Installation setup, software updates, reference libraries and documentation are also available from the Restricted Area of the website **www.eliwell.it** once you have registered.

ISO 9001

Eliwell Controls Srl
Via dell’ Industria, 15 Z. I. Paludi
32010 Pieve d’ Alpago (BL) - Italy
Telephone +39 (0) 437 986 111
Facsimile +39 (0) 437 989 066
Sales:
+39 (0) 437 986 100 (Italy)
+39 (0) 437 986 200 (other countries)
saleseliwell@invensys.com
Technical helpline: +39 (0) 437 986 250
eliwell.freeway@invensys.com
www.eliwell.it

DISCLAIMER

Eliwell Controls srl declines all responsibility for injury or damage deriving from the installation or use of this software other than for the purpose and in the manner described in this manual.
Every care has been taken in preparing this document; however, Eliwell Controls srl cannot accept liability for any damage resulting from its use.
Within the limits established by law, Eliwell Controls srl shall under no circumstances be held responsible for specific, accidental, direct or indirect damage (including but not restricted to damage resulting from loss or absence of earnings, interruption of activity, loss of data or any other economic loss) deriving from use of or failure to use this software product or from the supply of or failure to supply the associated technical support service, even in cases in which Eliwell Controls srl has been previously notified of the risk of such damage.

9MA10043 12/10

eliwell

free STUDIO

Development software for FREE

eliwell

Studio - Application

IEC 61131-3 development environment

PROGRAMMABLE CONTROLLERS

The software solution in an integrated Suite for quick, easy programming

Quick start Developer

invensys Controls

SUMMARY

INTRODUCTION

GENERAL DESCRIPTION

FREE Studio is a quick, effective development tool for creating and customising new programs for all types of IEC application, which is particularly suited to HVAC/R requirements.
The use of several different programming languages in accordance with **IEC61131-3** regulations (programming standard for industrial control), makes it possible to develop new algorithms or entire programs totally unassisted, which can then be uploaded to

- **FREE Smart** via PC or Multi Function key
- **FREE Evolution** via PC or USB

guaranteeing the utmost confidentiality with appropriate security.

COMPONENTS

All basic components, interfaces, converters and accessories are described below[1].

FREE Studio software

The **FREE Studio**[2] software suite and its functions will be illustrated in this document.
The **FREE Studio** software suite consists of 4 development environments for programming the **FREE Smart & FREE Evolution** programmable controllers:

- **FREE Studio Application**, dedicated to developers, for creating and managing libraries, IEC applications and diagnostics
- **FREE Studio Device**, dedicated to users, for managing previously developed IEC applications, downloading them to the **target**[3] device and modifying device parameters from a serial port.

The following 2 development environments are only available for **FREE Evolution:**

- **FREE Studio Connection**[4], dedicated to software developers, for creating networks
- **FREE Studio User Interface**, dedicated to software developers, for personalising the graphical interface on built-in terminals and remote terminals.

DM Interface (DMI)

Component required only for **FREE Smart:**
Component required for connection to PC
The USB/TTL-I2C hardware interface to be used in association with the software suite enables you to:

- use the software itself.
- connect to the **target** device in order to control it.
- connect to the Multi Function Key (**MFK**) component.

The Multi Function key (MFK)

Component required for **FREE Smart** only:
This is a memory support, which allows you to:

1. update the firmware of the **target** device.
2. update the IEC application of the **target** device.
3. update the parameter values of the **target** device.
4. download the parameter values from the **target** device.

Connection cables

Components required for **FREE Smart** only:

- "Yellow" cable with JST – molex terminals, see Connection Modes chapter for use instructions
- "Blue" cable with JST – JST terminals, see Connection Modes chapter for use instructions
- USB-A/A extension lead, 2 m

Converters

Components required for **FREE Evolution** only:

- USB/485 adapter
- USB/CAN adapter

Note: alternatively if there is a **RS232** serial port, **Evolution** can be connected to the PC using an RS485/RS232 adapter

[1] the interfaces and converters are dedicated to Smart or Evolution.
[2] Software suite or package. See abbreviations and definitions
[3] programmable controller. See abbreviations and definitions
[4] For Evolution only: Connection is the entry point of the software suite - see Target chapter

Abbreviations and definitions

- **Application, Device, Connection, UserInterface**: abbreviations of **FREE (Studio) Application, FREE Device, FREE Connection and FREE UserInterface** respectively. Software suite
- **IEC application**: an application made to **IEC61131-3** (industrial control programming standards) by means of the **Application** development environment (tool) to be downloaded to the **target** using **Application** or **Device**
- **Target device, Target**: name given to the **FREE Smart** or **FREE Evolution** programmable controller or "instrument"
- **DMI**: abbreviation of DM Interface
- **HMI**: acronym of **Human Machine Interface**. Graphical interface developed with **UserInterface**
- **Instance**: object of a predefined object class (function block, **template**, etc)
- **IEC language**: programming language to **IEC61131-3**
- **BIOS menu, BIOS**: factory-set BIOS parameters menu. Not modifiable[5]
- **MFK**: abbreviation of Multi Function Key
- **Quick Start**: this document
- **Smart**: abbreviation of **FREE Smart**; **Evolution**: abbreviation of **FREE Evolution**
- **Studio**: abbreviation of **FREE Studio**. The software suite described in this document
- **Tab**. The work environment is divided into sections or **panels**. Each panel may in turn be subdivided into tabs (e.g. Resources tab)
- **Template**: in this document it denotes a page template created with **UserInterface**
- **Panel** or **section**: see **Tab** (e.g. Project panel)

Note: Many definitions and abbreviations are standard information technology and/or PLC terms and are not listed here For example a Function is a standard term. Other terms, such as **Function Block**, will be described in the relevant paragraphs.

IMPORTANT:
This **Quick Start**:

- is aimed at designers and developers and requires a knowledge of one or more IEC61131-3 standard programming languages.
- is designed to provide a first-level overview of the installation, functions and use of **FREE Studio**
- requires a reading of **FREE Smart** installation manual 9MA**x**00036 and/or **FREE Evolution** manual 9MA**x**00042 (x = 0: IT; 1: EN; 3: ES; 5: DE) available in electronic format only (PDF) from **www.eliwell.it** subject to registration with the restricted area
- is available in electronic format only (PDF) from **www.eliwell.it** subject to registration with the restricted area

SYSTEM RESOURCES

The developer has the following resources available on the **target** devices for IEC programming

	Smart	Evolution
CPU	14.7 MHz	72 MHz, 32MB RAM
Available memory for **Application**	190KByte	1MByte
Available memory for **User Interface**	-	1MByte
FLASH memory data	-	126MByte
RAM Memory*	2300Byte	512KByte
RAM Memory*	1024Byte	500 word
EEPROM variables	1024Byte	4000 word
* automatic mapping ** Modbus mapping		

[5] The BIOS cannot be modified but can be updated.

OPERATING SYSTEMS

- Windows XP Home SP2 or SP3 Italian, English, Spanish and German.
- Windows XP Professional SP2 or SP3 Italian, English, Spanish and German.
- Windows 7 Home Premium Italian, English, Spanish and German.
- Windows 7 Professional Italian, English, Spanish and German.
- Windows 7 Ultimate Italian, English, Spanish and German.[1].

OTHER SOFTWARE

(FREE Smart only)
Driver AT90USBxxx CDC USB to UART MGM for use of the DM interface[2]

See DM Interface Installation

MINIMUM HARDWARE

- 1024x768 graphic resolution
- 700MHz CPU
- 1 GB RAM
- HD with 5GB of disk space
- 1 USB port[3]
- Mouse or other pointing device.
- 1 CD-RW drive

Note: A typical installation (available in English, standard libraries) requires about 500 Mbyte of disk space.
Allow a further 1GByte of disk space for installing libraries and dedicated applications

To download the application, you need the following components:

FREE SMART HARDWARE AND ACCESSORIES

- 1 **FREE Smart SMD5500/C/S** or a device from the **FREE Smart** family
- 1 **DMI 100-3 Manufacturer** + yellow TTL cable[4]
- 1 optional **MFK** + TTL blue cable
- **FREE Smart power cables and transformer**[5]

FREE EVOLUTION HARDWARE AND ACCESSORIES

- 1 **FREE Evolution EVD7500/U** or request the demo case (**Demo Case**)
- 1 **USB/RS485** or **USB/CANOpen** or **plug-in Ethernet converter**
- **35VA** transformer for power supply to **FREE Evolution**

INTERNET

The installation set-up is supplied on CD-RW.
The installation set-up, software updates, libraries and documentation are also available from **www.eliwell.it** subject to registration with the Restricted Area (see Technical Support).
Make sure an Internet connection is available for access to software and manual documentation.

[1] For all: other languages not explicitly tested

[2] available in the installation package

[3] For Evolution it is also possible to use a RS232 port by means of a RS232/RS485 converter
For Smart (/S versions only) it is possible to use the RS232 port by means of RS232/RS485 converter with maximum speed 19200 baud. TTL cannot be used in this case.

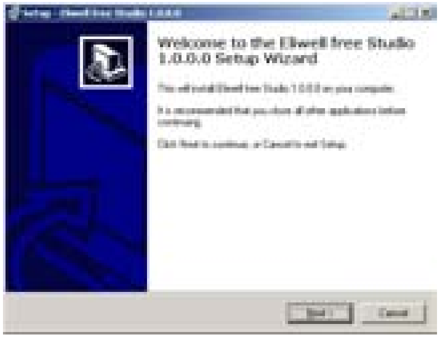
[4] see Connection

[5] FREE Smart does not need to be powered separately if it is connected to a PC by means of DMI. Use yellow TTL cable

INSTALLING STUDIO

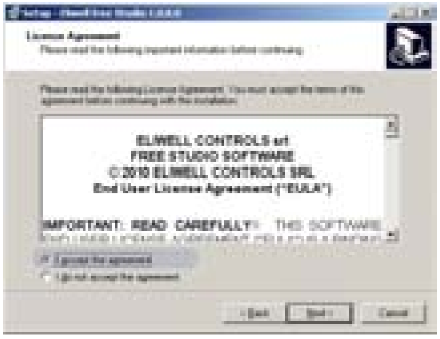
Run **EliwellFREEStudio_xxx.exe**[1] supplied by Eliwell and then follow the installation wizard as indicated in the following screenshots.

1. Click '**Next**'
2. Click '**I accept the agreement**'[2]
3. Click '**Next**'. When installation is complete, the **FREE Studio** icon will appear on the desktop.



Welcome to the Eliwell free Studio 1.0.0.0 Setup Wizard

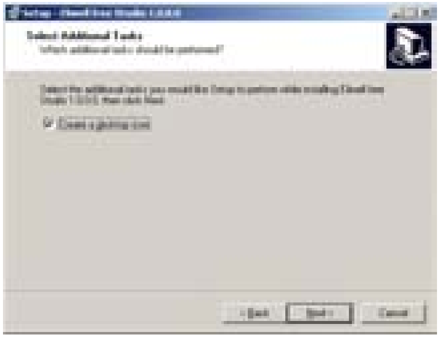
1



ELIWELL CONTROLS srl
FREE STUDIO SOFTWARE
© 2010 ELIWELL CONTROLS SRL
End User License Agreement ("EULA")

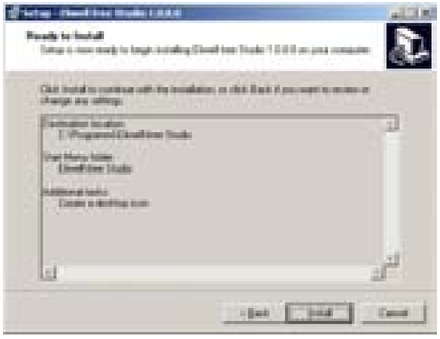
IMPORTANT: READ CAREFULLY!! THIS SOFTWARE

2

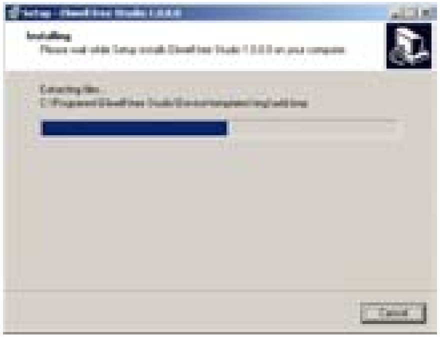


3

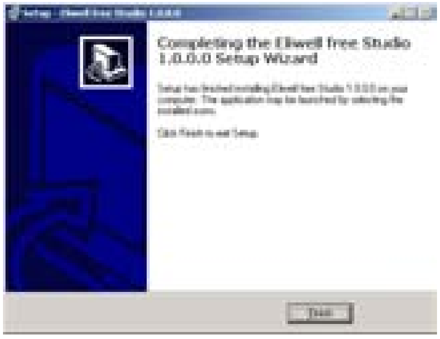
4. Click '**Install**'.
5. Installation in progress
6. Click '**Finish**'.



4



5



Completing the Eliwell free Studio 1.0.0.0 Setup Wizard

6

APPLICATION ICONS

Click the **FREE Studio** icon on the desktop to access a folder containing the following links.

free Smart free Evolution		free Evolution	
icons (links)	description	icons (links)	description
Ap	Application development tool: for Smart & Evolution	Co	Connection development tool for Evolution
De	Device development tool for Smart & Evolution	UI	User Interface development tool for Evolution

[1] Check the software version in advance with Eliwell Technical Support

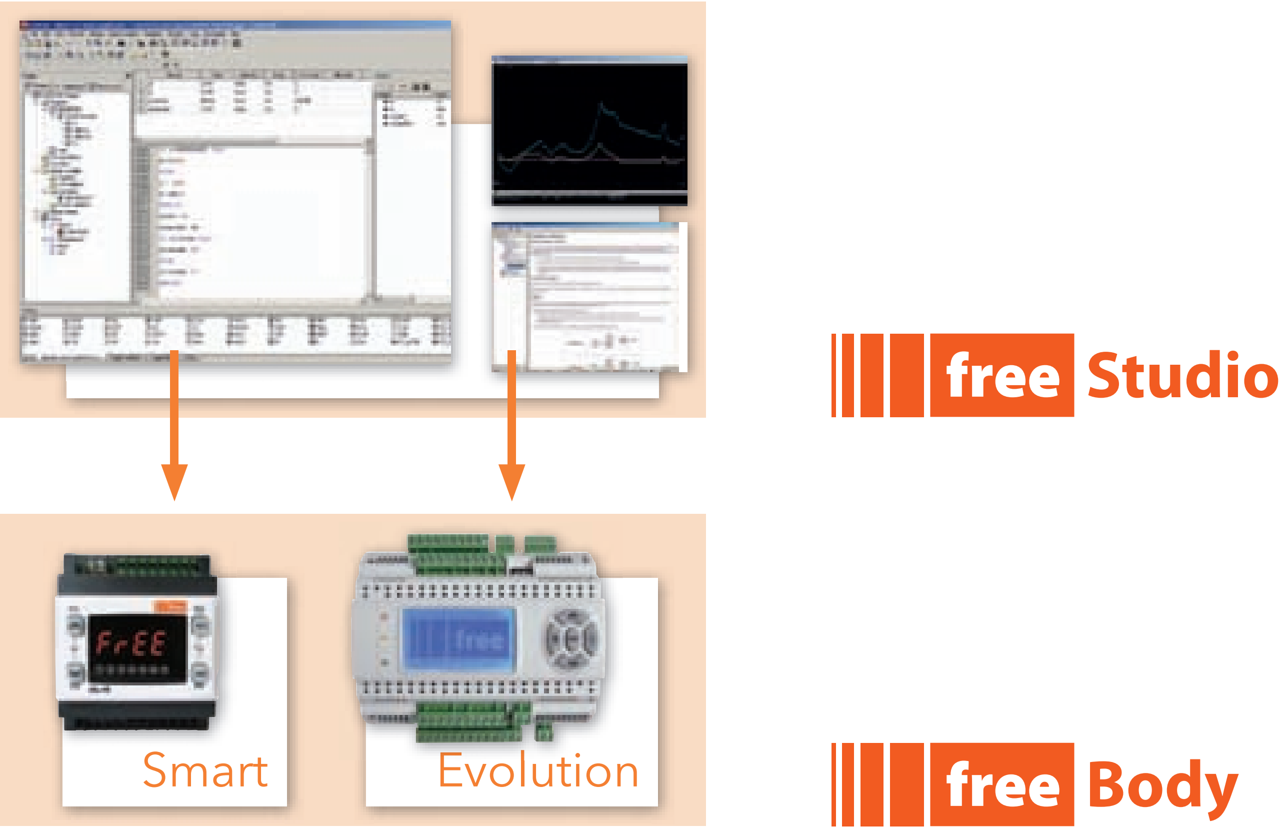
[2] To continue with installation, you must accept the License Agreement. Carefully read the End User License Agreement (EULA) before continuing. The EULA is also available on the website http://www.eliwell.it/content.aspx?id=4533

TARGET

The **Studio** software suite is designed for use with **Smart** and **Evolution target devices.**
The selection of the most appropriate target for the needs of the developer is indicated in the chapter **Defining Hardware Requirements.**

FREE WAY

The **Studio** software suite is unique for **Smart** and **Evolution.**
The entire system is named **FREE Way: FREE Studio** software + **FREE Body** hardware: **Smart** and **Evolution.**



FREE BODY: SMART

The **Smart** developer will only be able to use the **Application** and **Device** work environments
The **Smart** developer will be able to launch **Application** to start programming.
The **Smart** user can use **Device** to download the IEC application previously compiled by the developer.
From **Application** it will always be possible to open **Device** directly without having to launch the program using the **FREE Studio** icon.

FREE BODY: EVOLUTION

The **Evolution** developer will be able to use all work environments.
The **Evolution** developer will be able to launch **Connection** to get started. **Connection** is the entry point for all development activities, from PLC programming to creation of the HMI menu and downloading onto the target using **Device.**
From **Connection** it will always be possible to directly open **User Interface**, **Application** and **Device** without having to launch the relative program using the **FREE Studio** icon.

INSTALLING DM INTERFACE (DMI)[1]

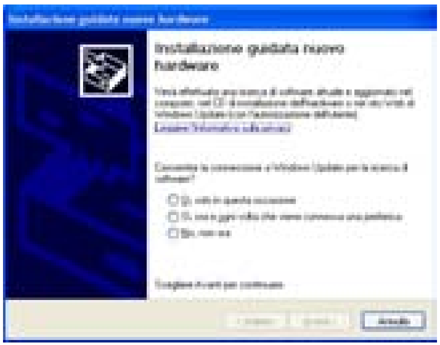
CONNECTING THE DM INTERFACE TO THE PC

Connect the DM interface to one of the USB ports on the PC.
You are advised to use the supplied USB extension lead for the connection.

DM INTERFACE SETUP

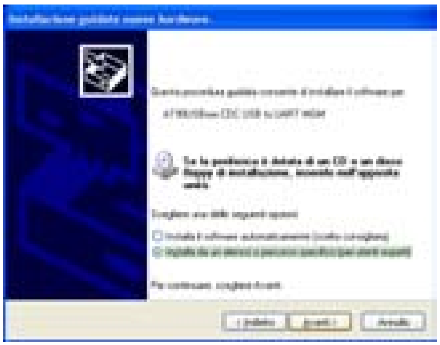
As soon as the DM interface is connected, the Windows XP®[2] operating system installation wizard will start the installation procedure. The steps to be followed are described below.

1. On the initial page, select the "No, not this time" option and click on "Next".
2. Set the second option as indicated and click "Next".
   The path you need to specify is the path of the program installation directory.
3. Click 'Next' and/or search for the indicated path with 'Browse'.



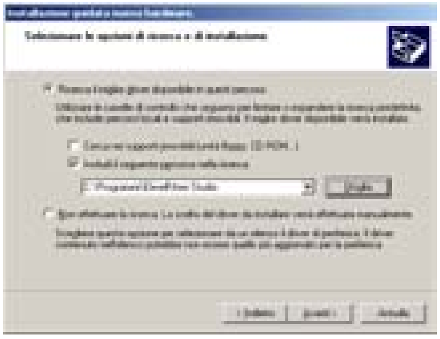
Installazione guidata nuovo hardware

1



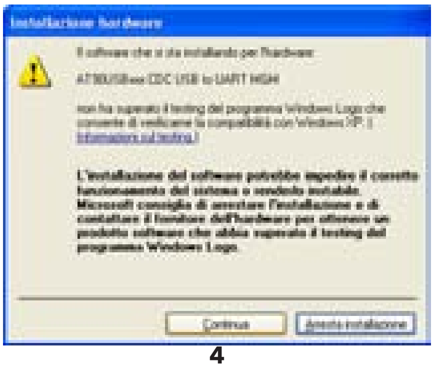
AT90USBxxx CDC USB to UART MGM

2



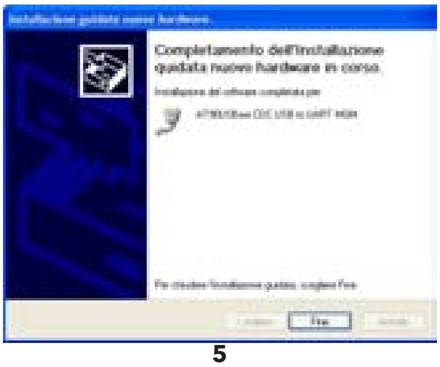
3

4. The operating system will prompt for confirmation to install the drivers.
   Click on "Continue"[3]
5. When the operating system recognises the device (driver) as "AT90USBxxx CDC USB to UART MGM[4]", click on "Finish".



AT90USBxxx CDC USB to UART MGM

4



Completamento dell'installazione guidata nuovo hardware in corso

AT90USBxxx CDC USB to UART MGM

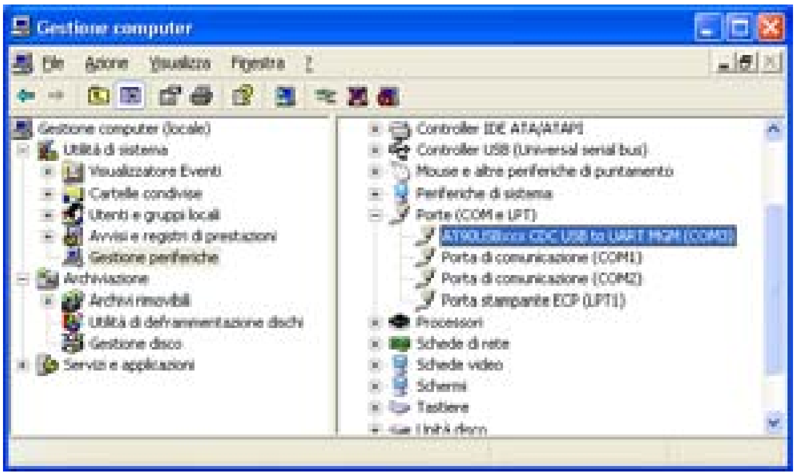
5

[1] This operation must be carried out after installing Studio. Installation required exclusively for Smart
[2] The same installation procedure applies to all other operating systems.
[3] ignore the warning message about compatibility with Windows XP.
[4] the file at90usbxxx_cdc.inf containing the driver is saved to C:\Program Files\Eliwell\FREE Studio

Reading the DM interface COM port

Follow the steps below to check which COM port the operating system has assigned to the DM interface.

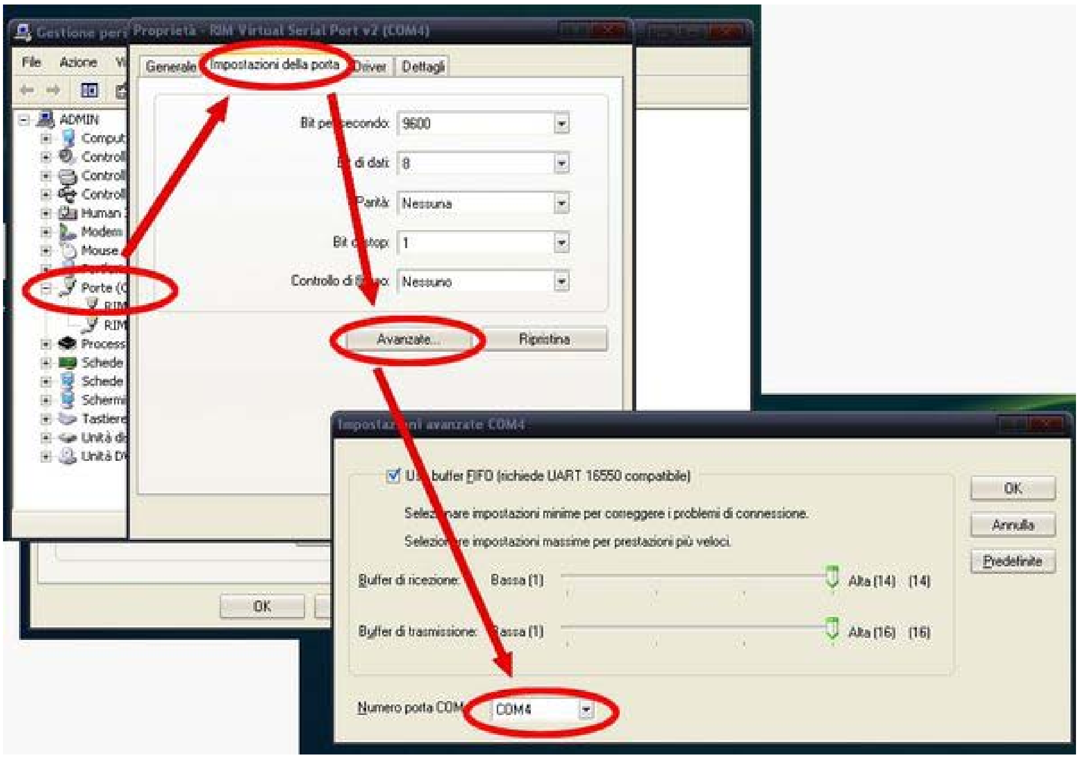
1. Click the right mouse button on the “Resources” icon.
2. Select the "Computer Management" option in the contextual menu.
3. Click on "Peripherals Management" in the left window.
4. Open the “Ports (COM & LPT)” options in the right window.
5. Read the port setting at the end of the “AT90USBxxx CDC USB to UART MGM” string or the string giving the DM interface data.
6. In the example shown in the illustration, the setting is: COM 3.



Changing the COM port setting

If there are contradictions, or for any other reason, another port can be assigned to the interface, as show in the illustration below.

- 1. Right-click on the COM port in question.
- 2. Click on "Properties".
- 3. Select the Port settings tab.
- 4. Click on the Advanced button.
- 5. Change the COM port number as desired.



Proprietà - RIM Virtual Serial Port v2 (COM4)

Bit per secondo: 9600

Bit di dati: 8

Parità: Nessuna

Bit di stop: 1

Controllo di flusso: Nessuno

Impostazioni avanzate COM4

☑ Usa buffer FIFO (richiede UART 16550 compatibile)

Selezionare impostazioni minime per correggere i problemi di connessione.

Selezionare impostazioni massime per prestazioni più veloci.

Buffer di ricezione: Bassa (1) Alta (14) (14)

Buffer di trasmissione: Bassa (1) Alta (16) (16)

Numero porta COM: COM4

SMALL APPLICATIONS

FREE Smart was designed for small plants and in particular for

Manufacturers of:

- A.T.U.s (Air Treatment Units)
- Chillers & Heat Pumps
- Rooftops
- Compressor Rooms

Installers/integrators of:

- Air systems
- Hydronic systems
- Combi systems (air/water)

FREE SMART SELECTION CRITERIA

The **FREE Smart** models are available in both a DIN rail-mounted version, which significantly reduces wiring times, or in the regular Eliwell 32x74 format for panel-mounting.

FREE Smart SMP has 2 models, giving you 6 digital inputs, up to 5 relay outputs, a TRIAC output, 2 PWM analogue outputs, 3 configurable 0…10V/0…20mA/4…20mA analogue outputs and an Open Collector digital output for an external relay.

The standard Eliwell 32x74mm format ensures versatility and ease of installation.

FREE Smart SMD – SMC comes in a variety of models, giving you 6 digital inputs, up to 5 relay outputs, up to 2 TRIAC outputs, up to 2 PWM analogue outputs, up to 3 configurable 0…10V/0…20mA/4…20mA analogue outputs and up to 2 Open Collector digital outputs for an external relay.

The 4DIN format guarantees maximum flexibility and easy installation.

All inputs and outputs are independent and configurable, meaning they can be adapted to fit any system.
It runs on 12-24V~ or 12-24V~/24V⎓ power supplies.

For further details, see the **FREE Smart** installation manual.

CONTINUITY WITH ENERGY FLEX[1]

FREE Smart inherits from the **Energy Flex** series the characteristics of the I/O and the electrical connections of the various models, to be combined with the corresponding expansions and keypads. The two platforms also use the same accessories.

FREE Smart also inherits the interface with Modbus RTU serial communication standard from the Flex range and the option of downloading parameter maps and applications via **MFK**.

[1] **Note that continuity with Energy Flex DOES NOT include the applications.
The applications (e.g. Air Treatment Units) are developed by Eliwell and available for download from the site.
Visit the site and/or contact Technical Support for further details.**

SELECTING AN INTERFACE

FREE Smart is available in 3 versions: SMP for panel mounting and SMD/SMC for DIN rail mounting.

SMP 32x74

SMD 4DIN with display

SMC 4DIN without display

The IEC application created by **Application** can be loaded on every version.
Menu navigation is identical for all models.
The SMP and SMD models have a 7-segment, 4-digit LED display.
The SMC model has no display so you need a remote terminal for configuration in the field.
For a description of the keys, user interface and connections, refer to installation manual **9MAx0036**[2]

SELECTING OPTIONAL MODULES

Some of the accessories used by **Smart** are listed below:

SKP 10 remote terminal 32x74

SKW22 remote terminal with LCD display without backlighting

SKW22L remote terminal with LCD display with backlighting

SKP22 remote terminal with LCD display without backlighting

SKP22L remote terminal with LCD display with backlighting

expansion modules (various models available) SE600 4DIN without display

[2] (x = 0: IT; 1: EN; 3: ES; 5: DE)

- The SKP 10 remote terminal for panel-mounting has the same interface as the main **Smart** module.
- The SKW22 remote terminal for wall-mounting has a dedicated interface with LCD display with or without backlighting. For further details, see installation manual **8MAx0218**[3]
- The SKP22 remote terminal for panel-mounting has a dedicated interface with LCD display with or without backlighting. For further details, see installation manual **8MAx0218**[4]
- The SE600 expansion modules - available in various models - have I/O similar to those of **SMD**

For details of connections and further modules and accessories, see installation manual **9MAx0036**[5]

COMPLEX APPLICATIONS

FREE Evolution was designed for medium and large plants and in particular for

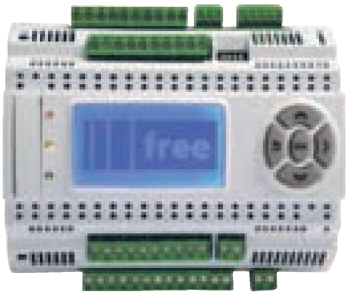
Manufacturers of:

- A.T.U.s (Air Treatment Units)
- Chillers & Heat Pumps
- Rooftops
- Compressor Rooms

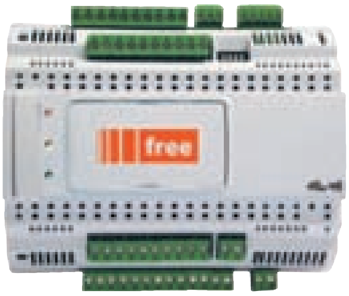
Installers/integrators of:

- Air systems
- Hydronic systems
- Combi systems (air/water)

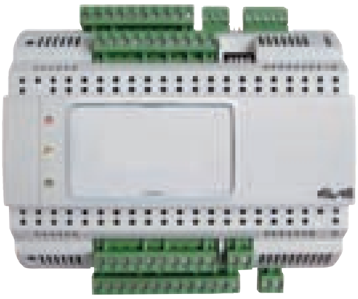
FREE EVOLUTION SELECTION CRITERIA



EVD



EVC



EVE expansion module

FREE Evolution models are available in the 8 DIN rail-mounted version, with display (**EVD** model) or without display (**EVC** model), both supplied with removable screw terminals for quick and easy installation.

FREE Evolution EVD/EVC comes in a variety of models based on which you can mount 9 digital inputs, up to 7 relay outputs, 5 configurable 0…10V/4…20mA analogue outputs

The **/U** models are also equipped as standard with one USB port and one mini USB port.

The FREE Evolution EVE expansion module, without display, has the same format and the same I/O as the main module

The 8DIN format guarantees maximum flexibility and easy installation.

All inputs and outputs are independent and configurable, meaning they can be adapted to fit any system.
24V~/⎓ or 48Vc power supply

For further details, see the **FREE Evolution installation manual. 9MAx0042**[6]

3-4-5 **(x = 0: IT; 1: EN; 3: ES; 5: DE)**

CONNECTIVITY

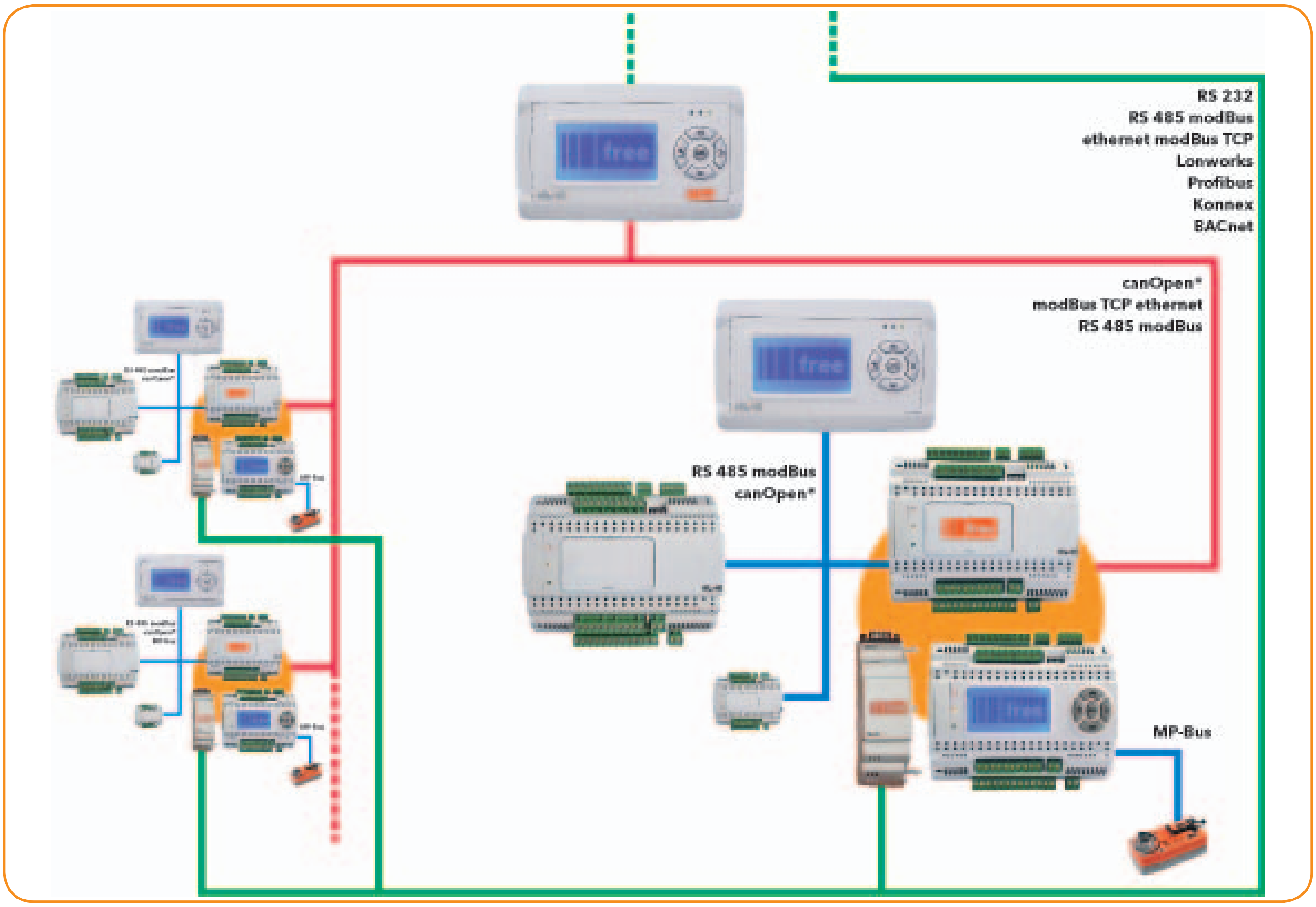
The essential feature of **Evolution** is its versatile **connectivity**
The **EVD/EVC** models are all available as standard with CANOpen.
Integrated RS485 or MPBUS serials are also available as standard, depending on the model.
Each **EVC/EVC** model can be connected to 12 **EVE** expansion modules via CANOpen or RS485 serial.
Integration into industrial systems, BMS and Ethernet networks is permitted thanks to the plug-in modules, available in 2DIN format, that connect quickly and intuitively to the main modules.
Each **Evolution** can be connected to up to two remote terminals to view the integrated machine display. Using the RS485 network, a **Smart** system acting as a slave can be connected in a network in which **Evolution** acts as master.

EVK terminal

EVK/N terminal

EVS plug-in

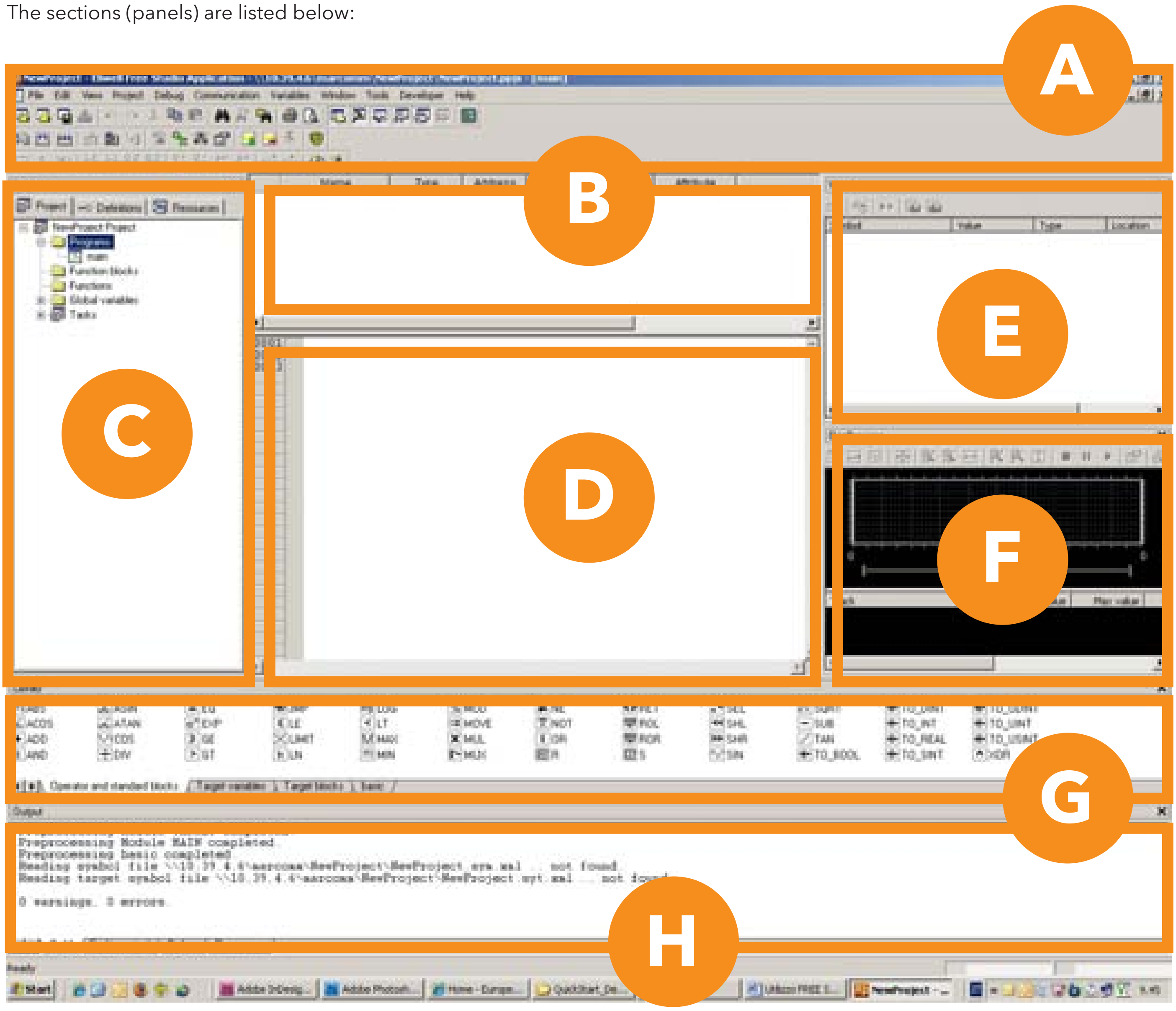
The **EVK** terminal is available with CANOpen as standard. The **EVK** terminal can read menus from **EVD/EVC**
The **EVK/N** terminal is available with CANOpen, RS485 and ETHERNET as standard. The **EVK** terminal is fully programmable using the dedicated menu
Integrated RS485 or MPBUS are also available as standard, depending on the model.



APPLICATION PANEL DEFINITIONS

The **Application** work environment has various sections for developing the IEC application, e.g. programming in IEC 61131-3 compatible language, testing, debugging and IEC application downloading to the target device.
N.B.: Application only downloads the IEC application.
Parameters (BIOS and PLC) and their default values are downloaded by Device and not by Application.

The sections (panels) are listed below:



```
Preprocessing Module MAIN completed
Preprocessing basic completed
Reading symbol file \\10.39.4.4\aarcoma\NewProject\NewProject.sym.xml ... not found
Reading target symbol file \\10.39.4.4\aarcoma\NewProject\NewProject.syt.xml ... not found

0 warnings, 0 errors
```

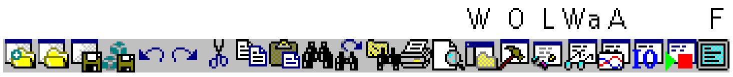
Section	Panel /Section name Application	Panel	Description
A	/	Toolbars	This section shows the work tools in the form of icons. To display or hide the toolbars, simply right-click and select or deselect the desired tools. Some tools can be selected by means of key combinations (**shortcuts**). See relevant chapter for details
B	/	Variables	The global and local variables of the code displayed in section **D** '**Edit**' (programs, function blocks and functions) appear in this section

Section	Panel /Section name Application	Panel	Description
C	**Project**	**Project**	The Project panel is divided into 3 **tabs**: • **Project** - for managing the application code • **Definitions** - for managing and defining complex variables defined by the developer • **Resources** - for managing the target device menu
D	**/**	**Edit**	This section enables you to write the program
E	**Watch**	**Display variables with application running**	This panel enables you to manage variables debugging by displaying their status in numerical format when the application is running and connected to the target device
F	**Oscilloscope**	**Display variables chart**	Like the **Watch** panel, this panel enables you to manage variables debugging by displaying their status in graph format when the application is running and connected to the target device
G	**Library**	**Functions library**	This tool enables you to manage default functions libraries or functions libraries created by the developer. The panel is divided into various **tabs**, one for each library. The tabs relating to the following are always available: • **Operator and standard blocks**: standard operators (AND, OR, etc) • **Target variables**: specific variables of the target **device** • **Target blocks**: specific functions of the target **device** Any additional tabs are managed by means of the drop-down menu **Project > Library manager**
H	**Output**	**Messages**	This panel shows the messages relating to the development of the project (file opening, compilation errors, status of connection to target device, etc.) **Note**: the connection to the target device is also visible in the bottom right of the screen by means of coloured messages (see Application Download)
The panels appear or remain hidden (see **Shortcuts**) and can be organised as the user wants in terms of size and layout. **Studio** saves the last settings made.			

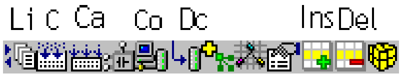
ICONS

The work tools are available via the drop-down menus or **shortcuts.**
Panel A shows a series of icons that help the developer to create the application.
The toolbars are displayed via the menu **View > Toolbars**
Below are the most frequently used icons in the **Main** and **Project toolbars.**

Main Toolbar



Project Toolbar

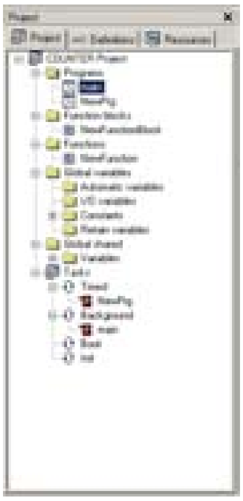


The icons are self-explanatory. Apart from the first few icons, which are very similar to standard icons (Open project, Save project, Copy, Paste etc.) there are icons with letters above them:

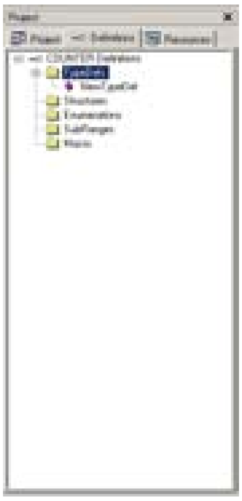
	Toolbar	Function	Function	Shortcut
F	Main	Full screen	Full Screen	CTRL+U
W	Main	workspace	WORKSPACE	CTRL+W
O	Main	show / hide message panel	OUTPUT	CTRL+R
L	Main	show / hide library panel	LIBRARY	CTRL+L
Wa	Main	show / hide watch window	WATCH WINDOW	CTRL+T
A	Main	show / hide asynch graphic window	ASYNCH GRAPHIC WINDOW	CTRL+K
C	PROJECT	COMPILE only selected project	COMPILE	F7
Ca	PROJECT	COMPILE ALL	COMPILE ALL	-
Co	PROJECT	CONNECTS TO THE TARGET	CONNECTS TO THE TARGET	F5
Dc	PROJECT	Download code	DOWNLOAD CODE	-
Ins	PROJECT	INSERT RECORD (variable, constant, alarm etc.)	INSERT RECORD	-
Del	PROJECT	DELETE RECORD	DELETE RECORD	-

Project panel

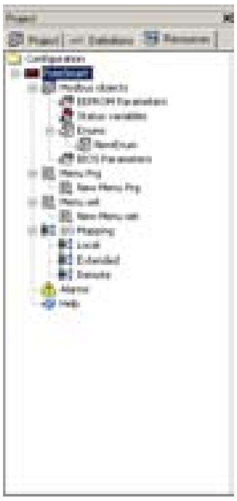
Detailed display of **Panel C / Project**[1]



Project tab



Definitions tab



Resources tab

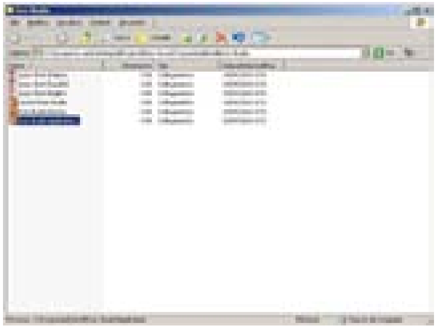
[1] view for Smart

free Smart

Ap

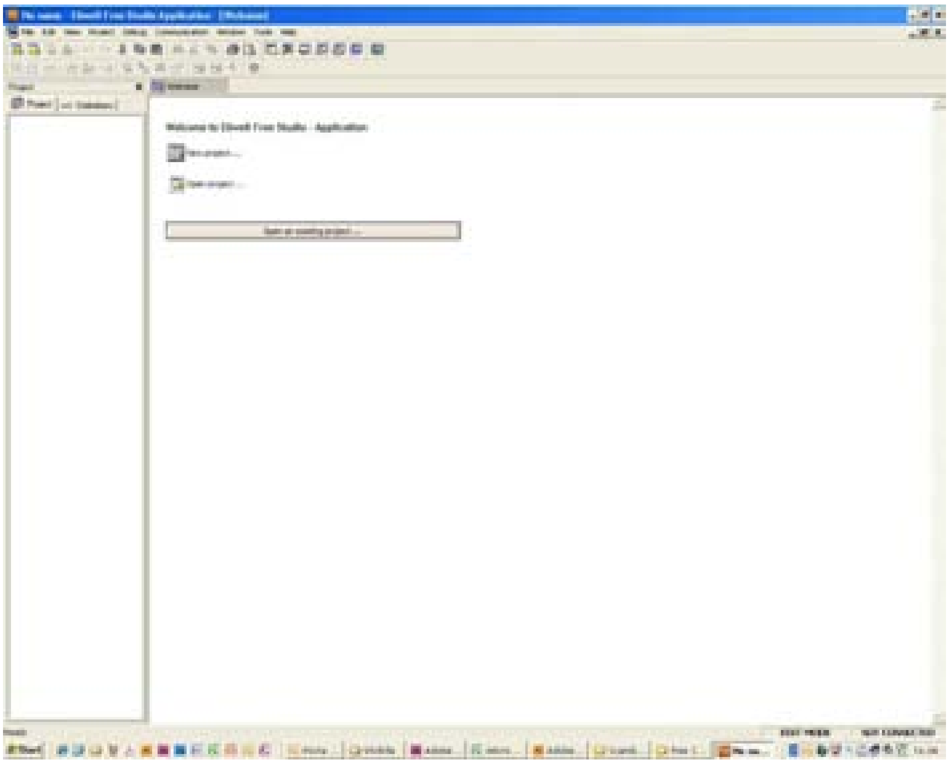
FIRST START-UP[1]

1. Click on the **Launch FREE Studio** icon on your Desktop
2. Click on the **FREE Studio Application** icon.



1 2

The program appears as shown in the picture:


Welcome to Eliwell Free Studio - Application
New project ...
Open project ...
Open an existing project ...

You can now:

- Create a new project '**New Project ...**'
- Open an existing project ' **Open project ...**'
- Open recent projects '**Open an existing project ...**' by selecting from a list

Existing projects have the extension **.ppjs**

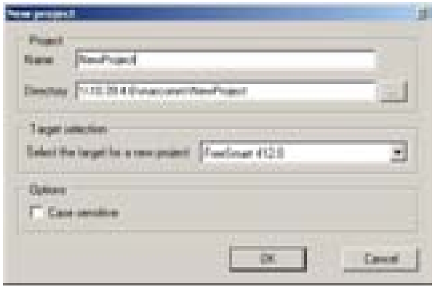
The IEC application is made up of one **project** and one or more **programs** written in IEC code

[1] **The 'free Smart' logo at the start of this chapter indicates that the Application tool is available for Smart. All tools (including Application) are available for Evolution. For Evolution you are advised - for First Start-up and Creation of a new program - to always start from the Connection tool (see relevant chapter).**

CREATING A NEW PROJECT

- Click **New Project ...** or
- select **File > New Project** from the drop-down menu[2]

The dialogue box shown below appears, in which you can create, name and save a new project:



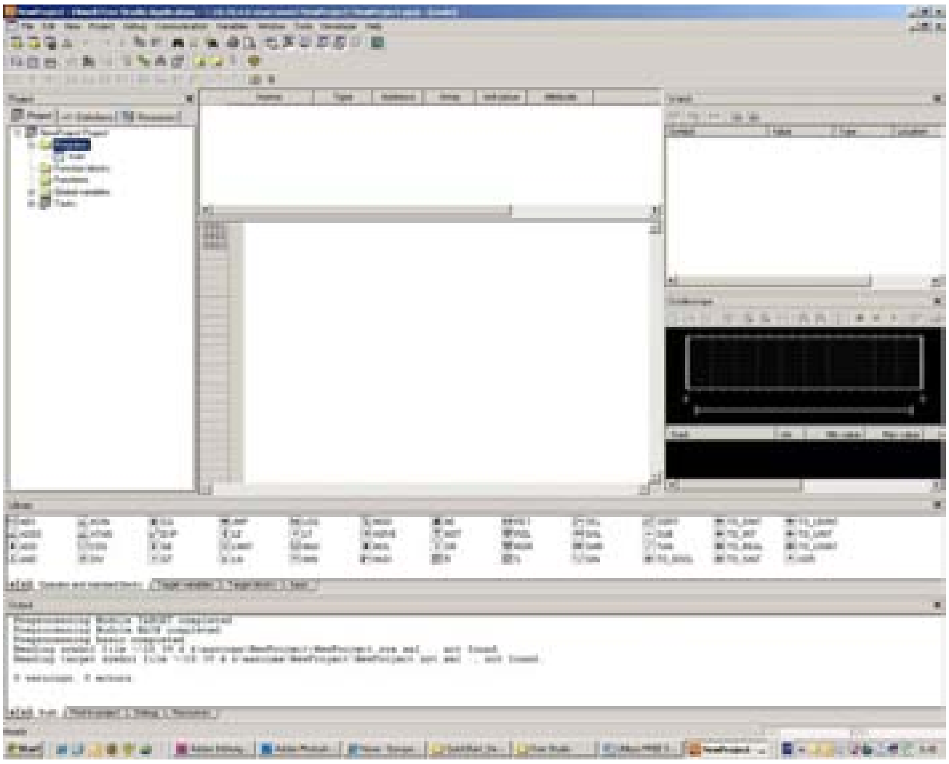
Name: NewProject

Directory: [illegible]

Select the target for a new project: FreeSmart 412.0

New project

The new project appears as shown in the figure:



- For a description of the icons, see **Shortcuts**
- A description of the programming languages is available in the **Online Help** - see 'Support'
- The panels in the standard development environment are described in the next chapter

The pages that follow explain how to create, develop and test a new application, with simple examples of libraries, function blocks and elementary programs.

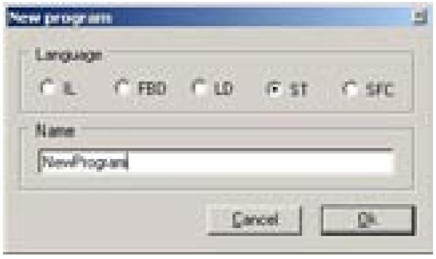
[2] **The figure shows an example of a project for Smart. The same procedure applies for Evolution. Exceptions will be pointed out as appropriate. However you are advised in this case to always start from the Connection tool and to follow the procedure described in the relevant chapter.**

CREATING A NEW PROGRAM[3]

Project Panel / Programs Folder.
Right-click and select **New Program**.
Every project contains by default the program **main**

SELECTING THE IEC LANGUAGE

The following dialogue box appears:



New program

Language

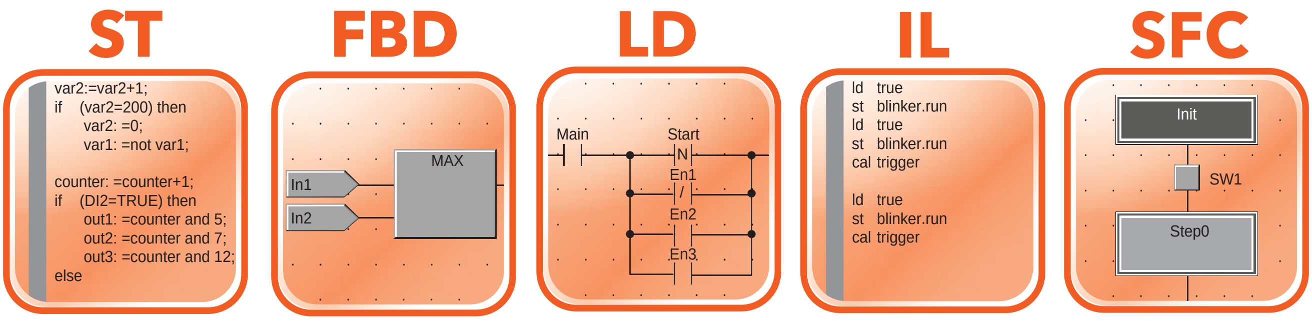
○ IL ○ FBD ○ LD ● ST ○ SFC

Name

NewProgram

New program in ST language

Select the desired programming language and enter a name.
The **FREE STUDIO** platform is compatible with all 5 standard programming languages (**IEC 61131-3**).



```
var2:=var2+1;
if   (var2=200) then
      var2: =0;
      var1: =not var1;

counter: =counter+1;
if   (DI2=TRUE) then
      out1: =counter and 5;
      out2: =counter and 7;
      out3: =counter and 12;
else
```

```
ld   true
st   blinker.run
ld   true
st   blinker.run
cal  trigger

ld   true
st   blinker.run
cal  trigger
```

Each project may consist of several programs.
The developer may use one or more languages in the same project. Each new program actually offers the choice of 5 programming languages, 2 text-based and 3 graphics-based:

• **ST**, **Structured Text** language	text
• **FBD**, **Functional Block Diagram** language	graphical
• **LD**, **Ladder** language	graphical
• **IL**, **Instruction List** language	text
• **SFC**, **Sequential Function Chart** language	graphical

OPENING AN EXISTING PROGRAM

If the developer already has an existing IEC application, simply open a **.ppjs** project from the menu **File > Open Project** or select it from the list.

[3] Creating a new program

WRITING A PROGRAM

In the **Edit** panel, write the program that you intend to develop.
The elements of the development tools are presented generically below.
The appendix give some examples of simple programs which you can use to create a basic, functioning project.

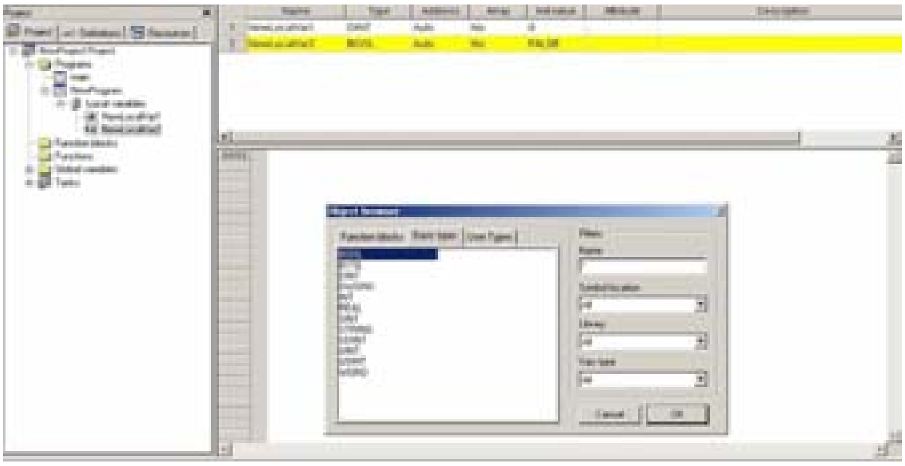
PROJECT PANEL / PROJECT TAB

Creating local variables

Using the drop-down menu, select **Variables > Insert** (new record) or click on the respective icon
The variable appears in yellow in the central section, and you can define its characteristics by clicking on the respective boxes.
Where multiple choices exist, you can choose from a set of options by clicking on the three dots (…).
A dialogue box will open

- Name
- Type, to choose from one of the preset options or variables defined by the user (see picture)
- Address: the default setting is automatic
- Array: defines whether the variable is array type (if so, define its dimension) or not
- Init value: initial value
- Description

In the navigation tree below the program folder, the local variables will appear, identified by an icon
Note: Local variables are **NOT** static



Creating a function block / function

In the same way as we create a program, it is possible to create

- a **function**
- a **function block**

in the desired programming language.
Note: Creation of a function is available in 4 programming languages. **SFC** language is not supported for functions.

A function or function block is a (sub)program with inputs and outputs:

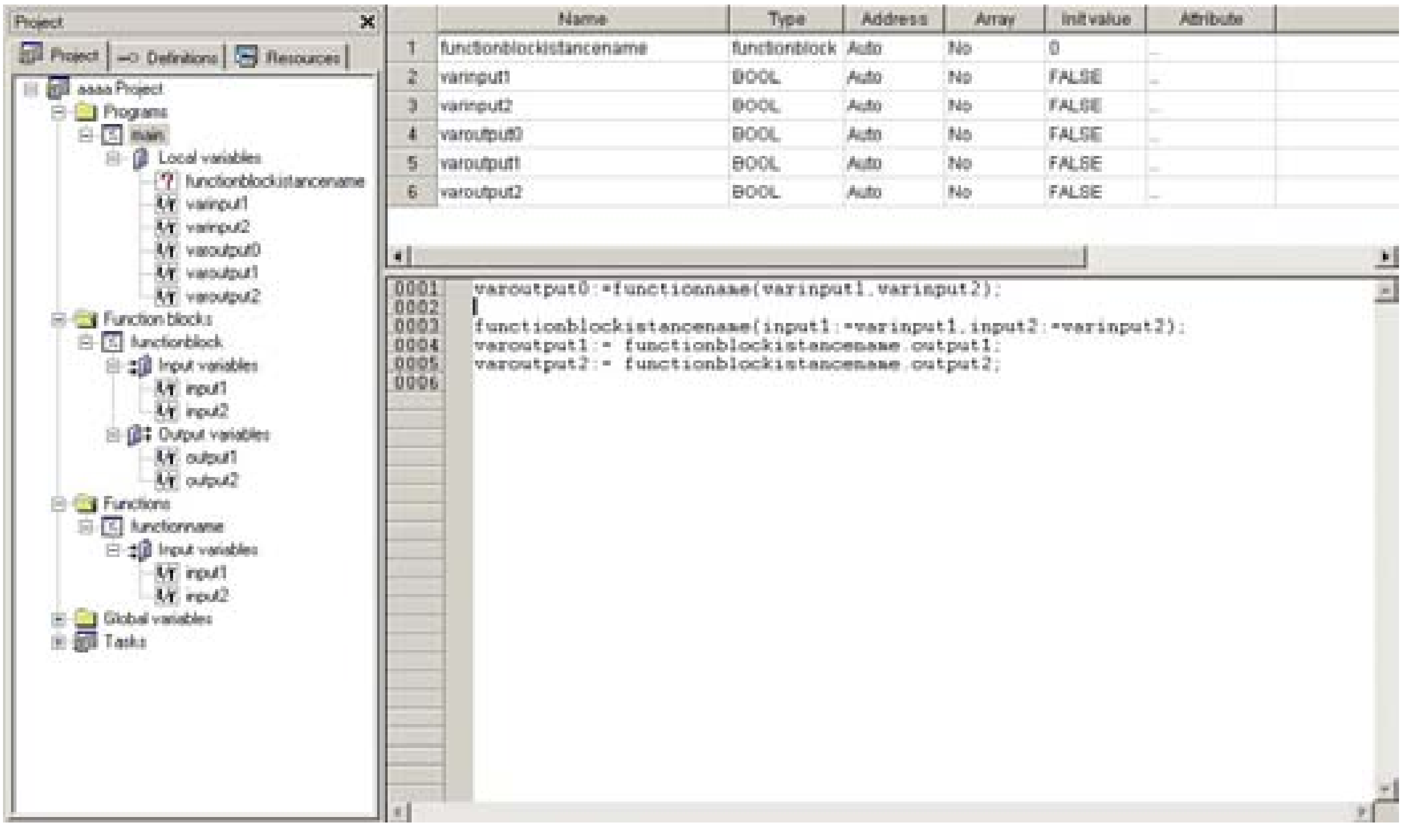
- a **function** requires **n** inputs and a single output (**RESULT)** with the same name as the function. The function's local memory is initialised each time the function is called
- a **function block** requires **n** inputs and **m** outputs. The local memory of each instance of the function block is kept between one call and the next (static memory)

Each function or function block can be used within a program simply by dragging and dropping the icon in the **Edit** section of the program.

The function is used within the program by passing the input variables.

The function block is used within the program as an **instance** in the same way as the declaration of a variable.

Example in ST language of call to function and function block



	Name	Type	Address	Array	Init value	Attribute
1	functionblockistancename	functionblock	Auto	No	0	...
2	varinput1	BOOL	Auto	No	FALSE	...
3	varinput2	BOOL	Auto	No	FALSE	...
4	varoutput0	BOOL	Auto	No	FALSE	...
5	varoutput1	BOOL	Auto	No	FALSE	...
6	varoutput2	BOOL	Auto	No	FALSE	...

```
varoutput0:=functionname(varinput1,varinput2);

functionblockistancename(input1:=varinput1,input2:=varinput2);
varoutput1:= functionblockistancename.output1;
varoutput2:= functionblockistancename.output2;
```

Creating global variables

The navigation tree contains the folder **Global Variables**
From here you can create new

- Global variables
- Mapped variables
- Constants
- Retain variables[4]

Shared global resources

The shared global resources will appear in this section but will be defined in the **Resources panel**

[4] not currently supported on Smart and Evolution targets.

ASSOCIATING A PROGRAM TO A TASK

For a program to run, it must be associated to a **task**.
There are various types of task:

- **BOOT** Task runs once only at system start-up
- **Init**. Task runs each time the application is downloaded
- **Timed**. Task runs at regular intervals which can be set by the developer. The default setting is every 100ms
- **Background.** Task runs with low priority after the **Timed** tasks (between the end of one **Timed** task and the start of the next).
  Note Each new project has the **main** program associated to the background task (the **main** program can still be eliminated and/or associated to other tasks).

To activate a task, go to the task you want, right-click and select **Add program**

PROJECT PANEL / RESOURCES TAB

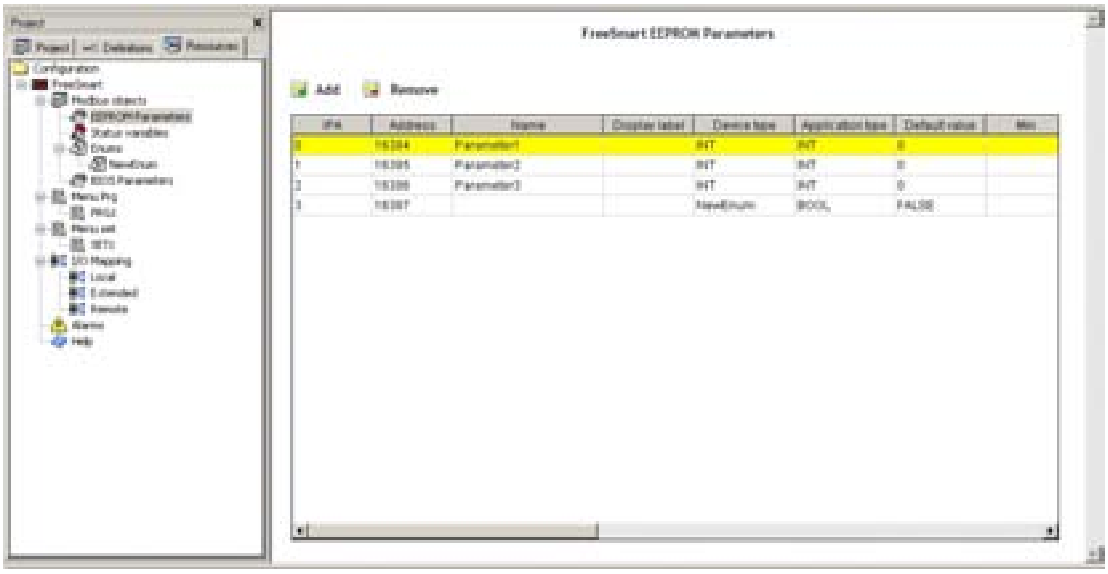
The **Resources** tab enables you to define the EEPROM parameters and status variables which may then be used in the application code (they will appear in the section **Project>Global Shared**) and allows you to construct the menu **Prg** (Programming), the menu Set Status[5] and Alarms of the **FREE Smart / Evolution target** as well as the interface of the current project, which will be displayed in **FREE Studio Device**

The **Resources** tab includes the following sections:

MODBUS OBJECTS

This section enables you to define EEPROM parameters and variables, which can be displayed on the **target, Device** and read using the Modbus® RTU protocol
For all sections, it is possible to add or remove parameters/variables in the same way as for variables in the **Project** tab

free Smart



FreeSmart EEPROM Parameters

IPA	Address	Name	Display label	Device type	Application type	Default value	Min
0	16384	Parameter1		INT	INT	0	
1	16385	Parameter2		INT	INT	0	
2	16386	Parameter3		INT	INT	0	
3	16387			NewEnum	BOOL	FALSE	

EEPROM Parameters

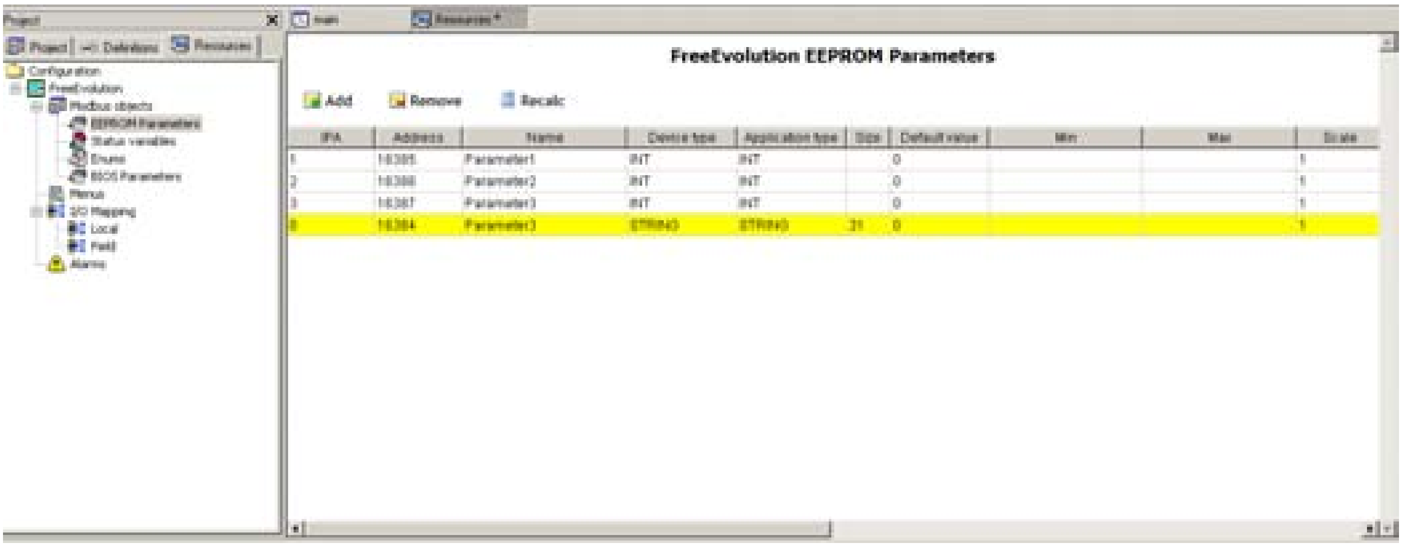
This section is for creating the variables which the developer intends to save even if the power to the **target** device is switched off.
Note: the user of parameters in the code is read-only; to write, use the appropriate functions of **FREE Smart.**

Status Variables

In this section we define the status variables which can be displayed in the menu of the **target** device.
Each variable will have a transcoding on the instrument due to the 4 digit / 7-segment display. In the Display label box, you can select the transcoding and see a preview of how it looks on the display by clicking on the 3 dots (…)
Note: some letters will not be displayed (e.g. **x** and **z**) so there will be a blank space on the display.
If the display label is **zzz**, nothing will appear on the display.
If the display label is **SET**, **SEt** will appear on the display.

[5] **only for Smart. See Installation Manual 9MAx0036 (x = 0: IT; 1: EN; 3: ES; 5: DE)**
For Evolution the Menu is defined from UserInterface

free Evolution



FreeEvolution EEPROM Parameters

IPA	Address	Name	Device type	Application type	Size	Default value	Min	Max	Scale
1	16385	Parameter1	INT	INT		0			1
2	16386	Parameter2	INT	INT		0			1
3	16387	Parameter3	INT	INT		0			1
0	16384	Parameter3	STRING	STRING	31	0			1

EEPROM Parameters

Same as **Smart**

Status Variables

In this section we define the status variables which can be displayed in the menu of the **target** device.

free Smart   free Evolution

In the **target** column, only columns that are visible from the **Smart or Evolution target** are indicated.
Refers to both unless indicated

target	column	Description
	IPA	Pre-assigned index
	address	Resource Modbus® address
	Name	Resource name which may be used by the developer in the IEC application code
Smart	**display label**	Name displayed in the application menu of the target (4-digit)
	Device Type	Type of data displayed on **target** and **Device**
	Default Value /Min/Max	Minimum/maximum default values of the resource displayed on **target** and **Device**
	Format	Display format of **Default Value /Min/Max** e.g. **XXX.Y** display of integer with decimal point
	Application Type	Type of data used in IEC application code
	Scale/Offset	Conversion coefficients between **Device Type** and **Application Type** **Application Type = scale* Device Type + offset**
Evolution	**Size**	Significant only in the case of **STRING** type. Dimension (Length) of the string. Default and max= **31** characters
	Unit	Unit of measurement of **Device Type** displayed on **Device** and if available with icon on **target**
Smart	**Access Level**	See Visibility table. This column does not apply to **Evolution**
	Read only	Enables/disables editing of Status variables.
	Description/Note	Free text displayed on **Device**

Enums

In this section we define the types of additional variable which can be used in the **Device Type column**[6]

BIOS parameters

In this section we define any variations in the default BIOS parameters map which is factory-set by Eliwell

free Smart

TARGET MENU

The target menu consists of a BIOS menu + an Application menu
The BIOS menu is factory-set[7].
The tables below defines the main functions of the keys / LEDs of the **target** device

key	press	Description
F5	**short**	switch from BIOS menu to Application menu and vice versa
F1 or F3	**short**	navigate folders and edit values
F2	**short**	cancel operation (ESC function)
F4	**short**	access to set menu
F2+F4	**short**	access to Prg menu
F1/F2/F3/F4	**long**	managed by developer (by means of **target variable sysKeyFunctions[]**)
The LEDs are managed by the developer by means of **target variable sysLocalLeds[**])		

Prg menu[8]

In this section we define the Prg Menu and the folders/variables of which it is composed.
The Prg Menu can consist of one or more folders, defined by the developer, into which are inserted

- EEPROM parameters
- Status variables
- BIOS parameters
- inputs and outputs

Set menu[9]

In this section we define the Set Menu and the folders/variables of which it is composed.
The set menu is created in the same way as the Prg menu
The set menu contains the AL folder - see below

Visibility of menu resources

The visibility of the resources created by the developer is indicated in the following table:

Access Level column	visibility on Device	visibility on target	Notes
Always visible	YES	YES	object assigned to a Prg or set menu
Level 1	YES	Yes level 1	
Level 2	YES	Yes level 2	
Never visible	YES	NO	
Never visible	YES Visible in the folder **ALL PARAMETERS**	NO	object NOT assigned to any Prg or Set menu

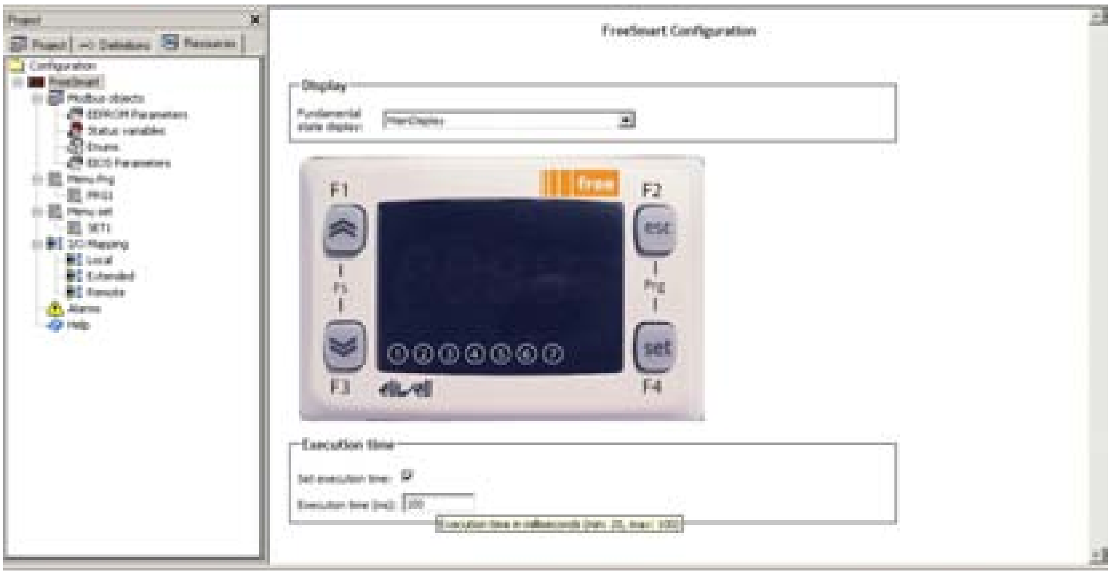
Alarms

It is possible to define alarm variables whose status must be managed by the developer.
If the variable assumes a value **other than zero**, the label will be displayed in the Alarms folder (AL) of the **set** menu in **FREE Smart**
In the **Evolution** target it is only a Global type **USINT** declaration

[6] **see EEPROM parameters and Status Variables table**
[7] **see installation manual**
[8] **the elements entered in the table in this section will be displayed on Device**
[9] **the elements entered in the table in this section will be displayed on Device**

FREE Smart configuration

From the **Resources** tab, click on **FREE Smart** to access the following panel **FREE Smart Configuration**



FreeSmart Configuration

Display

Fundamental state display: [illegible]

Execution time

Set execution time: ☑

Execution time (ms): 100

Execution time in milliseconds (min: 20, max: 100)

Using the drop-down menu you can set the value of the variable / parameter you intend to show on **FREE Smart** as the main display (**Fundamental state display**). If you set no value (**none**) the message **PLC** will be displayed.
The values available are all the **I/O[10] +** the values defined by the developer in the **Set menu**

Execution time

From the same panel it is possible to set the execution time of the project in milliseconds (**ms**).
The default setting is 100ms. The available range is from **20ms** to **100ms[11].**

I/O mapping

In this section we define the links between variables and physical I/O of FREE **Smart**.

- **Local**: local variables of the SMD/SMC base module
- **Extended**: variables of the SE expansion module
- **Remote**: variables on the SKW / SKP remote terminals

free Evolution

TARGET MENU

The target menu must be created by means of the **UserInterface.**
The BIOS menu is factory-set**[12]** and is visible from **Device**.
The main functions of the keys / LEDs of the **target** device can be programmed by means of the **UserInterface[13]**

Menus

In this section we define a 'dummy' Menu that does not appear on the display and the folders/variables of which it is composed.
The Menu can consist of one or more folders, defined by the developer, into which are entered

- EEPROM parameters
- Status variables

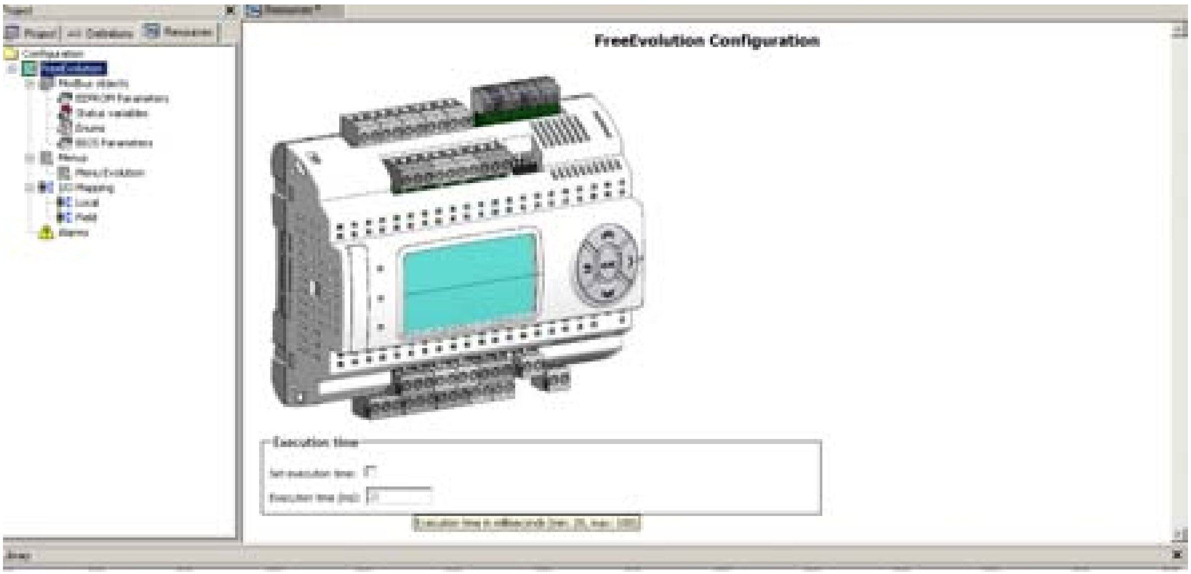
Alarms

The alarms for **Evolution** are only defined to enable the portability of a **Smart** project

[10] the I/O are all the local and remote inputs and outputs, including those of the remote terminals. See installation manual for details
[11] Also see Tasks
[12] see installation manual
[13] LEDs also programmable from Application

FREE Evolution configuration

From the **Resources** tab, click on **FREE Evolution** to access the following panel **FREE Evolution Configuration**



Execution time

Set execution time: □

Execution time (ms):

Execution time in milliseconds (min.: 20, max.: 100)

Execution time

From the same panel it is possible to set the execution time of the project in milliseconds (**ms**).
The default setting is 20ms. The available range is from **20ms** to **100ms**[14]**.**

I/O mapping

In this section we define the links between variables and physical I/O of FREE **Evolution**.

- **Local**: local variables of the **EVD/EVC** base module
- **Field**: variables and/or binding of third-party controllers

Note

If correctly defined[15] all the variables defined in **Resources** will automatically be located in the **Global Shared** section.

[14] Also see Tasks

[15] the project must be saved without errors for the variables to be visible in the Project tab in Global Shared

SAVING PROJECTS

Once you have completed the project you can save it in various forms and send it to third-party **Developers** or **Users**.

File types

Target	File type	Application	Description
Smart	**.PPJS**	**Application**	Project created with **Application** to be opened with **Application**
Evolution	**.PPJS**	**Application** **Connection**	Project created with **Application** that can be added to an **Evolution** device with **Connection**
	.PLL	**Application**	Library created by the developer to be used in **Application** or **Device**
	.RSM	**Application**	Compressed **.PPJS** project
Evolution	**.PAJX**	**User Interface**	Project created with **UserInterface** to be opened with **UserInterface**
Evolution	**.PPJX**	**User Interface**	As for .PPJS. Compressed **.PAJX** project
Smart	**.CFN**	**Device**	Project created with **Application** to be opened with **Device** Note. The source code is not available but the project can be downloaded to the **target** device
Evolution	**.CFN**	**Device**	Project created with **Application** or **Connection** to be opened with **Device.** If the device is created with **Connection** it contains both the **PLC** application and the **HMI** menu Note. The source code is not available but the project can be downloaded to the **target** device

TRANSFERRING PROJECTS AND FILES

The table below shows several operations involving the transfer of projects from one developer to another or from a developer to a user. The data transfer involves the sending of one or more files defined in the previous section

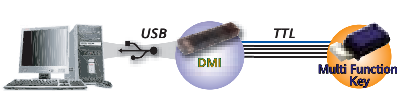
Project	From -->To	Application	File type	Notes
The **developer** wants to transfer a project to another developer The developer sends a **.ppjs** file The developer who receives it can open and edit it with **Device Application**; The developer connects to the **target**, and compiles and downloads the application.	**Developer** ↓ **Developer**	**Application**	**.PPJS**	
The developer sends a **.ppjs** file to another developer The developer who receives it can open and edit it with **Application**; The developer connects to the **target** device and compiles the application. After compilation, the developer selects the menu heading **Developer > Open with FREE Studio Device** and configures the **target** device BIOS (parameterisation of the **target** device)	**Developer** ↓ **Developer**	**Application** **Device**	**.PPJS**	
The developer wants to transfer a project to a user The developer sends a .CFN file The user who receives it can open it with **Device**; The user connects to the **target** device, downloads the application and configures the target device BIOS (parameterisation)	**Developer** ↓ **User**	**Device**	**.CFN**	From **Device** the user CANNOT compile the application

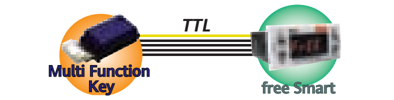
Project	From -->To	Application	File type	Notes
The developer wants to debug an IEC application already present on the target device, but does not have the source code. The developer receives the entire project folder (which includes the binary files and debug symbols) used for downloading the IEC application onto the **target** device. **Note**: the project has not been edited (no change to source and no repeat compilation and/or download)	**Developer** ↓ **Developer**	**Application**	**project folder**	

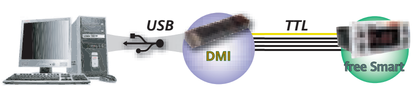
CONNECTION

free Smart

To download the IEC applications of **Studio** from the personal computer to the **Smart target** device, several additional modules are necessary.
The tables below show possible operations:

Multi Function Key / DMI **PC → ← MFK**		
		
use the blue TTL cable for DMI - MFK connection		
direction **Data downloading**	**→**	**←**
Parameter map	-	-
IEC application	✓	-
BIOS	✓	-

Multi Function Key **MFK → ← Smart**		
		
use the yellow TTL cable for MFK - target connection		
direction **Data downloading**	**→**	**←**
Parameter map	✓	✓
IEC application	✓	-
BIOS	✓	-

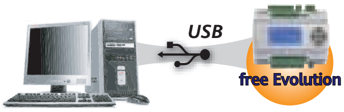
Network **PC → ← Smart**		
		
use the yellow TTL cable for DMI - target connection		
direction **Data downloading**	**→**	**←**
Parameter map	✓	✓
IEC application	✓	-
BIOS	✓	-

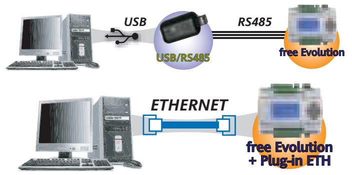
Note: in the case of Network **Smart** must not be connected to earth

free Evolution

To download the IEC applications of **Studio** from the personal computer to the **Evolution target** device, several additional modules are necessary.
The tables below show possible operations:

USB Host **USB → ← Evolution**		
direction **Data downloading**	**→**	**←**
Parameter map	✓	✓
IEC application	✓	-
HMI application	✓	-
Data archive	-	-
BIOS	✓	-

USB device **PC → ← Evolution**		
		
direction **Data downloading**	**→**	**←**
Parameter map	-	-
IEC application	✓	✓
HMI application	✓	✓
Data archive	✓	✓
BIOS	-	-

USB-RS485 **USB-CANOpen** **ETHERNET** **PC → ← Evolution**		
 USB RS485 USB/RS485 free Evolution ETHERNET free Evolution + Plug-in ETH		
direction **Data downloading**	**→**	**←**
Parameter map	✓	✓
IEC application	✓	-
HMI application	✓	-
Data archive	-	-
BIOS	✓	-

DOWNLOAD IEC APPLICATION ONTO TARGET

Connection and application downloading operations for **Smart** and **Evolution** are different.
The connection procedures are illustrated below (see relevant paragraph)

- Network for Smart
- USB-R85 for Evolution

The actual download adopts the same procedure for both targets

free Smart

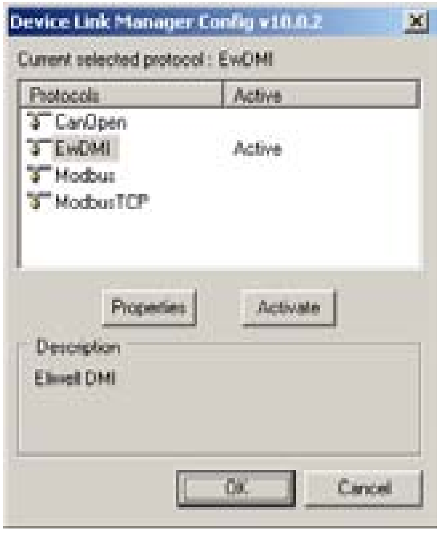
PRELIMINARY OPERATIONS

In order to download the application correctly, the user must:

1. connect the DM hardware interface to the PC.
2. Make sure that the driver has been installed - see DM interface Set-up.

COM settings[1]

From the drop-down menu, select **Communication > Settings**
The COM port must previously have been read/set in "Peripherals Management" (see Reading the DM interface COM port) for it to be recognised. If there are errors, refer to the paragraph "**DM interface connection error**".



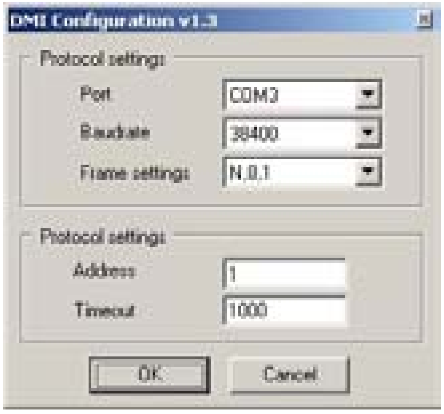
Device Link Manager Config v10.0.2

Current selected protocol : EwDMI

Protocols	Active
CanOpen	
EwDMI	Active
Modbus	
ModbusTCP	

Description: Eliwell DMI

For **Smart** select **EWDMI** or **Modbus**[2]. If the protocol is not activated press the **Activate button**
The value selected for the COM port will be saved and will reappear each time the program is accessed, until it is changed.
The properties are visible and can be edited from the panel **Communication > Settings > Properties**[3]



DMI Configuration v1.3

Protocol settings
Port: COM3
Baudrate: 38400
Frame settings: N,8,1

Protocol settings
Address: 1
Timeout: 1000

[1] the COM settings must be set on all of the work environments: Application, Device and UserInterface
[2] in the case of Modbus for /S models only with maximum speed 19200 baud. TTL not for use.
[3] obviously, the protocol must be activated beforehand

DM interface connection error

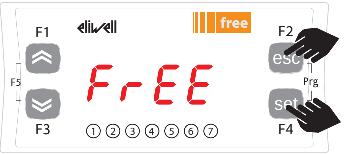
Error opening serial port
If the "Error opening serial port" message appears, proceed as follows:

1. Check that the COM port setting in the program is the same as one read in the COM port reading by the **DM** interface.
2. If they are the same, physically disconnect and reconnect the **DM** interface from the USB port. This should make the operating system recognise the interface.
3. Repeat the **DMI** Detection function.

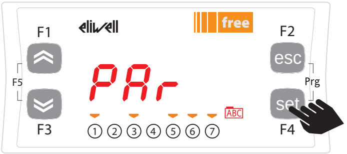
SMART TARGET

Smart has certain parameters in the CF folder for managing the connection between the **target** and **Studio**

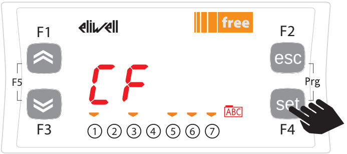
If the **target** is 'empty'[4], e.g. there is no **IEC application** on the device, **Smart** will display the message **FrEE:**



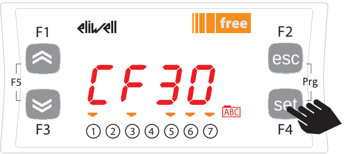
To view the parameter menu, press the Esc and Set keys at the same time. This will open the PAr menu.



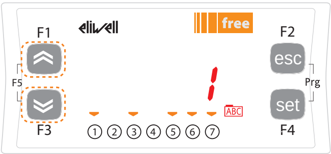
The parameters menu PAr contains all controller folders. Press the set key to view folders.



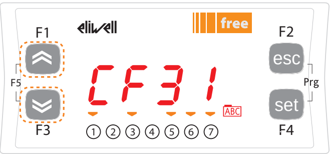
The first folder shown is the CF configuration folder. Press the set key to view the folder parameters.



The first parameter shown is CF30. To view the value of the parameter press the set key.



Use the UP and DOWN keys to change the value if necessary. To confirm the value press the set key.
To exit press Esc



Use the UP and DOWN keys to scroll the other parameters and repeat the procedure to view the values and - if necessary - edit them.

Parameters needed for correct connection between the **Smart target** and **Studio**.[5]

parameter	description	values	default	visibility	notes
CF01	Select COM1 (TTL) protocol	0 = Eliwell 1 = Modbus	1	2	Must be set to 1
CF30	Modbus protocol controller address	1...255	1	3	Check that the set values correspond to those defined by the panel **Communication > Settings > Properties**
CF31	Modbus protocol baud rate	0,1, 2 = not used 3 = 9600 baud 4 = 19200 baud 5 = 38400 baud 6 = 5700 baud 7 = 115200 baud	3	3	
CF32	Modbus protocol controller parity	1= EVEN 2 = NONE 3 = ODD	1	3	

[4] otherwise (IEC application exists on Smart) the message PLC will appear on the display. Simultaneously press the UP and DOWN keys to view the message FrEE

[5] for other parameters and to manage visibility levels see the Smart installation manual

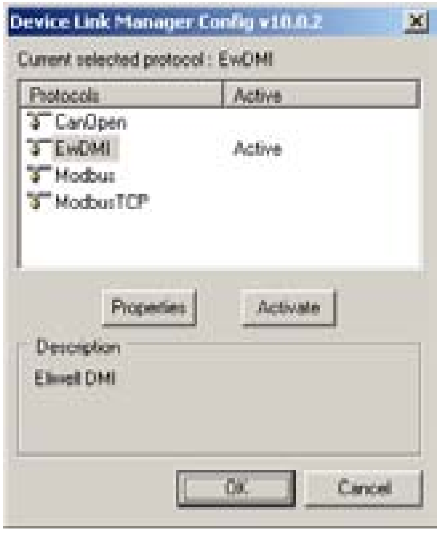
free Evolution

COM settings[6]

The settings are identical to those given for **Smart**:
panel **Communication > Settings**
<u>For **Evolution EWDMI** must NOT be selected</u>

- Select **CAnOpen** in the case of USB/CAN connection
- Select **Modbus** in the case of USB/RS485 connection
- Select **Modbus TCP** in the case of Ethernet connection using the relevant plug-in

If the protocol is not activated press the **Activate** button



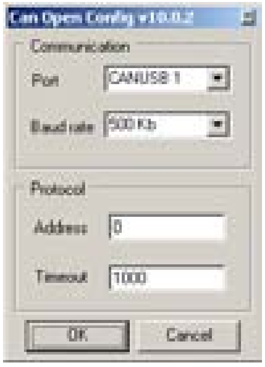
Device Link Manager Config v10.0.2

Current selected protocol : EwDMI

Protocols	Active
CanOpen	
EwDMI	Active
Modbus	
ModbusTCP	

Description: Eliwell DMI

Also in this case, the properties are visible and can be edited from the panel **Communication > Settings > Properties** **Communication > Settings > Properties**[7]. The panel displayed depends on which protocol is activated:



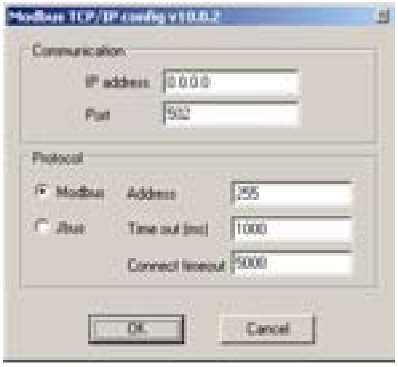
Can Open Config v10.0.2

Communication
Port: CANUSB 1
Baud rate: 500 Kb

Protocol
Address: 0
Timeout: 1000

CANOpen

ModBus



Modbus TCP/IP config v10.0.2

Communication
IP address: 0.0.0.0
Port: 502

Protocol
● Modbus Address: 255
○ Jbus Time out (ms): 1000
Connect timeout: 5000

ModbusTCP/IP

EVOLUTION TARGET

Evolution has a number of BIOS parameters for managing the connection between the **target** and **Studio** but, unlike **Smart**, it does not have a default menu shown on the on-board or remote display.[8]
Therefore in order to view the parameters you need to open **Device**
Parameters needed for correct connection between the **Smart target** and **Studio**[9]:

The two protocols available on-board are shown. For the protocols available on the plug-in module, see the installation manual

[6] **the COM settings are valid on both Application and Device**
[7] **obviously, the protocol must be activated beforehand**
[8] **Eliwell provides examples of HMI menus for configuration of BIOS parameters**
[9] **for other parameters see the Evolution installation manual**

On-board RS485

parameter	description	values	default	notes
Baud_RS485_OB	On-board Modbus protocol baud rate	0=9600 baud 1=19200 baud 2=38400 baud 3=58600 baud 4=76800 baud 5=115200 baud	2	
Addr_RS485_OB	On-board RS485 serial address	1...255	1	The actual address is determined by the sum of this value + the value of the DIP switch
Proto_RS485_OB	On-board RS485 protocol selection	2 = uNET 3 = Modbus/RTU	3	
Databit_RS485_OB	On-board RS485 data bit number		8	Fixed at 8
Stopbit_RS485_OB	On-board RS485 stop bit number	1= 1 stop bit 2= 2 stop bit	1	
Parity_RS485_OB	On-board RS485 protocol parity	0= NULL 1= ODD 2= EVEN	2	

On-board CAN

parameter	description	values	default	notes
Baud_CAN_OB	On-board CAN protocol baud rate	2=500 Kbaud 3=250 Kbaud 4=125 Kbaud 5=125 Kbaud 6=50 Kbaud	2	
Addr_CAN_OB	On-board CAN serial address	1...127	1	The actual address is determined by the sum of this value + the value of the DIP switch

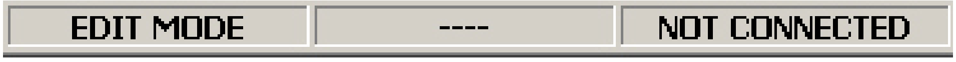
CONNECTING TO THE TARGET

Before connecting, the bottom right of the screen displays the message **NOT CONNECTED**
From the drop-down menu select **Communication > Connect**

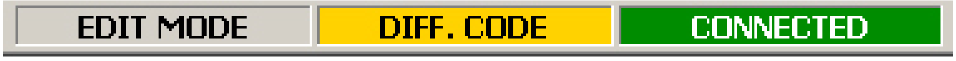
If connection is successful, the following screen (minimised) appears in the menu bar:

This screen will remain present as long as the **target** device is connected. To interrupt the connection, repeat the **Communication > Connect** procedure

At the bottom right of the panel is a connection palette divided into 3 sections
Before connecting to the target the panel shows that there is no connection

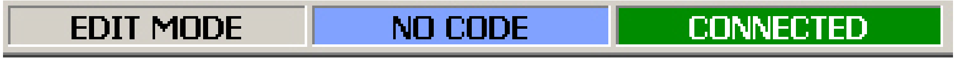


When connected to the **target** there are various situations shown below

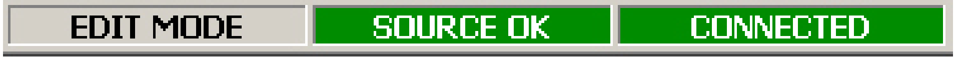


- **CONNECTED** (in green): if the connection is established and the **target** application coincides with the **Studio application**

- **CONNECTED** (in yellow): if the connection is established and the **target** application does NOT coincide with the **Studio application**



- **CONNECTED** (in green): if the connection is established
- **NO CODE** (in light blue): if the connection is established and there is no application on the **target**



- **CONNECTED** (in green): if the connection is established and the target application coincides with the **Studio application**
- **SOURCE OK** (in green): if the connection is established and the target application coincides with the **Studio application**

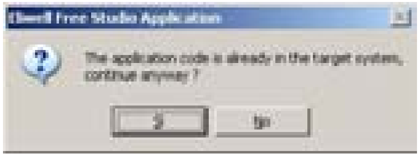
- **ERROR** (in red): if there is a communication error

DOWNLOAD

To download the application, simply press **F5** or select **Communication > Download code** from the drop-down menu

The following dialogue boxes will appear

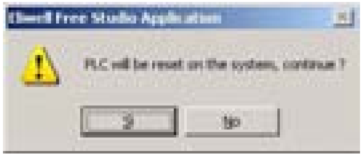
A. If the application is already present, you will be asked for confirmation to overwrite the previous version



Eliwell Free Studio Application

The application code is already in the target system, continue anyway ?

B. Warning message that the PLC will be reset on the target

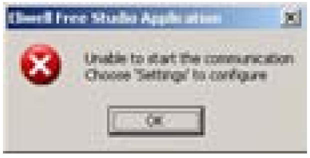


Eliwell Free Studio Application

PLC will be reset on the system, continue ?

If you confirm with 'Yes' the application will be downloaded to the target. The process may take a few seconds: the '**Download in progress**' window appears to show the status of the download.

In the event of a connection error, the following dialogue box appears:



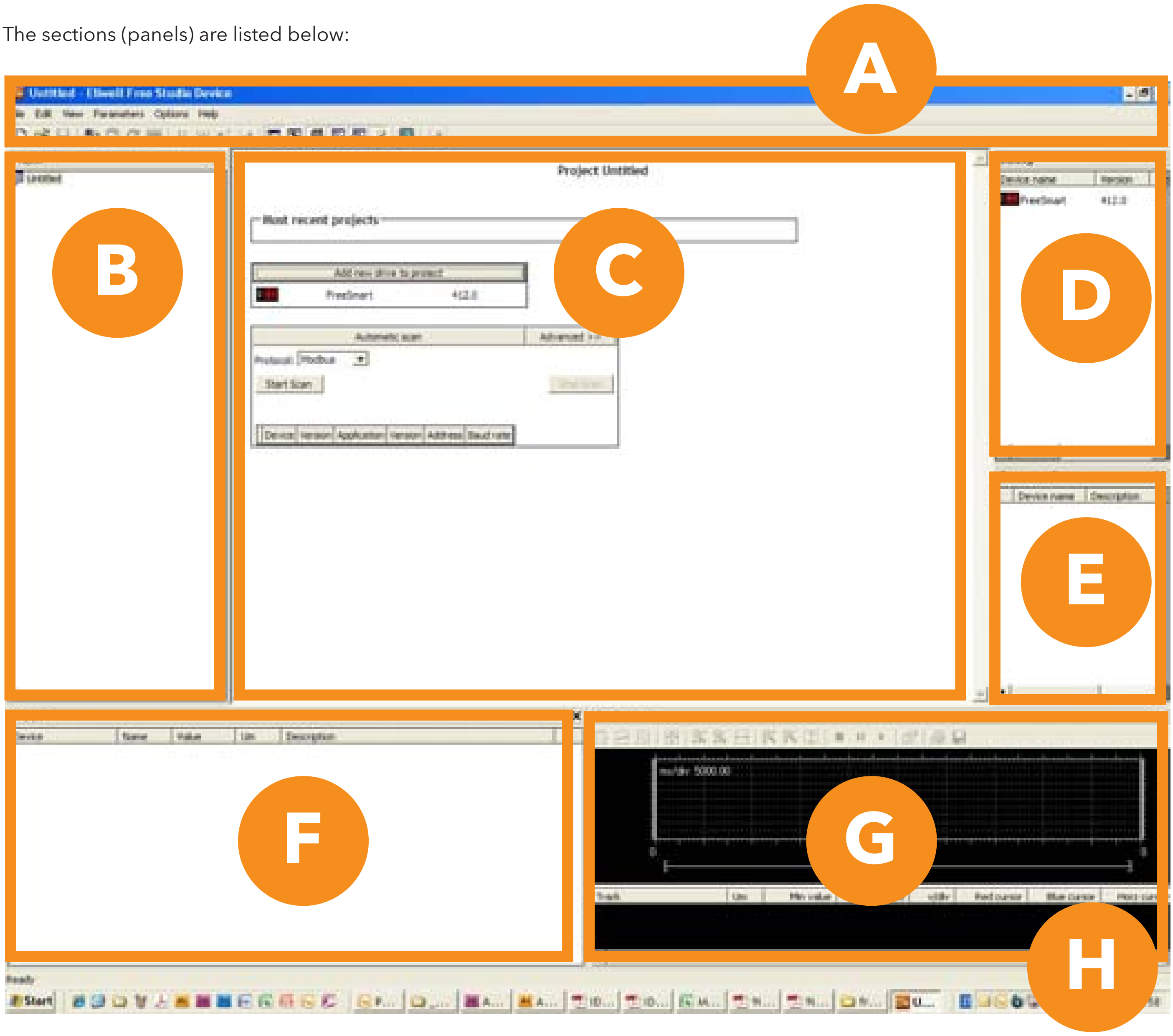
Eliwell Free Studio Application

Unable to start the communication
Choose 'Settings' to configure

free Smart

The **Device** work environment is shown below with screenshots and descriptions for **Smart**. Where necessary, the differences compared to **Evolution** are highlighted.
The **Device** work environment has various sections for managing an IEC application generated by a developer, e.g. **FREE Smart** parameter configuration, testing, BIOS updating and application downloading to the target device.
Note: the source code of the application is NOT visible from **Device**

The sections (panels) are listed below:



Section	Panel /Section name Application	Panel	Description
A	/	**Configurator bar**	This section shows the work tools in the form of icons. To display or hide the toolbars, simply right-click and select or deselect the desired tools. Some tools can be selected by means of key combinations (**shortcuts**). See relevant chapter for details

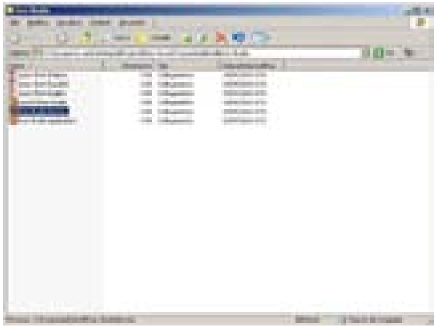
Section	Panel /Section name Application	Panel	Description
B	Project	Project	The Project panel enables you to manage one or more **FREE Smart** devices.
C	/	Edit panel	According to the selection made in the Project panel, the following appear in this section: • BIOS parameters (parameters + I/O values) • parameters and variables defined by the developer in **Application** associated to the menu
D	Catalog	Applications catalogue	This panel shows all the applications available from the catalogue
E	Connection Status		This panel shows the connection status of all the targets in the **Project** panel
F	Watch	Display variables with application running	This panel enables you to manage variables debugging by displaying their status in **number/letter** format when the application is running and connected to the target device
G	Oscilloscope	Display variables graph	Like the **Watch** panel, this panel enables you to manage variables debugging by displaying their status in **graph** format when the application is running and connected to the target device
H*	Output	Messages	This panel shows messages relating to the development of the project (file opening, reading/writing errors, status of connection to device, etc.) **Note**: the connection to the target device is also visible in the bottom right of the screen by means of coloured messages (see Application Download)

*panel not visible at first start-up.
To display it, simply check the heading in the drop down menu **View > Output**

The panels appear or remain hidden (see **Shortcuts**) and can be organised as the user wants in terms of size and layout. **Studio** saves the last settings made free Smart

FIRST START-UP[1]

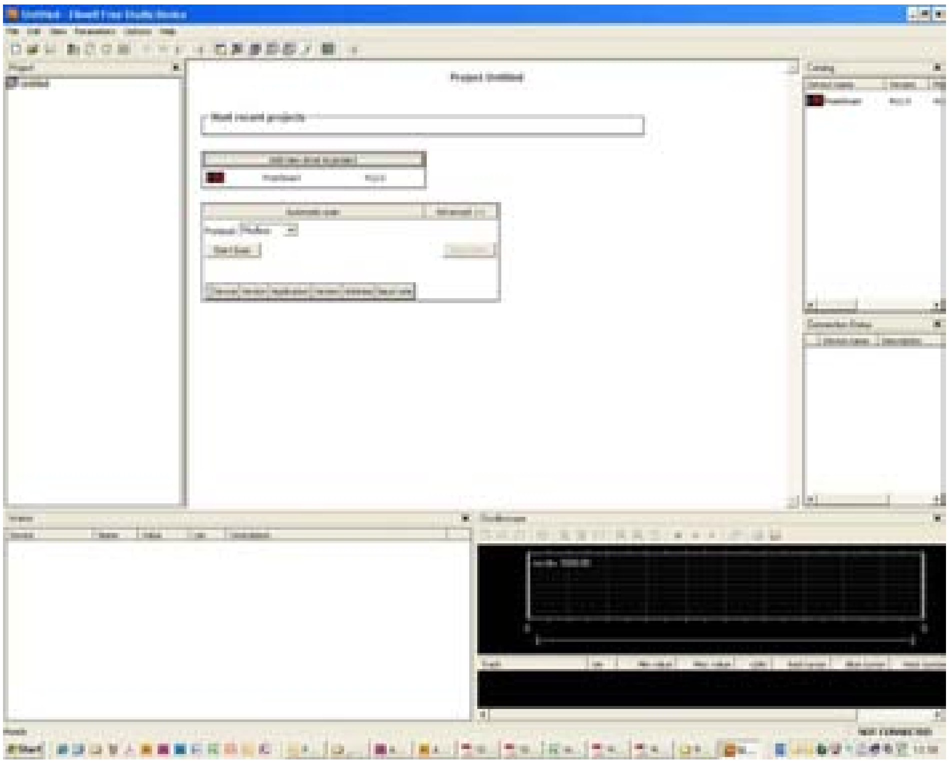
1. Click on the **Launch FREE Studio** icon on your Desktop
2. Click on the **FREE Studio Application[2] icon.**



1

2

The program appears as shown in the picture:



You can now:

- Create a new project **.CFN** '**New Project ...**'[3]
- Open an existing project ' **Open project ...**'
- Open recent projects '**Most recent projects ...**' by selecting from a list

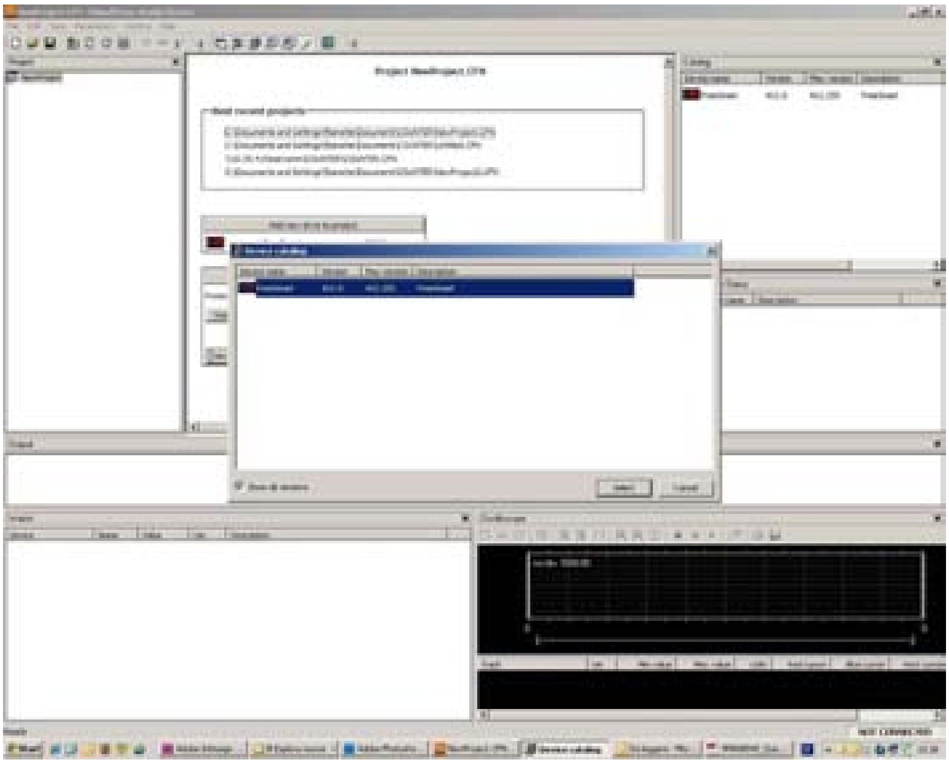
Device projects have the extension **.CFN**
In the same project, you can add more than one drive and associate to each drive an IEC application created with **Application**

[1] the Device tool is available for Smart. All tools (including Application) are available for Evolution. For Evolution you are advised - for First Start-up - to always start from the Connection tool (see relevant chapter)

[2] a project can also be opened with Device from Application from the drop-down menu Developer > Open with FREE Studio Device

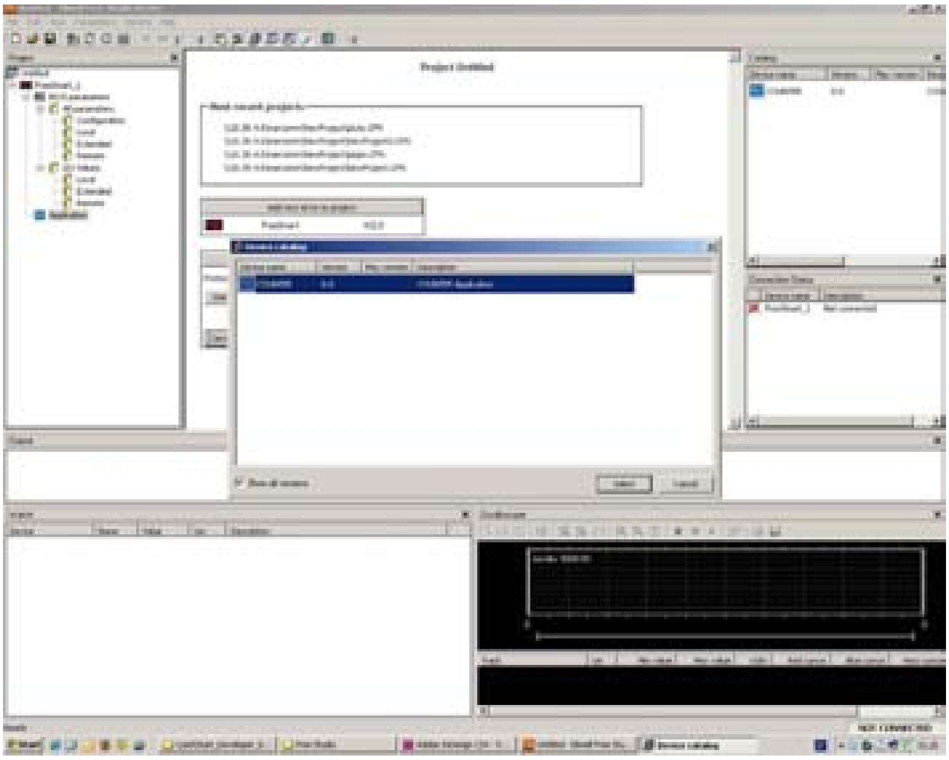
[3] for a new project, you must add a drive listed in the catalogue by right-clicking and selecting Add. See screenshot

A new **Untitled** project is empty. Simply right-click to add a new drive to the existing project in the catalogue by selecting



OPENING THE IEC APPLICATION

In the same way you can add the respective **PLC** IEC application present in the catalogue.
The catalogue can contain various applications but only one can be associated to a project
The IEC application will be added to the drive with name_1



Existing projects can be opened as described previously or directly from **Application:**
the menu **Developer > Open with free Device** will start the **Device** containing the project created by the developer with the development environment **Application**

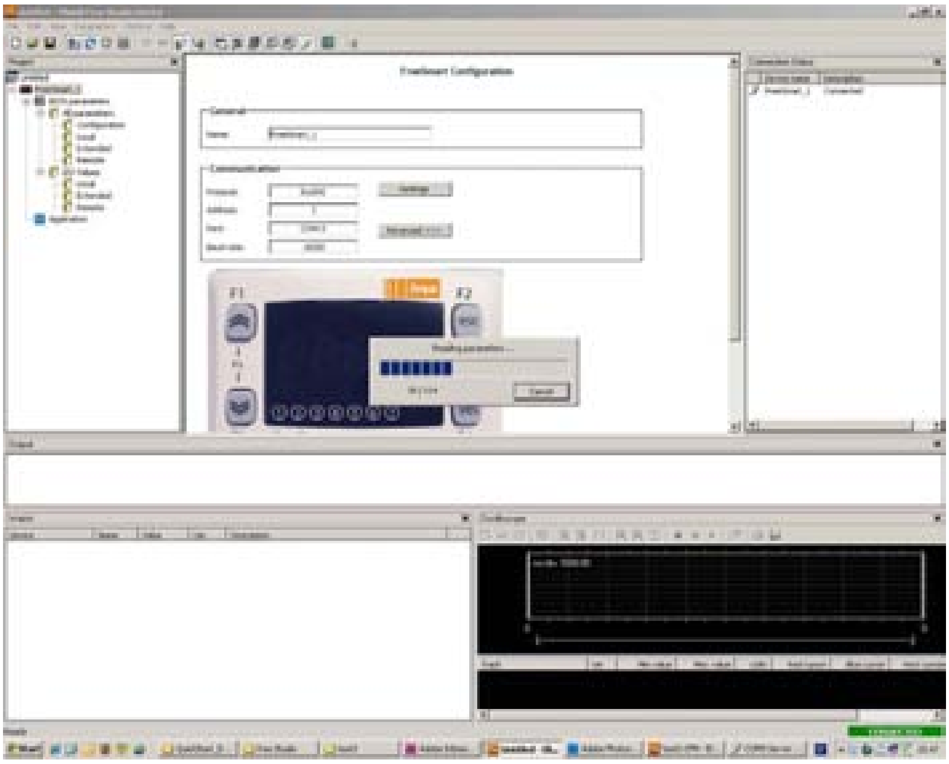
CONNECTING TO THE TARGET DEVICE

Device requires you to connect to the target device again (click on the respective icon). The central panel shows the protocol references, address, etc. The default name of the target device is FREESmart_1[4].
When the target device is connected, you can read or write parameters

READING PARAMETERS[5]

By default, the project contains a list of the BIOS parameters and I/O values[6] with their respective default settings.
From the target device, the user can read all or just some of the parameters[7]

Suppose we want to read all the parameters.
From the drop-down menu **Parameters>Read all device parameters** it is possible to read all the parameters of the connected target device.
The operation takes a few seconds.



If there are configuration parameters which have default values other than those present on the target device, they will be highlighted in an appropriate colour (blue[8])

WRITING PARAMETERS

The user can download all or some parameters to the target device, either with their default values or with modified values inserted by the user

Suppose we want to download all the CL parameters (local BIOS) and no others.
Select all the parameters - drop-down menu **Parameters>Select all** (CTRL+A) or using the respective icon. The parameters are all highlighted in yellow. Write drop-down menu **Parameters>Write selected** (CTRL+SHIFT+W) (or using the icon **W**)
All the CL parameters will be downloaded to the target device[9].

As explained in the notes, you can also download the values of the I/O variables set by the user or their default values.

[4] **by simply using the copy/paste function, you can create a twin project FREESmart_2, change the BIOS parameters and download them to the target device or a second target device**

[5] **in reality, in this case, the term 'parameter' refers to any resource present on the target device, in other words genuine parameters in their own right, I/O and variables of the application present on the target**

[6] **the default value of the analogue inputs and outputs is 0.0; the digital inputs and outputs are set to FALSE**

[7] **to read a parameter, simply select it with the mouse and click on the icon R.**
Multiple selections are permitted (CTRL for non-consecutive selections, SHIFT to select part of the block)
You can select the entire block by clicking on the respective icon
The selection is highlighted with a yellow background

[8] **see parameter values table**

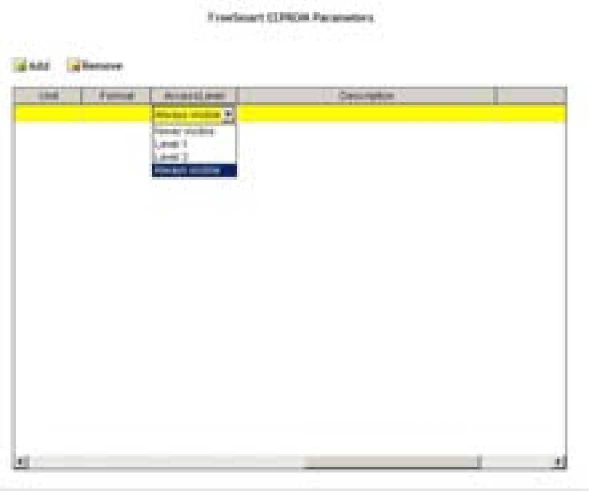
[9] **see the specific chapter containing project development examples**

Downloading **all** the parameters (drop-down menu **Parameters>Write all device parameters**[10]) replaces all the values present on the target device with the values listed in the Value column.
Note: the inputs are read-only and are shown in grey. See the following table:

Parameter values table

colour	column	description	cases
black	value	value aligned with **default**	**target** already read
blue	default	default column value different from value in **value** column	target already read
red	value	value NOT aligned with **default**	• **Device** just opened • value changed by user
grey	name	read-only parameters	I/O values analogue inputs AI and digital inputs DI
green	name	parameters not visible on **target** display	See visibility table

When declaring the status variables and EEPROM parameters in **Application** you can also define their level of visibility in the **Smart target**[11] menu as shown in the figure:



BIOS Download

The user can also upgrade the **Smart BIOS** from Device. Select the name of the **target** and click the right mouse button. Select **BIOS download**[12] and open the **.fwf** file that you want to download
The **Direct** mode is executed via **DMI** directly on PC whereas the MultiFunctionKey mode requires passing through **MFK** and then downloading onto **Smart**

free Evolution

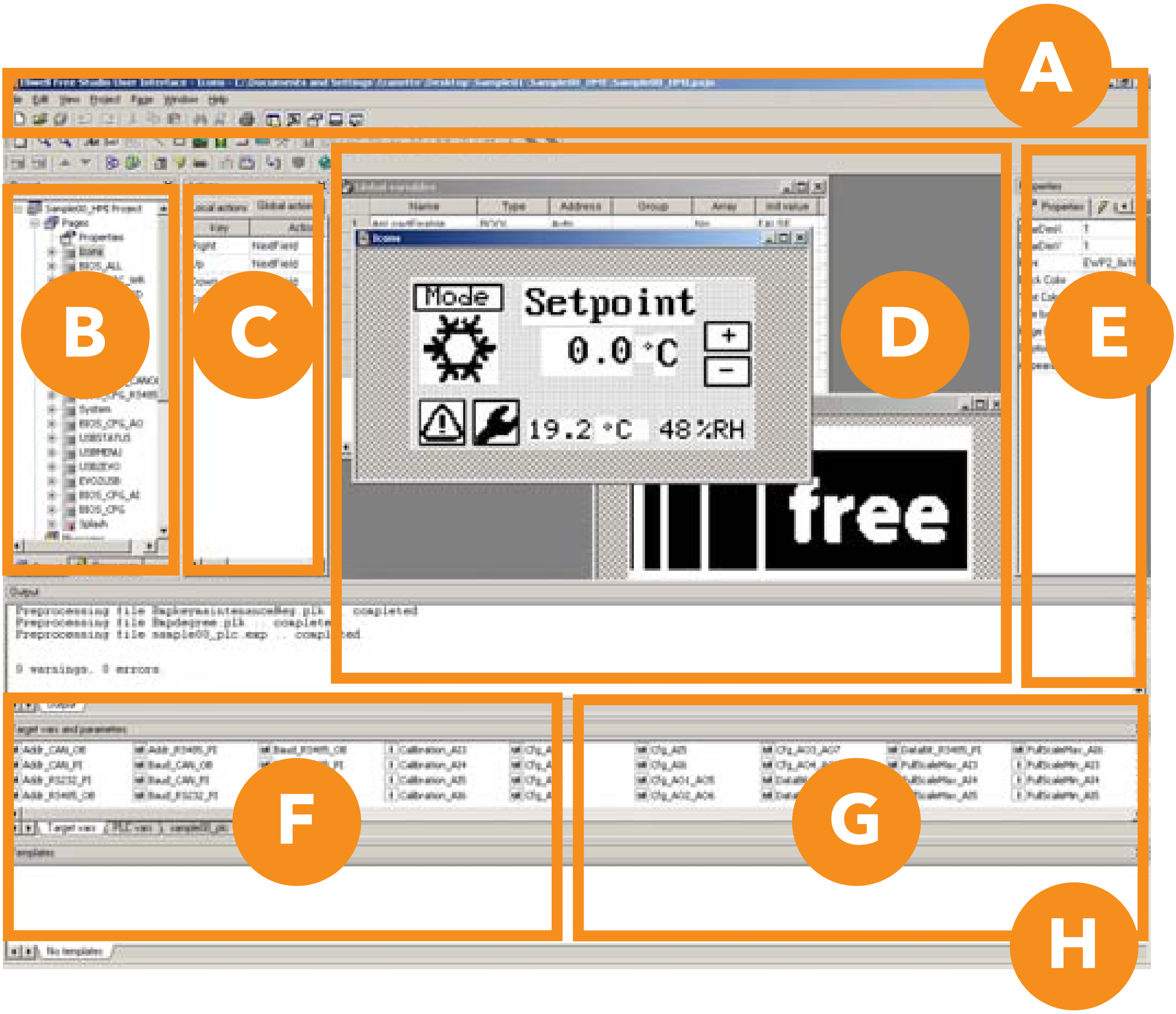
The default BIOS menu of **Evolution** is visible from the folder **All Parameters**.
In the **All Parameters** folder you can also view user parameters defined with **Application**
Downloading the **PLC** application and the **HMI** menu is only performed from **Device** but is described in the **Connection** chapter

[10] if you select Parameters>Write all default values, all the default values including I/O will be downloaded to the target device
[11] has no effect on Evolution
[12] see installation manual. BIOS Download not available via RS485.
Do NOT power up Smart externally (only power up via DMI)

DEFINITIONS OF USER INTERFACE PANELS

The **User Interface** work environment dedicated to developers features various sections for creating and managing customised pages, e.g. defining the graphical interface of **FREE Evolution,** using and creating templates and downloading **HMI** onto the target device.

The sections (panels) are listed below:



```
Preprocessing file Bapkeyresistenceconfig.plk ... completed
Preprocessing file Bapdegree.plk ... completed
Preprocessing file sample00_plc.exp ... completed

0 warnings, 0 errors
```

Section	Panel /Section name User Interface	Panel	Description
A	/	**Configurator bar**	This section shows the work tools in the form of icons. Some tools can be selected by means of key combinations (**shortcuts**). See relevant chapter for details
B	**Project**	**Project**	The Project panel is divided into 2 tabs: • **Project:** for managing the pages that will be shown on the **Evolution** graphical terminals • **Resources:** for managing images, lists and strings that may be used in the pages
C	**Actions**		According to the selection made in the tab, the following appear in this section: • **Local Actions** • **Global Actions** actions linked to the User Interface keys
D	**Document bar**	**Edit panel**	This panel is used to configure the graphics of the pages
E	**Properties**	**View properties**	This panel is used to define the properties of the pages or of their constituent elements
F	**Output**	**Messages**	This panel shows messages relating to the development of the project (file opening, compilation errors, etc.)
G	**Target vars & parameters**	**View target variables and parameters**	The panel is divided into 2 or more tabs: • **Target vars** contains the variables/BIOS parameters of **Evolution** • **PLC vars** contains the variables/parameters defined in the PLC application of **Application** installed in the Evolution base module where the **HMI** will be downloaded. • Other tabs display variables/parameters with Modbus address to be shown on the HM interface. Note: these applications have been downloaded using Studio onto other **Evolution** modules connected in a network to the **Evolution** onto which the HMI that you intend to develop will be downloaded
H	**Templates**	**View available templates**	This panel shows all of the pages created as 'templates', e.g. as reusable models available in the catalogue

The panels may be visible or hidden at first start-up.
To view them, simply check the relative heading in the drop-down menu. For example: **View > Templates**
The panels can be organised as the user wants in terms of size and layout.
Studio saves the last settings made.

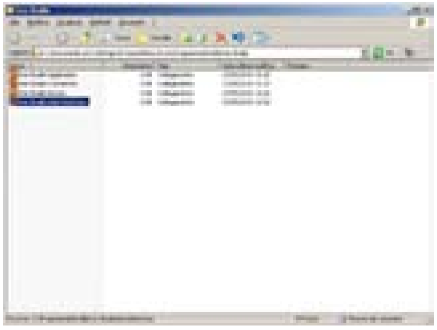
free Evolution

USERINTERFACE VS CONNECTION

As already mentioned in the **TARGET** chapter and as described in the **Connection** chapter, **UserInterface** is typically opened from **Connection** by clicking on the HMI node
You are therefore advised to read the **Connection** chapter before developing a menu in **UserInterface**
To get started and create a simple **stand-alone** menu[1] follow the outline illustrated below

FIRST START-UP

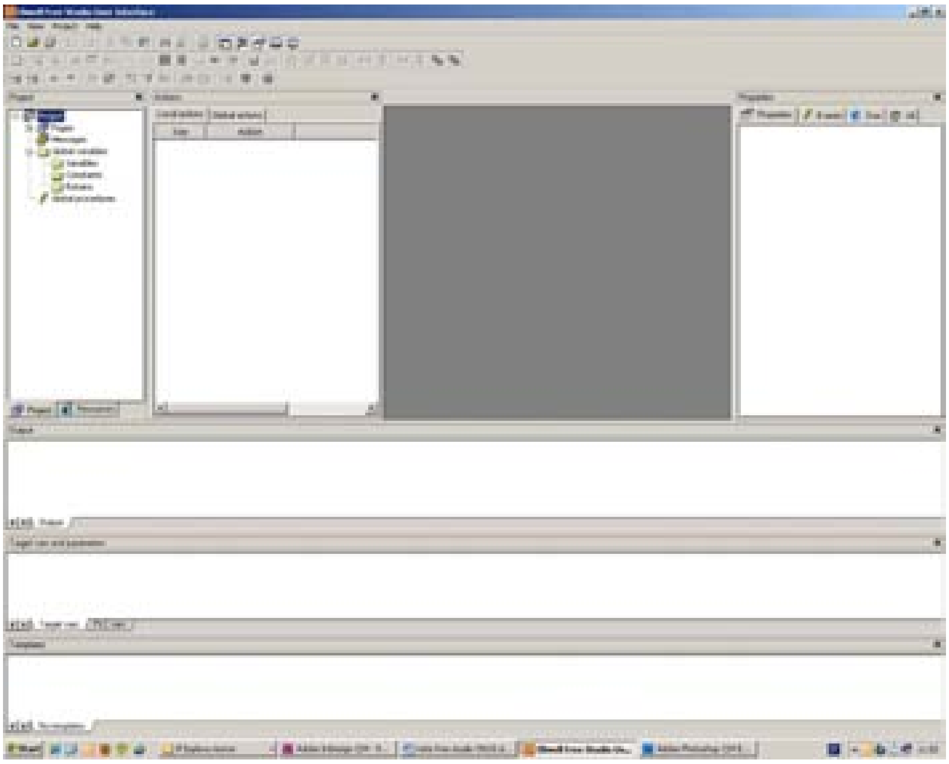
1. Click on the **Launch FREE Studio** icon on your Desktop
2. Click on the **FREE Studio User Interface** icon.



eliwell
free Studio
USER INTERFACE

1 2

The program appears as shown in the picture:



You can now:

- Create a new project '**New Project ...**' - **CTRL+N**
- Open an existing project ' **Open project ...**' - **CTRL+O**

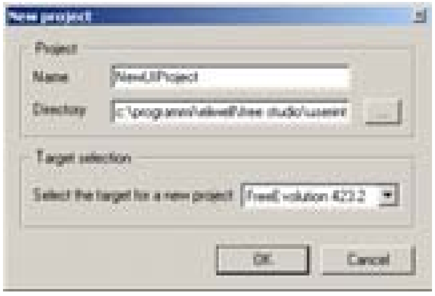
Existing projects have the extension **.pajx**

[1] **a menu created in this way is subsequently linked to a Connection project. See corresponding chapter.**

CREATING A NEW PROJECT

- Click **New Project ...** or
- Select **File > New Project (CTRL+N)** from the drop-down menu

The dialogue box shown below appears, in which you can create, name and save a new project:



New project

Project

Name: NewUIProject

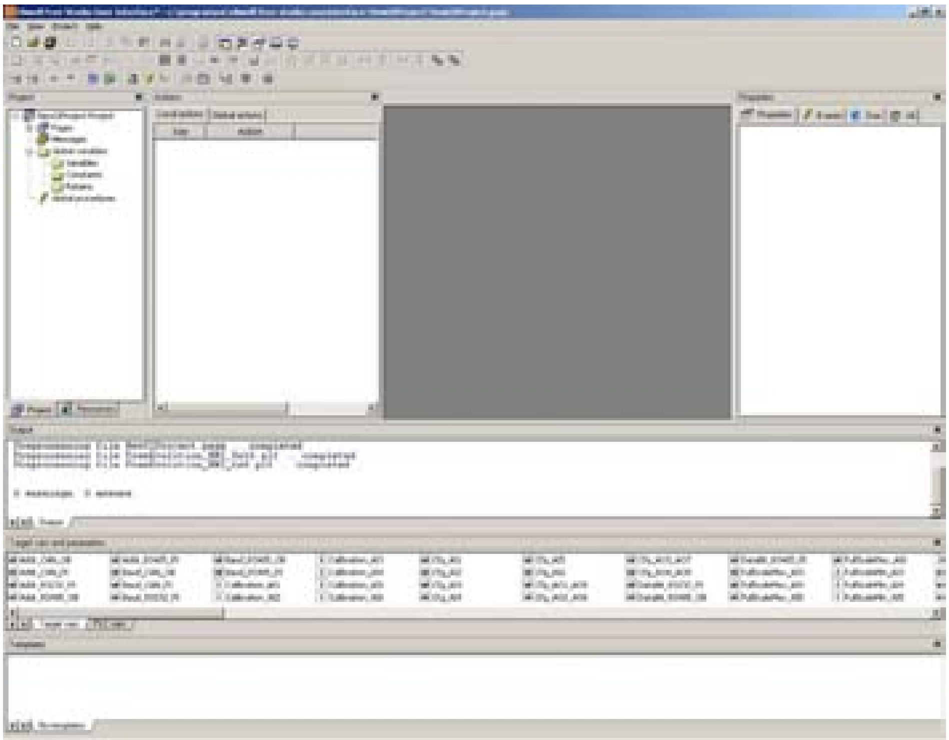
Directory: c:\programmi\eliwell\free studio\userin

Target selection

Select the target for a new project: FreeEvolution 423.2

New Project

The new project appears as shown in the figure:



- A description of the icons is available in the **Online Help** - see 'Support'
- The standard development environment described previously is developed in this chapter

The pages that follow explain how to create, develop and test a new menu, with simple examples of pages, templates and elementary programs.

CREATING PAGES

All of the created pages that will comprise the menu shown on the instrument display will appear in the **Document bar** panel. The elements for creating pages are presented generically below.
The Appendix gives some simple examples of menus that you can use to define a basic, functioning menu.

PROJECT PANEL / PROJECT TAB

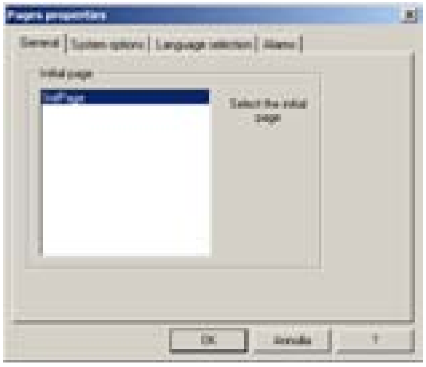
In this tab the following elements (nodes) can be used:

- **Pages** for managing pages (create, delete, import/export etc.)
- **Messages** for managing message pages
- **Global variables** for managing 'variables' common to all pages
- **Global procedures** for managing 'procedures' common to all pages. Note: procedures are written in ST language only

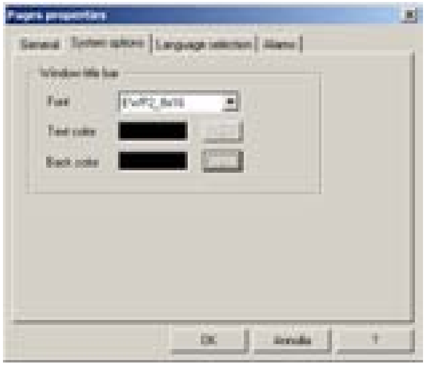
PAGES

Properties

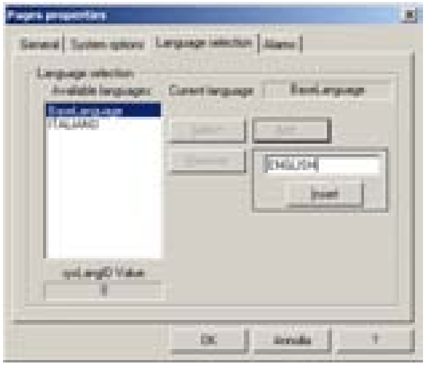
From the **Project** tab, **Pages** node, by right-clicking on **Properties** you can edit the properties of[1] pages as indicated:



General



System options



Language selection

1. From the **General** tab you can select which page to set as the initial page
2. From the **System options** tab you can select the type of font[2], the text colour and the background colour (choice of black or white)
3. From the **Language selection** tab you can manage the language settings:
   • click **Select** to choose the current language (indicated in the **Current language** box),
   • click **Add** to add a new language
   • click **Delete** to delete a language
   • click **Export** to export and import the **.txt** text file to be translated[3]
   • click **Import** to import the translated **.txt** text file[4]
4. from the tab **Global On timer** the developer can add specially created global 'on timer' procedures

Creating pages

Using the drop-down menu, select **Project > Objects** (insert page) or click on the respective icon **New Page**
A window will open where you can insert the name of the page[5]. The created page appears in the **Document bar** section (where it can be edited and changed)
Note: the page can be set as a **pop-up page**[6]

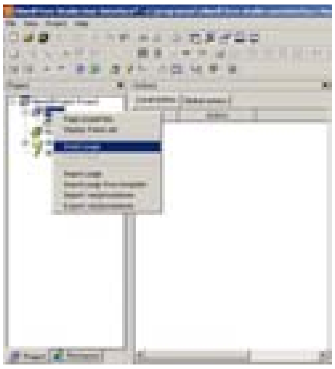
[1] **these properties are not those of the individual page but are common to all pages**
[2] **also visible from the tab Resources > Fonts**
[3] **and from the drop-down menu Project > Export language to file...**
[4] **and from the drop-down menu Project > Import language from file...**
[5] **the page name must not contain spaces (e.g. New Page returns an error; NewPage (with no spaces) or New_Page (with underscore) are acceptable names. No other characters are accepted (E.g. /, <,>, etc)**
[6] **only one page is shown on the display at any one time. A pop-up page is a page that is appears on top of another page**

Initial page

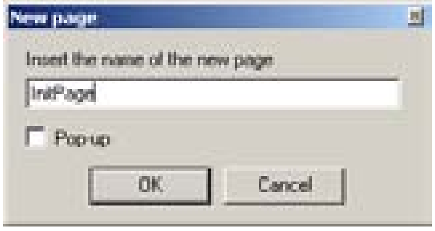
The property of 'initial' page **must** be assigned[7]
The initial page can be the page used for the 'fundamental state display' e.g. the default page that appears when the instrument is not in use. The fundamental state display can therefore be an image[8] (e.g. logo or company name) or a page that displays important values (e.g. temperature, pressure, setpoint etc.) for easy reference.
Note: if the initial page is not set, a compilation error will be returned.

Managing pages

Click the right mouse button to rename, copy or delete a page, etc.



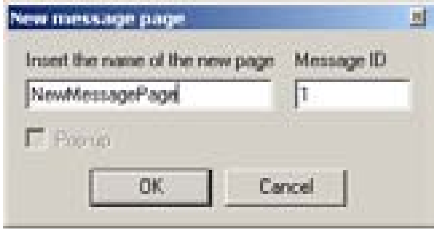
Insert page

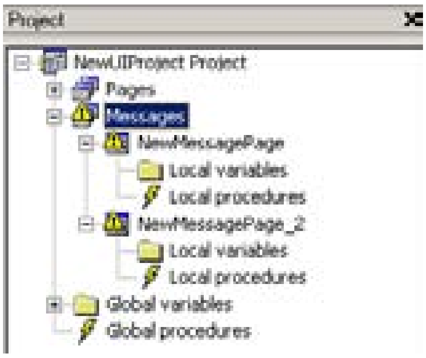


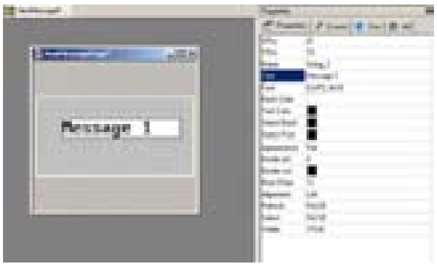
New page

Messages

From the **Project** panel, right-click the **Messages** node and select **Insert new message page** to create a new message page[9]. Message pages are similar to other pages but each one bears a unique **ID**





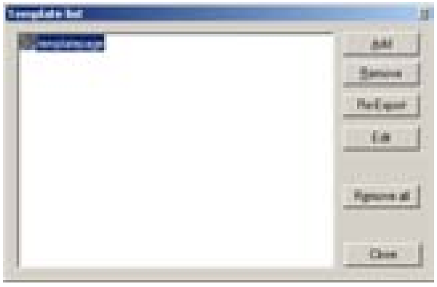


Message page **example of message pages tree** **example of message 1 page**

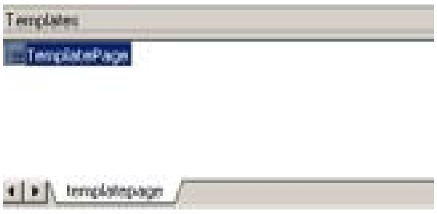
Template

From the **Project** panel, by right-clicking on a page you can export the page (ending in **.pex**), or select **Export page as template** to save the page as a template page[10]. (ending in **.petx**)
From the drop-down menu **Project > Template management** you can manage templates by adding .petx pages to the **Template List**. The added pages will also appear in the **Templates** panel as a library and can be used by dragging and dropping them in the project, creating the desired template from time to time.

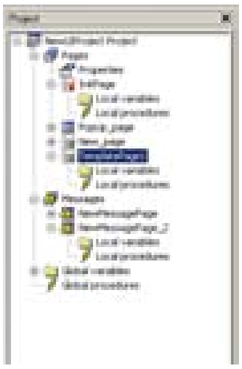
[7] it can be modified from the Pages section (node). The initial page is identified by a 'play' icon (red triangle).
[8] technically known as a 'splash'
[9] identified in the tree as a Warning triangle on a yellow background
[10] a template is a model or form that must be compiled based on a predefined structure



Template list



Template library



Instance of a template page

PROPERTIES PANEL

Every object that will be drawn on the page has properties that can be set from the **Properties** tab.
Depending on the object selected from the **Page bar** in the **Properties** tab, the elements required to correctly manage the object on the page will be shown or hidden
The complete list can be consulted in the **Online Help**. The most frequently used ones are listed here.

PROPERTIES TAB[11]

Each individual page has properties listed as follows:

	Default	Description			Default	Description
CharDimX	6	Defines the 'guides' of the page grid min(1,1) for positioning objects		Title bar	NO	Inserts a title with a preset width
CharDimY	8			Page border	NO	Inserts a page border
Font	EWP2_8x16	Character size		Caption	*	Defines the content of the **title bar**
Back color	'white'	Background colour		Appearance	Flat	Defines the page border
Text color	'black'	Text colour				
Each time that these fields are modified, an asterisk (*) appears next to the name of the page. To save the change, always use the RETURN key.						

ACTIONS PANEL

This panel enables you to define the 'actions' assigned to the **Evolution** keys for navigating the menu on the display.
Actions are divided into

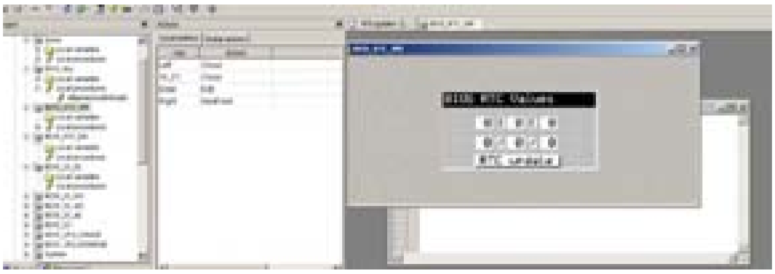
- **Local Actions**: actions that are specific to the individual page concerned
- **Global Actions**: actions that are common to all pages in the menu/project

Evolution has 4 keys: Right, Left, Up, Down and Enter. You can also associate an action to a long press of each of the 4 keys[12] (identified by the prefix **Long**, e.g. **Long**Up).

In addition to the 4 keys there are 10 virtual keys identified with **VK_F1**, ..., **VKF10**[13]
The virtual keys are associated to a **procedure** written in **ST** language

[11] do not confuse with the corresponding item 'Properties' in the Project tab, Pages section (node)
[12] long press of about 3-5 seconds
[13] also see UserInterface manual (Online Help)

For example, suppose that we want to set the time of the RTC clock. We create the relative page



We associate the page to a virtual key **VK_F1 (Close)** in Local Actions.
The procedure in **ST** language is as follows:

```
sysClock_update := TRUE;
uint_ret := Video_SendEvent[14](kWM_KEY,kKEY_VK_F1);
```

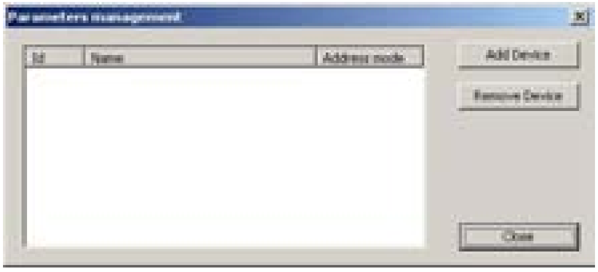
To insert an action, click the right mouse button and select **Insert action**. The selected action is highlighted yellow. Selecting **Remove action** will delete the action highlighted in yellow
Each **key** can be associated to the following actions

Action column	Description	Notes
Call	call of a procedure	to call a procedure simply write in the **Link** column the name of the procedure you want to call
OpenPage	opens a page	write in the **Link** column the name of the page you want to open
Close	closes the page	pair with the relevant key. Note: automatically returns to the start page
NextField	moves to the next value in a field	
PrevField	moves to the previous value in a field	
Edit	enables you to edit, e.g. write a value in a field	

[14] the standard functions predefined as VideoSentEvent, etc. (to be handwritten) are described in the UserInterface manual

APPLICATION FIELDS

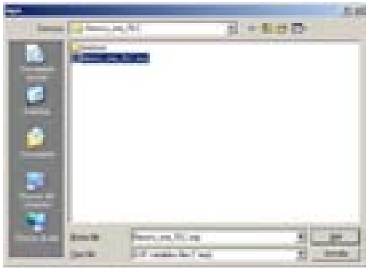
From the menu **Project > Parameter management**[15] you can link the parameters of an application developed in **Application**



Click **Add Device** to import the desired parameters file (**.parx**). Modbus or EWDMI addressing can be associated to each set of parameters.
The set of parameters will be shown in the appropriate tab of the **Target vars and parameters** panel

PLC VARIABLES

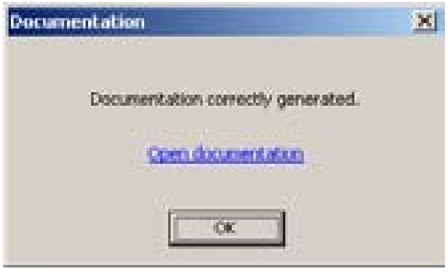
From the menu **Project > Parameter management** you can link the variables of a PLC application developed in Application



Import the desired 'EXP variable' file (**.exp**). The set of parameters will be shown in the relative **PLC vars** tab in the **Target vars and parameters** panel

GENERATING DOCUMENTATION

Using the icon **Generate Doc.** you can create documentation linked to the **Doc** fields (**Properties** panel, **Doc** tab)



Documentation correctly generated.

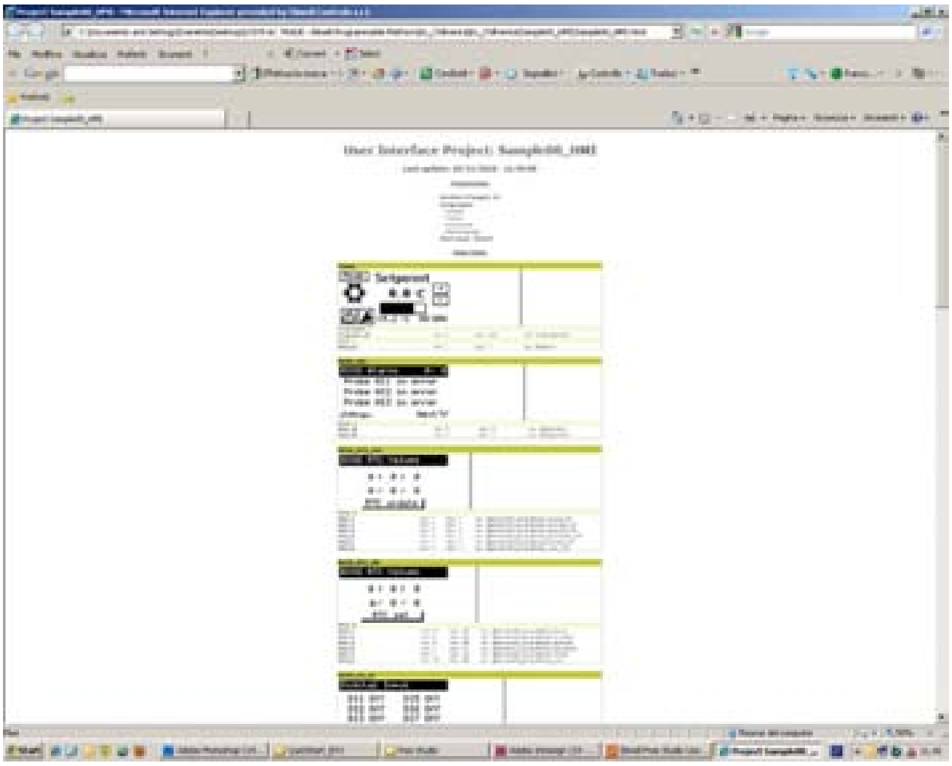
Through a **browser**[16] you can view the generated documentation by clicking on the link Open documentation.
the documentation gives all information relating to the project:

- number of pages
- program languages
- initial page (name of initial page)
- visual list of pages (as they are presented on the display)

[15] or from the corresponding Parameter management icon

[16] Internet Explorer or equivalent

Example of documentation generated using the icon **Generate Doc.**



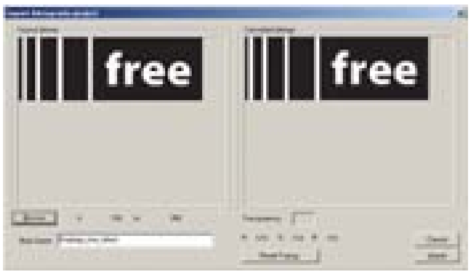
PROJECT PANEL / RESOURCES TAB

In this tab the following elements (nodes) can be used:

- **Fonts** for managing pages (create, delete, import/export etc.)
- **Bitmaps** for managing images to be inserted in the pages
- **String table** for managing strings to be used in the various pages
- **Enumeratives** for managing enumerable sets to be used in the various pages
- **Image lists**
- **Sets** for managing sets of strings / parameters to be used in the various pages

Bitmaps

UserInterface can be used to import images (specifically **bitmaps**, such as icons, logos etc.) to be used in the menu pages. Click on the node **Bitmaps > Import bitmap.** The corresponding window appears for importing the image. Click the **browse** button to select the image and the Import button to import it. Example of import of **free** logo.



We recommend that you import black/white images of suitable size. The display can support an image of approximately 100x50px. The imported bitmap will be in the **Bitmaps node**[17]

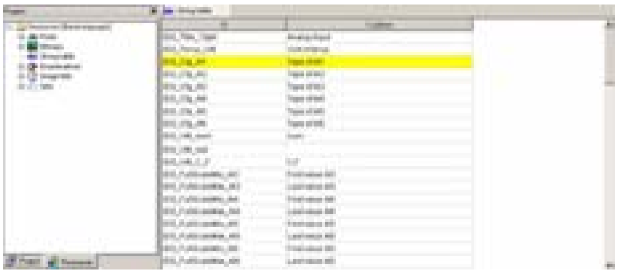
To insert the bitmap in a page:

- insert an 'image' box (menu Page > New Image or use the corresponding icon)
- define the dimensions of the image from the **Properties** panel using **XDim** and **YDim** (e.g. 100x50px)
- again from the **Properties** panel select the bitmap from the field of the same name

[17] **if not renamed it will be assigned a default name consisting of the image name preceded by the prefix bmp (e.g. logofree-> bmplogofree)**

String table

UserInterface allows you to define strings that can be used when creating pages.
Using the menu **String>Insert** or the respective icon **Insert Record** you can create an association[18] between a string and its corresponding description.
In the **ID** field the string is defined as **IDS**_stringname. [19]
In the **Caption** field enter a description of the string.



By selecting the page from the menu **Project/Project** you can associate the desired string to a text field using the menu **Properties>Caption**[20]. The text that will appear on the display will be the text inserted in the description.[21]

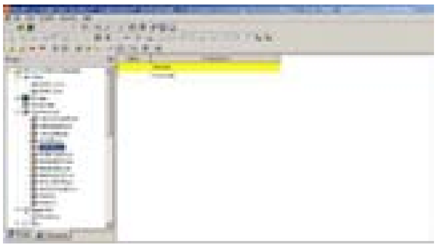
Enumeratives

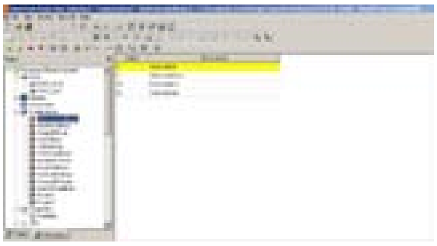
UserInterface also allows you to define strings by means of enumerable sets that can be used when creating pages.
An enumerable set is an array of values that are not necessarily consecutive. Each value is associated to a description that will be shown on the **Evolution** display.
Move onto the **Enumeratives** node and use the menu **Record>Insert** or the respective icon **Insert Record** to create the sets that you want to use[22]
Each set will be shown as a subnode of **Enumeratives**.
Move onto the corresponding node and use the menu **Record>Insert** or the respective icon **Insert Record** to enter the array values
After creating (or defining) the enumerative from the **Resources** tab, open the **Project** tab and, selecting the page / element of the desired page, you need to associate the corresponding **Enum** to the Format field
The three examples show

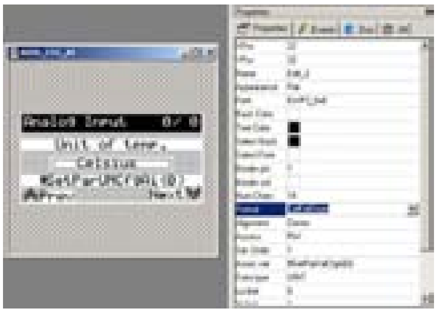
- **definition** of an enumerative **CelFarEnum** consisting of 2 values for defining the degrees of temperature (0=Celsius, 1=Fahrenheit).
- **definition** of an enumerative consisting of 4 non-consecutive values for defining USB commands (values 8,9,10,12)
- use of **CelFarEnum** on a page

The order of the elements can be changed with the icons **Move Up ▲** / **Move Down ▼**

[18] technically referred to as 'transcoding'
[19] you are recommended to use this prefix to prevent the descriptions that appear on the display being confused with the transcoding in UserInterface
[20] in practice it is also possible to associate images but for the sake of simplicity we have only dealt with text fields here
[21] if the Description field is empty, the string IDS_xxx will appear
[22] the default name is Enum1, Enum2, etc. Select the box to change the name







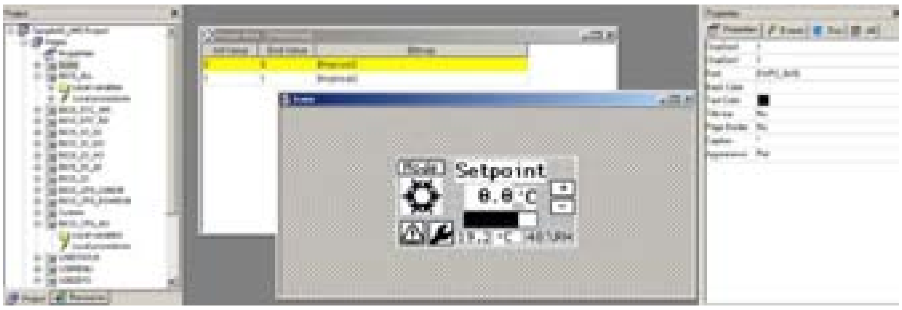
definition of CelFarEnum **Enum with non-consecutive values** **use of CelFarEnum**

Image Lists

Similarly to **Enum** it is possible to create lists of images associated to ranges of values. The image shown on the display will change depending on the state of the variable associated to the list.
A list is created and a **bitmap** (see corresponding node) is associated to each image, indicating its reference range.
After creating (or defining) the list from the **Resources** tab, open the **Project** tab and, selecting the page / element (created with '**New animation**') of the desired page, you need to associate the bitmap list to the Imagelist field
The three examples show

- **definition** of a **Modeimg** list of 2 bitmaps for displaying Cool mode and Heat mode. The range of values is reduced to one value in this case (0 for Cool, 1 for Heat)
- use of **Modeimg** on an 'Icons' page, associating the **Modeimg** list with the animation. A state variable must obviously be associated, depending on which the image will change

However a static bitmap must be inserted as an image via **Page > New Image** or using the respective icon



definition of ModeImg **use of ModeImg**

Sets

Often there is the need to create a group of very similar pages (for example a scroll menu for showing a set of parameters in the same folder)
UserInterface allows you to define **sets** that can be used when creating this type of page, without having to replicate the page **n** times.
A set is an array of strings or variables/parameters[23]
Move onto the **Sets** node and use the menu **Record>Insert** or the respective icon **Insert Record** to create the sets that you want to use[24]
The type of set can be

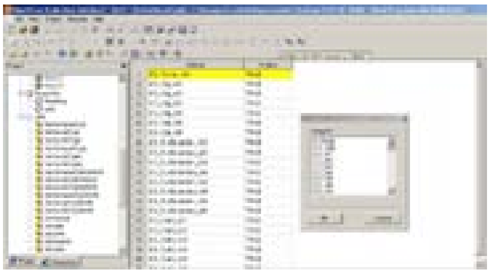
- **String** e.g. a string defined in a **string table**
- **Variant** e.g. a variable/parameter

Each set will be shown as a subnode of **Sets**.
Move onto the corresponding node and use the menu **Record>Insert** or the respective icon **Insert Record** to enter new items in the array

[23] including any alarms
[24] the default name is Set1, Set2, etc. Select the box to change the name

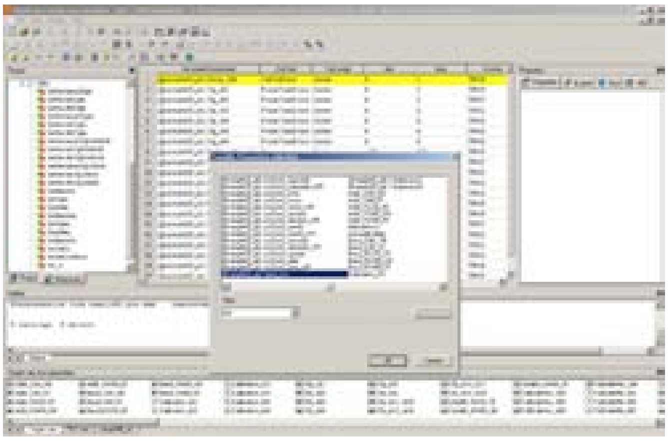
Strings

In the case of a **String** type **Set** the string is selected by moving onto the box and clicking on the three dots (...)
The **Resource string selection** window appears, which shows the available strings[25] that can be associated to the array
The visibility (**Visible** column) for strings is TRUE - not modifiable



Variant

In the case of a **Variant** type **Set**, the variable is selected[26] in the same way by moving onto the column **Variable/Parameter**
The **Variable/parameter selection** window appears, which shows the available variables that can be associated to the array



Using the Filter option you can filter the display of the variables/parameters

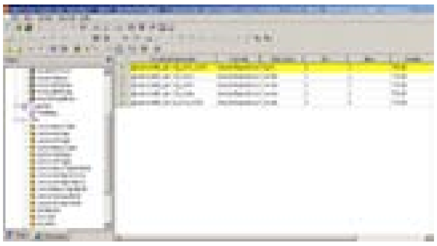
- The **Format** column shows the default U.M. of the parameter. The user can manually associate an **Enum**
- Similarly the Min Max columns show the default range of the parameter, which can be modified by the user[27]
- The **Text align** column aligns the string to the left, right or centre.
-

After creating the Set from the **Resources** tab, open the **Project** tab
Previously a page was created that you want to replicate using **Sets**
(for example in order to manage the analogue outputs **BIOS_CFG_AO**)
Selecting the appropriate field, you need to associate the corresponding Set to the field **Assoc_var** using the prefix **#,** and with the suffix **(0)** to indicate the first element of the array. Note: The field must be of the **Edit** type
(for example in order to manage the analogue outputs **#SetParValCfgA0(0)**)

The three examples show
- **definition** of a Variant Set **SetParValCfgA0** of size 5 to define array of output configuration parameters
- **definition** of an enumerative **AnaOutTypeEnum** of 3 values (voltage, current, ON/OFF)
- **use** of **#SetParValCfgA0(0)** on a page (which results in the production of 5 pages that differ in terms of the configuration of parameter **SetParValCfgA0(1),..., SetParValCfgA0(5)**[28]
- note that the Edit field 'IO Modulated' is associated by means of the Format field to the Enumerative **AnaOutTypeEnum**

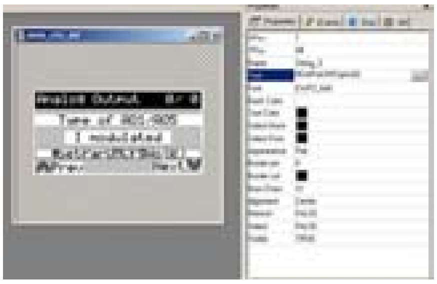
Note that in the example there is another Edit field associated to a **Set**.
It is possible to use multiple **Sets** on the same page. You are advised to define **Sets** of the same size if used on the same page.

[25] previously defined in the string table (see)
[26] in this case the term 'variable' refers to genuine variables in their own right, PLC parameters, BIOS, alarms etc.
[27] e.g. for a parameter that defines a COOL, HEAT and AUTO machine mode it is possible to hide one or more values at UI level even if at PLC (or application level) there are 3 usage modes
[28] note that the array starts from the value (0) whereas when defining Sets the first value is 1, therefore Set(1) corresponds to #Set(0)



SetParValCfgA0

AnaOutTypeEnum



use of #SetParValCfgA0()

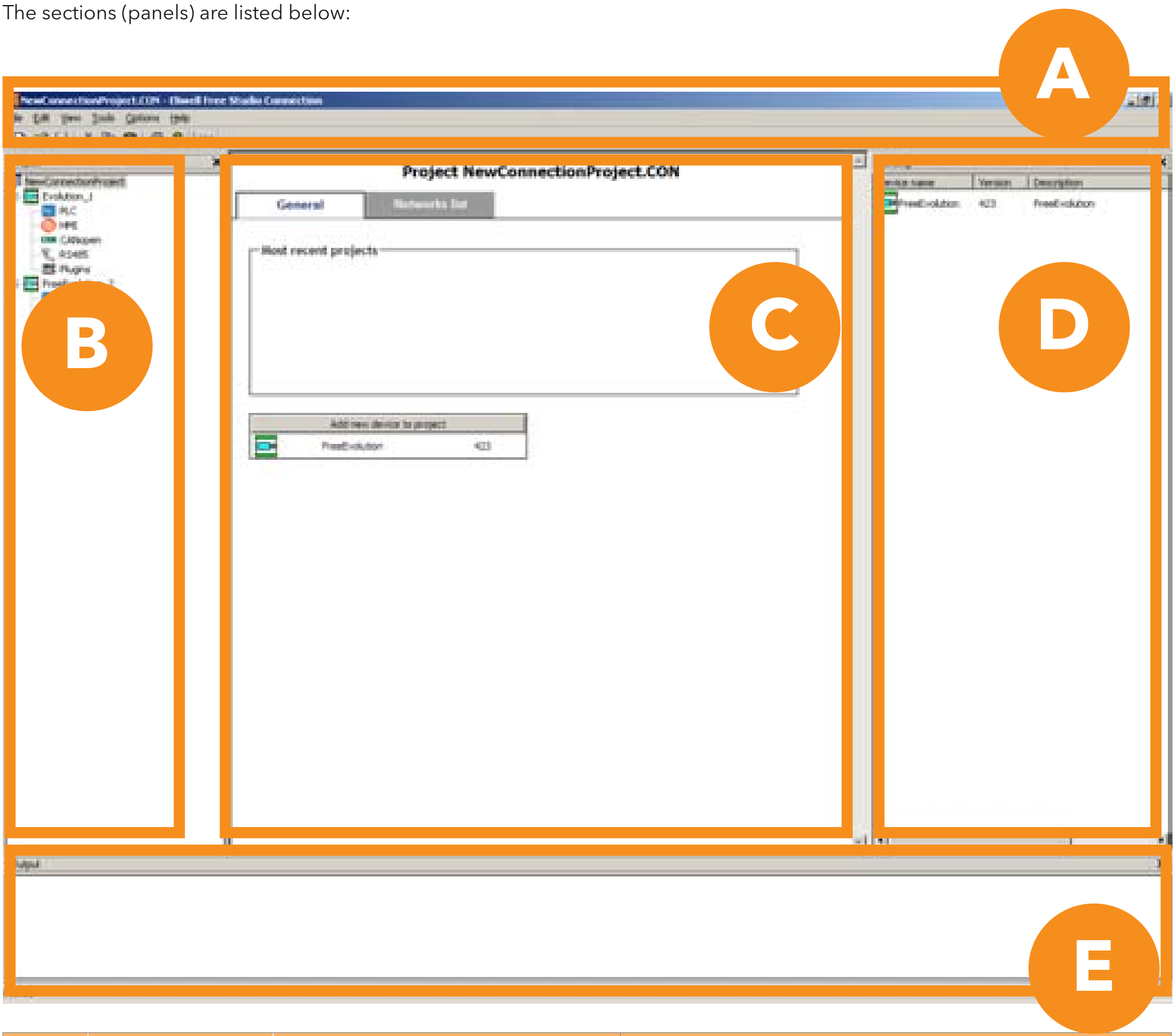
CONNECTION PANEL DEFINITIONS

free Evolution

The **Connection** work environment dedicated to developers has various sections for creating and managing **Evolution** devices, as well as for compiling the entire project, e.g. as mentioned in **Target**

- creating the **PLC** application using **Application**
- defining the graphical interface of **Evolution** using **UserInterface**
- compiling the **Build project**
- downloading the project via **Device**

The sections (panels) are listed below:


A
NewConnectionProject.CON - Eliwell Free Studio Connection
File Edit View Tools Options Help
B
NewConnectionProject
Evolution_1
PLC
HMI
CANopen
RS485
Plugins
FreeEvolution_1
Project NewConnectionProject.CON
General
Networks list
Most recent projects
Add new device to project
FreeEvolution 423
C
Device name
Version
Description
FreeEvolution 423 FreeEvolution
D
E

Section	Panel /Section name Application	Panel	Description
A	/	**Toolbar (Main bar)**	This section shows the work tools in the form of icons. Some tools can be selected by means of key combinations (**shortcuts**). See relevant chapter for details
B	**Project**	**Project (Workspace)**	The **Project** panel can be used to manage the project and one or more **Evolution*** devices • from the project you can manage the device network (**Network list**) • The various protocols are configured In each **device** - see panel **C**

Section	Panel /Section name Application	Panel	Description
C	/	Configuration panel (Edit)	This section contains • **General** project management tab • **Network list** tab Depends on the selected **Evolution** device • **General Evolution** configuration tab In the **Evolution** device • **PLC** for configuration of IEC application developed with **Application** • **HMI** for configuration of **User Interface menu** • **CANopen** for configuration of corresponding protocol • **RS485** for configuration of corresponding protocol • **Plugins** for configuration of protocols using plug-in
D	Catalog	Device catalogue	This panel shows all the devices available from the catalogue. Note: Dynamic visibility of devices based on selections (e.g. **plugins**)
E	Output	Messages	This panel shows the messages relating to the development of the project (file opening, reading/writing errors, status of connection to device, etc.) **Note**: the connection to the target device is also visible in the bottom right of the screen by means of coloured messages (see Application Download)

***Connection** manages one or more **Evolution** devices
Smart can be inserted in the network as a **slave** device via **RS485**

The display of the panels can be set from the drop-down menu **View > panel name** and the panels can be organised as the user wants in terms of size and layout.
Studio saves the last settings made

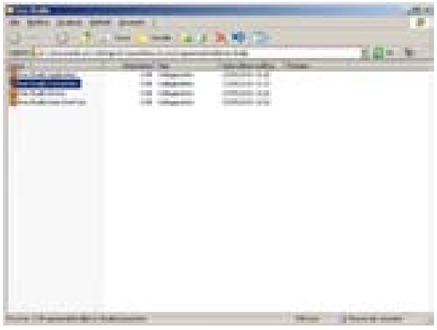
The panel **View > Compiler toolbar** is an icon that is visible or hidden in the **Main menu**

TO START CONNECTION

free Evolution

FIRST START-UP

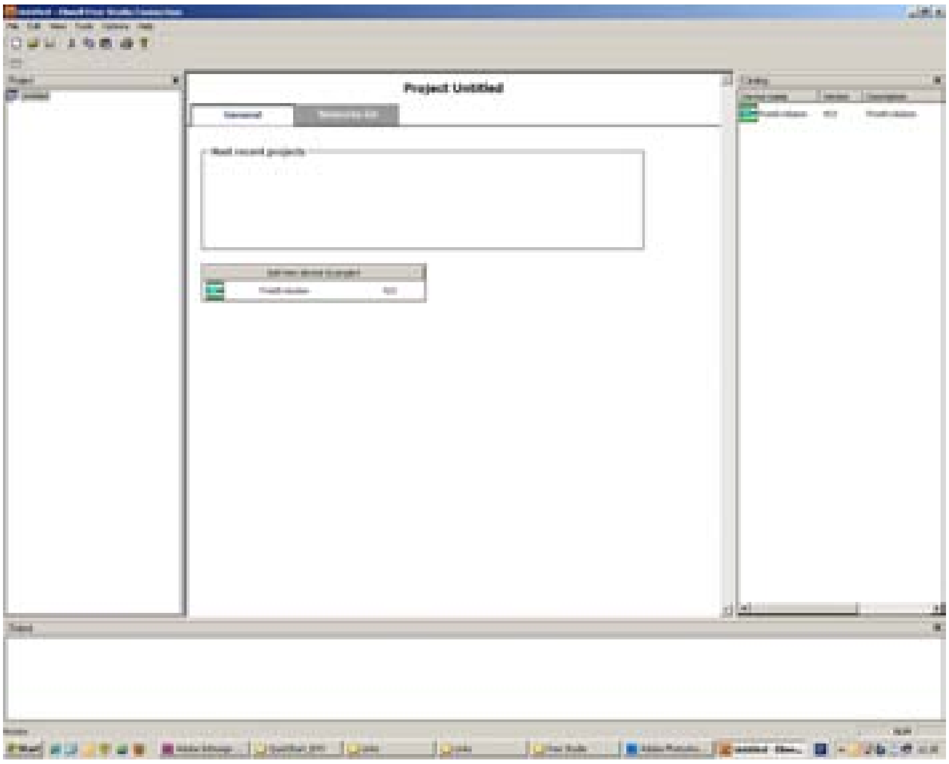
1. Click on the **Launch FREE Studio** icon on your Desktop
2. Click on the **FREE Studio Connection** icon.



1

2

The program appears as shown in the picture:



You can now:

- Add a new **device** to the project
- Add/remove a network from the **Network list**

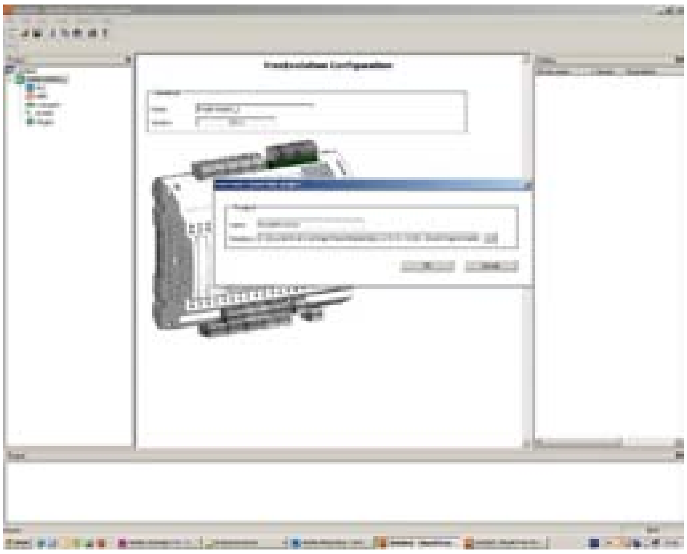
Existing projects have the extension **.con**

CREATING A NEW PROJECT (WORKSPACE)

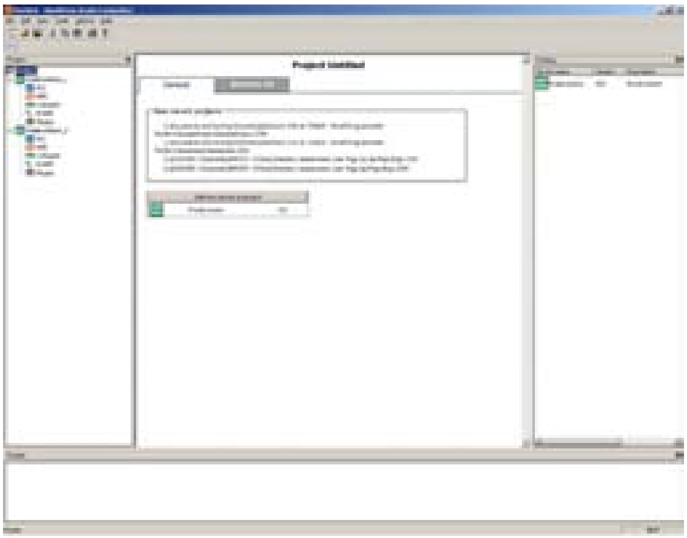
To create a new project, simply save the project Untitled[1] with a name of your choosing **(CTRL+S)**
To add a device to the project, click **Add new device to the project** or simply drag and drop the **FreeEvolution** icon from the **Catalog** panel onto the project name.
You are advised to rename each **Evolution** device with an appropriate name (the default name is **FreeEvolution1**,

[1] **saving the project creates in your PC a new folder with the chosen name that functions as the Workspace**

FreeEvolution2, etc.)
In the workspace you can see the name of the device and the version)[2]



creating a new project CTRL+S



adding Evolution to the project
Note example of adding 2 Evolution devices

PROJECT PANEL

Each **Evolution** device has the following resources, which are shown as nodes of the **target**

colour	description	what to do	Notes
PLC	PLC project	move onto the node and click the right mouse button to open **Application**	see Getting Started (**Application**)
HMI	menu for creating pages	move onto the node and click the right mouse button to open **User Interface**	see Getting Started (**User Interface**)
CANOpen*	on-board **Evolution** serial as standard	select the **Mode** and add the device from the catalogue	always available
RS485*	on-board **Evolution** serial models **7500** and expansion modules **EVE75xx**		check availability
Plugins**	optional modules available separately		check availability
* also available as serials on plug-in module			
e.g. **Ethernet plug-in			

You can create both **PLC** and **HMI** from scratch or import existing projects and projects from the catalogue

PLC project

From the panel **Project > PLC node** click the right mouse button and select Open with **Free Studio Application**
By default you will be prompted to save the project **FreeEvolution_n_PLC**
See **Application** for information on development. To summarise:

- develop the PLC code with **Application**
- create parameters and state variables of the PLC application
- compile and download application onto **Evolution** and **debug** if necessary
- close **Application** and return to **Connection**

The saved project .**ppjs** is located in the **Connection** folder / workspace

[2] the catalogue can contain multiple Evolutions with different versions

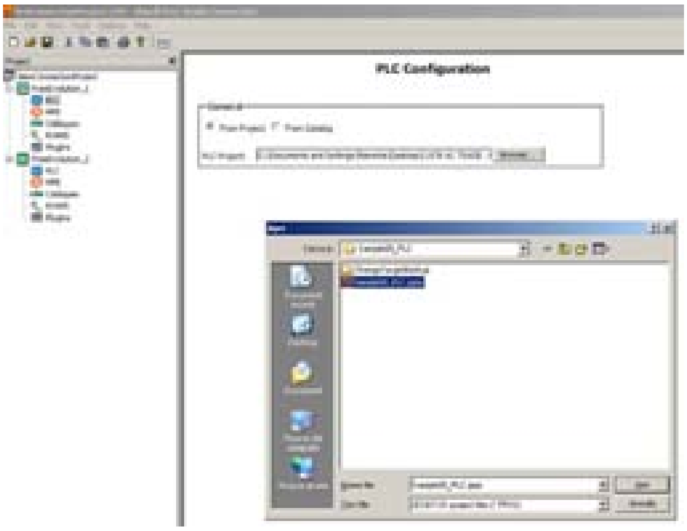
HMI project

From the panel **Project > PLC node** click the right mouse button and select Open with **Free Studio UserInterface**
By default you will be prompted to save the project **FreeEvolution_n_HMI**
See **User Interface** for information on development. To summarise:

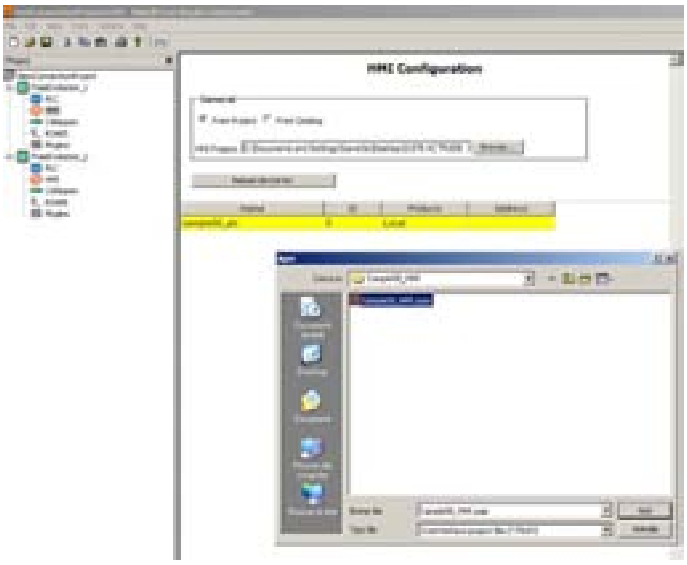
- develop the **HMI** menu or pages with **UserInterface**
- the parameters and state variables of the previously created **PLC** application link to the menu and will be visible in section **H** of **UserInterface Target vars & parameters**
- check the connection mode, appropriately configuring the connection parameters
- compile the **HMI** pages and download them onto **Evolution**
- close **UserInterface** and return to **Connection**

The saved project .**pajx** is located in the **Connection** folder / workspace

The device will be added to a list (highlighted in yellow)
Press the Reload Device List button to populate the table with the list of devices connected to the **HMI** project and to use the pages remotely if so required



PLC Configuration
adding an existing project



HMI configuration
adding an existing project and reloading devices list

Download project

Use the menu **Tools>Build** or **F7** to create the .**CFN** configuration file
From the panel **Tools > Open with Free Studio Device**, open **Device**
After configuring the device, e.g. the **BIOS** communication parameters, you will be able to download:

- **PLC** application
- **HMI** pages
- configuration file **Cfg files**

using the **Download All** button
Note: each of the 3 files can be downloaded independently from the other two by moving the mouse pointer onto the respective icon, clicking the right mouse button and running the corresponding **download**.

ELIWELL SUPPORT

ELIWELL TECHNICAL SUPPORT

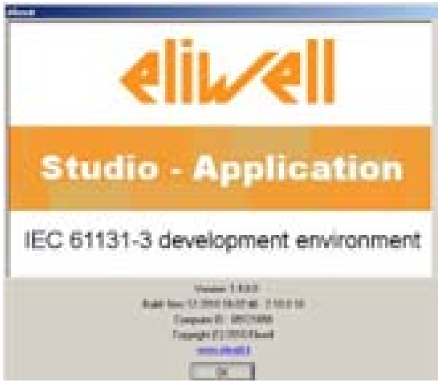
Eliwell provides access to fully trained, professional technical personnel to answer its customers' questions precisely and reliably.
Our technicians are on hand every day, to respond to any technical requirements relating to the installation and operation of all Eliwell instruments.

If you have any questions or require support, contact Eliwell Technical Support:

Tel. +39 0437 986 250
E-mail: **eliwell.freeway@invensys.com**

Before contacting Eliwell Technical Support please check the software version of Eliwell **FREE Studio**.

From the drop-down menu select **Help > About**
The following screen appears[1]:



Studio - Application

IEC 61131-3 development environment

Quote software **Version**, Build and Computer ID

ONLINE HELP

During each program development stage, pressing the **F1** key opens a **pdf** document containing the entire Online Help **in printable format**

free STUDIO

Development software for free Smart

PROGRAMMABLE CONTROLLERS

The software solution in an integrated suite for quick, easy programming

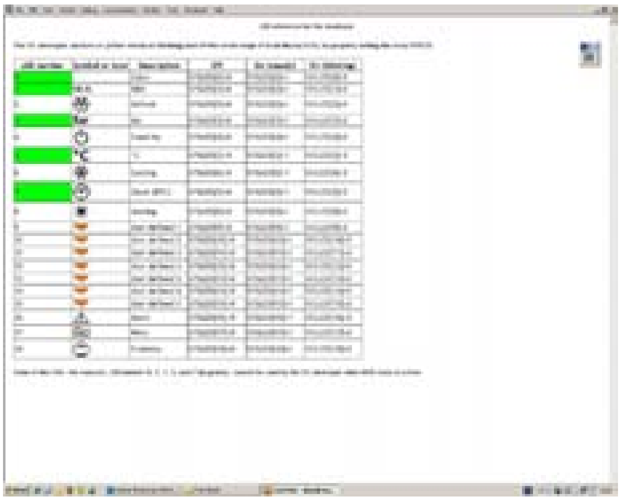
Help On Line

FREE Studio offers developers On-line Help to assist programmers with each stage of program development

[1] **Example for Application. Same screenshot for Device. For Evolution users, the screenshot is the same for UserInterface and Connection.**

ONLINE HELP - LED[2]

From the panel **Project>Resources>Help** access the section **LED reference for the developer[3]** which contains a description of the characteristics of the icons used in **FREE Smart** and how to manage them using **target variable sysLocalLeds[**])



WEBSITE RESTRICTED AREA

The Eliwell website **www.eliwell.it** includes a restricted area, access to which is by password only.

Once accredited to access the restricted area with a personal Username and Password, users can view documents, information, product updates and software which is not openly visible on the site.

Click **here** to visit the respective web page and obtain all the necessary information for registering

The FREE Studio documentation is available to registered users in the section **Applications > Programmables > FREE WAY > FREE Studio**

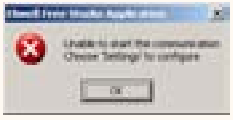
Software updates are available to registered users in the section **Technical Support > Programmables Area > FREE WAY > FREE Studio**

[2] only available for Smart.
[3] see full screen (CTRL+U)

TROUBLESHOOTING

The tables below show some examples of error messages and the corrective action to be taken.
For messages not shown here, contact Eliwell Technical Support or consult **www.eliwell.it**

APPLICATION

Error message	Panel	Effect	Resolution
 Eliwell Free Studio Application Unable to start the communication Choose Settings to configure OK	Communication	target not connected	Check settings of COM port using drop-down menu Communication > Settings EWDMI Properties
Invalid memory image file. Please upload memory image from the target	Output	compilation error	connect **FREE Smart** target device to PC by means of DMI

DEVICE

Error message	Panel	Effect	Resolution
invalid connection string	Connection Status	target not connected	Check communication settings using menu Communication > Settings EWDMI Properties or by changing string in menu Communication > Advanced

SHORTCUTS

The table shows some of the key combinations (**shortcuts**) available for enabling/disabling the main functions of **Application**.
For **Device** see next table and/or Notes column.
Connection and **User Interface** have shortcuts for opening, saving and compiling projects
The '**toggle**' function indicates that the function is enabled if disabled and vice versa by using the same sequence of keys.
For example: CTRL+U enables Full Screen; pressing CTRL+U disables it again[1]

The Online Help is activated using the **F1** key

Menu	Function	Function	Shortcut	Notes
ALL	**ONLINE HELP**	**HELP**	**F1**	**ALWAYS AVAILABLE printable HELP in PDF format**

APPLICATION SHORTCUTS

Menu	Function	Function	Shortcut	Notes
Edit	Print	Print	CTRL+P	***
Edit	Cut	Cut	CTRL+X	***
Edit	Copy	Copy	CTRL+C	***
Edit	Paste	Paste	CTRL+V	***
Edit	Save	Save	CTRL+S	***
View > Toolbars	Full screen	Full Screen	CTRL+U	Toggle***
View > Toolbars	FBD bar	FBD BAR	CTRL+D	Toggle
View > Toolbars	LD bar	LD BAR	CTRL+A	Toggle
View > Toolbars	SFC bar	SFC BAR	CTRL+Q	Toggle
View > Toolbars	project bar	PROJECT BAR	CTRL+J	Toggle
View > Toolbars	network	NETWORK	CTRL+M	Toggle
View > Toolbars	document bar	DOCUMENT BAR	CTRL+N	Toggle
View > Tool Windows	workspace	WORKSPACE	CTRL+W	Toggle
View > Tool Windows	output	OUTPUT	CTRL+R	Toggle***
View > Tool Windows	library	LIBRARY	CTRL+L	Toggle
View > Tool Windows	watch window	WATCH WINDOW	CTRL+T	Toggle***
View > Tool Windows	asynch graphic window	ASYNCH GRAPHIC WINDOW	CTRL+K	Toggle***
PROJECT	COMPILE	COMPILE	F7	if more than one project, compiles only selected project
COMMUNICATION	download code	DOWNLOAD CODE	F5	
DEBUG	add symbol to watch	ADD SYMBOL TO WATCH	F8	when variable is selected (drag & drop not necessary)
DEBUG	Inserts new items into watch	INSERTS NEW ITEMS INTO WATCH	SHIFT + F8	chosen from a list

[1] in Device, the display of the panels is defined by the View menu, by checking individual headings or leaving them unchecked

Menu	Function	Function	Shortcut	Notes
DEBUG		ADD SYMBOL TO A DEBUG WINDOW	F10	
DEBUG		INSERTS NEW ITEMS INTO A DEBUG WINDOW	SHIFT + F10	
DEBUG		ADD/REMOVE BREAK POINT	F12	
DEBUG		ADD/REMOVE TEXT TRIGGER	F9	
DEBUG		ADD/REMOVE GRAPHIC TRIGGER	SHIFT + F9	
DEBUG		REMOVE ALL TRIGGERS	CTRL+SHIFT + F9	
DEBUG		Trigger list	CTRL+I	
*** also applies to **Device**				

DEVICE SHORTCUTS

Menu	Function	Function	Shortcut	Notes
Edit	Print	Print	CTRL+P	***
Edit	Cut	Cut	CTRL+X	***
Edit	Copy	Copy	CTRL+C	***
Edit	Paste	Paste	CTRL+V	***
View > Toolbars	Full screen	Full Screen	CTRL+U	Toggle***
View > Tool Windows	output	OUTPUT	CTRL+R	Toggle***
View > Tool Windows	watch window	WATCH WINDOW	CTRL+T	Toggle***
View > Tool Windows	asynch graphic window	ASYNCH GRAPHIC WINDOW	CTRL+K	Toggle***
PARAMETERS	read selected	READ SELECTED	CTRL+SHIFT+R	The selected parameters are highlighted in yellow
PARAMETERS	write selected	WRITE SELECTED	CTRL+SHIFT+R	
PARAMETERS	select all	SELECT ALL	CTRL+A	Operation to be performed with care The next operation could be writing all parameters and/or resetting the default values
*** also applies to **Application**				

USER INTERFACE SHORTCUTS

Menu	Function	Function	Shortcut	Notes
CONFIGURATOR BAR	new project	NEW PROJECT	CTRL+N	***
CONFIGURATOR BAR	open project	OPEN PROJECT	CTRL+O	***
CONFIGURATOR BAR	generate code (compile)	GENERATE CODE (COMPILE)	F7	
*** also applies to **Connection**				

CONNECTION SHORTCUTS

Menu	Function	Function	Shortcut	Notes
MAIN BAR	new project	NEW PROJECT	CTRL+N	***
MAIN BAR	open project	OPEN PROJECT	CTRL+O	***
MAIN BAR	Save	Save	CTRL+S	
*** also applies to **User Interface**				

PROJECT 1 - COUNTER

In the pages that follow, we will guide you through the procedure for writing, testing and downloading to **FREE Smart** a simple COUNTER program written in ST language, which performs a count and displays a 'saw-tooth' on the oscilloscope.

The COUNTER program counts the local variable **a** from 0 to MAXSCALE1. When **a** reaches MAXSCALE1 it will be re-initialised to -100.

As previously described in **First Start-up**,
- start **Application**,
- create a new **COUNTER** project
- create a **COUNTER** program in **ST** language

In the central **Edit** section, write the **COUNTER** program code in **ST** language.

COUNTER program

```
if a>MAXSCALE1 then

a:=-100;

else

a:=+1;

b:=a*10;

end_if;

count:=a;

counter1:=b;
```

Local variables

In the variables section we create the DINT type local variables **a** and **b** (use the drop-down menu **Variables > Insert**, or the icon **Insert record**) and associate the **Init Value** 0 (zero)
The variables **a** and **b** are of no use outside the COUNTER program, and are therefore defined as local

Variables and global constants

In the program, we have also included a constant which we will define as 'global':
- the constant MAXSCALE1 must be created in the panel **Project >Global Variables Constants**, which we will create here in the same way as we create a local variable. MAXSCALE1 has a preset value of -100 so it is not necessary to define it as a variable but as a constant

Global shared variables

Should you want to display the result of the counter on the **target**, in the program we have also included 1 variable which we will define as **Global shared**[1]:
The variable **count,** which we will create, is obtained from the panel **Resources>Modbus Objects > Status variables** and not from the section Edit/Variables, when we enter a new record.
In this case it will be necessary to also define how the name of the variable will appear on the **FREE Smart** 7-segment display (e.g. label of up to 4 digits, for example **CNT** for `count`) and mapping between **Application Type** and **Device Type**

In the COUNTER program we have included another variable, `counter1`,[2] which we will define in the same way.
In this case, we do not define how it appears on the display. The software will propose **COUn** by default (instead of COUnter), in other words it will use the first 4 letters on the 4-segment display.

[1] **note that the folder 'Global shared' appears if, and only if, we define the variable in the tab Resources>Modbus Objects > Status variables**

[2] **note that if we had created a variable** `counter` **and not** `counter1` **at the compilation stage, we would have had an error (name conflict between COUNTER program and variable)**

Fundamental state display

By clicking on **Resources>FREE Smart** we choose what to show as the fundamental state display[3]
We choose the variable count (which is the value of the local variable 'a')
We update the program by entering an always active or always inactive alarm

```
if alarm then

alarm1:=0;

else

alarm1:=1;

end_if;
```

The alarm variable must be defined as a local variable - of BOOL type (Boolean).
FALSE is proposed by default as the initial value - to change it simply key in **TRUE**.
The alarm will be always active. We can enter a description such as 'Alarm always active'
Alarm1 must be defined in the panel Resources>Alarms. As a label we enter ALM1

Menu creation

At this point we can create

- the customised menu PRG1 for entering any parameters defined by the developer (or use the existing BiOS parameters)
- the customised menu SET1 for entering the 2 variables count and counter1

Task

If we go back to the Project tab and select the task Timed we can add the program COUNTER.

Compilation

We compile the project - drop-down menu **Project >Compile** (or F7)
Note: the project is compiled without errors and without the need to use a **FREE Smart** connected to the PC

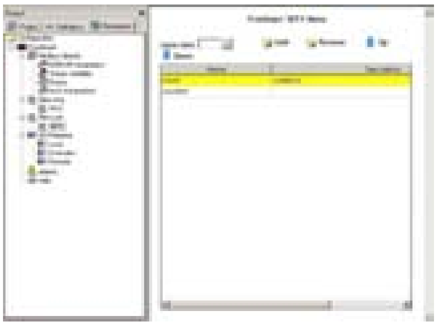
To summarise...
... see screenshots

COUNTER program - Project tab

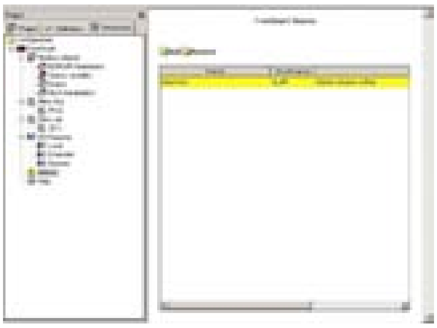
COUNTER program - Resources tab fundamental state display

COUNTER program - Resources tab define variables

[3] as you can see, the choice includes the default I/O + the variables created by the developer



COUNTER program - Resources tab
Set menu (SET1)



COUNTER program - Resources tab
Alarms menu

COUNTER program - Resources tab
definition of Variables

Notes

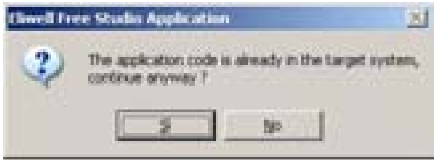
in the Project tab, the global shared variables appear only after they have been declared in the Resources tab. In the example:

- Alarms ➔ alarm1
- Parameters➔ PAR1
- Variables ➔ count and counter1

Connecting to FREE Smart and downloading the application

After connecting FREE Smart to the DMI and PC on the USB port and making the appropriate checks on the COM and baud rate (see relevant chapter) we can connect to the target:

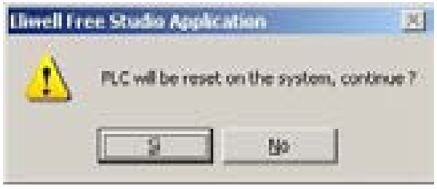
- from the drop-down menu **Communication > Connect** or icon **Connects to the target** connect to **FREE Smart**
- if connection is successful, the message CONNECTED appears in green at the bottom right of the screen[4]
- from the drop-down menu **Communication > Download code (F5)** or icon **Code download** now download the application to **FREE Smart**
- one or more dialogue boxes[5] will appear, indicating that the application will be downloaded. Downloading could take a few seconds



Eliwell Free Studio Application

The application code is already in the target system, continue anyway ?

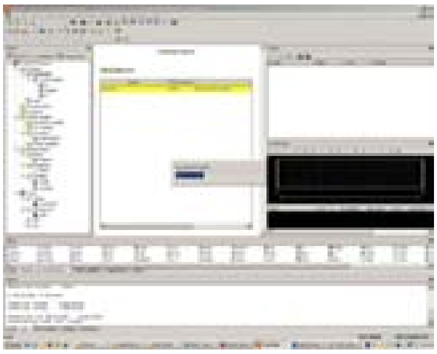
IEC application already on target
confirm to continue



Eliwell Free Studio Application

PLC will be reset on the system, continue ?

PLC to be reset confirm replacement



application download in progress

The application is not operating on FREE Smart

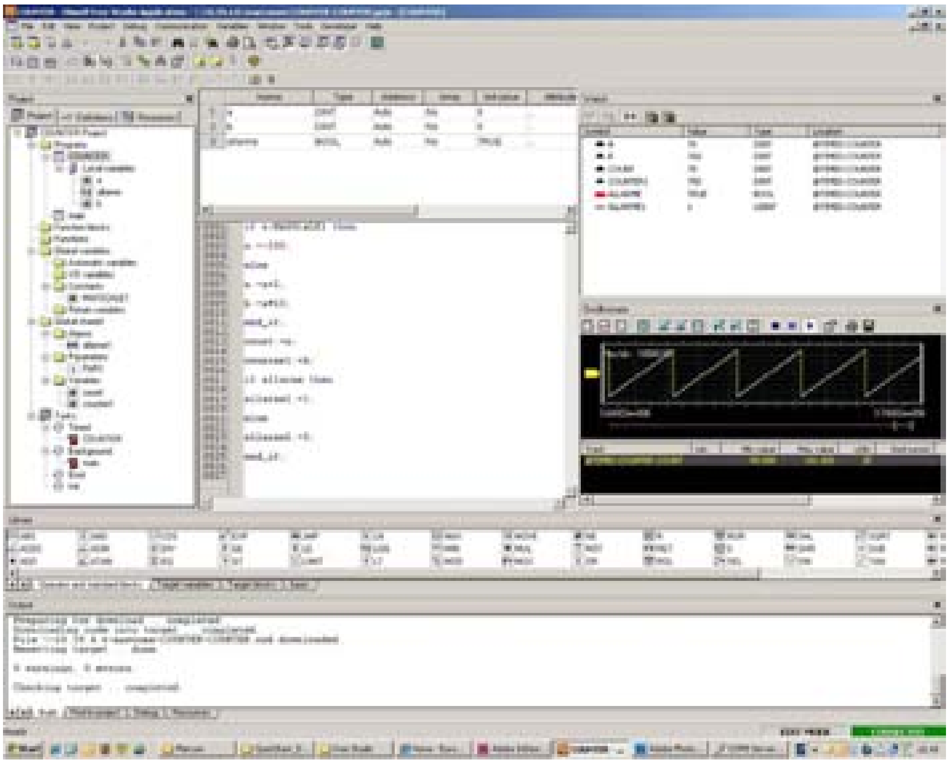
[4] should this not be the case, see Download on target or Troubleshooting
[5] depending on whether the target has already been programmed or not

Watch and Oscilloscope

On the display of the target device, the count value appears[6] which ranges from -100 to +100.
Note that the alarm LED appears because the alarm is active
In the Status menu, we can check for the presence

- of the folder SET1 with the variables CNT and COUNt
- of the folder AL with the alarm ALM1 to 1

On **Application** we can test the value of the variables and display them as text or graphics simply by dragging the variable from the code in the panel **Watch** and **Oscilloscope**
See screenshot below



- Note how the local variables **a** and **b** cycle from -100 to 100 (x 10 for variable b), as do the corresponding variables count and counter1
- Alarm is an always true Boolean variable (TRUE shown in red[7]) and alarm1 is fixed at 1
- The variable count is displayed in the oscilloscope and we can see the 'saw-tooth'

Saving the project

We can save the project as counter.ppjs or a name of our choice, generate an .rsm file, password-protect it and export it as a library object (.pll)

DEVICE

The project can be opened with **Device:**
we open the tool of the same name, from which we can read or write one or more BIOS parameters, change them and/or download the default again.
It is possible to read the current value of the variables defined by the developer but not change them

Device does not require the user to know any programming language; the source code is not visible.
Device requires you to connect to the target device again (click on the respective icon). The central panel shows the protocol references, address, etc. The default name of the target device is FREESmart_1[8].
When the target device is connected, you can read the parameters or write the default or changed parameters

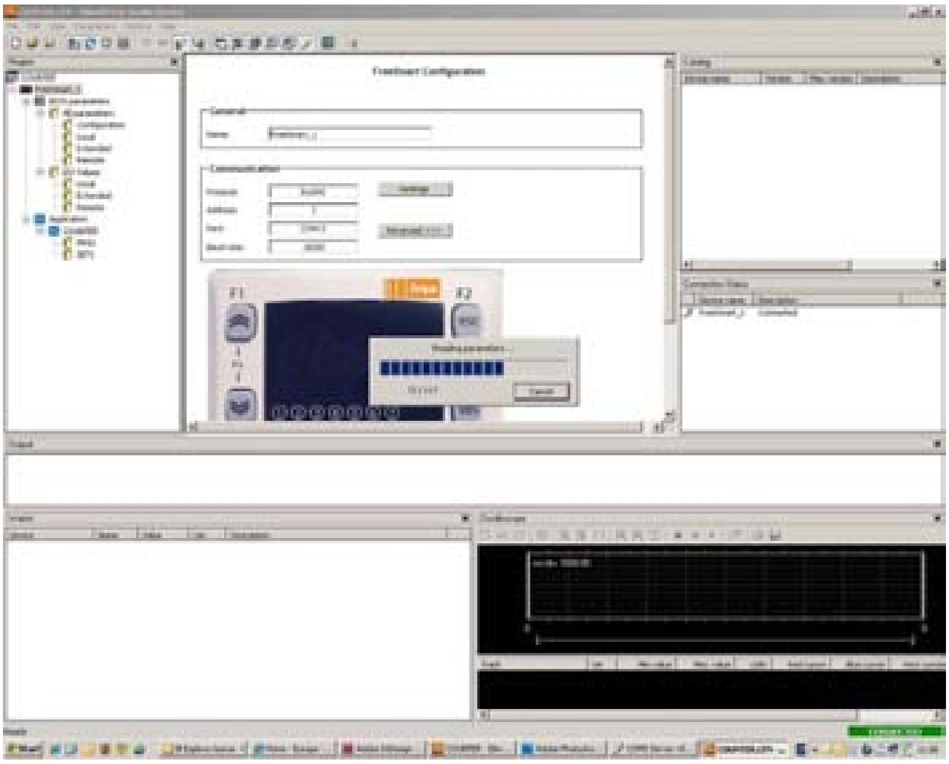
[6] **count is set as a fundamental display. If you do not set any variable, the message PLC appears. By pressing F5, FREE will appear and you can access the variables and parameters of BIOS**

[7] **if you change Alarm from TRUE to FALSE, re-compile and download the new application, the alarm LED will switch off and the variable will be indicated in green in the Watch panel**

[8] **by simply using the copy/paste function, you can create a twin project FREESmart_2, change the BIOS parameters and download them to the target device or a second target device**

Reading parameters

Suppose we cannot read all the parameters (Read all device parameters)



As we can see, there are configuration parameters which have default values other than those present on the target device (highlighted in blue)

Displaying variables

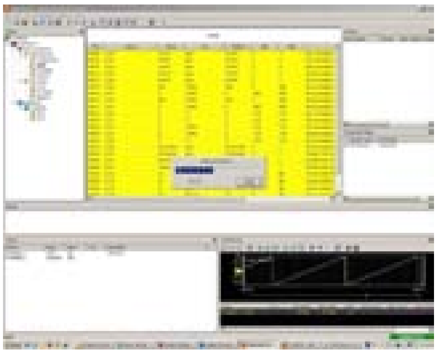
As with **Application**, with **Device** it is also possible to display the variables simply by dragging them to the Watch and Oscilloscope panels

Writing parameters

Suppose we want to download all the CL parameters (local BIOS) and no others.
Select all the parameters - drop-down menu **Parameters>Select all** (CTRL+A) or using the respective icon ▦. The parameters are all highlighted in yellow. Write drop-down menu **Parameters>Write selected** (CTRL+SHIFT+W) (or using the icon **W**)
All the CL parameters will be downloaded to the target device

CF Folder parameters CF31 and Ui26 with different default values from the application

Watch / Oscilloscope variables count and counter1



CL parameters (all) download in progress

We can save the project as Counter.CFN or a name of our choice

PROJECT 2 - COUNTER2 FUNCTION BLOCK

Starting from the COUNTER program described in the previous example, we create a function block and we modify the IEC application by adding code for the switching of a LED / relay

Saving the project

We save the COUNTER project by renaming it COUNTER2:
from the drop-down menu **File>Save project as** proposes the COUNTER work folder.
Go up one level, create a new COUNTER2 folder and save the project as COUNTER2.

Creating a function block

Create a new function block **COUNTERST** in **ST** language.
Copy/paste the COUNTER program into the Edit panel of the function block with several variants

COUNTERST program (function block)

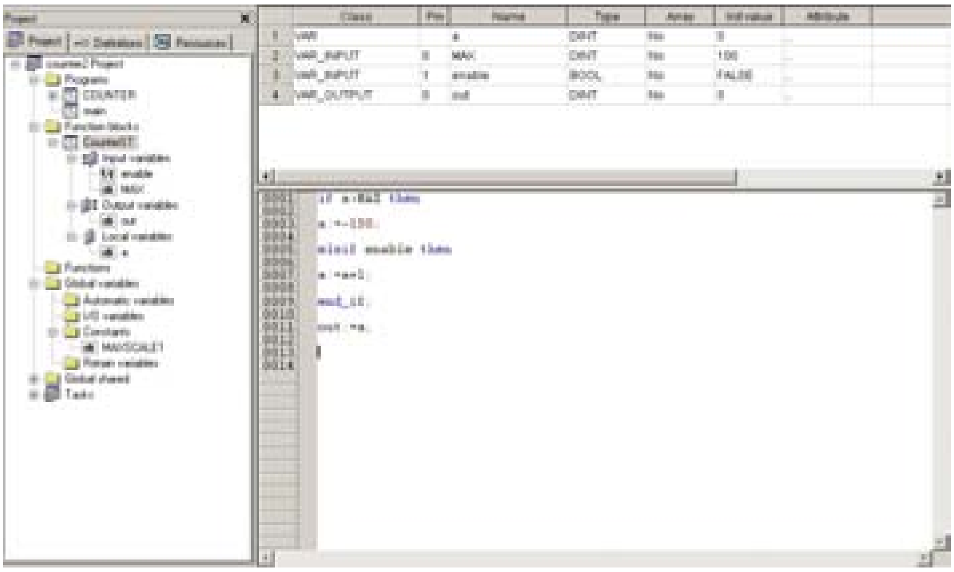
```
if a>MAX then

a:=-100;

elsif enable then

a:=+1;

end_if;

out:=a;
```

Local input and output variables

In section B[1] **variables** we create the following variables:

- **a** as local variable
- **MAX** and **enable** as input variables to the function block
- **out** as output variable from the function block

The function block will therefore have **2 input variables, one output variable and one local variable** which will save the status of the counter in each instance of the function block



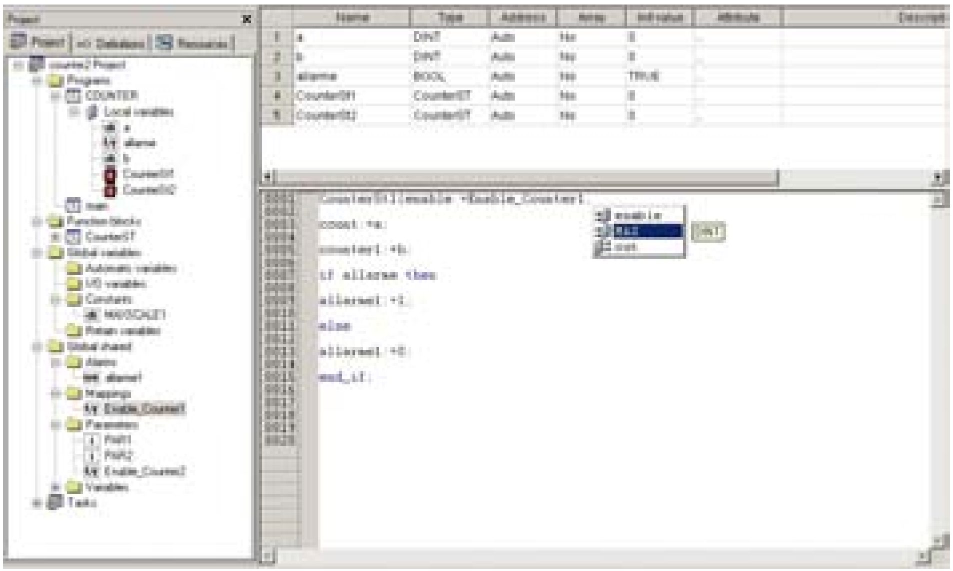
	Class	Pin	Name	Type	Array	Init value	Attribute
1	VAR		a	DINT	No	0	...
2	VAR_INPUT	0	MAX	DINT	No	100	...
3	VAR_INPUT	1	enable	BOOL	No	FALSE	...
4	VAR_OUTPUT	0	out	DINT	No	0	...

```
if a>MAX then

a:=-100;

elsif enable then

a:=a+1;

end_if;

out:=a;
```

[1] see Application Panel Definitions

We now write the COUNTER2 program in which the functional block will be called.
The function block will have 2 instances `CounterSt1, CounterSt2` defined as local variables of CounterSt type

As we write the code, note how **Application** guides the developer writing the code by proposing the variables to pass to the function block[2]



	Name	Type	Address	Array	Init value	Attribute	Description
1	a	DINT	Auto	No	0	...	
2	b	DINT	Auto	No	0	...	
3	allarme	BOOL	Auto	No	TRUE	...	
4	CounterSt1	CounterST	Auto	No	0	...	
5	CounterSt2	CounterST	Auto	No	0	...	

```
CounterSt1(enable :=Enable_Counter1

count :=a;

counter1 :=b;

if allarme then

allarme1 :=1;

else

allarme1 :=0;

end_if;
```

The Boolean variable in this case will serve to switch

- a suitably configured relay
- an LED[3] of the target on the basis of a defined condition

We now write the new code COUNTER2

COUNTER2 program

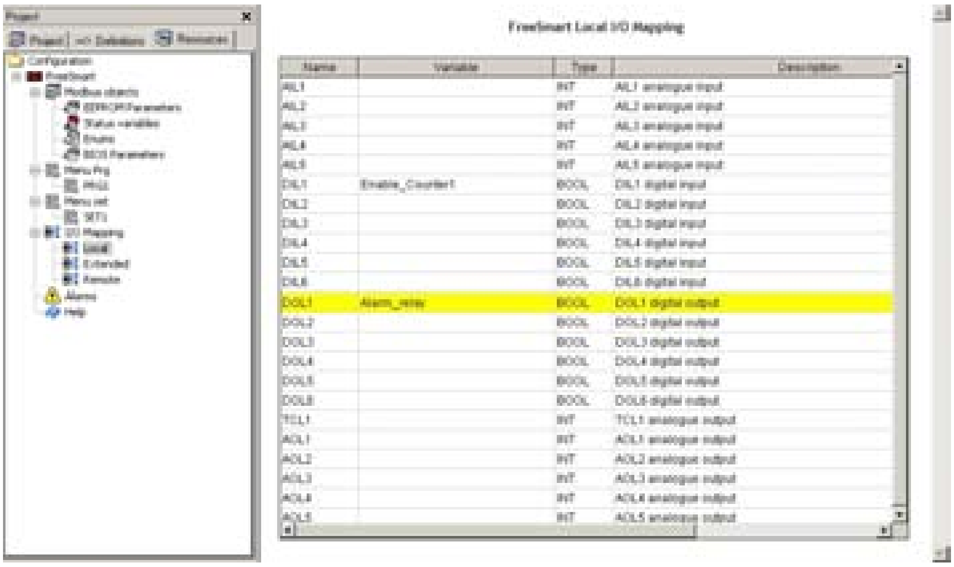
```
CounterSt1(enable:=not Enable_Counter1,MAX:=MAXSCALE1);
counter1:=CounterSt1.out;

CounterSt2(enable:=Enable_Counter2,MAX:=MAXSCALE1/2);
counter2:=CounterSt2.out;

alarm:=(counter1>counter2);

if alarm then

alarm1:=1;
Alarm_relay := TRUE;

else

alarm1:=0;
Alarm_relay := FALSE;

end_if;

if (counter1<=counter2) then
      sysLocalLeds[6]:=1;
else
      sysLocalLeds[6]:=0;
end_if;
```

[2] in the example 2 input variables (identified by an icon with the arrows on the left, and the output with the arrows on the right)

[3] we use the variable target `sysLocalLeds.` See relevant section

In the **Resources** tab, it is necessary to create and/or rename, from the previous project, the EEPROM parameters, status variables, etc.
It is also necessary, if desired, to add the resources to the **Prg menu** and **set menu**

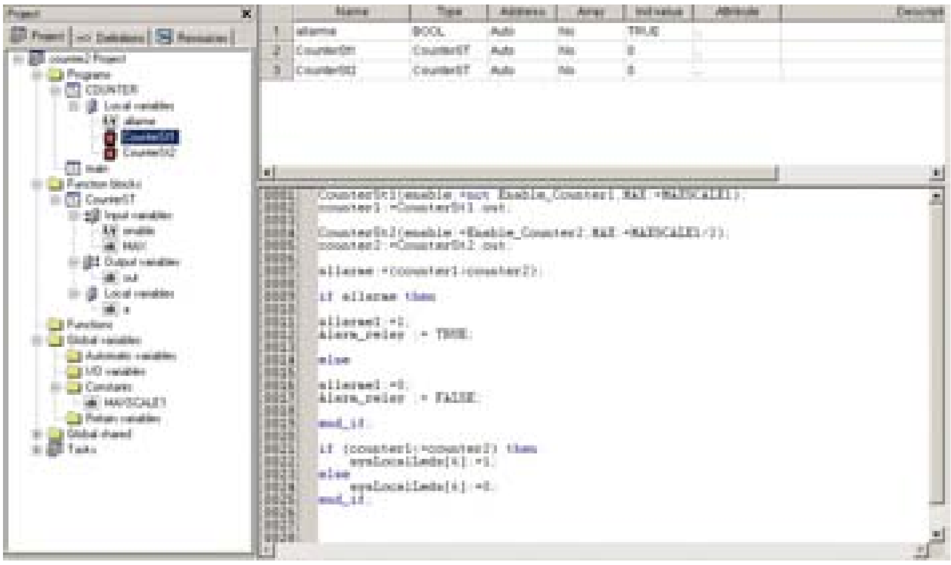
In the **I/O mapping** menu, lastly, it is possible to associate resources to the inputs/outputs
We associate a digital input `Enable_Counter1` and `Alarm_relay` to an output



FreeSmart Local I/O Mapping

Name	Variable	Type	Description
AIL1		INT	AIL1 analogue input
AIL2		INT	AIL2 analogue input
AIL3		INT	AIL3 analogue input
AIL4		INT	AIL4 analogue input
AIL5		INT	AIL5 analogue input
DIL1	Enable_Counter1	BOOL	DIL1 digital input
DIL2		BOOL	DIL2 digital input
DIL3		BOOL	DIL3 digital input
DIL4		BOOL	DIL4 digital input
DIL5		BOOL	DIL5 digital input
DIL6		BOOL	DIL6 digital input
DOL1	Alarm_relay	BOOL	DOL1 digital output
DOL2		BOOL	DOL2 digital output
DOL3		BOOL	DOL3 digital output
DOL4		BOOL	DOL4 digital output
DOL5		BOOL	DOL5 digital output
DOL6		BOOL	DOL6 digital output
TCL1		INT	TCL1 analogue output
AOL1		INT	AOL1 analogue output
AOL2		INT	AOL2 analogue output
AOL3		INT	AOL3 analogue output
AOL4		INT	AOL4 analogue output
AOL5		INT	AOL5 analogue output

The program is now completed



	Name	Type	Address	Array	Init value	Attribute	Description
1	alarme	BOOL	Auto	No	TRUE	...	
2	CounterSt1	CounterST	Auto	No	0	...	
3	CounterSt2	CounterST	Auto	No	0	...	

```
CounterSt1(enable :=not Enable_Counter1,MAX :=MAXSCALE1);
counter1 :=CounterSt1.out;

CounterSt2(enable :=Enable_Counter2,MAX :=MAXSCALE1/2);
counter2 :=CounterSt2.out;

allarme :=(counter1>counter2);

if allarme then

allarme1 :=1;
Alarm_relay  := TRUE;

else

allarme1 :=0;
Alarm_relay  := FALSE;

end_if;

if (counter1>=counter2) then
    sysLocalLeds[6] :=1;
else
    sysLocalLeds[6] :=0;
end_if;
```

Exporting to target device

From the drop-down menu **Developer>Export application to catalog** remember to export the project to catalogue so as to enable **Device** to re-use it for downloading to the **target**

Before exporting to catalogue, use the drop-down menu **Project>Options** to assign to the project a release (e.g. the same name **counter2**), a version (e.g. **1.0**) and compile the project again

This will enable **Device** to recognise the IEC application on the **target** when it is downloaded

Compile the project and download it to **target** by means of **Application** or **Device**

The counter will display a count from -100 to 100 in the same way as the IEC COUNTER application previously described
In this case, the Alarm LED will switch to the compressor LED[4] once the defined threshold has been reached

[4] SysLocalLeds[6], The alarm LED is SysLocalLeds[16]