Easy Configurator user manual   rev 1.0

1 – The Easy Configurator

The Easy Configurator is a software tool that allows you to customize the numbers of bytes exchanged between the EasyCAT and the master, as well as their names and data types, without having to dig into the complexity of the EtherCAT® specifications.

To do this, it creates the three files described in chapter **"3 – How the EasyCAT is configured"**, I.E. the .h the .xml and the .bin , and it can also program the EasyCAT EEPROM with the .bin file, as described in chapter **"5 - The embedded programmer"**.

When in this document we refer to the "EasyCAT", we mean not only the EasyCAT shield for Arduino, but all the AB&T boards belonging to the EasyCAT family.

Important note – The Easy Configurator is not suitable for EtherCAT® slaves from other manufacturers.
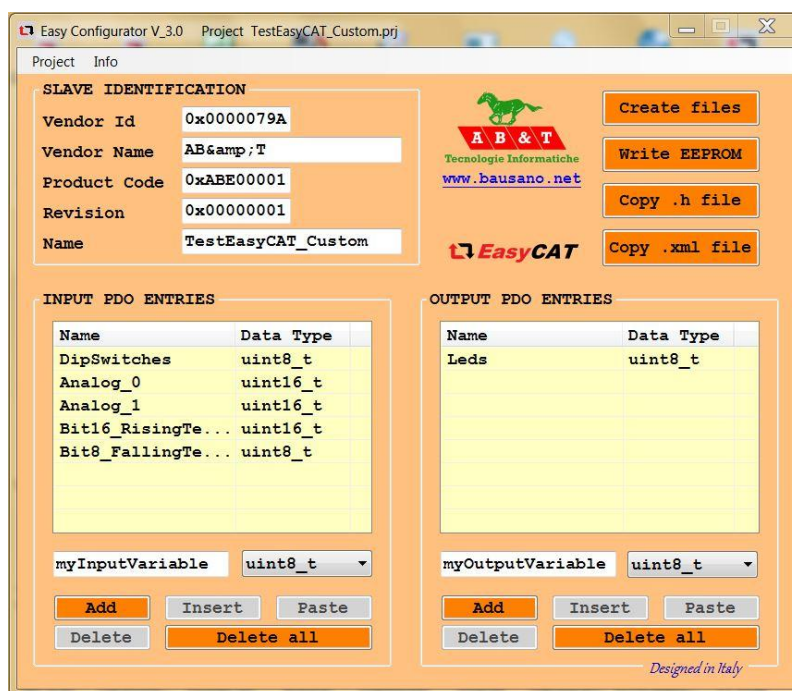

Fig-1   The Easy Configurator window

1.1 – System requirement

The Easy Configurator runs on any Windows PC, from XP onward, equipped with a standard Ethernet interface 100BASE-TX.

Please note that the Easy Configurator embedded programmer needs the WinPcap library to be installed on your PC. The library is freely downloadable from the official WinPCAP website .

May be that WinPcap is already installed on your PC, as it is the capture engine used by many popular network tools, as the widespread WireShark: in this case you don't have to install it again.

Further be sure that the EasyCAT library release V_2_0 or above is installed on your PC.

1.2 – Installation

The Easy Configurator is a portable program, I.E. it doesn't need an installation: you just have to download the *EasyConfigurator.zip* file from the AB&T website and unzip it somewhere on your hard disk.

Here you can see the folders structure that will be created.
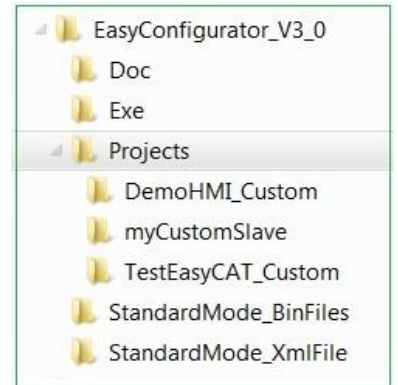

Fig-2 Folders structure

Under the *Exe* folder there is the *EasyCAT_Config_GUI.exe* file, that is the one you have to launch to start the Easy Configurator.

The *EasyCATConfig.exe* is the configuration engine and the *EEPROMprogrammer.exe* is the programming engine, and they are launched automatically by the Easy Configurator.
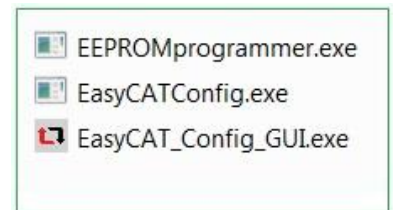

Fig-3 Exe folder

Under the *Projects* folder there are the subfolders that contain same project examples.
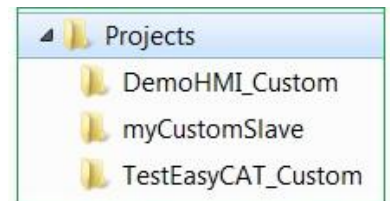It is recommended to put here every project that you will create.

Fig-4 Projects folder

Under each project folder there is the .prj file , that contains the project configuration as defined by the user.
The .h , .xml and .bin files, are created by the Easy Configurator and will be used to customize the EasyCAT slave, as described later.
The *log_configuration.txt* and the *log_programmer.txt* are log files that can be examined from the menu "Info → Config log" and "Info → Prog log".
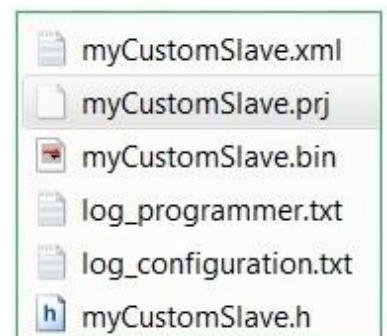

Fig-5 Project files

The *StandardMode_BinFiles* folder contains the four .bin files, useful if you chose to use the *Standard Mode* , as described in the paragraph **"4.2 – Using the *Standard Mode*"**.

Fig-6 StandardMode_BinFiles folder

The *Doc* folder contains the Easy Configurator user manual, that is the document that you are reading just now.
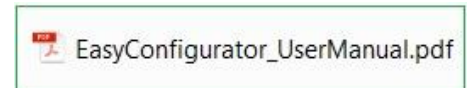It is accessible from the menu "Info → Help"

Fig-7 Doc folder

2 - The EasyCAT board default configuration

The EasyCAT board is shipped configured for 32+32 bytes, I.E. it is capable to exchange, out of the box, 32 bytes in input and 32 bytes in output, with an EtherCAT® master.
The PDO entries, I.E. the 32+32 bytes can be accessed from the user application as follows;

```
EASYCAT.BufferIn.Byte[0] = SomeInValue;          // The application can write the bytes that
EASYCAT.BufferIn.Byte[1] = SomeOtherInValue;     // will be transmitted to the EtherCAT master
.....                                            // in this way.
.....                                            //
.....                                            //
.....
EASYCAT.BufferIn.Byte[31] = TheLastInValue;
```

Fig-8   These are the Inputs, I.E. the data transmitted from the EasyCAT to the master.

```
SomeOutValue = EASYCAT.BufferOut.Byte[0];         // The application can read the bytes that
SomeOtherOutValue = EASYCAT.BufferOut.Byte[1];    // have been received from the EtherCAT master
.....                                             // in this way.
.....                                             //
.....                                             //
.....
TheLastOutValue = EASYCAT.BufferOut.Byte[31];
```

Fig-9   These are the Outputs, I.E. the data transmitted from the master to the EasyCAT.

This is very simple, as everything is already defined and the user has nothing to do other than accessing the input/output variables in his application, but, in some situations, it may be useful to modify this fixed pattern, to better meet our needs.
For instance it could be required to exchange more than 32+32 bytes, or to customize the names and the data types of the variables, in order to use them in a more comfortable way in our application.

To do this, first we have to understand which are the elements that define the EasyCAT configuration.
This is described in the chapter "**3 – How the EasyCAT is configured**" .

3 – How the EasyCAT is configured

3.1 - The ESI .xml file

Any EtherCAT® slave must have an .xml file that describes its resources as defined in the ETG.2000 EtherCAT slave information (ESI) specification document.

This .xml file is provided by the slave manufacturer and it is used by the EtherCAT® master to know how to use the resources available on the slave.

3.2 - The SSI .bin file

The essential informations contained in the .xml file are also written, in binary format, to an EEPROM, present on board of the EasyCAT.

The rules to generate this .bin file are described in the ETG.2010 EtherCAT slave information interface (SSI) document.

The contents of the EEPROM is used at power up, to configure some features of the Asic that implements the communication protocol, and further, an EtherCAT® master can read the EEPROM to retrieve the information it needs, in the event that the .xml file is not available.

3.3 - The include .h file

Also the slave user application, I.E. the firmware that runs on the Arduino or other uP, must be aware of the resources available, and this is done through an include .h file, that is part of the application firmware project.

If all this seems complex is because it is.
But don't worry, the Easy Configurator has been designed to make it easy.

4 – How to change the EasyCAT configuration

4.1 – *Standard Mode* and *Custom Mode*

If you want to change the EasyCAT default configuration there are two different options: the *Standard Mode* and the *Custom Mode.*

The *Standard Mode* is the simplest, but also the less powerful, as it allows you to change only the number of bytes exchanged with the master, choosing between four precooked configurations: 16+16, 32+32 (the shipping default), 64+64 and 128+128.
The second option, the *Custom Mode*, is more powerful as it gives you the freedom to change the number of the variables exchanged with the master as well as their names and data types.

4.2 – Using the *Standard Mode*

If you decide to use the *Standard Mode* all the files that you need are already available, as explained here:

1) The *EasyCAT.xml* file is downloadable from the AB&T website, and it already contains the precooked configurations that allow the EasyCAT to exchange the default 32+32 as well as 16+16, 64+64 and 128+128 bytes.

2) The .bin file is included in the Easy Configurator under the folder *StandardMode_BinFiles* but you have to select the one you need, as each of the precooked configurations has its own .bin file, I.E. :

EasyCAT_16_16.bin   EasyCAT_32_32.bin   EasyCAT_64_64.bin   EasyCAT_128_128.bin

The chosen file must be used to program the EasyCAT EEPROM, and how to do this is explained in chapter "**5 - The Embedded programmer**" .

3) The .h file is the *EasyCAT.h* file, already present in the EasyCAT library, but, to change the 32+32 default configuration to the one requested, you have to do a simple operation: in the slave user application, I.E. the firmware that runs on the Arduino or other uP, define the constant "BYTE_NUM" with the number of bytes required, I.E. 16, 32, 64 or 128.
If "BYTE_NUM" is not defined the default 32+32 byte configuration is applied.

```
#define BYTE_NUM 64              // Standard mode 64+64
                                // The file EasyCAT_64_64.bin must be loaded into the EEPROM

#include "EasyCAT.h"            // EasyCAT library to interface the LAN9252
#include <SPI.h>               // SPI library

EasyCAT EASYCAT;               // EasyCAT istantiation
```

Fig-10   User application configured for *Standard Mode* 64+64

4.2.1 – PDO entries visibility in *Standard Mode*

In *Standard Mode* the PDO entries are accessible from the user applications in the same way as the default configuration, as the default configuration is nothing more than *Standard Mode* 32+32. So, as an example, if you configure the EasyCAT in *Standard mode* 64+64 the PDO entries will visible from the application as follows:

```
EASYCAT.BufferIn.Byte[0] = SomeInValue;          // The application can write the bytes that
EASYCAT.BufferIn.Byte[1] = SomeOtherInValue;     // will be transmitted to the EtherCAT master
.....                                            // in this way.
.....                                            //
.....                                            //
.....
EASYCAT.BufferIn.Byte[63] = TheLastInValue;
```

Fig-11   These are the Inputs in *Standard Mode* 64+64, I.E. the data transmitted from the EasyCAT to   the master.

```
SomeOutValue = EASYCAT.BufferOut.Byte[0];        // The application can read the bytes that
SomeOtherOutValue = EASYCAT.BufferOut.Byte[1];   // have been received from the EtherCAT master
.....                                            // in this way.
.....                                            //
.....                                            //
.....
TheLastOutValue = EASYCAT.BufferOut.Byte[63];
```

Fig-12   These are the Outputs in *Standard Mode* 64+64, I.E. the data transmitted from the master to the EasyCAT.

4.3 – Using the *Custom Mode*

If you decide to use the *Custom Mode* the Easy Configuration will create for you the needed files. You just have to type in a few fields that identify the slave, and the Input and Output PDO entries that define the names and the data types of your custom variables.

4.3.1 – The slave identification fields

These fields identify the board and the manufacturer:

Fig-13   Slave identification fields

Vendor ID   = an Hex number assigned by the ETG.

Vendor Name = the official name of the Company that owns the Vendor ID.

Product Code = an Hex number that identifies the product.

Revision   = an Hex number that identifies the revision level of the product.

Name   = an ASCII string that identifies the product in a readable way.

Please note that, if you use your EasyCAT board for purposes other than educational and experimental, you have to request a Vendor ID to the EtherCAT technology group (ETG) , free of charge, and to be compliant with the ETG rules.

4.3.2 – The Input and Output PDO Entries

Here you have to type in the names and the data types of the variables that will be exchanged between the EasyCAT and the master.

You can type in the names that you like, using the textbox, and associate them to the data type chosen, using the combobox, from the ten available, I.E :

uint8_t	unsigned 8 bit
uint16_t	unsigned 8 bit
uint32_t	unsigned 8 bit
uint64_t	unsigned 8 bit
int8_t	signed 8 bit
int16_t	signed 8 bit
int32_t	signed 8 bit
int64_t	signed 8 bit
float	floating point 32 bit
double	floating point 64 bit

Fig-14   Creating a new entry

Using the "Add" button the new entry will be added to the list.

Proceeding in this way, you can add all the entries that you need, for the inputs as well for the outputs.
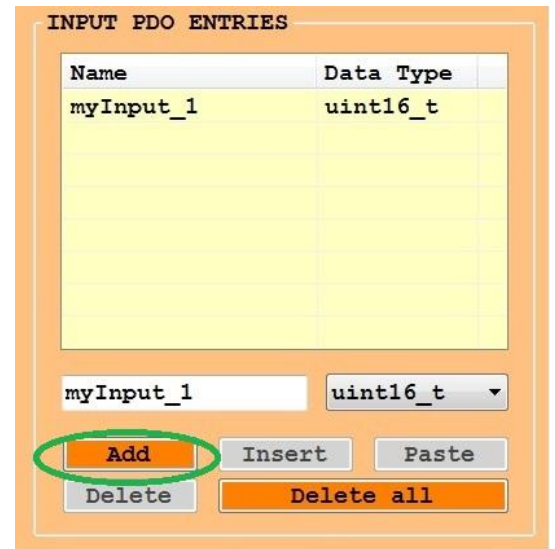

Fig-15   Adding an entry

If you need to edit an entry in the list, first you have to select it with the left button of the mouse, then right click it.
Left click on the "Copy to edit box" option and the entry will be copied in the textbox and in the combobox, highlighted with the green mark.

Here you can edit it and then you have the choice to bring it back in the same position with the "Paste" button, to insert it in the upper position, with the "Insert" button, or to append it to the list, with the "Add" button.
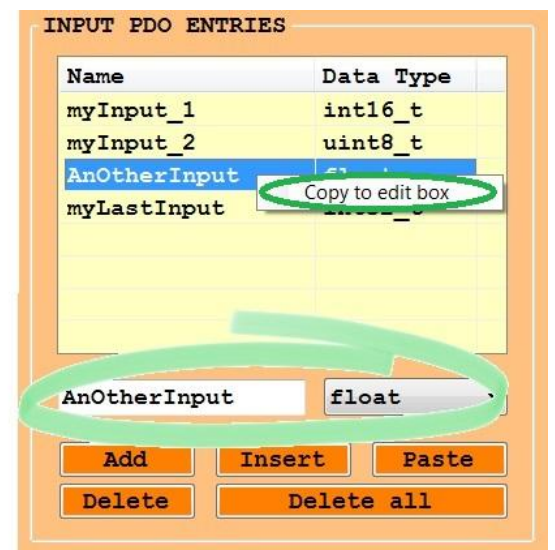
Fig-16   Editing an entry

4.3.3 – **Saving a project**
When you are satisfied with all the data that you typed in, you have to save the project, using the menu "Project → Save as".
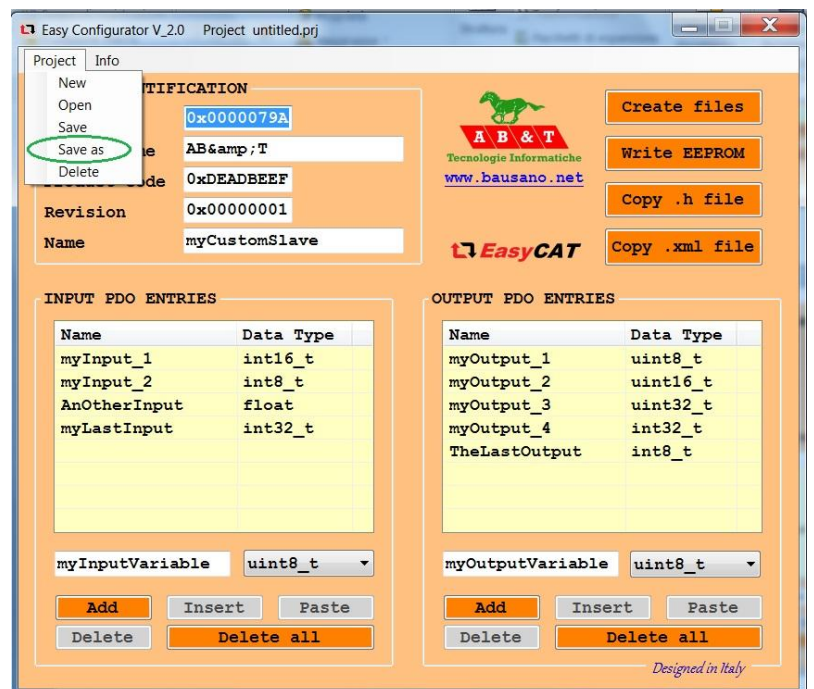

Fig-17   Saving a project

Rename the project with a suitable name, in this example *myCustomSlave.prj* , and save it.

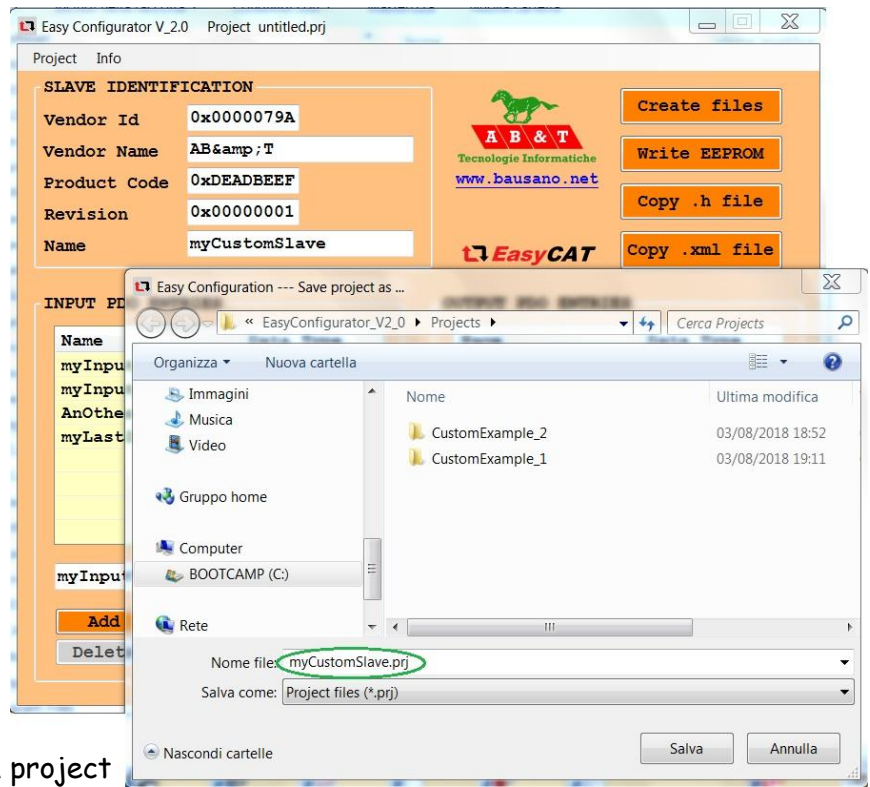It is recommended to save all the projects in the *Projects* folder , besides the examples provided by AB&T.

Fig-18 Renaming and locating a project

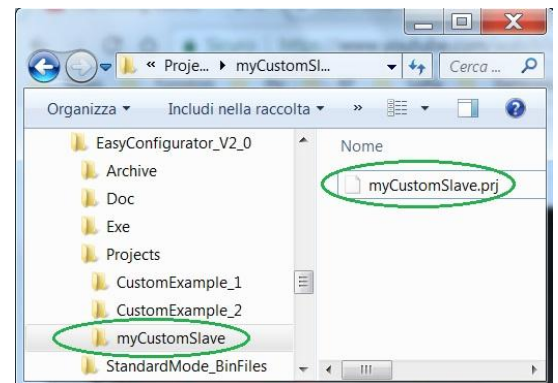A subfolder with the project name will be created and, inside it, the project file, with the .prj extension.

Fig-19 Project created

4.3.4 – Creating the custom files
Now you can create the custom files that will be used to configure your EasyCAT.

To do this click on the button "CreateFiles": a message box will appear confirming that the files :
myCustomSlave.h
myCustomSlave.Xml
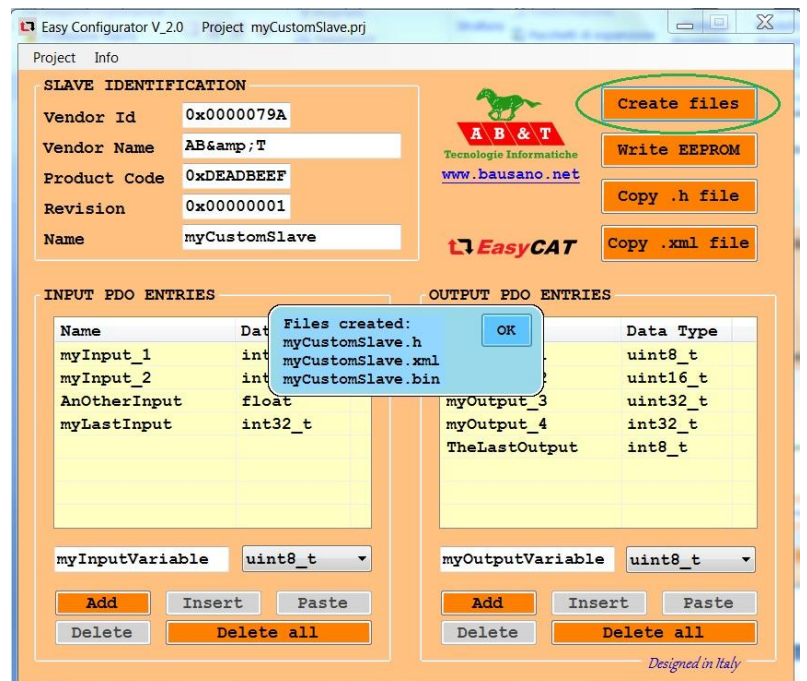*myCustomSlave.bin*
have been created:

Fig-20 Custom files created

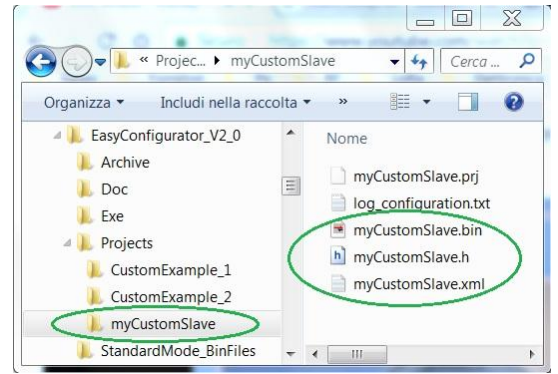The custom files are located in the project folder I.E. , in this example , in the *myCustomSlave* folder.

Fig-21 The custom files

4.3.5 – How to use the custom files

4.3.5.1 – Using the .xml file

The .xml file is used by the EtherCAT® master and so what to do with it depends on the master that you are using. As an example, here we show what you have to do if you are using the Beckhoff Configurator.

The "Copy .xml file" button allows you to copy the file to the clipboard without having to look for it in the hard disk.
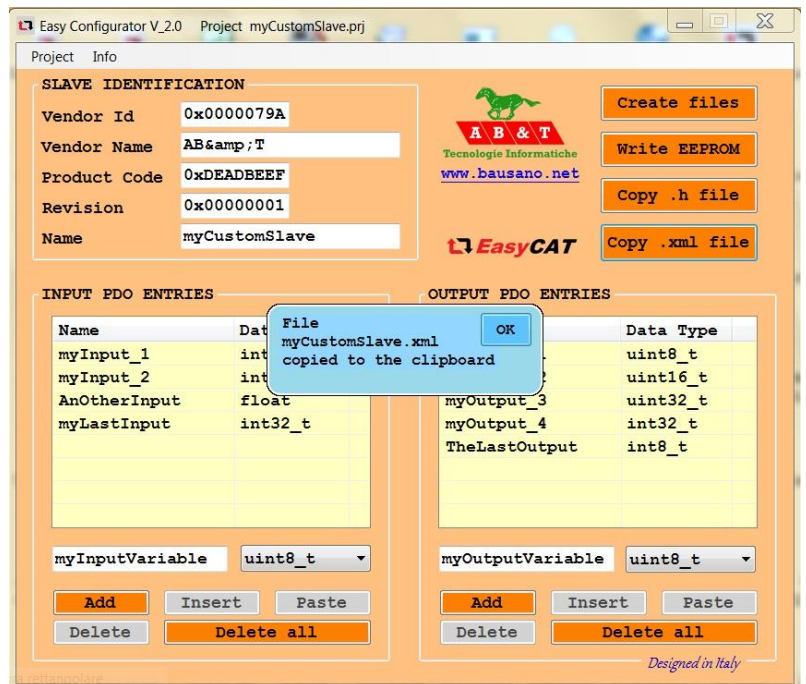

Fig-22 Copying the .xml file

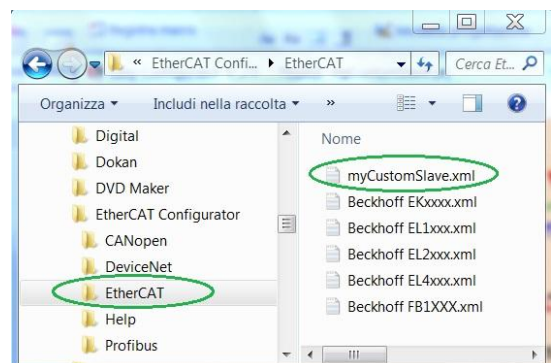Then paste the file in the folder *EtherCAT,* under the Configurator installation root.

Fig-23 Pasting the .xml file

4.3.5.2 – Using the .h files

The .h files must be copied in the EasyCAT user application project, I.E. the project of the firmware that runs on the Arduino or other uP.

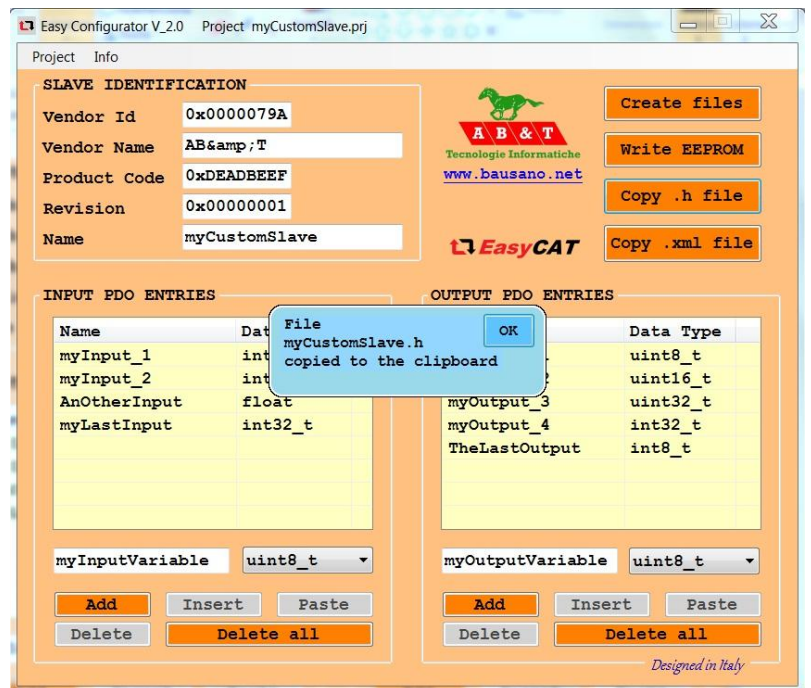You can copy the file to the clipboard clicking on the "Copy .h file" button.

Fig-24 Copying the .h file

If you are using the Arduino Ide the file must be pasted in the project folder, beside the .ino file.
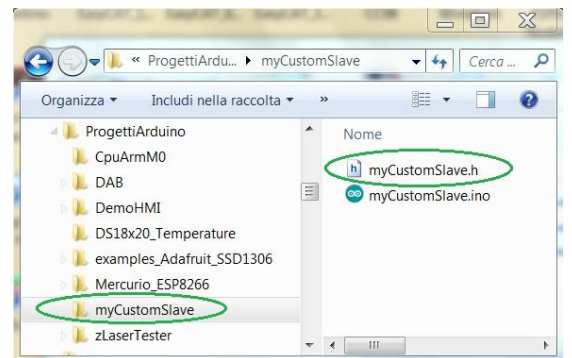

Fig-25 Pasting the .h file

To make it usable from the user application, you must define "CUSTOM" and include your custom .h file before including the *EasyCAT.h* library file.

Fare clic qui per immettere testo.

```
#define CUSTOM                      // Custom mode
#include "myCustomSlave.h";         // This file has been created by the Easy Configurator
                                    // and must be located in the Arduino project folder
                                    // The file myCustomSlave.bin must be loaded into the EEPROM

#include "EasyCAT.h"                // EasyCAT library to interface the LAN9252
#include <SPI.h>                    // SPI library

EasyCAT EASYCAT;                    // EasyCAT istantiation
```

Fig-26   User application configured for *Custom Mode*

4.3.5.3 – Using the .bin files

The .bin file must be downloaded to the EasyCAT EEPROM, using the programmer embedded in the Easy Configurator, as explained in chapter "5 - The Embedded programmer".

4.3.6 – PDO entries visibility in Custom mode

If you have defined the custom PDO entries as beside, they will be accessible from the application as follows:

INPUT PDO ENTRIES		OUTPUT PDO ENTRIES	
Name	Data Type	Name	Data Type
myInput_1	int16_t	myOutput_1	uint8_t
myInput_2	uint8_t	myOutput_2	uint16_t
AnOtherInput	float	myOutput_3	uint32_t
myLastInput	int32_t	myOutput_4	int32_t
		TheLastOutput	int8_t

Fig-27   Custom PDO entries

```
aSigned_16_bit_in     = EASYCAT.BufferIn.Cust.myInput_1;
anUnSigned_8_bit_in   = EASYCAT.BufferIn.Cust.myInput_2;
aFloat_in             = EASYCAT.BufferIn.Cust.AnOtherInput;
aSigned 32_bit_in     = EASYCAT.BufferIn.Cust.myLastInput;
```

Fig-28 These are the Inputs in *Custom mode*, I.E. the data transmitted from the EasyCAT to the master.

```
EASYCAT.BufferOut.Cust.myOutput_1     = anUnSigned_8_bit_out;
EASYCAT.BufferOut.Cust.myOutput_2     = anUnSigned_16_bit_out;
EASYCAT.BufferOut.Cust.myOutput_3     = anUnSigned_32_bit_out;
EASYCAT.BufferOut.Cust.myOutput_4     = aSigned_32_bit_out;
EASYCAT.BufferOut.Cust.TheLastOutput  = aSigned_8_bit_out;
```

Fig-29 These are the Outputs in *Custom mode*, I.E. the data transmitted from the master to the EasyCAT.

5 – The Embedded programmer

As has been said previously, changing the EasyCAT configuration requires also to change the content of the on board EEPROM. To do this you can use the Embedded programmer, included in the Easy Configurator.

Important note – The Embedded programmer has been designed to be used with the AB&T EasyCAT line of boards and it is not suitable for EtherCAT® slaves from other manufacturers.

The EasyCAT board must be powered on and connected directly to the Ethernet interface of your PC, using the RJ45 connector marked "IN".
No other EtherCAT® devices must be present, I.E. the EasyCAT RJ45 "OUT" connector must be left unused. No network switches or other devices must be present between the PC and the EasyCAT.

When everything is ready click on the button "Write EEPROM": a file dialog will be opened.

If you are using the *Standard Mode,* navigate to the folder *StandardMod_BinFiles* and select one of the four .bin files available.
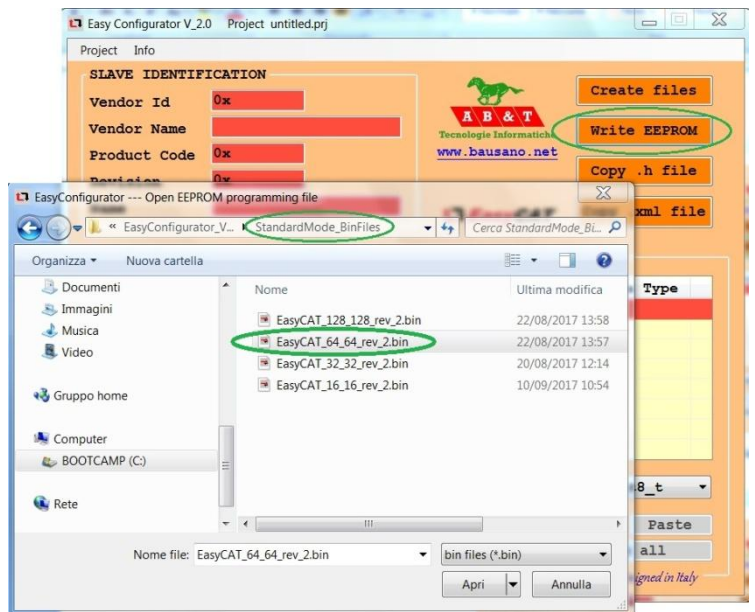

Fig-30 Choosing the programming file in *Standard Mode*

If you are using the *Custom Mode,* select the .bin file, in your project folder, that has been created when the last "Create files" procedure was executed.
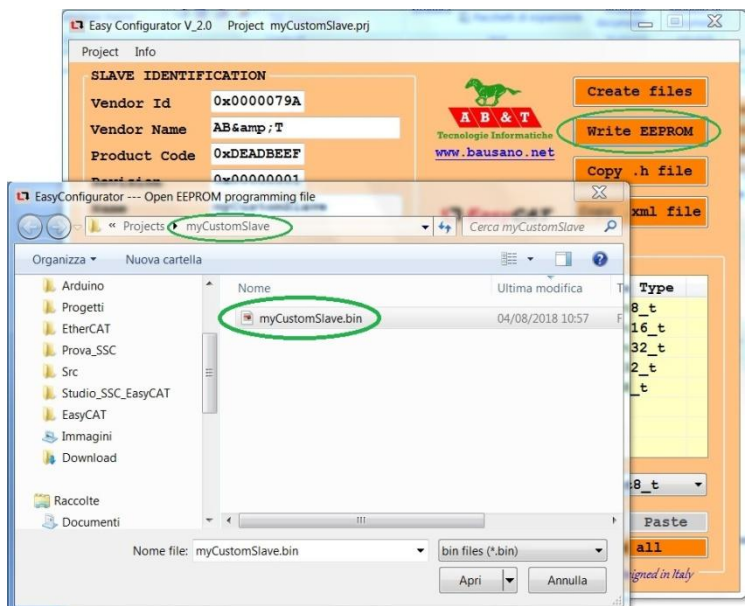

Fig-31   Choosing the programming file in *Custom Mode*

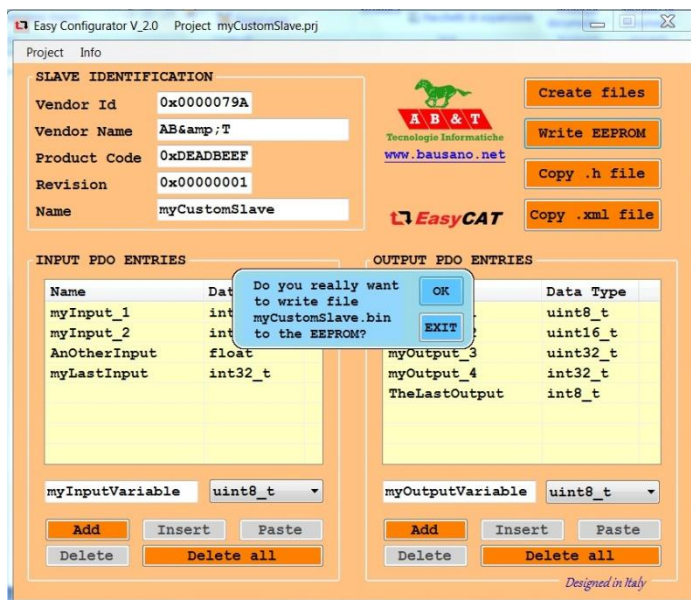Confirm the programming request, clicking "OK" in the message box.

Fig-32 Confirming the programming operation

Wait for the programming to complete: it usually takes about 30 seconds, but this time could be different as it depends on many factors.
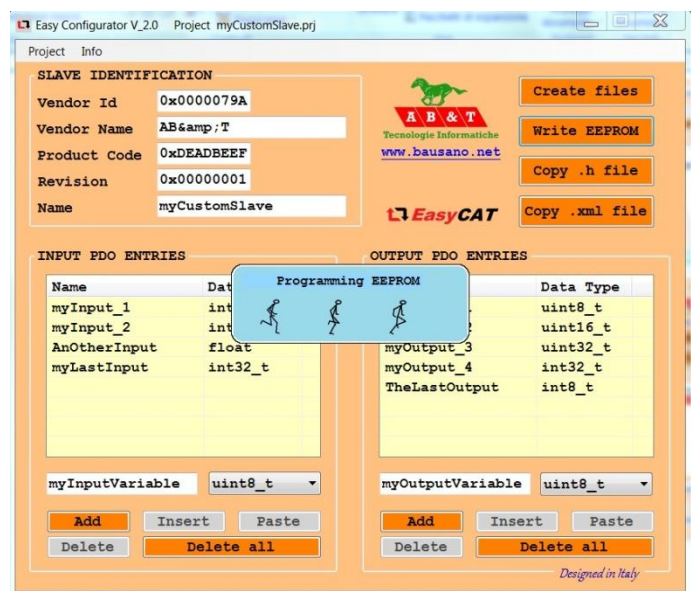

Fig-33 Programming the EEPROM

Your EasyCAT board has been successfully customized.
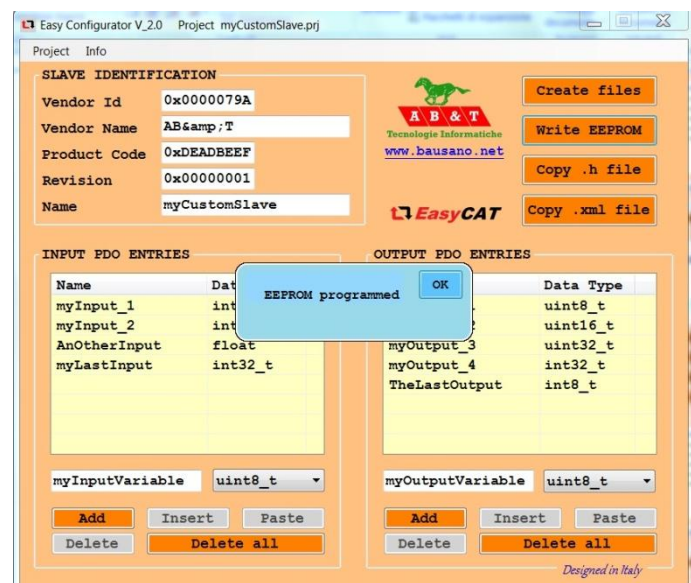It is a good idea to label the board in order to clearly identify it.

Fig-34 EEPROM programmed

------ **Appendix** --

Useful links

AB&T website
https://www.bausano.net/en/

EasyCAT shield page on the AB&T website
https://www.bausano.net/en/hardware/ethercat-e-arduino/easycat.html

WinPCAP website
https://www.winpcap.org/

EtherCAT Technology Group ETG
https://www.ethercat.org/default.htm

EasyCAT shield page on the ETG website
https://www.ethercat.org/en/products/791FFAA126AD43859920EA64384AD4FD.htm