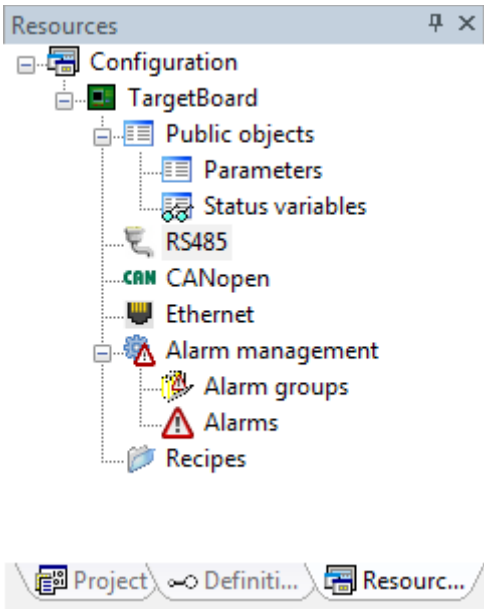
Index

1 MODBUS RTU

1.1 RESOURCES TAB



Resources tab allow you to define parameters and status variables to publish to other devices, field-bus network configuration, alarms definitions, recipes, I/O mappings and so on.

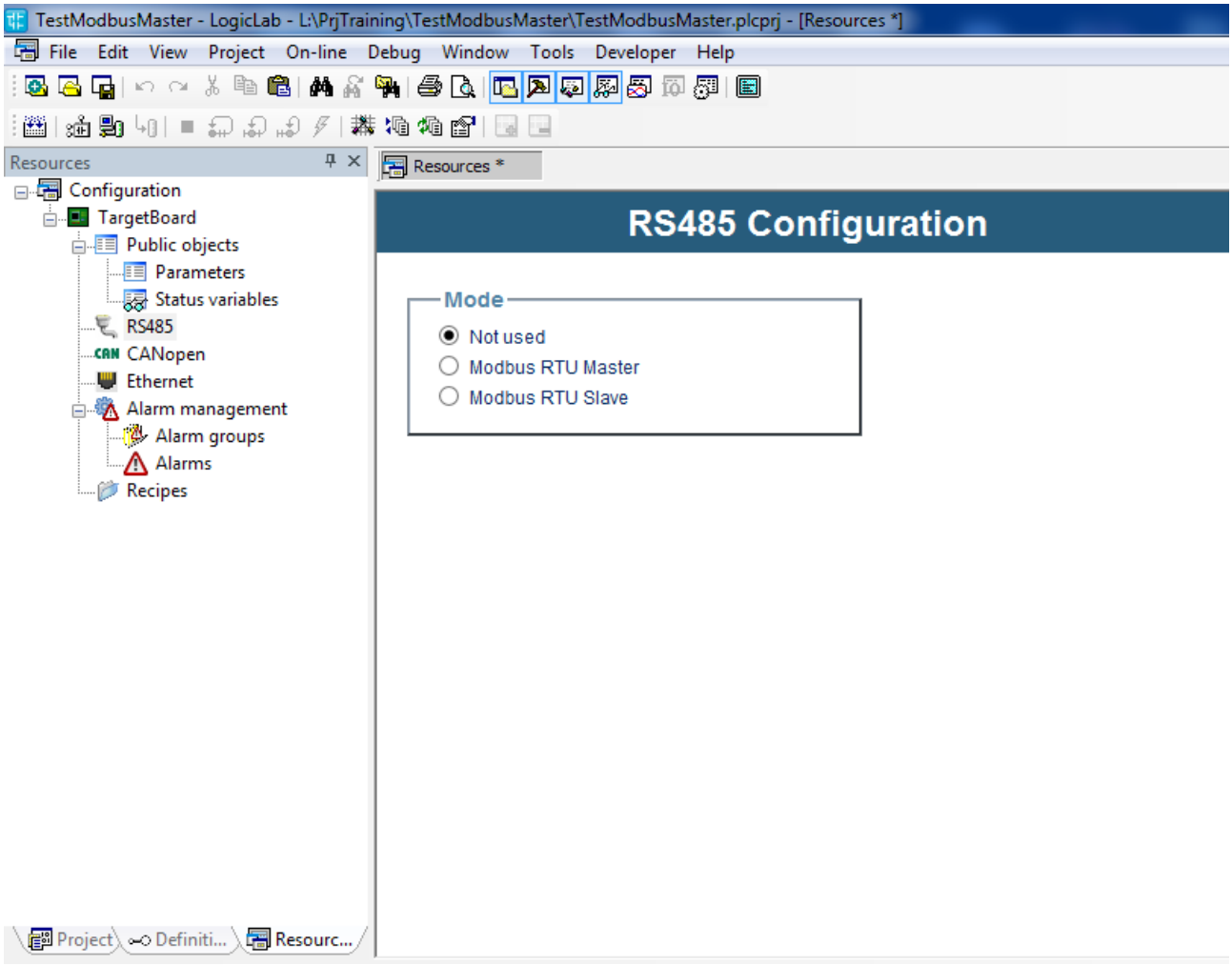
In your LogicLab release *Resources tab* could be quite different from the one described here (depending on the supported target features).

If your target board supports Modbus RTU you should have a serial port that can be configured to do so.

RS485 port can be used to configure your target as Modbus RTU Master (to control a network of slaves that answer your Modbus requests) or as Modbus RTU Slave (in this case another device makes requests to read or write parameters and status variables that you have previously defined as *Public objects*).

1.2 MODBUS RTU MASTER CONFIGURATION

Click RS485 to display the following page on the center screen:



You can decide how to configure the resource by choosing among:

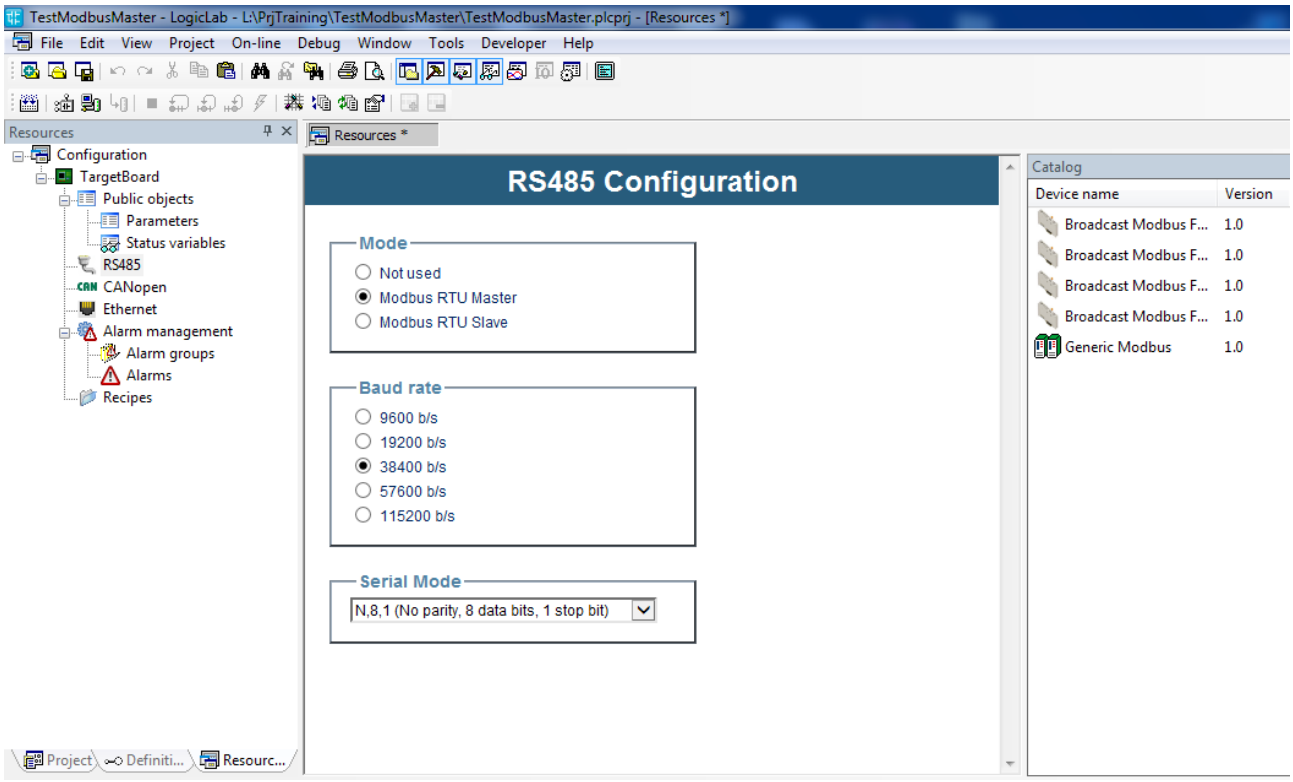
- Not used (default)
- Modbus RTU Master
- Modbus RTU Slave

Click the second option to configure the serial port as the Master in a Modbus RTU network.

Additional areas are shown on the center screen for setting:

- Baud rate
- Serial Mode (number of data bits, parity check, number of stop bit)

while the Catalog window lists all of the slave nodes that can be connected to the Master node.



If Catalog window is not visible click on menu Developer → View Catalog to show it.

Once Modbus RTU Master mode is set, you can start defining the device network:

- by dragging them from the Catalog window
- by adding them via Add on the context menu (right mouse key) assigned to the RS485 sheet

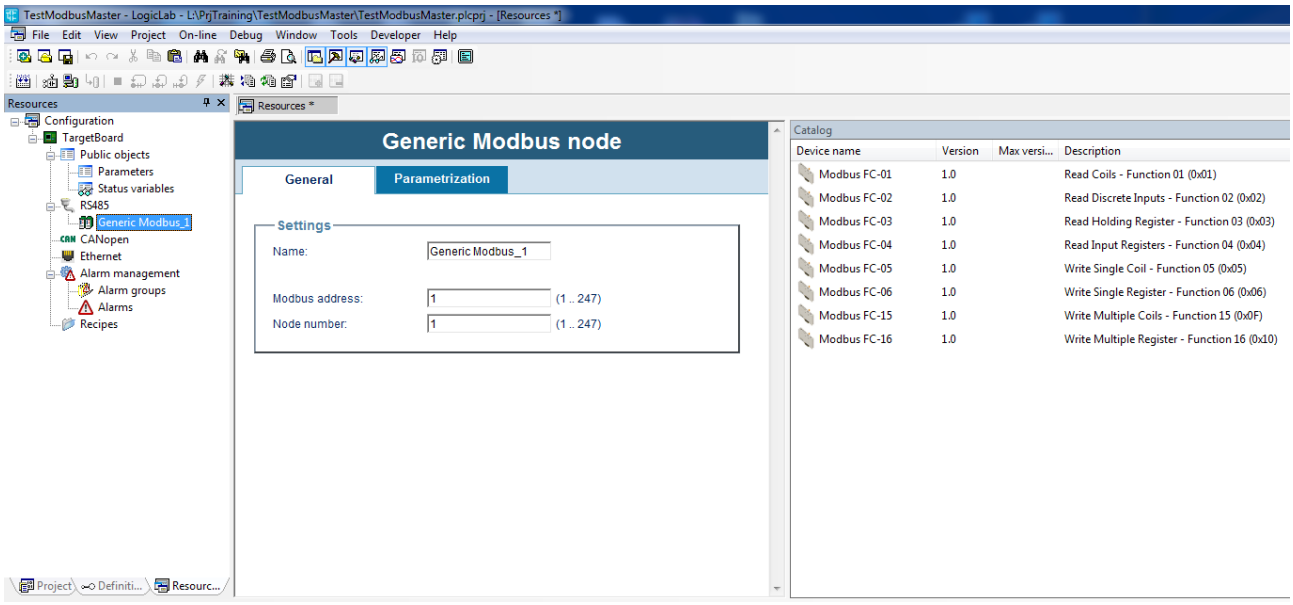
By dragging (or adding) the Generic Modbus node to the RS485 sheet, you can define a series of Modbus commands for exchanging, in read/write, one or more variables with a generic RTU slave device.

On the other hand, if you want to define your own custom device, you can do it by clicking Run ModbusCustomEditor on the Developer menu.

1.2.1 GENERIC MODBUS SLAVE

After inserting the Generic Modbus node on the RS485 sheet, click the node to display:

- the General tab and the Parameterization tab on the center screen
- the list of available Modbus commands in the Catalog window

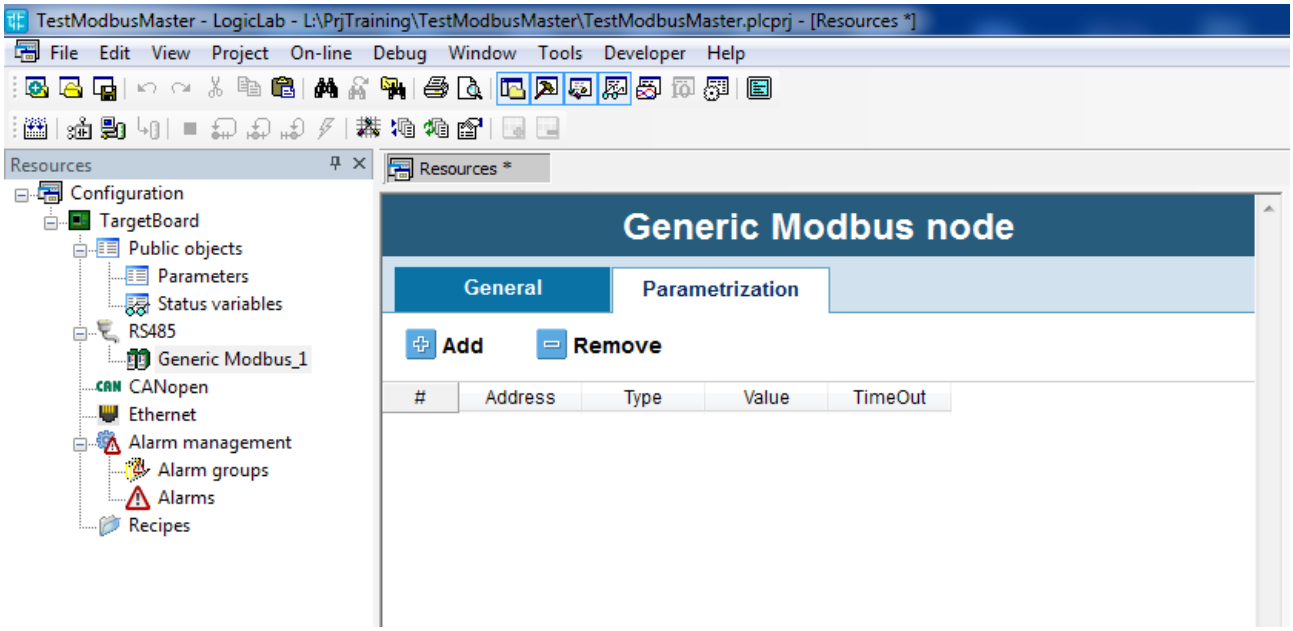


The General tab lets you set:

- Name: slave node name
- Modbus address: slave node MODBUS address
- Node number: value used by diagnostics (function blocks and system variables) to identify slave node

The Parameterization tab lets you define a series of Modbus objects sent to the slave node at each PLC restart.

The list of objects to be sent is built with the Add and Remove buttons that appear on the Tab.



Once a line (an object) is added, you have to set:

- Address: Modbus address assigned to object to be written
- Type: type assigned to object to be written
- Value: value to be assigned to Modbus object
- TimeOut: value in msec of TimeOut before operation is considered failed (default = 200 msec)

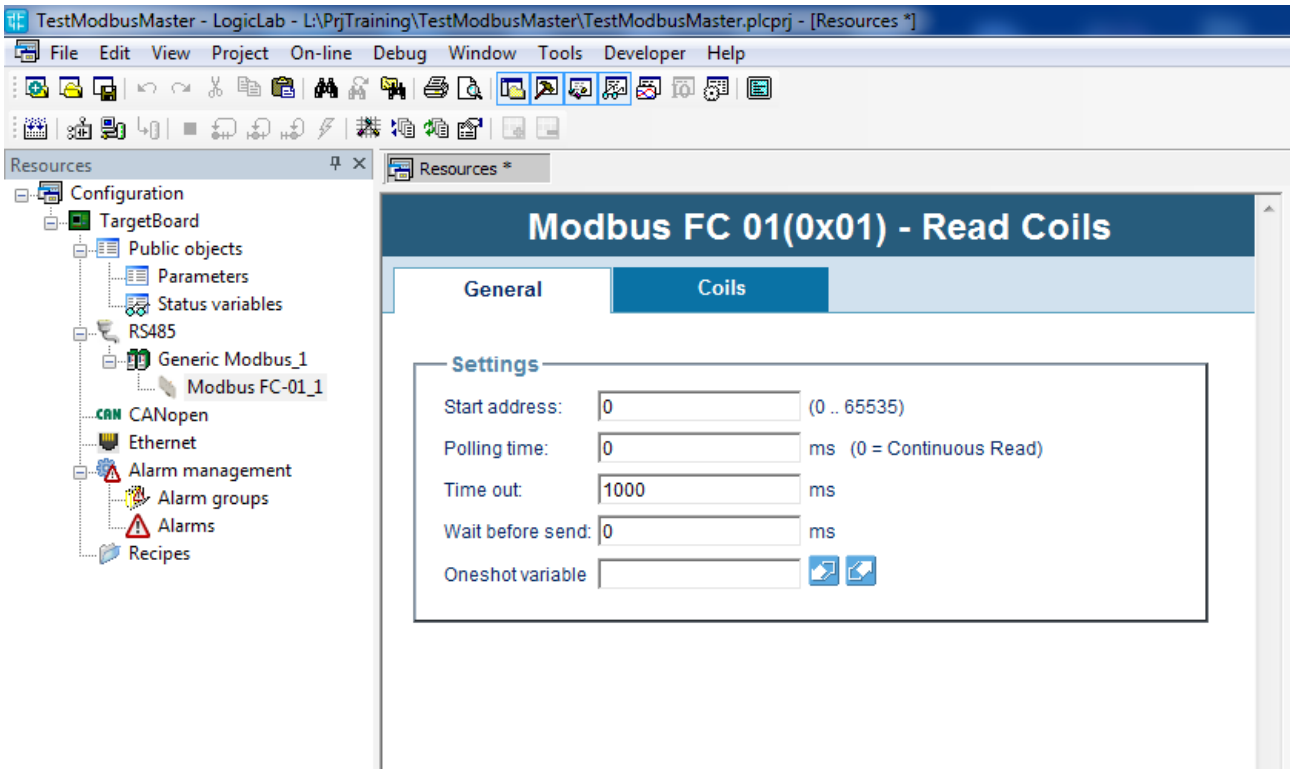
As mentioned above, by dragging or adding (via the Add window) one of the Modbus commands available in the Catalog window:

- Modbus FC-01(Read Coils): reading of one or more RW bits by slave node
- Modbus FC-02(Read Discrete Inputs): reading of one or more RO bits by slave node
- Modbus FC-03(Read Holding Registers): reading of one or more RW registers by slave node
- Modbus FC-04(Read Input Registers): reading of one or more RO registers by slave node
- Modbus FC-05(Write Single Coil): writing of a single RW bit on slave node
- Modbus FC-06(Write Single Register): writing of a single RW register on slave node
- Modbus FC-15(Write Multiple Coils): writing of one or more RW bits on slave node
- Modbus FC-16(Write Multiple registers): writing of one or more RW registers on slave node

You can define which and how many Modbus objects you want to exchange with the slave node.

Click the Modbus FC-01 command to display the following in the center window:

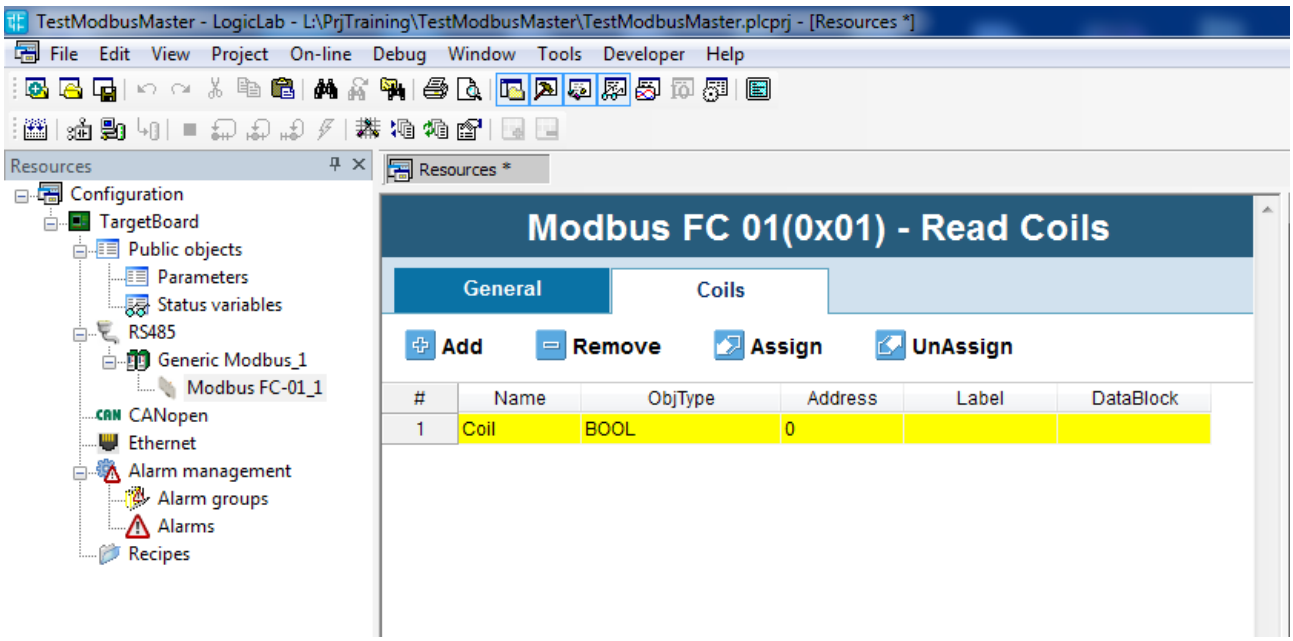
- General tab
- Coils tab



On the General tab, you have to set:

- Start address: MODBUS address of first bit to be read
- Polling time: interval in msec between two executions of MODBUS command:
  - =0 msec: continuous read, interval between two consecutive reads depends on PLC idle
  - ≠0 msec: wait value between two consecutive reads
- Time out: value in msec of timeout before operation is considered failed (default = 1000 msec)
- Wait before send: waiting time between two consecutive reads

  This time is added to value assigned to Polling time parameter. Default = 0
- Oneshot variable: you can specify a BOOL variable that is used to trigger the sending of the command. When this variable is set to TRUE the command is sent, then, when the command is actually served, the variable is automatically reset by Modbus RTU master.



On the Coils tab, you can decide (with the Add and Remove buttons), how many bits will be read by the slave starting from the Start address set on the General panel.

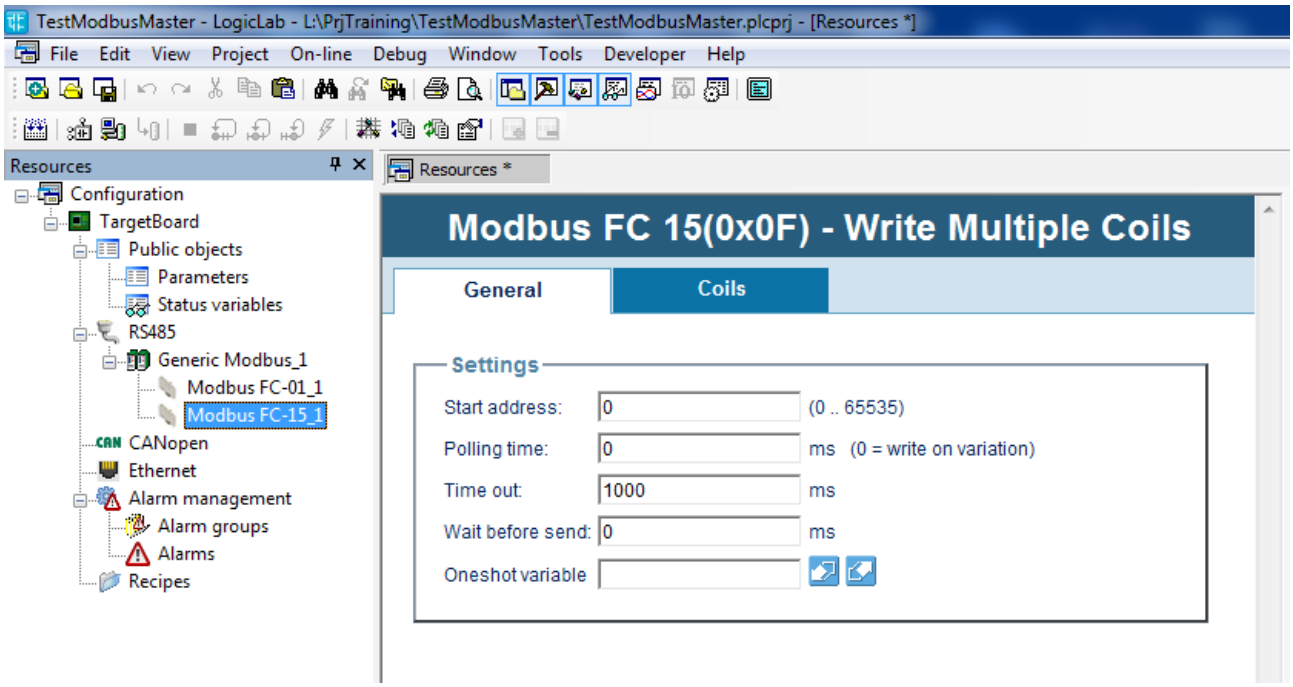
You have to assign a variable to each line inserted by using one of the modes described below:

- by means of the Assign button, click one of the Automatic (BOOL) variables previously declared on the PLC. The variable will no longer appear in the Automatic variables but will be mapped in the appropriate datablock.
- directly declare a BOOL variable in the Label column; the variable will be mapped in the appropriate datablock.

Note: the project must first be saved in order to run each of these two operations.

Click the Modbus FC-15 command to display the following in the center window:

- General tab
- Coils tab

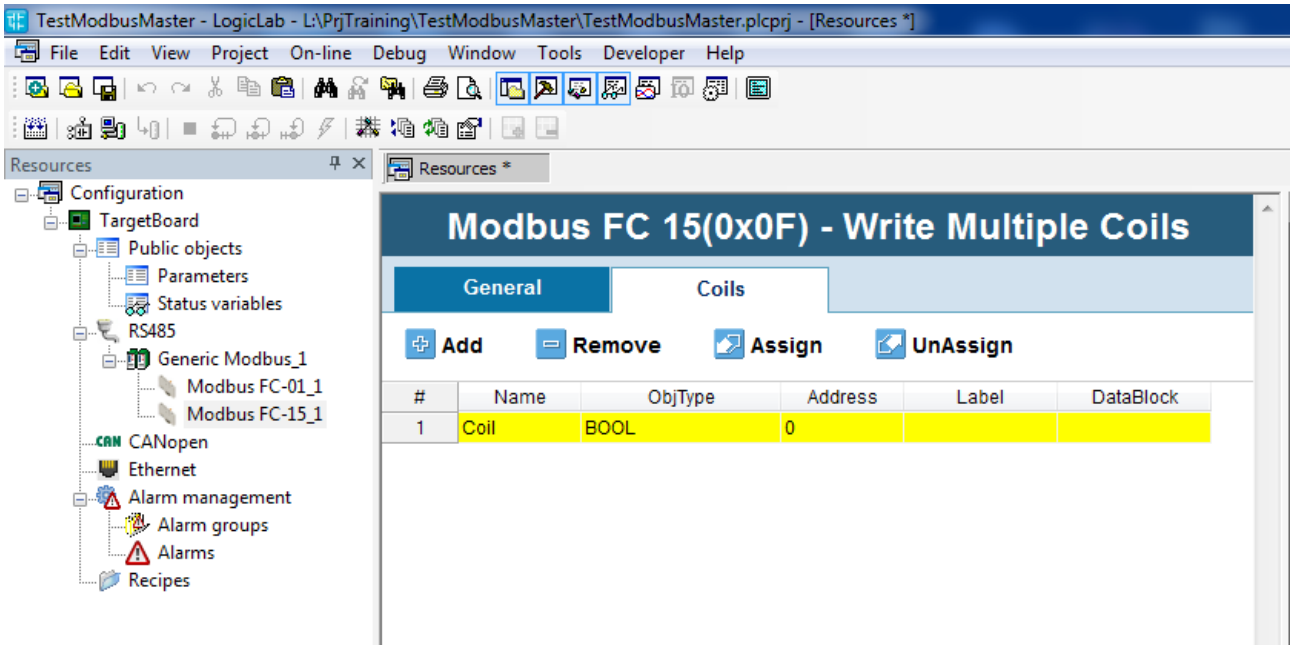


On the General tab, you have to set:

- Start address: MODBUS address of first bit to be written
- Polling time: interval in msec between two executions of MODBUS command:
  - =0 msec: write on variation
  - ≠0 msec: wait value between two consecutive writes
- Time out: value in msec of timeout before operation is considered failed (default = 1000 msec)
- Wait before send: waiting time between two consecutive writes.

  This time is added to value assigned to Polling time parameter. Default = 0.
- Oneshot variable: you can specify a BOOL variable that is used to trigger the sending of the command. When this variable is set to TRUE the command is sent, then, when the command is actually served, the variable is automatically reset by Modbus RTU master.

On the Coils tab, you can decide (with the Add and Remove buttons), how many bits will be written on the slave starting from the Start address set on the General panel.



You have to assign a variable to each line inserted by using one of the modes described below:

- by means of the Assign button, click one of the Automatic (BOOL) variables previously declared on the PLC. The variable will no longer appear in the Automatic variables but will be mapped in the appropriate datablock.
- directly declare a BOOL variable in the Label column; the variable will be mapped in the appropriate datablock.

Note: the project must first be saved in order to run each of these two operations.

1.2.2 MODBUS CUSTOM EDITOR

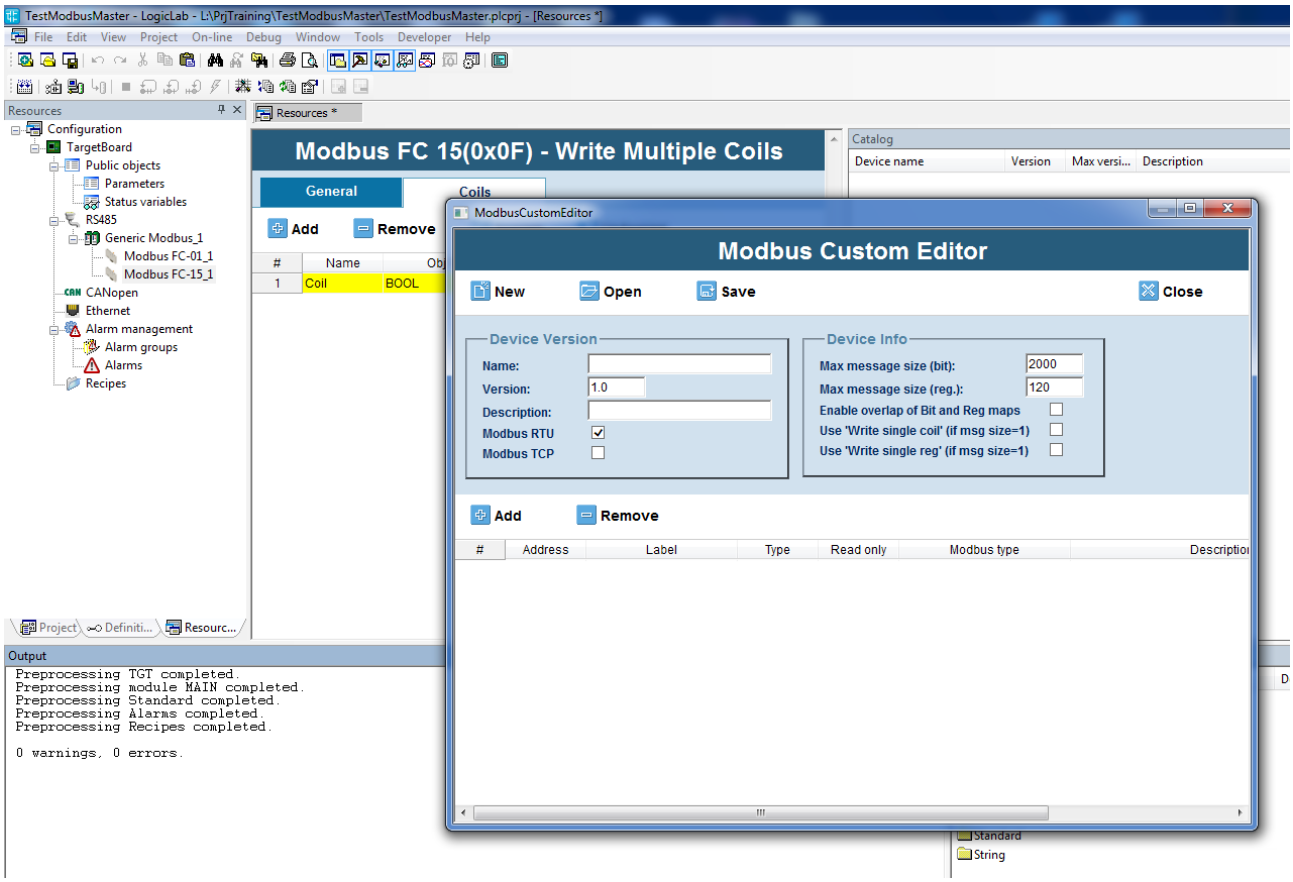
The Modbus Custom Editor tool is a graphics environment that lets you define the MODBUS map of a generic custom RTU and/or TCP device.

The file generated is automatically inserted in the LogicLab Catalog.

Note: LogicLab IDE functions are inhibited during all steps of configuration involving use of the Modbus Custom Editor tool.

As mentioned above, just click Run ModbusCustomEditor on the LogicLab Developer menu to launch the Modbus Custom Editor tool.

The following screen will appear:



Where:

- the New button lets you create a new empty custom file.
- the Open button lets you open a project.
- the Save button lets you save the open project.

  During this operation, the correctness of data inserted in the table is checked.
- the Close button lets you exit the tool.
- the Name field lets you set the name of the custom device to be added to the LogicLab Catalog

It is preferable to begin this field with a letter and not with a number. Otherwise, the name of the generated file will be preceded by a character "_".

- the Version field lets you set the version of the custom device.
- the Description field lets you set the description of the custom device.
- the Modbus RTU field (checkbox) lets you set whether or not the custom device can be used in a Modbus RTU (Serial) network.
- the Modbus TCP field (checkbox) lets you set whether or not the custom device can be used in a Modbus TCP (Ethernet) network.
- the Max data length bits field lets you set the maximum number of bits that the custom device will be able to exchange with the Master node via a single Modbus command.
- the Max data length reg field lets you set the maximum number of registers that the custom device will be able to exchange with the Master node via a single Modbus command.
- the Enable overlap of Bit and Reg maps field lets you set whether or not the bit map and the custom device registers overlap (i.e., whether or not they share addresses).
- the Add button lets you insert an object in the Modbus map of the custom device.
- the Remove button lets you remove selected objects (with yellow borders) from the Modbus map of the custom device.

Keep the CTRL button pushed to decide which elements to remove (even if not contiguous).

Keeping the SHIFT button pushed gives the same results, but in this case the selected elements MUST be contiguous.

In the displayed table:

- the Address column lets you set the Modbus address assigned to the object in question in the slave device.
- the Label column lets you set the name of the parameter assigned to the Modbus object in question.
- the Type column lets you set the type of parameter assigned to the Modbus object in question.

Available data types:

Type	Size
BOOL	bit
USINT	8 bit without sign
BYTE	8 bit (sign is insignificant)
INT	16 bit with sign
UINT	16 bit without sign
WORD	16 bit (sign is insignificant)
DINT	32 bit with sign
UDINT	32 bit without sign
DWORD	32 bit (sign is insignificant)
REAL	32 bit

- the Read only column lets you set the access mode to the object in question:
  - Variable in read only (Read only = TRUE)
  - Variable in read/write (Read only = FALSE)

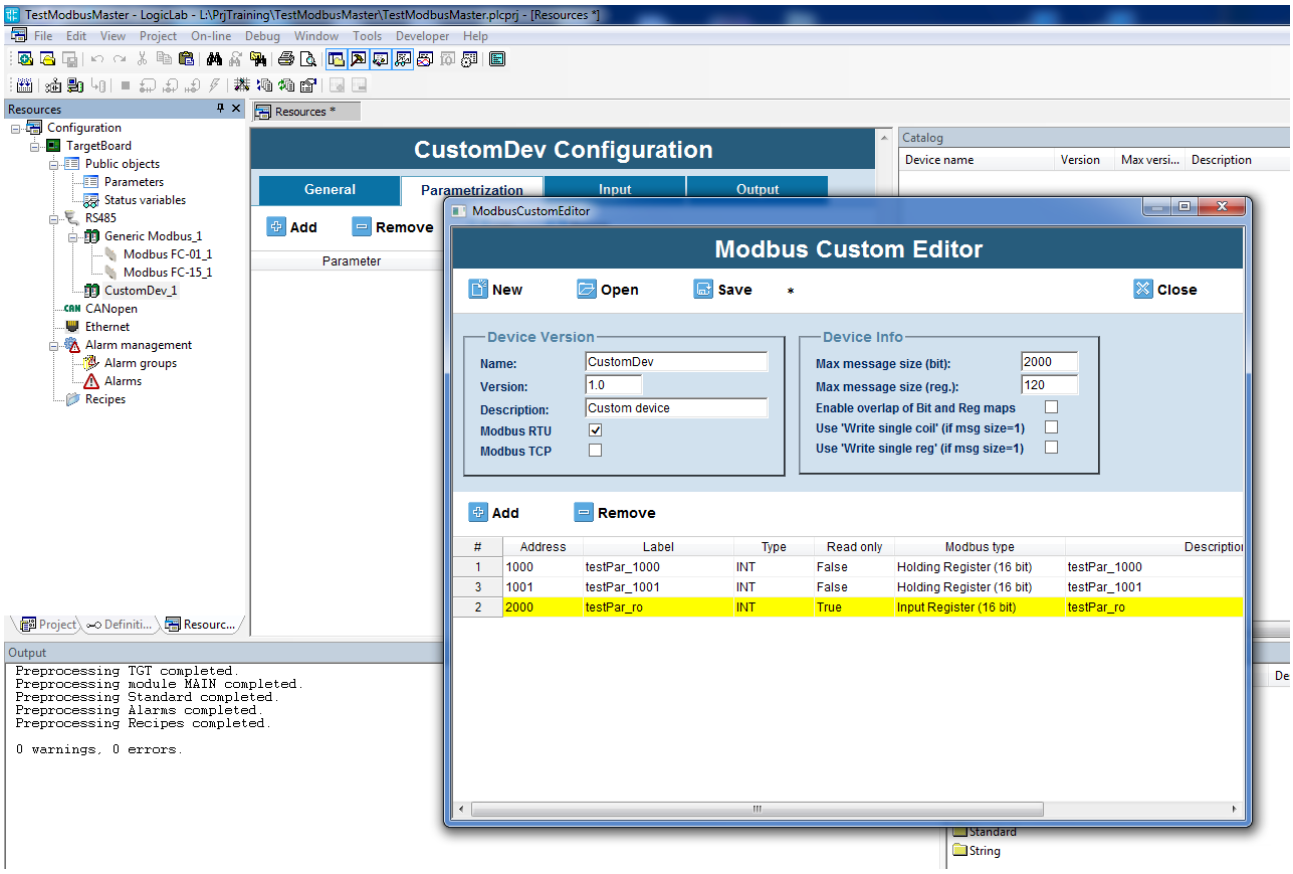
- the Modbus Type column lets you set the configuration of the Modbus object in question (Type and Read only columns), using the standard code provided by the Modbus communication protocol.

The configuration of Modbus type column and the Type and Read only columns are strictly correlated; any changes to one of these columns will cause automatic alignment of the others (see table below):

Type	Read only	Modbus Type
BOOL	TRUE	Discrete Input
BOOL	FALSE	Coil
USINT	TRUE	Input register (8 bit)
USINT	FALSE	Holding register (8 bit)
BYTE	TRUE	Input register (8 bit)
BYTE	FALSE	Holding register (8 bit)
INT	TRUE	Input register (16 bit)
INT	FALSE	Holding register (16 bit)
UINT	TRUE	Input register (16 bit)
UINT	FALSE	Holding register (16 bit)
WORD	TRUE	Input register (16 bit)
WORD	FALSE	Holding register (16 bit)
DINT	TRUE	Input register (32 bit)
DINT	FALSE	Holding register (32 bit)
UDINT	TRUE	Input register (32 bit)
UDINT	FALSE	Holding register (32 bit)
DWORD	TRUE	Input register (32 bit)
DWORD	FALSE	Holding register (32 bit)
REAL	TRUE	Input register (32 bit)
REAL	FALSE	Holding register (32 bit)

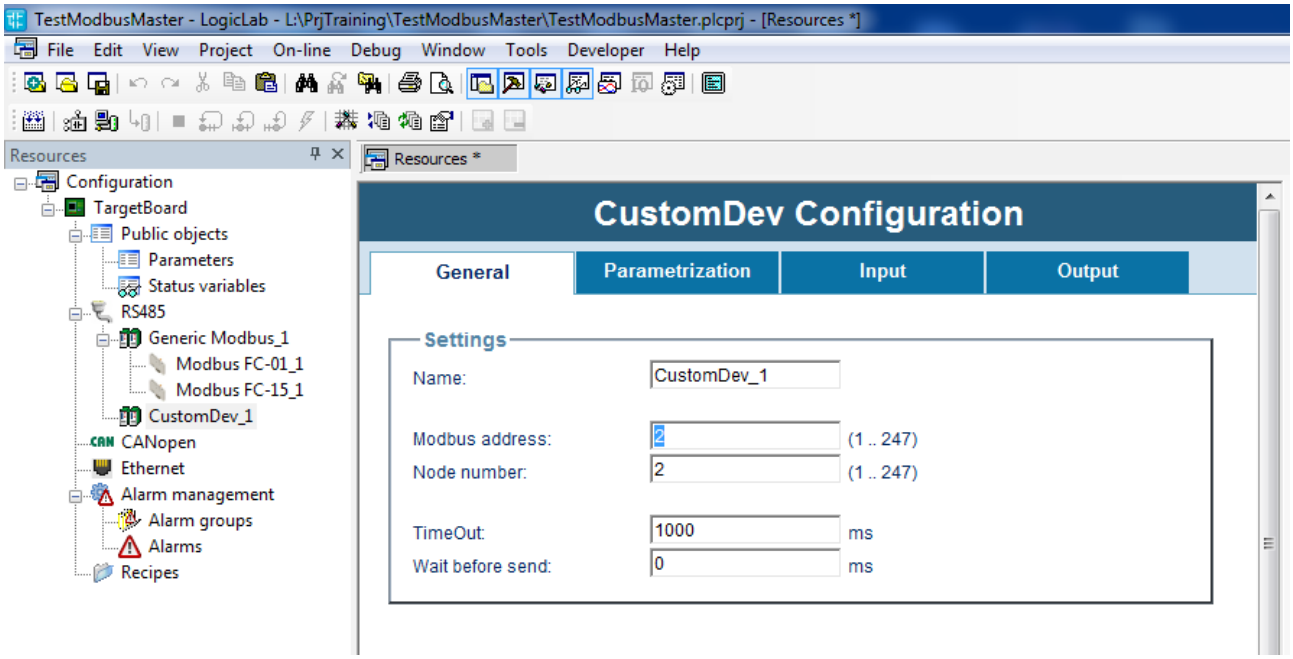
- the Description column lets you set the description of the Modbus object in question.

Here we define a CustomDev 1.0 sample slave device with just three parameters:



1.2.3 DEVICES IN CATALOG

By dragging or adding one of the Catalog devices (including a custom device you have defined) via the Add window (context menu control), the following screen appears:

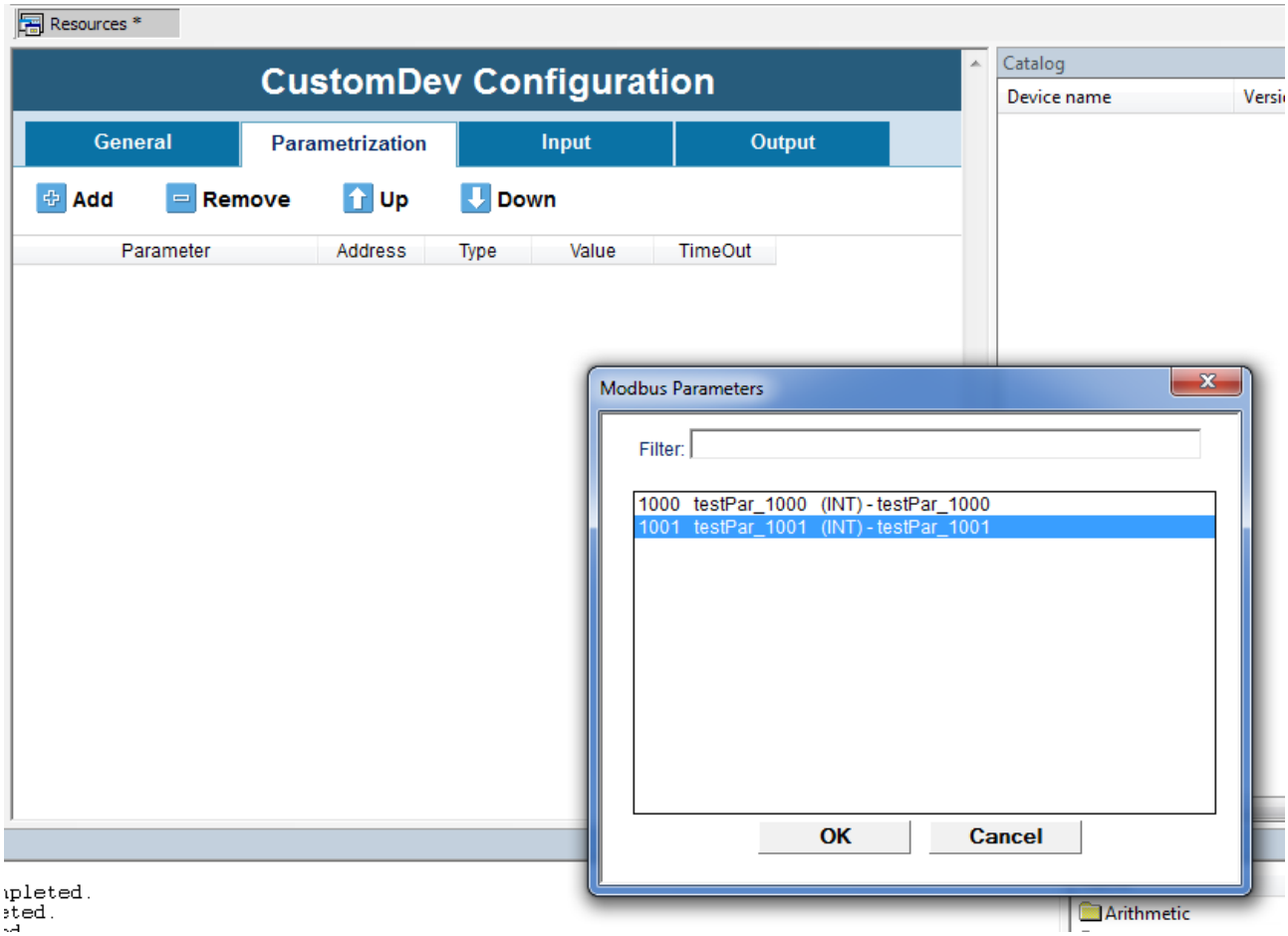


The General tab lets you set:

- Name: name of slave node.
- Modbus address: MODBUS address of slave node.
- Node number: value used by diagnostics (function block and system variables) to identify the slave node.
- TimeOut: value in msec of TimeOut before operation is considered failed (default = 1000 msec)
- Wait before send: waiting time between two consecutive operations. This time is added to value assigned to Polling time parameter. Default = 0.

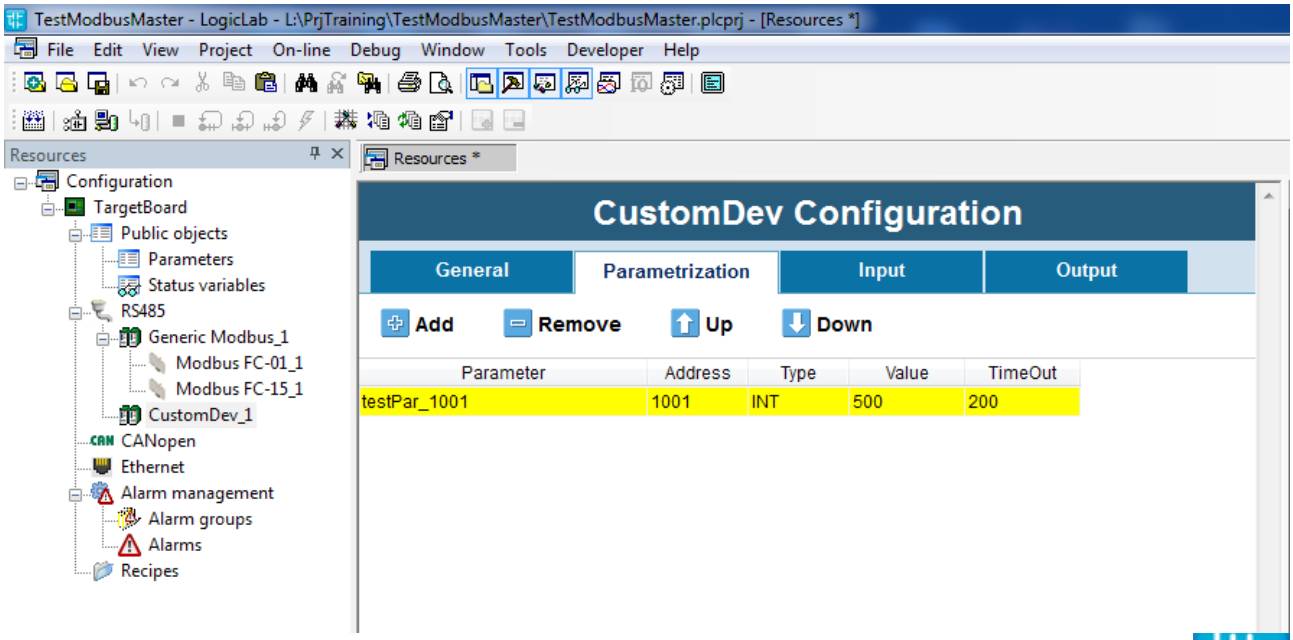
The Parameterization tab lets you define a series of Modbus objects sent to the slave node at each PLC restart.

The list of objects to be sent is built with the Add and Remove buttons that appear on the Tab.



The objects shown in the window that appears after the Add button is pushed are all (and only) RW parameters contained in the device Modbus dictionary.

You can set the order for sending parameters to the slave node by using the Up and Down buttons.



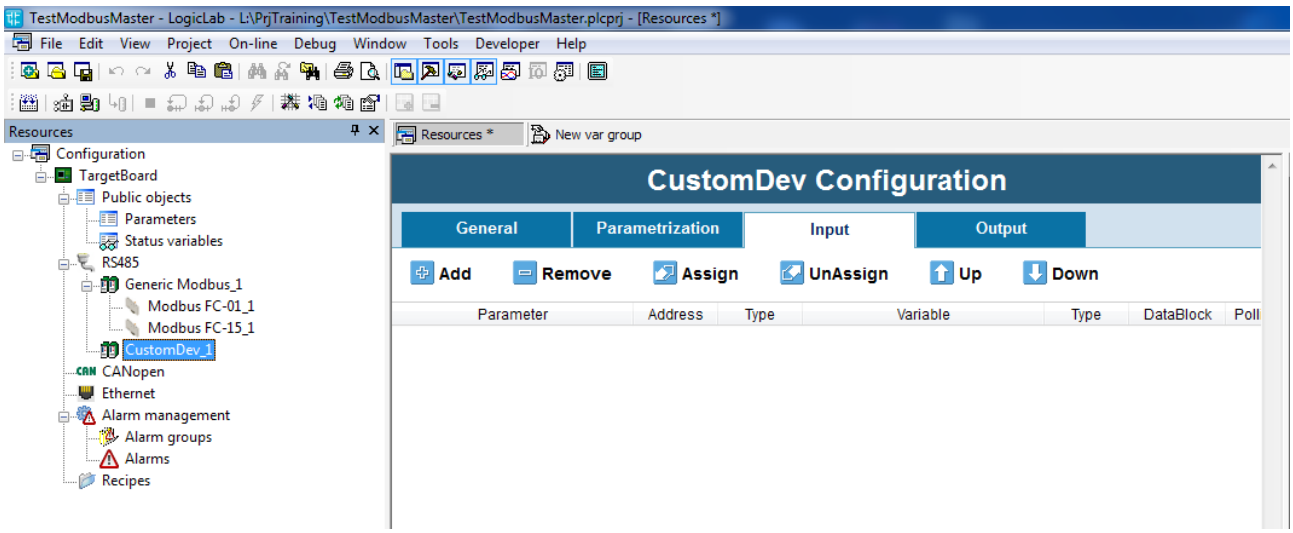
Once a line (an object) is added, you have to set:

- Value: value to be assigned to Modbus objectTimeOut: value in msec of TimeOut before operation is considered failed (default = 200 msec)

The Input tab lets you define (with the Add and Remove buttons) which and how many parameters will be read by the device.

The window that appears when the Add button is pushed shows all of the available objects in the device Modbus dictionary (both RO and RW).

You can set the order for reading parameters from the slave node by using the Up and Down buttons.



You have to assign a variable to each line inserted by using one of the modes described below:

- by means of the Assign button, click one of the Automatic variables (the type depends on the contents of the Type column) previously declared on the PLC.

The variable will no longer appear in the Automatic variables but will be mapped in the appropriate datablock.

- directly declare a variable in the Label column (the type depends on the contents of the Type column); the variable will be mapped in the appropriate datablock.

You can also set the Polling time parameter to define the interval in msec between two executions of MODBUS command:

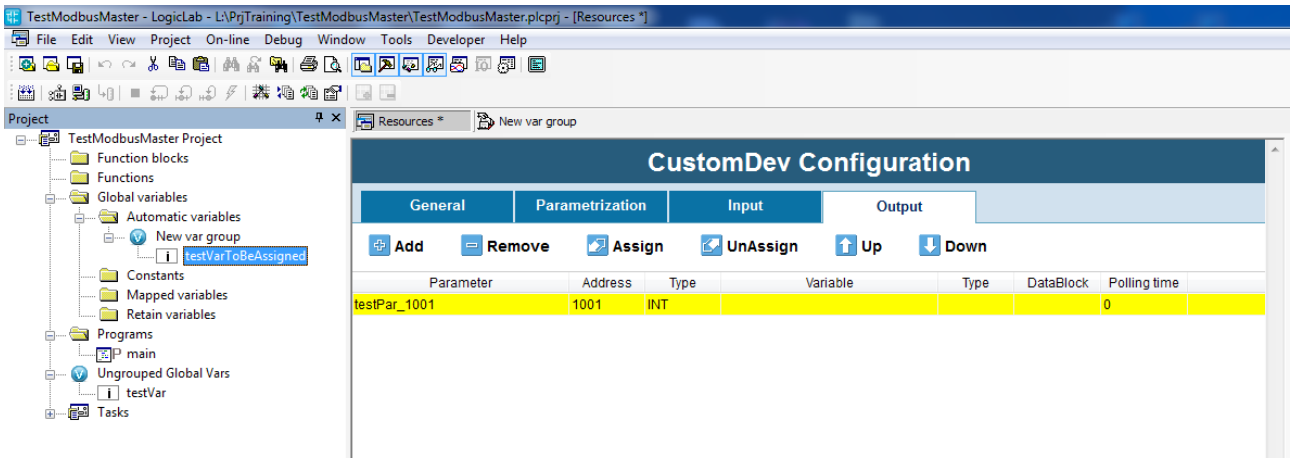
- =0 msec: continuous read, interval between two consecutive reads depends on PLC idle
- ≠0 msec: wait value between two consecutive reads

If the listed objects have:

- Contiguous Addresses
- Identical Polling time
- Homogeneous Type: USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD and REAL are homogeneous with regard to Modbus because they are all seen as one or more registers.

they will be grouped in a single Modbus request, subject to the maximum number of consecutive objects (bits or registers) that can be exchanged with the device in a single

request.



The Output tab lets you define (with the Add and Remove buttons) which and how many parameters will be written on the device.

The objects shown in the window that appears after the Add button is pushed are all (and only) RW objects contained in the device Modbus dictionary.

You can set the order for writing these parameters on the slave node by using the Up and Down buttons.

You have to assign a variable to each line inserted by using one of the modes described below:

- by means of the Assign button, click one of the Automatic variables (the type depends on the contents of the Type column) previously declared on the PLC.

The variable will no longer appear in the Automatic variables but will be mapped in the appropriate datablock.

- directly declare a variable in the Label column (the type depends on the contents of the Type column); the variable will be mapped in the appropriate datablock.

You can also set the Polling time parameter to define the interval in msec between two executions of MODBUS command:

- =0 msec: write on variation
- ≠0 msec: wait value between two consecutive writes

Note:

If the listed objects have:

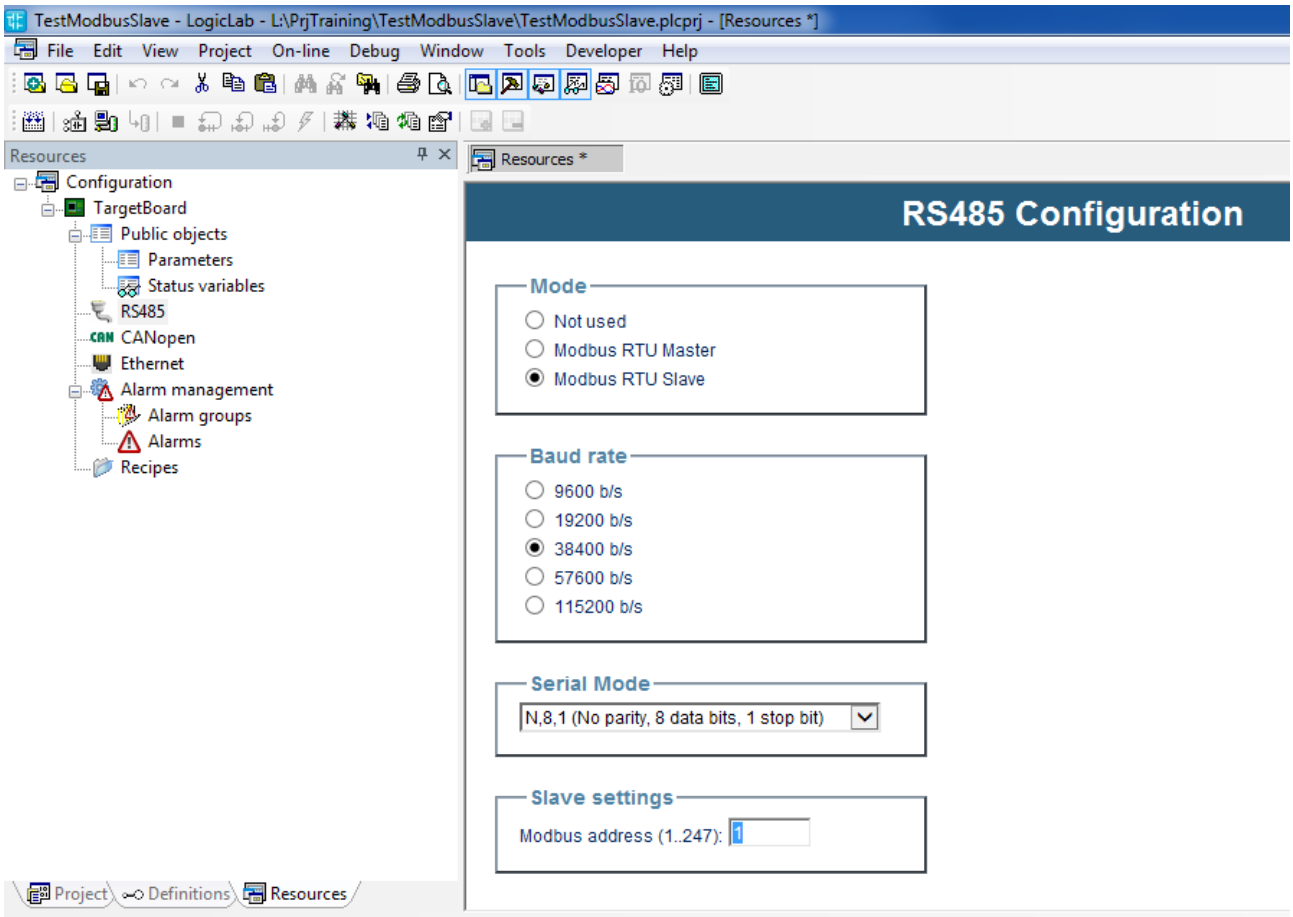
- Contiguous Modbus Addresses
- Identical Polling time
- Homogeneous Type: USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD and REAL are homogeneous with regard to Modbus because they are all seen as one or more registers.

they will be grouped in a single Modbus request, subject to the maximum number of consecutive objects (bits or registers) that can be exchanged with the device in a single request.

1.3 MODBUS RTU SLAVE CONFIGURATION

Configure the target board as Modbus RTU slave if you want your board to manage R/W request from a master.

From the RS485 window select the Modbus RTU Slave option and this screen will appear:



You can configure the serial port Slave in a Modbus RTU network.

In this case, additional areas are shown on the center screen for setting:

- Baud rate
- Serial Mode (number of data bits, parity check, number of stop bit)
- Slave settings (node number)

In this case, the Catalog window remains empty.

A Master on the network would be able to read/write parameters, status variables and local I/O depending on what is published by your slave.

1.4 MODBUS DATABASE DEFINITION

Slave should publish a database indexed using Modbus addresses.

On your target you can have:

- Embedded database

  always available on the target board

- Application database

  defined using LogicLab

- Both

  a range of Mobus addresses will be reserved for embedded parameters database definitions and a distinct range of addresses could be used for application database definitions.

1.4.1 TARGET EMBEDDED DATABASE

Provider of the target board has already defined a Modbus object dictionary and has established the address values and their meanings (match between address and corresponding parameter / status variable / input / coil) for the target.

If your target board has an on board Modbus database you can simply enable RTU slave mode to give access to the on board database.

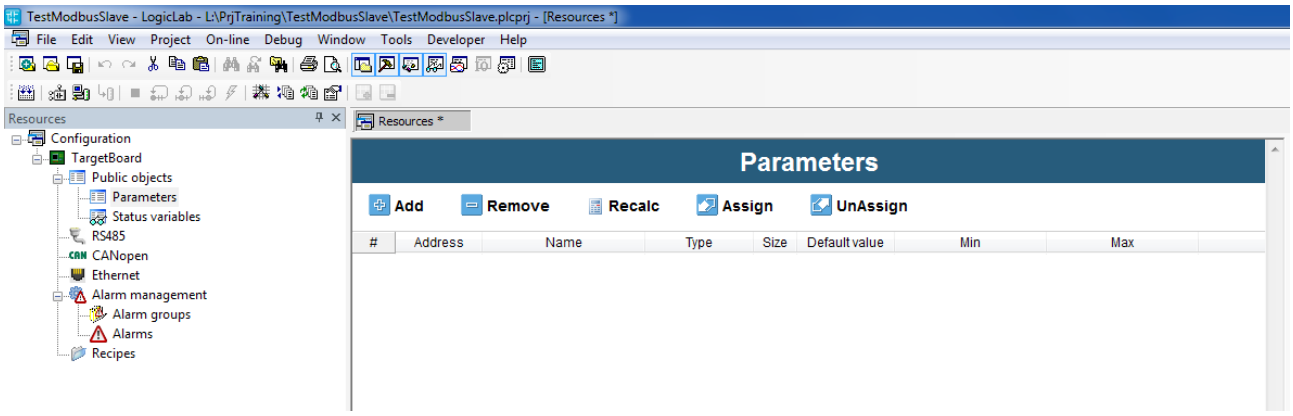
1.4.2 APPLICATION DATABASE (USING LLEXEC)

Objects that can be accessed by an external Master are defined using LogicLab itself.

You can begin defining the dictionary of published Modbus objects in the following areas:

- Parameters: where you can insert all of the objects you want to be written just once
- Status variables: where you can insert all of the objects that can be read/written multiple times

If you are using a target board that run Axel LLExec runtime the following screen will be displayed when you click the Parameters node:



where:

- you can use the Add button to add a new line to the table shown in the figure in order to receive the publication of one of the project variables. In the Address column of the table, the Add command will propose the first useful Modbus address, allowing you to change it.
- you can use the Remove button to remove one or more lines from the grid.
- you can use the Recalc button to force automatic recalculation of Modbus addresses of the selected lines, starting from address 1. This operation overwrites previous settings.
- you can use the Assign button to publish, in the Modbus address indicated in the Address column, one of the Automatic variables declared in the PLC project that will become part of the mapped variables (no longer on the Automatic list). If you insert (in the Name column of the table) a variable that is not one of those declared Automatic in the PLC, that variable will be inserted in the appropriate datablock.
- you can use the UnAssign button to remove an assignment between Modbus address and the variable mapped in the datablock; the variable will appear on the Automatic list of PLC project.

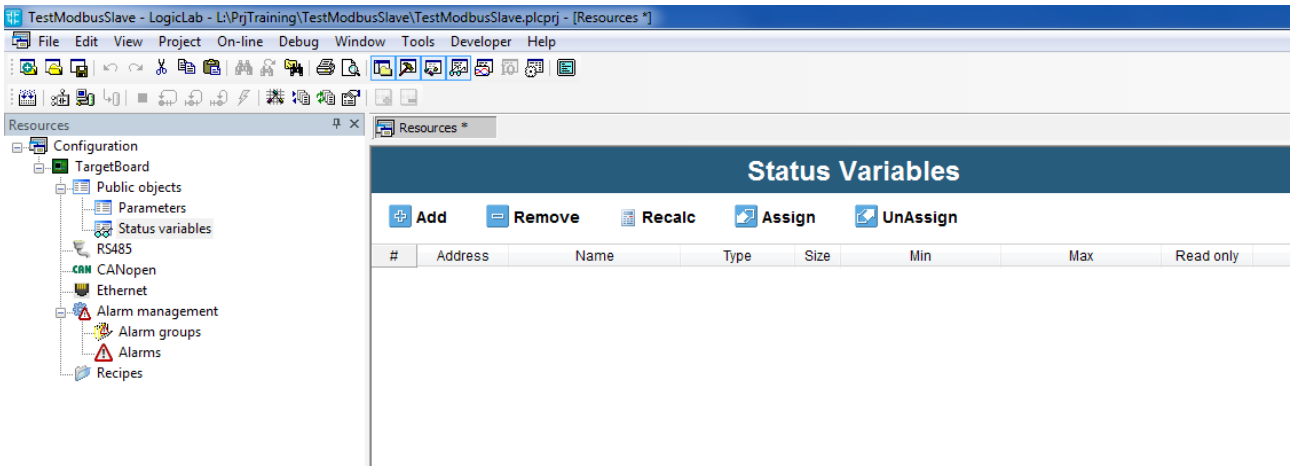
In addition to the Address and Name columns, you can also set the following columns:

- Default value: contains the initial value of the Modbus object.
- Min: contains the minimum value that can be assigned to the Modbus object in question by the Master by means of a write operation. This limit can be a numerical constant or a parameter (i.e., one of the Modbus objects inserted in the Parameters or Status variables table).
- Max: contains the maximum value that can be assigned to the Modbus object in question by the Master by means of a write operation. This limit can be a numerical constant or a parameter (i.e., one of the Modbus objects inserted in the Parameters or

Status variables table).

- Description: contains a brief description of the object.

Click the Status variables area to display the following screen:



With regard to the buttons and columns, see the description of the Parameters area.

In this case, since the listed objects are RW, you can decide (via the Read only column) whether the object can ONLY be read or READ and WRITTEN by the Master.

1.4.3 OTHER APPLICATION DATABASE IMPLEMENTATIONS

Database configuration interface depends on the database structure implemented on the real target.

If your target is not running Axel LLExec runtime database implementation and management could be quite different.

Your application database can be implemented in this way:

- On the target a datablock have been dedicated to be used for database definition.
- A convention between Modbus addresses and data access into this datablock is established.
- When you define a variable mapping it on the datablock its value can be accessed by modbus (using the established convention)

Let see it with an example:

- The target board has been implemented to publish an array of 1000 WORDs (this is your database).
- This array has been published to LogicLab as datablock.
- A Modbus address range from 50000 to 50999 has been reserved and dedicated to the access to the database (this is the established convention).
- If you map a WORD variable on the first element of the datablock its value would be accessed via Modbus at the address 50000, if you map it on the last element of the datablock it would be accessible at the address 50999.
- If you use LogicLab resources configurator (click on Parameters or Status variables node) you will be able to choose Modbus addresses, then corresponding variables will

be automatically mapped onto the right place on the datablock.

In this way you don't have to take care of where the variable is mapped onto the datablock. You can refer to the variable in the PLC using the name you have chosen and a Master can access to it using the assigned Modbus address.