Machine assuring products pallet pick and place with one rotate axis and one vertical axis for lifting and lowering movement.

- **Solution**
  - Two independent axes are needed one rotate and one vertical using servo-drive like Lexium15. MFB is used to control simultaneously these two axes.
- **Key functions**
  - MC_MoveAbsolute, to control linear and rotate axes.
  - TE_UploadDriveParam / TE_DownloadDriveParam
- **Key values**
  - Flexible solution => same program for different size of products
  - High dynamic Lexium15 with BSH motors

Material handling

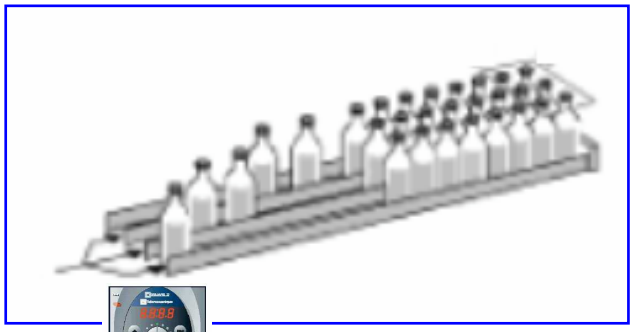


Goal of this application :

Pick and place Cartesian robot assuring products feed to the pallet products.
The robot has three independent linear axes for products positioning on the pallet layers.

- **Solution**
  - Classical gantry are widely used in material handling. At least three independent axes are required for translation and lifting movement using servo-drive like Lexium 05. MFB is used to control simultaneously these axes.
- **Key functions**
  - MC_MoveAbsolute / MC_ MoveAdditive
- **Key values**
  - No specialized motion module required
  - Real time positioning modification related to the product location on the pallet.

Material conveyor



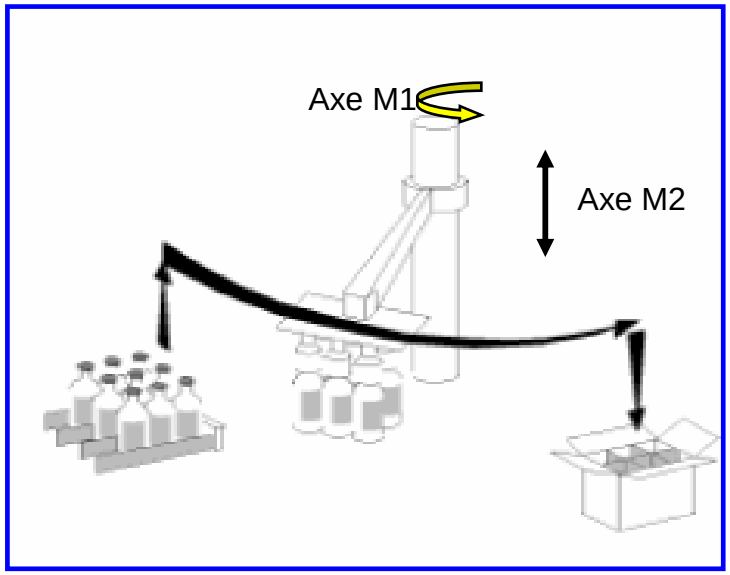
Conveyor motor

Goal of this application :

Feeder Belts Assuring products feed to the machine stacking products by a stopper.
Continuous operation and stop in case of jamming. Speed 130-140 m/min for the belt conveyors

- **Solution**
  - Most of conveyors are control by the inverters. ATV71 and ATV31 are the best products for that, compatible with MFB on CANopen.
- **Key functions**
  - MC_MoveVelocity
  - Speed modulation by modification of speed parameter
- **Key values**
  - Simple program adapted
  - Real time speed modulation

Boxing of products

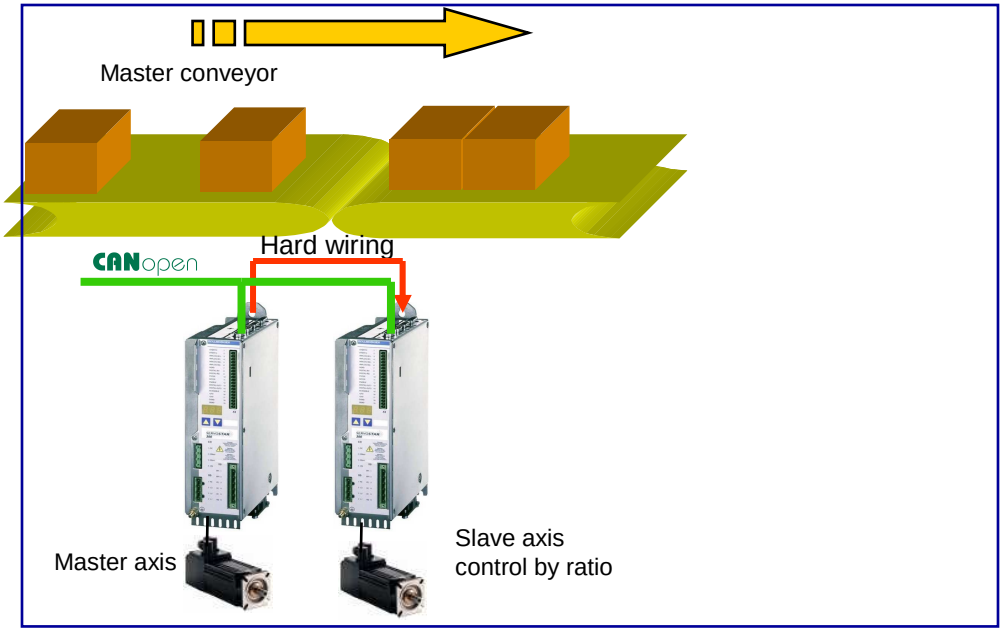


Goal of this application :

Boxing of product Assuring transfer and boxing of layers, using a Pick & Place unit (gripping is carried out by suction cups).

- **Solution**
  - Classical transfer machine are used in secondary packaging for boxing operation. The MFB allows to control simultaneously two coordinated axes for axis rotation and for lifting and Lowering movement.
- **Key functions**
  - MC_MoveAbsolute
- **Key values**
  - Best performances

Conveyor Grouping / Ungrouping systems



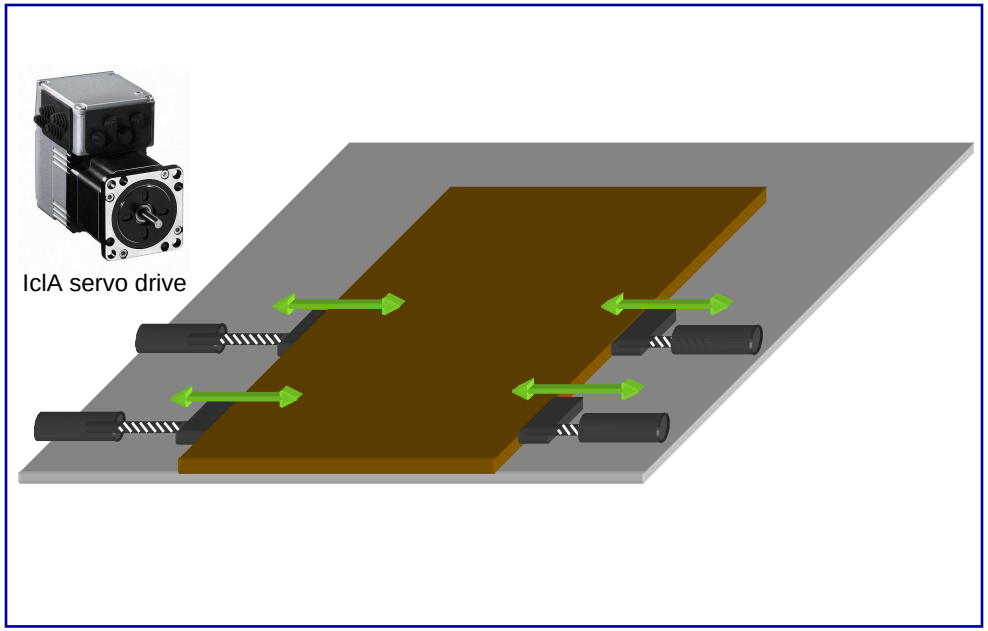
Goal of this application :

Conveyor Assuring grouping and ungrouping transfer of products.

The space between product are already constant. The Master/slave speed ratio determined the new space.

- **Solution**
  - Using the Master / Slave capability inside the Lexium15 drive, to manage the master and slave speed conveyor. The link between master and slave drives is done by cabling between I/O encoder emulation. The Lexium MFB are used to control de movement.
- **Key functions**
  - Lxm_GearPos to set the ratio values and linking to the master
  - MC_MoveVelocity for the master movement command.
- **Key values**
  - Simple motion program

Format adjustment in wood machines



Goal of this application :

Simple positioning system using servo-motor or stepper systems, assuring the machine setup in order to position guides for varying material sizes.

- **Solution**
  - Using MFB with IclA stepper or servo-drives for the positioning system. The position targeted could be adjusted by the HMI related to the type of product to do.
- **Key functions**
  - MC_MoveAbsolute
  - MC_ MoveRelative
- **Key values**
  - Simple positioning for auxiliaries axes
  - IclA and MFB is the best solution for flexible machine