

Baseline Software Platform

Easergy Builder

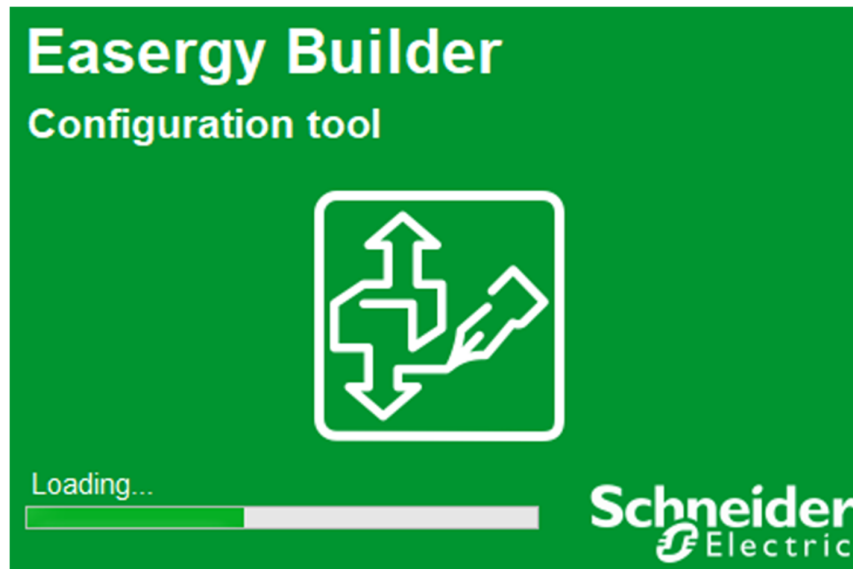
User Manual

This manual provides information for the use of the configuration tool included in the Schneider Electric Baseline Software Platform.

SE-S856-MSS

Publication Date (03/2023)

Read carefully the information contained in this manual before use this software tool for configuring the control system.



Change Control

Rev	Date	Description
1.0	09-08-2015	Initial revision
1.1	30-09-2015	Adapted to version 01.00.11 of Easergy Builder. Included Easergy T300 RTU (HU 250).
1.2	05-02-2016	Completed information about user rights.
1.3	05-07-2016	Updated with the revision 01.00.20 of Easergy Builder. Formula and Lioc devices have been included.
1.4	10-06-2017	Updated with the revision 1.1.6 of Easergy Builder. The two versions of SM_CPU866e are included: V0 and V1. Formula device is available for HU_A, HU_AF and SM_CPU866e (V0 and V1). The two versions of T300 are included: v1.0 and v1.2
1.5	01-03-2018	Update with the revision of 1.2.9 of Easergy Builder. GPS available for T300 IEC 61850 available for T300 Configuring Redundant new bus option (NO_ACCESS)
1.6	20-02-2019	Adaptation to Easergy Builder 1.4.5.
1.7	14-11-2019	Adaptation to Easergy Builder 1.6.0. Easergy T300 Devices have been removed and included in 'Easergy T300 Devices' user manual. Operating systems and minimum requirements for installation of Easergy Builder have been updated.
1.8	27-02-2020	Removed information about HU_A, HU_B, HU_BI and SM_CPU866. Information about sensitive data has been included. Paragraph 'Important information about Cybersecurity'. Included new supervision signal for T300 (RTU_AV_CPU_USAGE).
1.9	05-04-2021	Adaptation to Easergy Builder 1.7.5.0 (T300) and 1.6.7.1 (Saitel). Power supply warning levels for signals WARN_PSx updated. HST plugin has been included. String table included in coreDb
2.0	06-07-2021	Adaptation to Easergy Builder 1.7.9 (T300 and Saitel). Union of Saitel and T300 plugins, with only one for both platforms. SGCC profile has a new name: DLT634.5104. SpaBus master protocol has been included for Saitel.
3.0	12-08-2021	SNMP Manager section added.
3.1	18-10-2021	Revision for Easergy Builder 1.7.9 (T300 and Saitel).
4.0	16-03-2023	PowerLogic™ T300 Rebranding and new firmware release 2.8.2.

General Information

The Baseline Software Platform and all its components have been developed in accordance to the requirements for a quality management system, complying with the ISO 9001:2015 Norm.

Document code:	SE-S856-MSS
Revision/Date:	4.0 / 16-03-2023
File:	Easergy Builder – User Manual_EN_4.0.pdf
Retention period:	Permanent throughout its validation period + 3 years after its cancellation.

Reference Documents

User Manual	Document Code
Saitel DR Platform User Manual Platform User Manual	SE-F800-USRUSR
Configuration & Startup of Saitel DP	FTE-F700-CYP
Saitel DP Modules	FTE-F700-MOD
webApp User Manual	FTE-S856-WPP
webUI User Manual	SE-WUI-S854
IEC101 User Manual	SE-S854-I1D
IEC104 User Manual	SE-S854-I4D
IEC103 Master User Manual	FTE -S854-I3D
Modbus User Manual	SE-S854-MBD
ISaGRAF® User Manual	FTE-S854-ISD
DNP User Manual	FTE-S854-DNP
SOE User Manual	FTE-S854-SOE
TSV User Manual	FTE-TSV-S854
IEC61850 User Manual - Ed1	FTE-S854-IEC61-1
IEC61850 User Manual - Ed2	FTE-S854-IEC61-2
T300 Devices User Manual	FTE-S857-EBD
T300 User Manual	NT00378
T300 Quick Start Guide	NT00383
SM_CPU866e – User Manual	SE-M578-USR
HUe - User Manual	SE-M588-USR

Software in Easergy Builder

Easergy Builder installer includes the following software:

Table 1 – Software in Easergy Builder

Tool / Plugin	Version	Installation file
Easergy Builder	1.7.9.4	setup.exe
ATS	01.00.16	ATS_SPlugin.msi
Cilo	01.01.01	Cilo_SPlugin.msi
CLAQ	01.01.13	CLAQ_SPlugin.msi
CMON	01.00.18	CMON_SPlugin.msi
DNP Master	01.03.04	DNPM_SPlugin.msi
DNP Slave	01.03.04	DNPS_SPlugin.msi
HST	01.00.09	HST_SPlugin.msi
IEC101 Master	01.02.06	IEC101M_SPlugin.msi
IEC101 Slave	01.05.08	IEC101S_SPlugin.msi
IEC103 Master	01.00.12	IEC103M_SPlugin.msi
IEC104 Master	01.02.07	IEC104M_SPlugin.msi
IEC104 Slave	01.05.11	IEC104S_SPlugin.msi
IEC61850	01.02.19	IEC61850_SPlugin.msi
ISaGRAF 3	01.00.03	ISAGRAF_SPlugin.msi
ISaGRAF 5	01.01.03	ISAGRAF5_SPlugin.msi
LAQ	01.02.07	LAQ_SPlugin.msi
Lioc	01.01.11	Lioc_SPlugin.msi
Modbus Master	01.02.01	MDBM_SPlugin.msi
Modbus Slave	01.01.02	MDBS_SPlugin.msi
MICOM	01.00.07	MICOM_SPlugin.msi
SEPAM	01.01.04	SEPAM_SPlugin.msi
SOE	01.01.06	SOE_SPlugin.msi
SNMP Manager	01.00.08	SNMPM_SPlugin.msi
Terminal Server (TSV)	01.00.06	TSV_SPlugin.msi
webUI Editor	01.02.08	webUI Configuration
ZIGBEE	01.00.07	ZIGBEE_SPlugin.msi

Important information about Safety


Read these instructions carefully and look at the equipment to become familiar with it before trying to install, operate, service or maintain it. In this manual shows different types of messages associated with situations that have different level of risk for people and / or for equipment.




This symbol indicates '**DANGER**' or '**WARNING**'. This symbol informs of an electrical risk that will cause personal injuries if the instructions are not followed.



This symbol is associated to a safety alert. It is used to warn of possible personal injury hazards. The user must follow all instructions or messages associated to this symbol to avoid possible injuries.


 DANGER
DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.

 WARNING
WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.

NOTICE
NOTICE is used to address practices not related to physical injury. The safety alert symbol shall not be used with this signal word.

Important information about Cybersecurity

Easergy Builder allows to configure several unsecure protocols to communicate with field devices, such include IEC101, IEC104, IEC103, DNP, SpaBus, Modbus and IEC61850. Easergy Builder also allows to configure unsecure SNMP v1 and v2c protocols for management and monitoring of network-connected devices. The RTU does not have the capability to transmit data encrypted using these protocols.

 WARNING
If a malicious user gained access to the network, transmitted information could be disclosed or subject to tampering.

To avoid malicious use of the sensitive information that will be transmitted through an internal network, follow the instructions below:

- Network must be physically or logically segmented.
- The access to the network must be restricted using standard controls such as firewalls.
- For transmitting data over an external network, encrypt protocol transmissions over all external connections using an encrypted tunnel, TLS wrapper or a similar solution.

To Keep in Mind

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

Qualified personnel are individuals who:

- Have read and understood the information on the device and its user manual.
- Are familiar with the installation, commissioning, and operation of the equipment and of the system to which it is being connected.
- Have knowledge to perform switching operations in accordance with accepted safety engineering practices and are authorized to energize and de-energize equipment and to isolate, ground, and label it.
- Are trained in the care and use of safety apparatus in accordance with safety engineering practices.
- Are trained in emergency procedures (first aids).







It is necessary to consider that the documentation of the equipment collects the instructions for its installation, set up and operation. However, the manuals could not cover all the possible circumstances neither include specific information on all the details.





In case of questions or specific problems about a Schneider Electric equipment, contact with his sales office of Schneider Electric or with the customer care center to request the necessary information.

Symbols and Labels

Before the equipment is installed or commissioned, the user must understand the following symbols, which may be used on the equipment or referred to in the user documentation:

Table 2 – Symbols

Symbol	Associated Text	Description
	Possibility of electric shock	International Electrotechnical Commission (IEC) symbol associated to a DANGER or WARNING message indicating that there is an electrical risk. Failure to follow these instructions could cause damage to people or death.
	Caution, read the manual.	Symbol associated with a risk alert. The user must read the manual before handling the equipment.
	Possibility of electric shock	American National Standards Institute (ANSI) symbol associated to a DANGER or WARNING message indicating that there is an electrical risk. Failure to follow these instructions could cause damage to people or death.
	Protective earth connection	Associated symbol to the protective ground connection.
	Functional earth connection	Associated symbol to the functional ground connection.
	CE Mark	This symbol indicates that the equipment has been developed in compliance with all applicable European Directives.

Symbol	Associated Text	Description
	Electronic equipment. Special instructions must be followed for disposed.	This symbol indicates that, at the end of its life, this module must be disposed according to the WEEE Directive (Waste Electrical and Electronic Equipment).
	Compliant with RoHS.	The equipment has been designed and manufactured according to RoHS Directive (Restriction of Hazardous Substances).
	Direct Voltage	Symbol of direct voltage (V_{DC}).
	Alternate Voltage	Symbol of alternate voltage (V_{AC}).

Installation, Setup and Operation

Some devices use dangerous voltages ($> 50 V$), either the Schneider Electric device itself or some device connected to it. The user is responsible to check that the characteristics of each equipment are adapted and convenient for his installation. The user should read the instructions of installation before proceeding to the use or maintenance of the equipment.

 DANGER
Not following these instructions can be dangerous for the people and the equipment.

Devices that handle dangerous tensions are marked with a sticker on the front label (size: 12,5 mm). This label must be visible all the time while the module is installed on the DIN rail.

Because of the variety of uses of the product, the managers of the application and use of this controller device will have to take the measures to the fulfillment of all the safety requirements and provision of each application. These requirements are according to the applicable laws, regulations, codes and standard.

Content

<u>1</u>	<u>FIRST CONTACT WITH EASERGY BUILDER</u>	<u>9</u>
<u>2</u>	<u>WORKING WITH RTUS</u>	<u>31</u>
<u>3</u>	<u>COREDB – REAL-TIME DATABASE</u>	<u>77</u>
<u>4</u>	<u>SUPERVISION DEVICE</u>	<u>93</u>
<u>5</u>	<u>SNMP MANAGER</u>	<u>101</u>
<u>6</u>	<u>FORMULA DEVICE.....</u>	<u>107</u>

1 First Contact with Easergy Builder

Content

1	FIRST CONTACT WITH EASERGY BUILDER	9
1.1	BASELINE SOFTWARE PLATFORM	11
1.1.1	INTRODUCTION	11
1.1.2	COREDB – REAL-TIME DATABASE.....	13
1.1.3	DEVICES	13
1.1.4	PLUGINS – CONFIGURATION INTERFACES.....	14
1.1.5	SOFTWARE TOOLS	15
1.2	EASERGY BUILDER INSTALLING	16
1.2.1	INTRODUCTION	16
1.2.2	EASERGY BUILDER REQUIREMENTS.....	16
1.2.3	INSTALLING EASERGY BUILDER.....	18
1.2.4	UNINSTALLING EASERGY BUILDER	19
1.3	ENVIRONMENT DESCRIPTION.....	20
1.3.1	WORKSPACE MODE	22
1.3.2	CONFIGURATION MODE	26
1.3.3	WEBUI	28

1.1 Baseline Software Platform

1.1.1 Introduction

The Baseline Software Platform of Schneider Electric consists of:

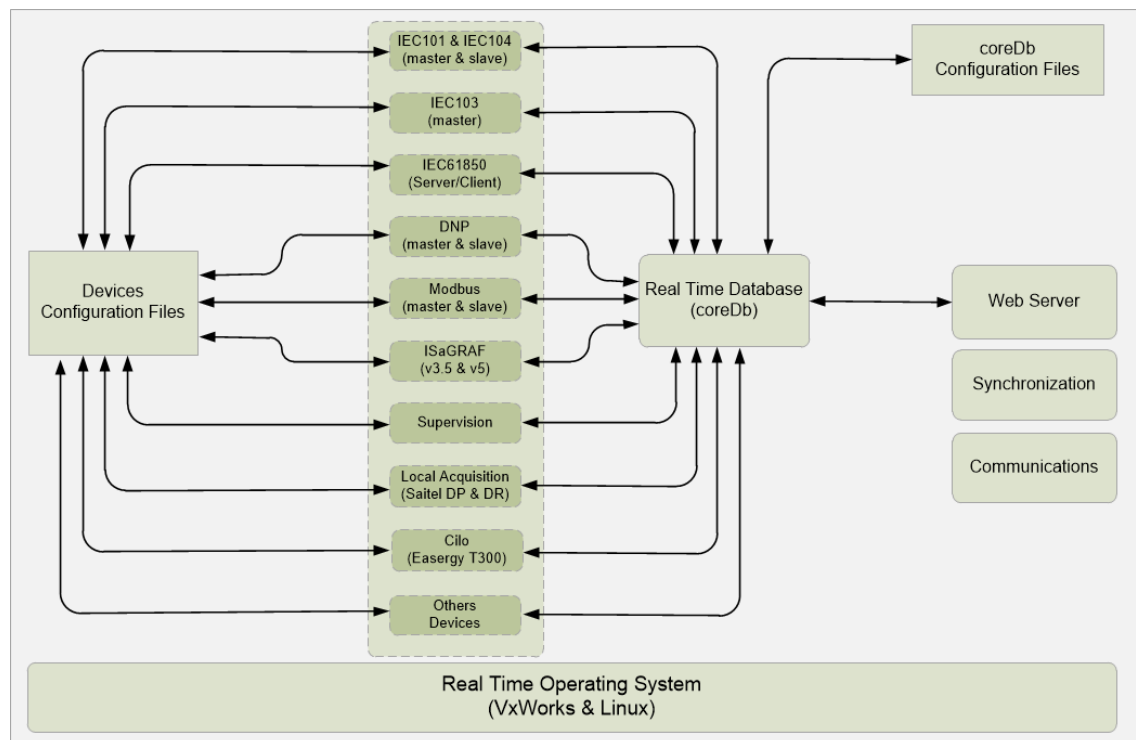
- Real-time operating system (RTOS): VxWorks, Linux,...
- Real-time applications and configuration files.
- Configuration, management, supervision and monitorization tools.

NOTICE

Saitel DR basic head units HU_B and HU_BI modules do not run VxWorks nor Linux. They operate with a tailored-made OS which includes the database and applications.

The following figure shows the different applications included in the software platform, as well as additional applications that implement new devices or protocols to upgrade Easergy Builder.

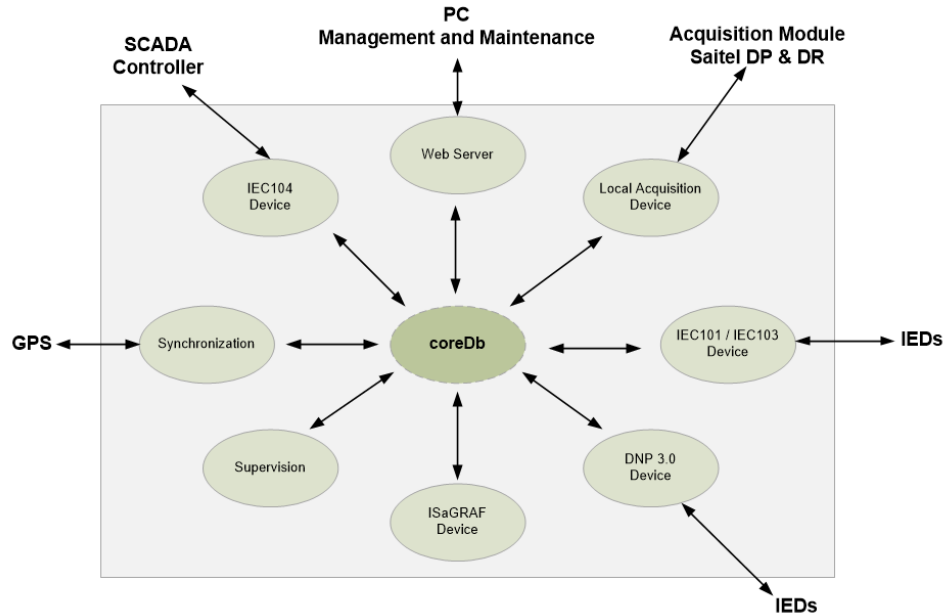
Figure 1 – Baseline Software Platform



The operating system abstracts the hardware from the software applications and manages the applications in real time. It integrates the basic protocols to access the remote unit (SFTP, SSH, etc.) and manage multiple users.

The real-time database, named coreDb, is probably the most important element. All the other elements are developed around coreDb:

Figure 2 – Relation between coreDb and other applications.



The following concepts are related to coreDb:

- **Device Controller** (also referred to as **Controller**): Real-time application that accesses coreDb. Each Controller acts as a producer and/or consumer of information managed by coreDb.
- **Point**: Each record of coreDb is a point. A point can be included in the table Status, Analog, Command, Setpoint or String.
- **Device**: A set of I/O points that share a common source/destination. A typical example of a Device is an IED that communicates with the RTU, or the representation of a SCADA exchanging information acquired or generated by the RTU. A Device is always associated to a type of Controller.
- **Source**: Origin of the value of a coreDb data point. Any coreDb data point can have several different sources (in one or several Devices). This means that a value of a database point can be configured to be updated by several different entities.

⚠ WARNING

It should be noted that any coreDb signal can be associated to more than one source; this is only applicable to Command and Setpoint tables. Allocating more than source to one point is **not recommended** in Status and Analog tables.

- **Destination**: Target of the value of a coreDb data point. coreDb data points can be configured to have several different destinations (in one or several Devices).
- **Coordinate**: Point identification within a Device. It is unique for each point and has a different structure for each Controller. It is described in detail in the appropriate manual of each Controller.
- **Configuration Plugin**: Specific Configuration plugins extend the Easergy Builder application to configure Device Controllers. Additional details about these plugins are provided in their manuals.

The user can modify the configuration of each Controller and Device using the appropriate Plugin. Once the database is completely configured, the files with the new information can be generated and transferred to the RTU, where they will be processed by the software on startup.

NOTICE

The configuration information exchanged between the RTU and Easergy Builder is not continuous, but performed through XML files under user request. **When the configuration is modified in Easergy Builder and the XML files are sent to the RTU, it is necessary to reboot the RTU.**

1.1.2 coreDb – Real-Time DataBase

coreDb is the real-time database backend on which BaseLine Software Platform is built. All the information controlled and managed by the system is stored in this database.

Thanks to this architecture, the system functionalities can be easily expanded to manage new protocols, customized controllers, etc. To accomplish this, trained developers only need to implement the required Device Controller and the associated Configuration plugin for Easergy Builder, allowing end users to configure the extended functionality.

coreDb records are also called data points or, simply points (this term will be used onwards to avoid confusing these with Device points).

coreDb points are organized in five tables: Status, Analog, Setpoint, String and Command to group the different types of points. These internal tables present the following differences.

- Depending on the **point type**: status, and command tables support integer values, whereas setpoint and analog tables manage floating values. String table has been designed for string values.
- Depending on the **treatment of the point**: status and analog points can be locked or reset to initial values, whereas the other two signal types cannot. All types can retain the value in a non-volatile memory.

1.1.3 Devices

Each type of device keeps a list of its associated points, identified by unique labels. These labels allow the identification of each device point unequivocally as source or destination of a coreDb data point.

Each point is a piece of information produced (or consumed) by a Device. Within a single Device, point identifiers (coordinates) are unique and cannot be used by two different points.

The following table shows the device configuration plugins included in Easergy Builder. Each plugin will be available or not depending on the hardware platforms:

Table 3 – Device compatibility for each platform

Device	SM_CPU866e	HUe	HU250
ATS	×	×	✓
Command Management (Cilo)	×	×	✓
Condition Monitoring (CMON)	×	×	✓
DNP Master & Slave protocols	✓	✓	✓
Formula	✓	✓	✓
Historical Sequence of Events (HST)	×	×	✓
IEC101 Master & Slave protocols	✓	✓	✓
IEC103 Master Protocol	✓	✓	×
IEC104 Master & Slave protocols	✓	✓	✓
IEC61850 Plugin	✓	✓	✓

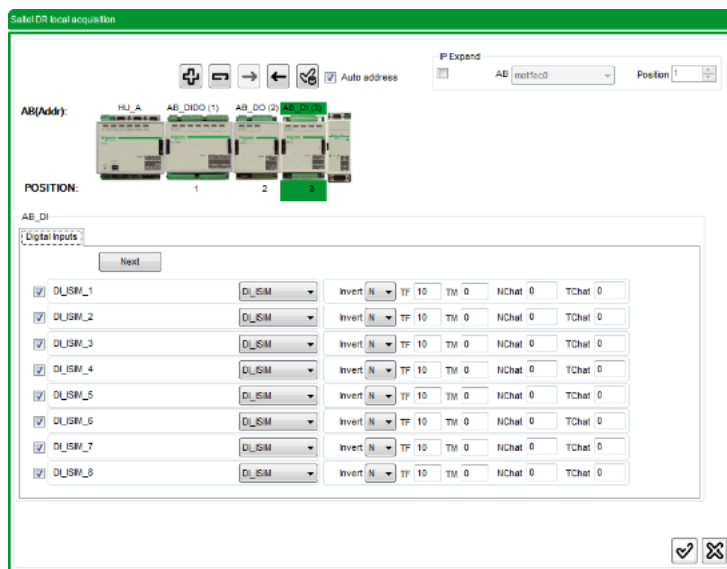
Device	SM_CPU866e	HUe	HU250
ISaGRAF 3	✓	✓	✗
ISaGRAF 5	✓	✓	✓
Local Input Output (Lioc)	✗	✗	✓
MICOM Master Protocol	✓	✓	✗
Modbus Master & Slave protocols	✓	✓	✓
Saitel DP local acquisition (LAQ)	✓	✗	✗
Saitel DR local acquisition (CLAQ)	✗	✓	✗
SEPAM Protocol	✓	✓	✓
Sequence of Events (SOE)	✓	✓	✓
SNMP Manager	✓	✗	✓
Supervision	✓	✓	✓
ZigBee	✗	✗	✓
Terminal Server (TSV)	✓	✗	✗
webUI	✓	✗	✗

1.1.4 Plugins – Configuration Interfaces

Easergy Builder has two types of interfaces to consult and modify the information stored in coreDb:

- **Graphical Interface (Plugin):** The picture below shows the interface to configure Saitel DR acquisition hardware. Using this graphical interface, the user can configure each local acquisition points with its corresponding point in coreDb. There is a graphical interface for each Device Controller.

Figure 3 – Saitel DR local acquisition plugin interface.



- **coreDb tables:** This interface enables access directly each point stored in coreDb. There is a tab available for each table: Status, Analog, Setpoint, String and Command.

Figure 4 – coreDb interface.

The screenshot shows the Easergy Builder software interface with the 'coreDb' tab selected. The interface includes a menu bar (File, View, Help, Add-Ons) and a toolbar with icons for search, refresh, and error handling. Below the toolbar is a search filter area with 'Name' and 'Source' dropdowns, an 'AND' operator, and a 'Destination' dropdown. The main area displays a table with the following columns: Name, Description, Source1 Device, Source1 Coordinates, Source1 Vmask, Destination1 Device, Destination1 Coordinates, Destination2 Device, Destination2 Coordinates, Init value, Blocked, Non volatile, Shared Publish, and Shared Subscribe. The table contains 30 rows of configuration points, including digital status points (M_0000 to M_0003), digital setpoints (D001_00000 to D001_00015), digital commands (D001_COMM...), digital diagnostics (D001_HW_DI...), and supervision points (WARN_BAT, FAIL_SYNC1, FAIL_SYNC2, FAIL_CONF, FAIL_RTU, DOING_WELL).

Name	Description	Source1 Device	Source1 Coordinates	Source1 Vmask	Destination1 Device	Destination1 Coordinates	Destination2 Device	Destination2 Coordinates	Init value	Blocked	Non volatile	Shared Publish	Shared Subscribe
M_0000	DI_SSM 1	claq	100002000							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
M_0001	DI_SSM 2	claq	100002001							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
M_0002	DI_SSM 3	claq	100002002							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
M_0003	DI_SSM 4	claq	100002003							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00000	DI_SSM_1	claq	100102000							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00001	DI_SSM_2	claq	100102001							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00002	DI_SSM_3	claq	100102002							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00003	DI_SSM_4	claq	100102003							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00004	DI_SSM_5	claq	100102004							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00005	DI_SSM_6	claq	100102005							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00006	DI_SSM_7	claq	100102006							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00007	DI_SSM_8	claq	100102007							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00008	DI_SSM_9	claq	100102008							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00009	DI_SSM_10	claq	100102009							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00010	DI_SSM_11	claq	100102010							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00011	DI_SSM_12	claq	100102011							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00012	DI_SSM_13	claq	100102012							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00013	DI_SSM_14	claq	100102013							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00014	DI_SSM_15	claq	100102014							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_00015	DI_SSM_16	claq	100102015							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_COMM...	COMM_DIAG	claq	100100000							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D001_HW_DI...	HW_DIAG	claq	100100001							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D002_COMM...	COMM_DIAG	claq	100200000							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D002_HW_DI...	HW_DIAG	claq	100200001							<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WARN_BAT	Low battery war...	supervision			WARN_BAT					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FAIL_SYNC1	Fail in primary sht...	supervision			FAIL_SYNC1					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FAIL_SYNC2	Fail in secondary...	supervision			FAIL_SYNC2					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FAIL_CONF	Fail in the config...	supervision			FAIL_CONF					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FAIL_RTU	FAIL_CONF is 1 ...	supervision								<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DOING_WELL	Signal for indicat...					supervision	DOING_WELL			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Once the configuration process is completed, Easergy Builder generates the necessary XML configuration files. These files must be transferred to the RTU to apply the new configuration. When the file transfer operation is completed without errors, the configuration changes can be activated in the RTU by rebooting it.

1.1.5 Software Tools

A basic configuration is included with HUE, which should be adapted to the requirement of the system.

- **Easergy Builder:** Engineering tool for the RTU OFFLINE configuration. It allows to include and adapt the different functions of the RTU to the system where it is being integrated. It is a software tool that needs to be installed on a PC.
- **CAE:** Engineering tool for defining the security policy and assigning roles to users. It allows defining a series of rights and responsibilities in the system for authorized users. It defines WHO, WHAT, WHEN and HOW can the user do it, according to the RBAC model. It is a software tool that needs to be installed on a PC.
- **webApp:** Web tool for online maintenance and monitoring of the RTU. Using the configuration defined in Easergy Builder and loaded in the CPU, the user can consult and/or change some parameters through the WEB server. Unlike Easergy Builder, webApp does NOT allow adding new features. Only the parameters included in the configuration can be changed.
- **Console:** This tool should only be used by advanced users with a wide knowledge of the system. Depending on the CPU type, connection can be made through a serial channel (PC COMx port) or using SSH through an Ethernet port. The console is a command-based tool. Each command will be available or not depending on the level of privileges assigned to the user.
- **webUI:** Only available for SM_CPU866e. Advanced tool for the design and operation of a local web-based substation user interface. It has two different environments:
 - **webUI (Real Time):** It allows to monitor graphically the information included in the different screens designed by the user through the editor included in Easergy Builder (webUI Add-Ons).

- **webUI Editor:** (webUI Add-): Editor included in Easergy Builder for creating and editing webUI screens. It allows to design new screens for the configuration loaded into the RTU or also modify the information displayed in those already available.

1.2 Easergy Builder Installing

1.2.1 Introduction

Easergy Builder is the configuration tool for Schneider Electric RTUs that use the BaseLine Software Platform. Among other features, it can be used to perform:

- Configuration of the general settings of an RTU: IP address, communication channels, etc.
- Design and maintenance of coreDb.
- Administration of the synchronization mechanisms.
- Configuration of the supervision and monitoring features.

Easergy Builder allows managing several RTUs from a single interface. For each type of RTU, several profiles or configurations can be defined. The user can choose the profile to be sent to the RTU.

Easergy Builder has two different working modes:

- **'WorkSpace'** mode: Used to administrate RTUs and their associated profiles and configurations. This mode is presented to the user on startup. In this work mode, the RTU administration tree is shown at the left in the tool.
- **'Configuration'** mode: Engineering interface for a specific configuration of an RTU. While the tool is in the WorkSpace mode, the Configuration mode is activated by double click on a Configuration in the RTU tree.

1.2.2 Easergy Builder Requirements

Supported Operating System

Easergy Builder can be used with the following operating systems:

- Microsoft Windows® 7 SP1 (x86 and x64).
- Microsoft Windows® 8 (x86 and x64).
- Microsoft Windows® 8.1 (x86 and x64).
- Microsoft Windows® 10.
- Microsoft Windows® Server 2008 R2 SP1 (x64).
- Microsoft Windows® Server 2012 (x64).
- Microsoft Windows® Server 2012 R2 (x64).

Minimum Hardware

The following hardware (minimum) is required to use Easergy Builder:

- 1 GHz or faster processor.
- 512 MB of RAM.
- 2.5 GB of available hard disk space (x86 and x64).

Setup File

It is recommended to use Easergy Builder on Microsoft Windows® 7 or upper. The file 'setup.exe' must be execute as administrator. To install this tool on Microsoft Windows® XP (not recommended) is necessary the file 'Easergy Builder.msi' file.

.NET Framework

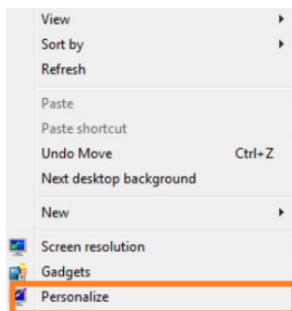
Microsoft .NET Framework 4.6.1 is required. This software can be downloaded from the Microsoft web site (<https://www.microsoft.com/en-US/download/details.aspx?id=49982>).

DPI on Windows 7

When Easergy Builder starts, the DPI parameter on Microsoft Windows® 7 must be 96 (100% of the screen resolution). If not, please follow these instructions:

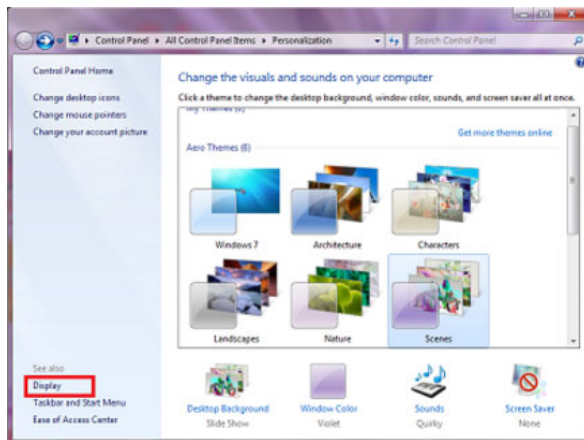
- Right click on an empty area of the desktop and choose 'Personalize':

Figure 5 – Display properties selection.



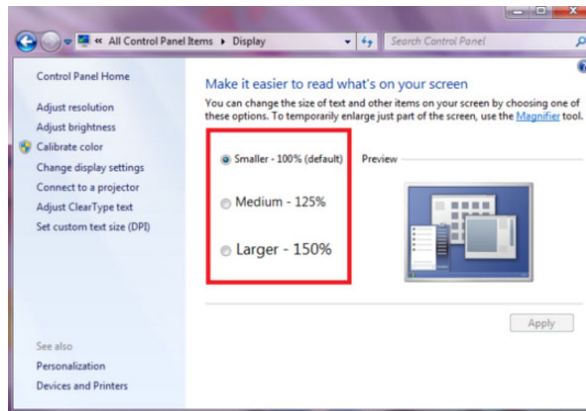
- Click on the Display link at the bottom left corner:

Figure 6 – Display configuration window.



- In the follow screen, please select 'Smaller – 100%'. This configuration is according to 96 DPI necessary for Easergy Builder execution.

Figure 7 – Display scale selection window.

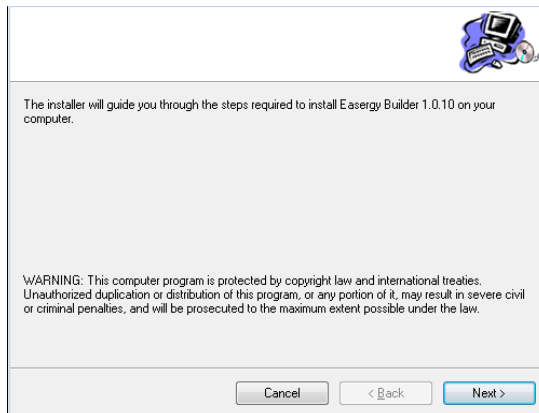


- Click on the Apply button and select 'Log off now'. Please, be sure to save your work before clicking on this.

1.2.3 Installing Easergy Builder

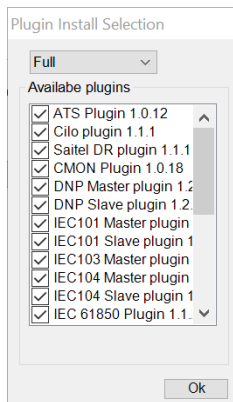
The next step is executing Easergy Builder installation program, 'Easergy Builder.msi' or 'setup.exe' depending on the OS.

Figure 8 – Startup of the Easergy Builder installation.



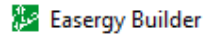
Follow the steps appearing on the screen to complete the software installation; during the process the user will be prompted to select the configuration plugins to install:

Figure 9 – Selecting plugins to be installed.



This screen will show the plugin installation files located in the same directory as the Easergy Builder installer. Each msi file contains a configuration plugin. Each plugin should be installed or not depending on the type of CPU. More information about available plugins for each CPU in Table 2 of this manual.

Select the plugins to be installed. Press 'OK' button and the installation process will complete. Once finished, the Microsoft Windows® startup menu will show the Easergy Builder icon under the 'Schneider Electric' folder.



⚠ WARNING

The installation of the Plugins is executed in background:

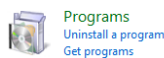
Please, be sure the following window is shown before running the application.

If Easergy Builder is installed in the PC, new plugins can be added executing its msi file. If the installation process is successfully, the new plugin will be available in Easergy Builder from now on.

1.2.4 Uninstalling Easergy Builder

The uninstall process of the tool depends on the host operating system in which Easergy Builder is running. This manual assumes that the user is working with Microsoft Windows® 7 Professional, so the process might differ considerably if the user is using a different operating system.

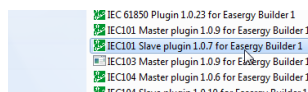
To update or uninstall Easergy Builder or a Configuration plugin on Microsoft Windows® 7 Professional, the user must access the control panel: **Start → Control Panel** and select 'Uninstall a program' in section 'Programs'.



This utility allows updating or uninstalling one or several Plugins and it allows updating or uninstalling Easergy Builder completely from the host PC.

Uninstalling a Plugin

To remove a Plugins from the PC, on the control panel, access to the software installed and select the plugin to be removed. Double-click on it and the uninstall process start.



If the user selects 'No' when prompted to confirm the operation, the uninstall process is cancelled. If confirmed, the process will complete with the elimination of the selected plugin.

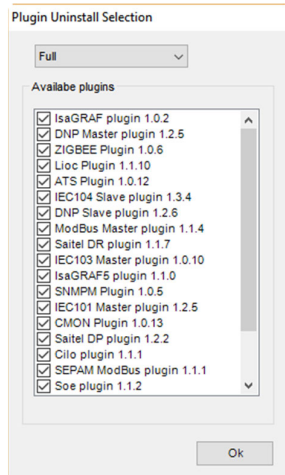
Uninstalling Easergy Builder

To remove Easergy Builder completely from the PC, it must be selected in the control panel program list and double-click on it.



The uninstall wizard is executed. Depending on Easergy Builder Version, the user could be prompted to select the plugins that must be removed from the PC. To remove Easergy Builder completely, select 'Full' in the plugin selection window. All plugins will be selected to uninstall them:

Figure 10 – Selecting plugins to be uninstalled.



NOTICE

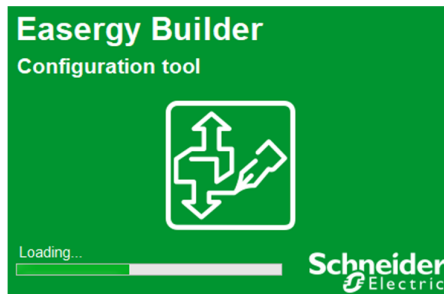
Note that the default uninstall process is set to 'Minimal'. This option uninstalls 'Easergy Builder' but all plugins that have not been selected will remain installed.

Press 'Ok' and the uninstall process will complete.

1.3 Environment Description

Easergy Builder can be launched from the Microsoft Windows® Start menu, or through the desktop shortcut created during the installation.

Figure 11 – Easergy Builder startup window.



There are some concepts that users must know before use Easergy Builder:

Operation Modes

In Easergy Builder there are two working modes:

- WorkSpace mode (initial mode when Easergy Builder starts).
- Configuration mode.

More information about these operation modes in below in this manual.

Devices

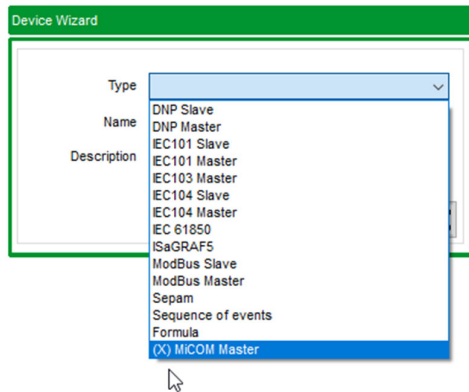
The data exchange between coreDb and external devices is managed by Device Controllers (software running in the CPU). Each type of Controller defines the rules that must be followed in this data exchange.

Each Device Controller is associated with an Easergy Builder 'Configuration Plugin'. Each plugin implements the rules to follow and offer user-friendly graphical interfaces to set the data exchange

with a Device (through its Device Controller). Much of the available plugins can be installed and uninstalled independently, but other ones are included in the core of the tool.

If a Plugin is not available, when a new Device is being created, this Plugin will be shown with (X).

Figure 12 – Plugin not available.



If this Device must be used in the RTU, the appropriated Plugin must be installed using its '.msi' file.

Communication Channels

Each port used to communicate with field devices must be configured as a communication channel. The number and type of these channels depends on the type of RTU and the communication modules installed (AB_SER, SM_SER or K7 modules). A general overview about communication channels is shown in this manual.

Add-Ons

An Add-on is an external application, implemented as a dynamic library (DLL). Add-ons can be integrated into Easergy Builder simply by placing the DLL file in a specific directory – **C:\Program Files\Schneider Electric\Easergy Builder\PlugsAddOn** - and restarting the application.

NOTICE

If the user modified the default directory during the installation of the tool, the path for Add-Ons should also change.

As an example of the use of this feature, suppose that an Add-on providing a calculator has been implemented and there is a DLL file containing the Add-on.

The user needs to create a new directory named, for example, 'Calculator' within the '**PlugsAddon**' folder under the main path where Easergy Builder software is installed and copy the DLL file to this folder.

NOTICE

Do not copy this DLL file in the 'PlugsAddon' directory, but in a folder under this directory, labeled as the Add-on name to be installed.

To activate the Add-on, restart Easergy Builder and the calculator will be available from the main menu '**Add-Ons → Calculator**'.

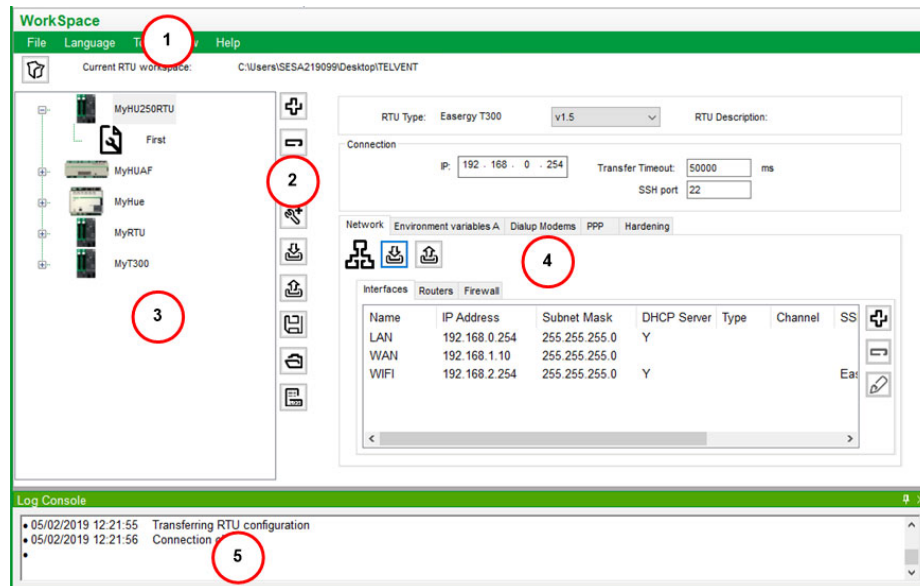
NOTICE

webUI editor has been implemented as an Add-on and it is available only for SM_CPU866e in version 1.6.7 and later of Easergy Builder.

For more information about webUI, please consult the webUI user manual.

1.3.1 WorkSpace Mode

Figure 13 – Easergy Builder in WorkSpace mode.



In the Easergy Builder WorkSpace mode there are four different areas:

- 1: Main menu
- 2: Administration toolbar
- 3: RTU tree
- 4: Edition area
- 5: Consoles information

In this mode the user can:

- Create new WorkSpaces or load other WorkSpaces into the PC storage device.
- Manage several RTU at the same time, with several configurations associated to each one.
- Select the preferred language (Deutsch, English and Spanish are available by default).
- Consult information about Easergy Builder (Help).

Zone 1. Main Menu

The main menu has the following options:


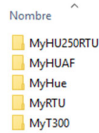
- **File:** This option allows creating and loading workspaces. A workspace is a set of RTUs stored in the same computer folder. All the RTUs defined in a workspace are shown in the RTU tree. The workspace path is shown under the main menu, next to the text '**Current RTU workspace:**'
 - **New WorkSpace:** Create a new workspace.
 - **Load WorkSpace** or **Load Recent WorkSpace:** Upload all RTUs defined in a workspace. The actual workspace can be reloaded using button . For example, the folder for the workspace shown previously is stored as follow:




Figure 14 – Workspace stored in folders.



- **Exit:** Exit Easergy Builder.
- **Language:** Language selection.
- **Tool:** Configuration of general parameters in Easergy Builder.
- **View:** Change between information in the Console section (Zone 5). Usually, Log Console, Changes Console are available.
- **Help:** Access to Easergy Builder embedded help.










Zone 2. Administration Toolbar

Buttons shown in this toolbar depend on the element selected. On start-up, available buttons in the administration toolbar are:



-  **Add RTU.** Create a new RTU inside the current workspace.
-  **Import an RTU from an EBC file.** This file should have been generated previously with Easergy Builder, using button .





These are the only available buttons when a workspace is opened for the first time. If there are not defined RTUs, no other operations can be done.

When an RTU is selected, the following additional buttons will be shown:

-  **'Remove RTU'.** Remove the selected RTU from the Workspace. All its configurations will be removed also.
-  **'Reboot RTU'.** Reboot the selected RTU
-  **'Create Configuration'.** Create a new configuration for the selected RTU.
-  **'Read Configuration'.** Transfers the configuration files from the selected RTU to the Workspace. All read data will be associated to the selected RTU. In the reading process, the user will be prompted about the data to be transferred: Networks configuration, environmental variables, modems, etc. The type of data available to be read are depending on the type of RTU selected
-  **'Completely Configure RTU'.** Send the complete configuration to the RTU (interfaces, coreDb, ...).
-  **Save the selected RTU.** Save all configurations for this RTU in a file in order to import into other WorkSpace using button . The file type is depending on the type of CPU to be set.
-  **Load a Configuration.** Use this button to load a configuration, RTU or a configuration project generated by CATconfig Tool. This option allows using Easergy Builder with legacy configuration projects.
-  **Syslog.** Load the sysLog file from the RTU to the PC.

If a Configuration is selected, the following buttons are available:

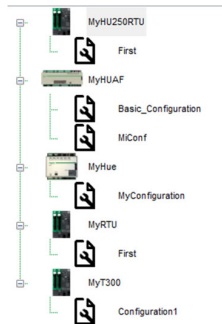
-  **Configure.** Switch to configuration mode to edit the selected RTU Configuration.
-  **Send Configuration to RTU.** Transfer the selected configuration to the RTU.

-  **Remove Configuration.** Remove the selected configuration from the RTU.
-  **Save the selected configuration.** Save this configuration to import into another workspace using button . The configuration can be saved as an RTU configuration file. This configuration will contain only the application part and not the system part (network, modems ...).
-  **Clone Configuration.** Copy the selected configuration with a different name in the same RTU. This is useful for versioning.








Zone 3. RTUs Area

In this area, all RTUs defined in the workspace are shown.






Figure 15 – RTU tree.



Right-clicking on an RTU, the following actions are available:

- Remove RTU ().
- Reboot RTU ().
- Create Configuration ().
- Read Configuration ().
- Completely configure RTU ().
- Save the selected RTU ().
- Load a configuration ().

Right-clicking on a Configuration, the following actions are available:

- Configure ().
- Send Configuration to RTU ().
- Remove Configuration ().
- Save the selected Configuration ().
- Clone Configuration ().

Zone 4. Edition Area

The content of this zone depends on the element selected (RTU or Configuration) and the type of this element (Saitel DR, Saitel DP or T300).


For example, if a Saitel DR-HUe is selected in the RTU tree, the following information will be shown in the edition area:

Figure 16 – Information on the edition area for a HUE RTU.

RTU Type: SaitelDR-HUE RTU Description: My RTU with HUE

Default configuration of new Configurations

AB_SER 0

RTU acquisition 

RTU Redundancy

Connection

IP: 10 . 1 . 1 . 1 Transfer Timeout: 50000 ms

Network Environment variables

Interfaces Routers Firewall

Name	IP Address	Subnet Mask	DHCP Server	Type	Channel	SS
LAN1	10.1.1.1	255.0.0.0				
LAN2	192.168.1.1	255.255.255.0				
MNT	192.168.2.1	255.255.255.0				

If the RTU selected is an PowerLogic T300 the editing area shows the following information:

Figure 17 – Information on the edition area for an PowerLogic T300 RTU.

RTU Type: Easergy T300 v1.5 RTU Description:

Connection

IP: 192 . 168 . 0 . 254 Transfer Timeout: 50000 ms

SSH port 22

Network Environment variables Dialup Modems PPP Hardening

Interfaces Routers Firewall

Name	IP Address	Subnet Mask	DHCP Server	Type	Channel	SS
LAN	192.168.0.254	255.255.255.0	Y			
WAN	192.168.1.10	255.255.255.0				
WIFI	192.168.2.254	255.255.255.0	Y			

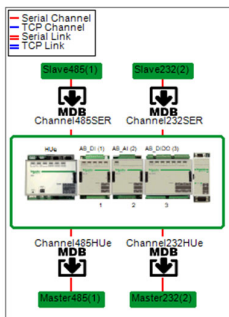
If a Configuration is selected, for example a Saitel DR RTU, the following information will be shown in the editing area:

Figure 18 – Information on the edition area for a Configuration.

Configuration

Redundant configuration

AB_SER 1

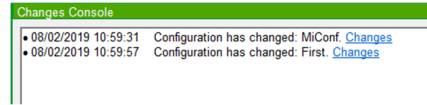


Zone 5. Console

The information in this area depends on the type of console that was selected at the bottom of the tool window or using option View in the Main menu.

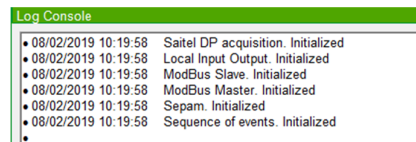
- **Changes Console.** Shows in chronological order the changes that have been made in the different configurations stored in the workspace. To visualize the information, select '**View → Changes Console**' in the main menu. Each message shows the configuration name that was changed, and more details are shown by clicking on the blue text.

Figure 19 – Information on the Changes Console.



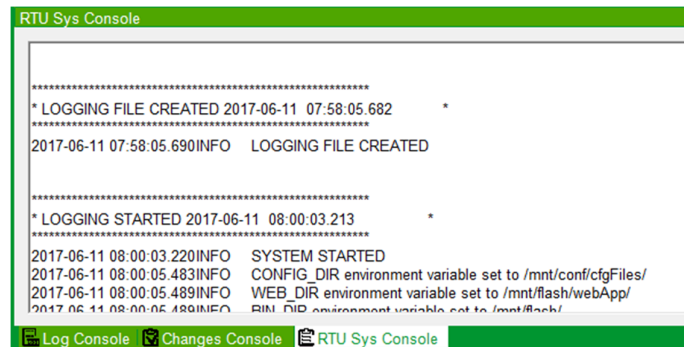
- **Log Console:** Shows Easergy Builder information messages. To visualize this information, select '**View → Log Console**' in the main menu or select the corresponding button in the bottom bar of the tool.

Figure 20 – Information on the Log Console.




- **RTU Sys Console:** Shows the last sysLog that has been read from the RTU. It will only be available if at least one sysLog has been read at some time since the last Easergy Builder boot.

Figure 21 – Information on the RTU Sys Console.



NOTICE

Messages in the 'RTU Sys Console' are not automatically refreshed in Easergy Builder. The information is read when the user use button  in the workspace mode. The information will remain static until the user requests the reading again.

1.3.2 Configuration Mode


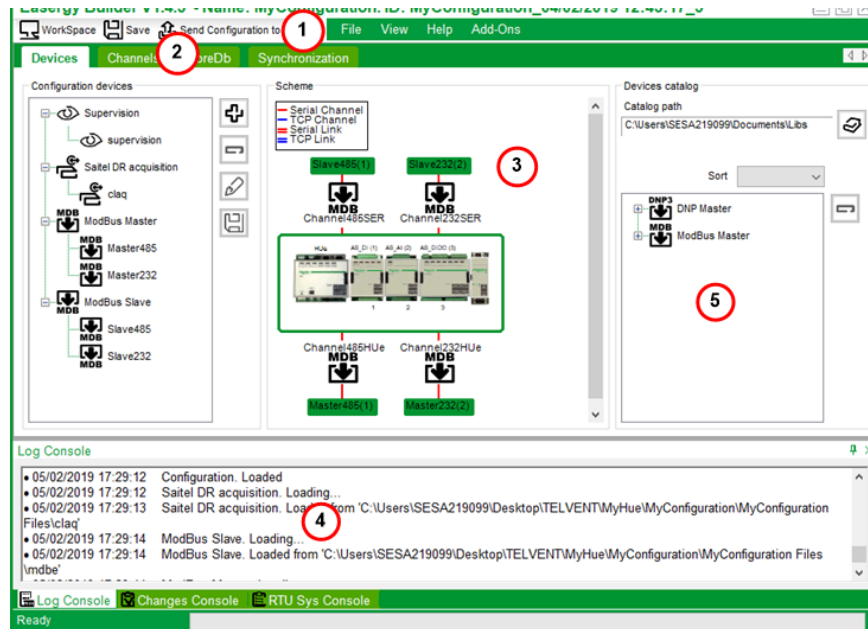
Easergy Builder activates the Configuration mode when the user double-clicks on a Configuration or selects a configuration and presses button .

Figure 22 – Configuration mode.



In the Configuration mode, the following areas are available:

- 1: Main menu
- 2: Section menu: Type of information to be shown on the editing zone. Devices, Channels, coreDb and Synchronization are available.
- 3: Edition zone
- 4: Console view
- 5: Devices catalog

In this mode, the user can:

- Configure de Devices to be used.
- Configure the communication channels.
- Manage the coreDb content.
- Configure the synchronization mechanisms.

Zone 1. Main Menu

The following options are available in the main menu:

- **WorkSpace**: Switch back to the Workspace mode.
- **Save**: Save the information of the current configuration to the PC.
- **Send Configuration to the RTU**: Send the current configuration to the RTU.
- **File**: Exit is the only available item:
 - **Exit**: Exit Easergy Builder.
- **View**: Allows to show 'Log Console' or 'sysLog Console' on the bottom of the Easergy Builder window.
- **Help**: General information about Easergy Builder and installed Plugins.

- **Add-Ons:** An Add-on is an external application, implemented as a dynamic library (DLL). This option shows all installed add-ons on the PC. To access webUI Editor, click on Add-Ons.

Zone 2. Section Menu

These tabs allow to the user access to the following information:

- **Devices:** A device is a set of information exchanged between coreDb and a generator/consumer of information.
- **Channels:** RTU communication channels.
- **coreDb:** Real-time database of the RTU. Information about the points associated to the configured Devices.
- **Synchronization:** Definition and configuration of synchronization devices.

Zone 3. Edition Area

The content of this zone depends on the tab selected in the Section Menu. More information below in this manual.

Zone 4. Console View

Information shown in this zone depends on the type of console that we have selected (bottom bar of Easergy Builder) and whether the RTU log has been previously read (syslog).

More information in paragraph 1.3.1

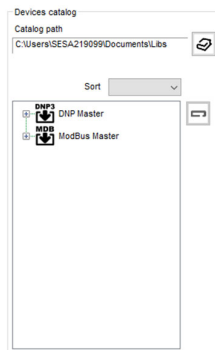
Zone 5. Device Catalog

The Device catalog is a library where Devices template are stored. The user can create new templates (stored in the PC) or re-use each one to create new Devices in several Configurations and RTUs.

A new template can be created right-clicking on a Device on the Device tree and selecting 'Create template'. A name is requested for the new template and it is generated with the same structure and settings as the Device used.

Available templates are shown in the Device catalog.

Figure 23 – Device catalog.



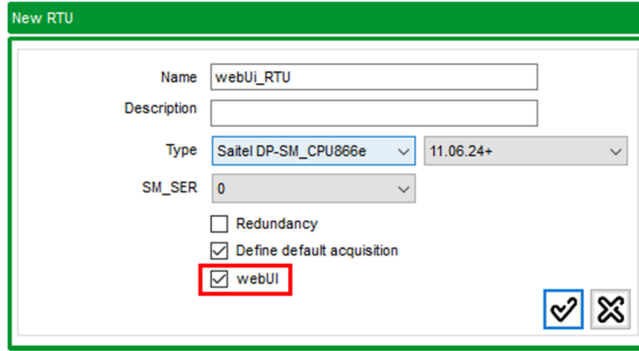
1.3.3 webUI

1.3.3.1 webUI Description

The webUI Editor is an Add-On (only available for SM_CPU866e) included in the Easergy Builder configuration tool that allows the design and edition of the graphics that are later displayed in real time in webUI.

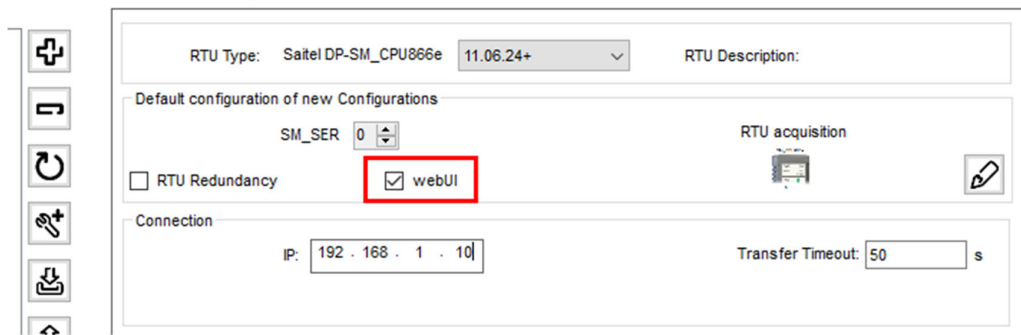
To access webUI Editor it must be selected in the configuration environment. When a new SM_CPU866e RTU is added to the workspace, user can select the webUi checkbox as shown in Figure 24.

Figure 24 – New RTU webUi.



If the above mentioned checkbox was not selected, it is possible to enable webUI by selecting the webUI checkbox in the Edition area of the Workspace as shown in Figure 25. Once the webUI checkbox is selected, it is possible to access the 'webUI Configuration' menu.

Figure 25 – webUi enabling.

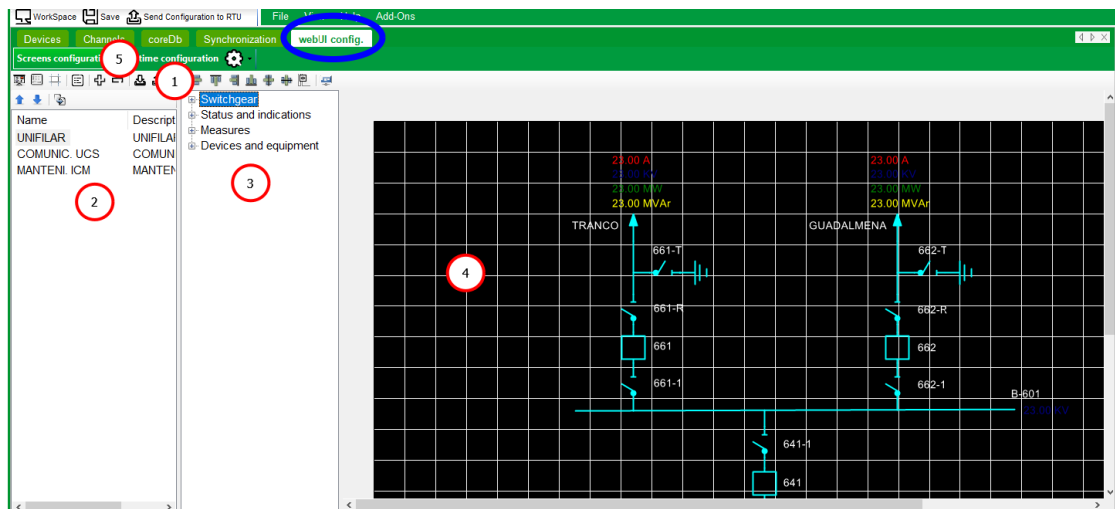


In the Easergy Builder main menu select Add-Ons and 'webUI Configuration'. If this option is not available, install the plugin 'webUI_SPlugin.msi' supplied with the Easergy Builder installation package.

webUI configuration for Easergy Builder completes the functionality of Easergy Builder with the necessary functions to perform the configuration of the embedded webUI system. This tool offers a series of windows and functions to make a set of allowing monitor and control the system.

With these considerations, webUI Configuration for Easergy Builder allows to configure the screens included in the system, as well as the graphical objects that will be represented in each of them.

Figure 26 – webUI main configuration screen.



The webUI Editor is shown as one more tab in Easergy Builder (marked in blue in the figure above).

This window has five different areas:

- **Zone 1:** A toolbar at the top that includes a series of buttons that allow:
 - Show or hide the different areas of the editor.
 - Show and hide the design grid.
 - Manage and use the graphical display repository.
 - Access to the editing help tools (object alignment).
 - Preview the selected screen.
- **Zone 2:** Name and description of the screens created in the system, where they are presented in a hierarchical way.
- **Zone 3:** Tree of graphic objects available to be used on the screens.
- **Zone 4:** Edition area, where the user can design the aspect that will show in real time the screen selected in zone 2.
- **Zona 5:** Two configuration tabs, one for screens and one for the database, and a settings tab.

For more information about webUI, please see user manual webUI.

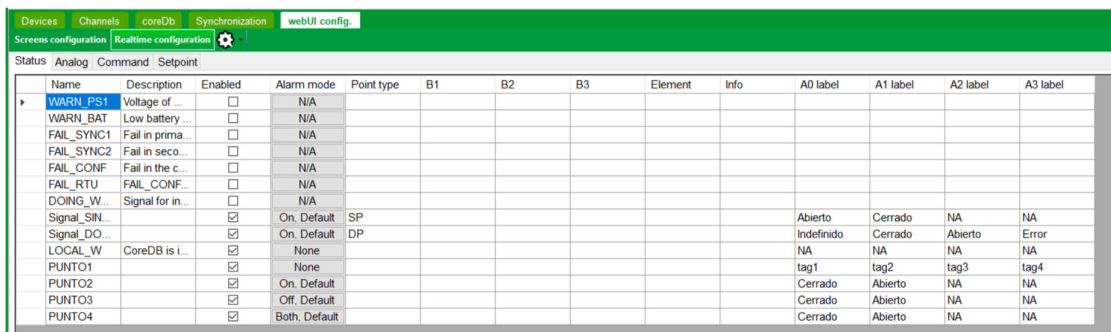
1.3.3.2 Real Time Configuration

From this view it can make the configuration of the points that will be managed by the webUI, indicating for each of them the special characteristics that configure them.

The presentation is divided into five tabs, Status, Analog, Command, String and Setpoint. Each tab presents the list of points of the indicated type registered in the coreDb database. For each point its name and description is indicated, as well as the parameters that apply to its webUI configuration.

For each point, use the associated enabled box to indicate if the point is managed by the webUI or not

Figure 27 – webUI real time configuration.



Name	Description	Enabled	Alarm mode	Point type	B1	B2	B3	Element	Info	A0 label	A1 label	A2 label	A3 label
WARN_PS1	Voltage of ...	<input type="checkbox"/>	N/A										
WARN_BAT	Low battery ...	<input type="checkbox"/>	N/A										
FAIL_SYNC1	Fail in prima...	<input type="checkbox"/>	N/A										
FAIL_SYNC2	Fail in seco...	<input type="checkbox"/>	N/A										
FAIL_CONF	Fail in the c...	<input type="checkbox"/>	N/A										
FAIL_RTU	FAIL_CONF...	<input type="checkbox"/>	N/A										
DOING_W...	Signal for in...	<input type="checkbox"/>	N/A										
Signal_SIN...		<input checked="" type="checkbox"/>	On. Default	SP						Abierto	Cerrado	NA	NA
Signal_DO...		<input checked="" type="checkbox"/>	On. Default	DP						Indefinido	Cerrado	Abierto	Error
LOCAL_W	CoreDB is l...	<input checked="" type="checkbox"/>	None							NA	NA	NA	NA
PUNTO1		<input checked="" type="checkbox"/>	None							tag1	tag2	tag3	tag4
PUNTO2		<input checked="" type="checkbox"/>	On. Default							Cerrado	Abierto	NA	NA
PUNTO3		<input checked="" type="checkbox"/>	Off. Default							Cerrado	Abierto	NA	NA
PUNTO4		<input checked="" type="checkbox"/>	Both. Default							Cerrado	Abierto	NA	NA

For more information about real time configuration, please see user manual webUI.

2 Working with RTUs

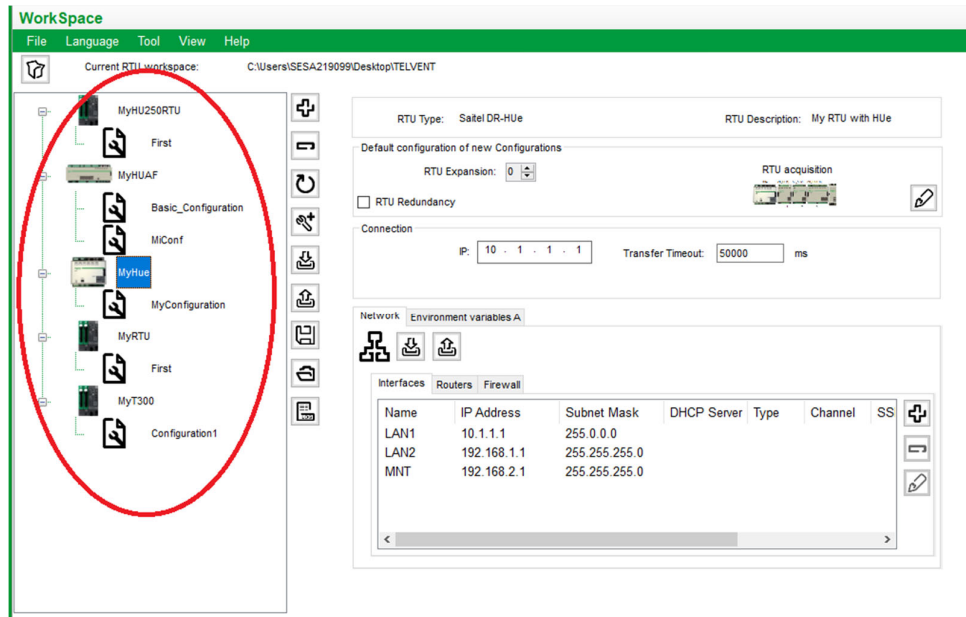
Content

2	WORKING WITH RTUS	31
2.1	INTRODUCTION	33
2.2	RTU BASIC ADMINISTRATION.....	34
2.2.1	CREATING A NEW RTU.....	34
2.2.2	REMOVING AN RTU.....	35
2.2.3	REBOOTING AN RTU	35
2.2.4	READING THE RTU CONFIGURATION.....	36
2.2.5	SENDING INFORMATION TO THE RTU	37
2.2.6	EXPORTING AN RTU OR CONFIGURATION.....	37
2.2.7	IMPORTING AN RTU OR CONFIGURATION	38
2.2.8	READING SYSLOG	39
2.2.9	CREATING A CONFIGURATION	39
2.3	CONFIGURING ARCHITECTURE	40
2.3.1	BASIC CONFIGURATION	41
2.3.2	CONFIGURING A DEFAULT LOCAL ACQUISITION.....	43
2.3.3	CONFIGURING RTU USERS.....	45
2.3.4	CONFIGURING NETWORK.....	46
2.3.5	HARDENING	51
2.3.6	ENVIRONMENT VARIABLES.....	52
2.3.7	CONFIGURING MODEMS.....	53
2.4	WORKING WITH CONFIGURATIONS.....	54
2.4.1	DEVICES	55
2.4.2	COMMUNICATION CHANNELS	56
2.4.3	SYNCHRONIZATION.....	62
2.5	CONFIGURING A SAITEL RTU.....	70
2.5.1	CREATING THE RTU INTO A WORKSPACE	70
2.5.2	RTU PARAMETERS.....	71
2.5.3	CREATING A CONFIGURATION	72
2.6	WORKING WITH POWERLOGIC T300.....	74
2.6.1	SENDING CONFIGURATION FILES TO THE RTU.....	74
2.6.2	CILO CONFIGURATION (COMMAND MANAGEMENT).....	75

2.1 Introduction

With the tool in WorkSpace mode, the user can manage several RTUs from a single interface. Different configurations can be assigned to each RTU, and the user can switch from one to another.

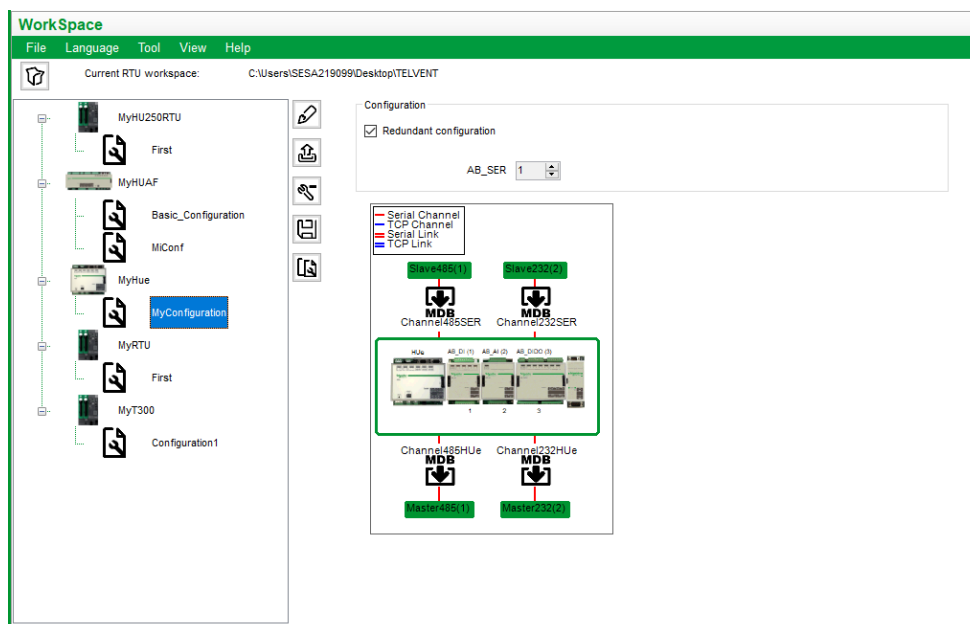
Figure 28 – Easergy Builder in WorkSpace mode (RTU parameters).



In the previous picture, there are several RTUs defined. All of them have one or more configuration assigned.










Depending on the element selected in the RTU tree, the editing zone will show the RTU information (MyHUe is selected in the previous picture) or the configuration information (a communication schematic diagram as picture below).

Figure 29 – Easergy Builder in WorkSpace mode (Configuration schematic).



2.2 RTU Basic Administration

In the Workspace mode, the following buttons are available:

- Create a new RTU ().
- Remove RTU ().
- Reboot RTU ().
- Create a new Configuration ().
- Read a Configuration ().
- Send the complete configuration to the RTU ().
- Export RTU ().
- Load RTU ().
- Read sysLog from the RTU ().

2.2.1 Creating a New RTU


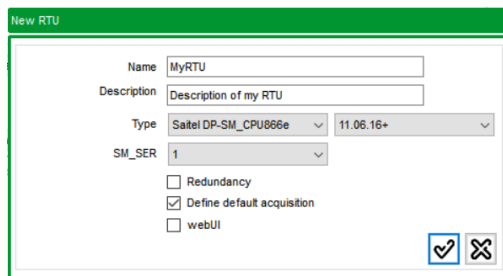
Press button  or right-click on the RTU tree area to add a new RTU by entering the required information in following fields:

Figure 30 – Creating a new RTU.



Where:

- **Name:** RTU name. The length cannot exceed 64 characters and cannot contain the characters \, /, :, *, ?, ', <, > or |) In the RTU tree, a new RTU will be identified with this name.
- **Description:** RTU description, 128 characters maximum (optional).
- **Type:** Depending on the CPU, Easergy Builder supports the following types of RTU:
 - PowerLogic T300: T300 using the HU250 as CPU. Versions available: v1.0, v1.2, v1.3, v1.4, v1.5, v1.6 and v1.7.
 - Saitel DP-SM_CPU866: Saitel DP RTU using SM_CPU866 module as CPU.
 - Saitel DP-SM_CPU866e: Saitel DP RTU using the SM_CPU866e module as CPU. Two versions available: V0 (Baseline 11.05.XX and previous) and V1 (Baseline 11.06.00 and later).
 - Saitel DR-HU120: T150 RTU using the HU120 module as CPU. This CPU is not available yet.
 - Saitel DR-HUe: Saitel DR RTU using the HUe module as CPU.
 - Saitel DR-HU_A: Saitel DR RTU using the HU_A module as CPU.

- Saitel DR-HU_AF: Saitel DR RTU using the HU_AF module as CPU.
- Saitel DR-HU_B: Saitel DR RTU using the HU_B module as CPU.
- Saitel DR-HU_BI: Saitel DR RTU using the HU_BI module as CPU.

NOTICE

For revision 1.8 and later of this manual, all information about old Saitel CPUs has been removed: HU_A, HU_AF, HU_B, HU_BI and SM_CPU866.

- **SM_SER or AB_SER:** Only for Saitel RTUs. Number of communication modules installed in the backplane or ITB. These modules will not be shown in Easergy Builder, but their ports will be available to be used as additional communication channels. This field depends on the CPU type:
 - SM_SER is available for SM_CPU866e.
 - AB_SER is available for HUE.
- **Redundancy:** Check this field if the RTU is CPU-redundant. Not available PowerLogic T300.
- **Define default acquisition:** When the RTU is created, the acquisition modules in the RTU should be included. These modules will be added by default when new configurations are created for this RTU. Check this field to include a set of acquisition modules as default. The following paragraph detail how to create a default configuration with Saitel DP and Saitel DR RTUs.
- **webUI:** Only available when SM_CPU866e is selected. Checking this field allows to use webUI in the configuration environment of the tool.

2.2.2 Removing an RTU


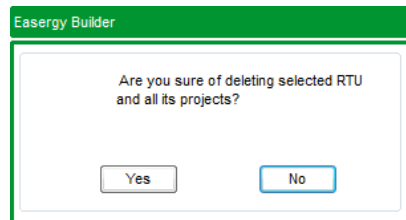
To remove an RTU and all its associated configurations from Easergy Builder, select the RTU and press the  button. A confirmation window will be shown:

Figure 31 – Removing an RTU.



Press 'Yes' and the RTU will disappear from the workspace. Press 'No' to cancel de operation.

2.2.3 Rebooting an RTU


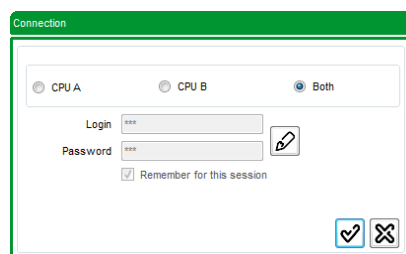
Use button  to reboot an RTU (It is like executing 'reboot' on the console tool). If the RTU is configured as redundant, Easergy Builder will ask which CPU is rebooted.

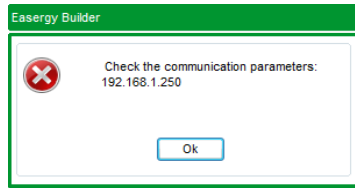
Figure 32 – Rebooting the RTU.



Select 'only CPU A', 'only CPU B' or 'Both'.


If Both is selected, CPU A will be rebooted first. If the reboot is completed successfully, Easergy Builder will try to reboot CPU B. If the reboot malfunctions (for example caused by a connection problem), the reboot operation will be stopped, and an error message shown:

Figure 33 – Error rebooting an RTU.



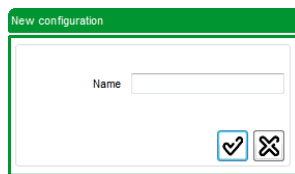
2.2.4 Reading the RTU Configuration

For an RTU, a new configuration is created using the information stored inside the RTU.

To accomplish this, select the RTU in the Workspace and press button  in the toolbar. Easergy Builder will ask for a valid user to retrieve the information. If the RTU is redundant, CPU A or CPU B must be selected.

Next, select a name for the new configuration:

Figure 34 – Importing the configuration from the RTU.



If the connection is established (see information in the Log Console), the user is asked for the type of information to import. The type of information to be imported will depend on the type of RTU.

Figure 35 – Selecting the information from an PowerLogic T300 RTU.

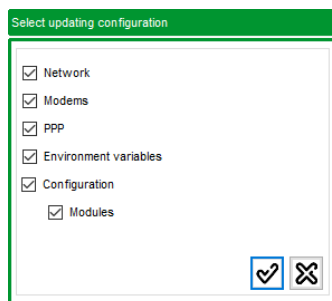
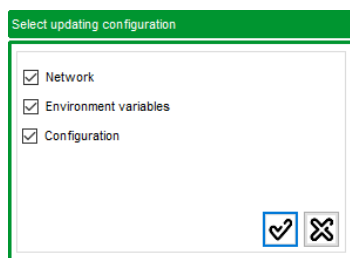


Figure 36 – Selecting the information from a Saitel DR (HUE).



Check all information to be imported and the file transfer starts (see information in the Log Console). All files containing the requested information will be transferred from the RTU to the PC. In the Workspace, the RTU tree view will add this new configuration associated to the selected RTU.


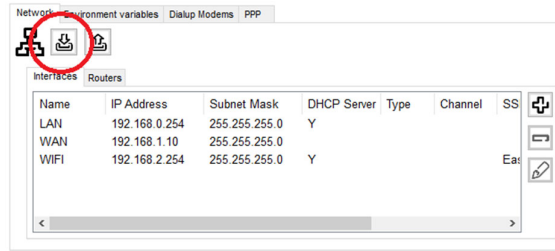
If button  on the tab is used, only the network information is transferred from the RTU. For example, to import the network configuration (Ethernet ports), use the following button:

Figure 37 – Importing only a type of information.



This button is available for Network, Environment variables, Dialup Modems, PPP and Hardening.

2.2.5 Sending Information to the RTU


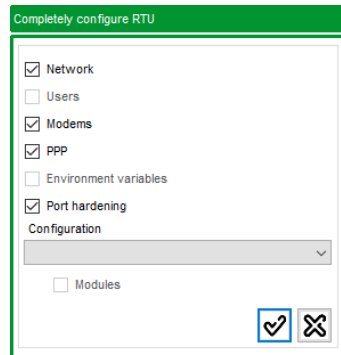
To send the information to the RTU, press button  and select the information to send. The information that is available to check depends on the type of RTU.

Figure 38 – Sending information to the RTU.



If a configuration is selected, all the files corresponding to that configuration will also be sent.

If only a type of information needs to be sent, use the button on the tab containing this information. To send only the configuration files, go into the Configuration mode and use the button '**Send Configuration to RTU**' as is explained in the next chapter.

2.2.6 Exporting an RTU or Configuration


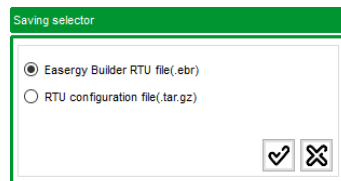
Button  allows exporting the RTU information to a file, including all its Configurations. This information can be exported to an Easergy Builder file (.EBR) or to a compressed file (.tar.gz).

Figure 39 – Type of file to generate in the export of the RTU.



Both files allow later to import the image from the Easergy Builder environment (see paragraph 2.2.7). The compressed file can be also used to import the information with the WEB tool (webApp).

NOTICE

Compressed file is only available for RTU using webApp (PowerLogic T300, SM_CPU866e V1 and HUE).


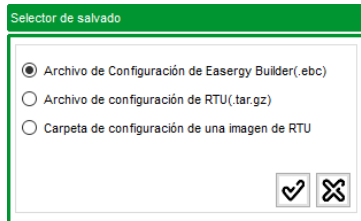
To save a Configuration, select it in the tree and press . The following options are available:

Figure 40 – Type of file to generate in the export of a Configuration.



Depending on selection, one of the following is generated:

- A file with proprietary format of Easergy Builder (.EBC).
- A compressed file(.tar.gz)
- A folder with the image of the selected Configuration, that is, it directly stores Configuration XML files in this folder. This option is only available for Saitel RTUs.

⚠ WARNING

When a folder is selected to save the image of the RTU, all the previous content in the folder will be deleted. If the folder is NOT empty, the user must confirm the operation before deleting the content.

2.2.7 Importing an RTU or Configuration


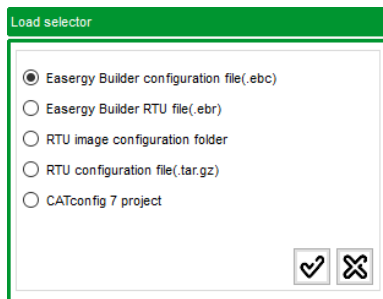

Button  allows importing information of an RTU or Configuration from a file or folder stored in the PC. This button is available only if an RTU is selected in the tree.

Figure 41 – Importing information from a file.



The information to be imported can be stored in different formats:

- A file with proprietary format of Easergy Builder (.EBC). This file only contents information about a Configuration (database, communication channels, synchronization, etc.). It does not include information of the RTU (users, network interfaces, ...). The new configuration is created under the selected RTU when the button  was pressed.

⚠ WARNING

Before importing a Configuration file (EBC), the user must be sure the information to import corresponds to the same RTU type that was selected in the tree. Easergy Builder **does not do** this check.

- A file with proprietary format of Easergy Builder (.EBR). This file was generated from Easergy Builder (see previous section) and it contains complete information of an RTU, including its Configurations.
- A folder containing Configuration XML files (see previous section).
- A compressed file (.tar.gz) containing the information to import, of an RTU or Configuration (see previous section).
- CATconfig 7 project. This option is very useful for working with old configuration that were generated with CATconfig tool.

2.2.8 Reading sysLog


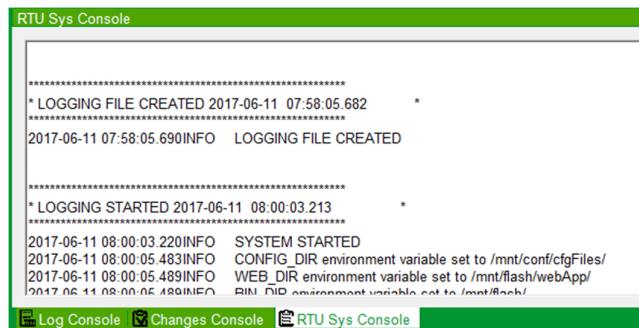
Button  allows reading the sysLog file from an RTU. An authorized user must be indicated, and the file will be transferred to the PC. The information is shown in the 'RTU Sys Console' (see Zone 5 in paragraph 1.3.1).

Figure 42 – sysLog imported from the RTU.



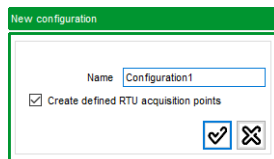
NOTICE

This information is a hardcopy of the moment when the sysLog was requested by the user from Easergy Builder. To update this information the user must request the sysLog again.

2.2.9 Creating a Configuration

Use button  to create a new configuration for an RTU:

Figure 43 – New configuration for an RTU.



Write the name of the new configuration.

If the RTU was created with an acquisition configuration by default (only for Saitel, see paragraph 2.3.2), select the field 'Create defined RTU acquisition points' and all points of the pre-defined I/O modules will be included in coreDb tables. (Only for Saitel).

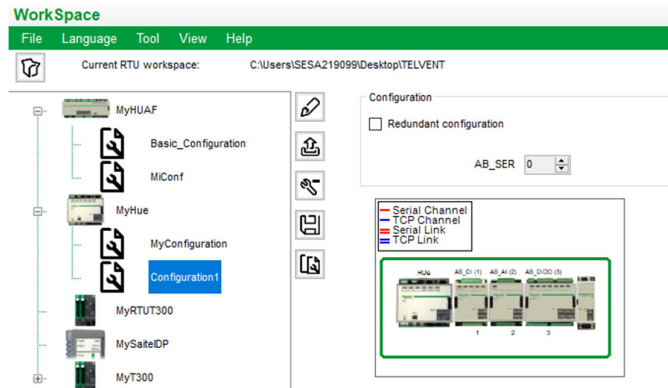
For example, if a HUE RTU was created with a default configuration including one AB_DI, one AB_AI and one AB_DIDO, selecting 'Create defined RTU acquisition point', in coreDb the following points will be included:

- 32 digital inputs in table Status (16 of AB_DI and 16 of AB_DIDO).
- 8 analog inputs in table Analog.
- 8 digital outputs in table Command.

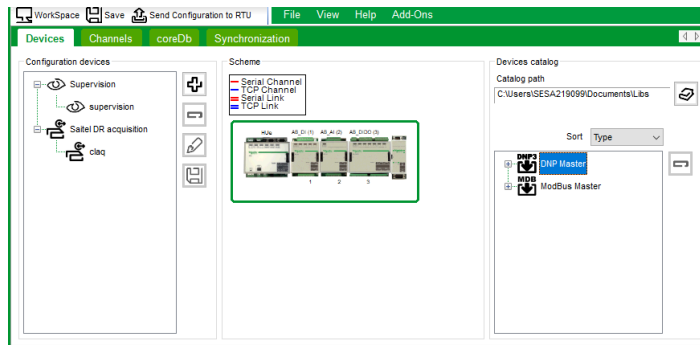
Other supervision and diagnostic points are automatically generated. For PowerLogic T300, only supervision points will be included in coreDb (Status, Command, Analog and String).

The new configuration is available for the RTU.

Figure 44 – New configuration for an RTU.



Double clicking on it will switch to Configuration mode where the new configuration can be edited.

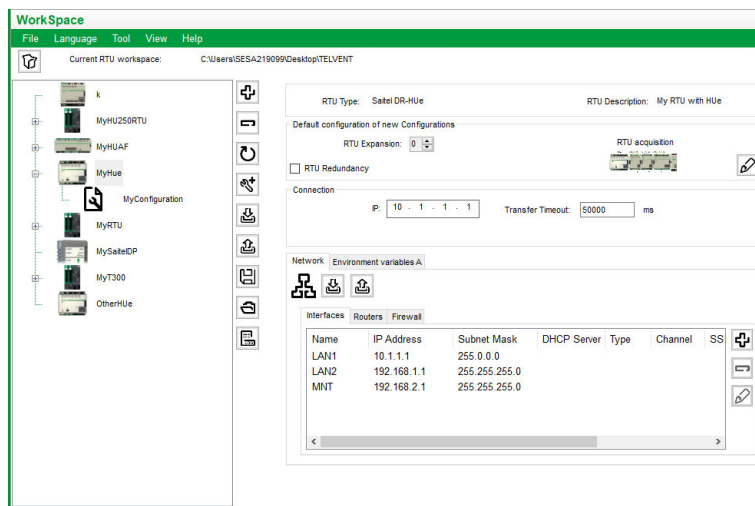


More information about this Configuration Mode in next chapter of this manual.

2.3 Configuring Architecture

Select an RTU in the tree and the following parameters can be set:

Figure 45 – RTU configuration parameters.



- RTU Type: Only available for SM_CPU866e and PowerLogic T300.
- Default values for new configurations (Number of communications modules, ITB or backplane composition and CPU redundancy).
- Connection parameters (IP address and transfer timeout)
- Network parameters.
- Environmental variables
- Other accessing parameters (only for PowerLogic T300):
 - Communication devices (Dialup modem).
 - PPP connections.
 - Hardening (only for v1.4 or later)

2.3.1 Basic Configuration

For a Saitel DR RTU, the following CPUs are available: HUE.

For a Saitel DP RTU, the following CPUs are available: SM_CPU866e (version V0 and V1) .

- Select SM_CPU866e V0 if the Baseline installed in the CPU is 11.05.XX or previous. These Baselines don not include the cybersecurity brick.
- Select SM_CPU866e V1 if the Baseline installed in the CPU is 11.06.00 or later. These Baselines include the cybersecurity brick.

For PowerLogic T300, the versions v1.0, v1.2, v1.3, v1.4, v1.5, v1.6 and v1.7 are available.

The CPU type is selected when the RTU is created and can NOT be modified later.

Select the number of communication modules to be included (AB_SER for Saitel DR or SM_SER for Saitel DP). These modules are not shown in the schematic picture of the ITB, but its communication ports will be created, and they can be used for configuring communication channels.

'**RTU Redundancy**' field only is available for Saitel RTUs. If '**RTU Redundancy**' is unselected (or it is not available) only the IP address (corresponding to a unique CPU) must be configured:

Figure 46 – Configuring the IP address.

The screenshot shows a configuration window titled "Connection". It contains three input fields:

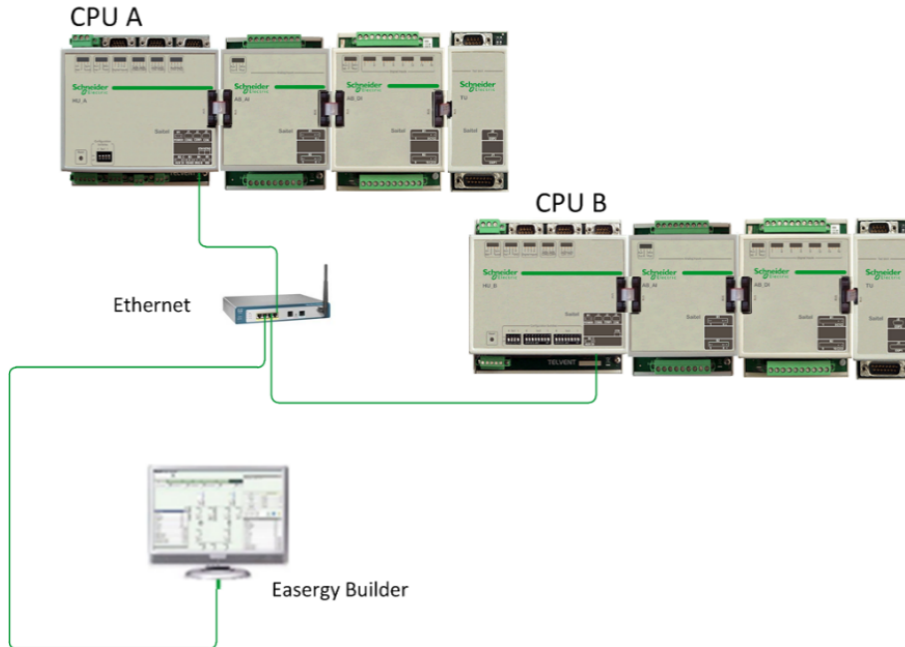
- IP:** A text box containing the IP address "192 . 168 . 0 . 254".
- Transfer Timeout:** A text box containing the value "50000" followed by the unit "ms".
- SSH port:** A text box containing the value "22".

'**Transfer Timeout**' is the time (in milliseconds) to consider the FTP connection broken when a configuration is being sent to the RTU (default value: 50 s).

'**SSH port**' is only available for PowerLogic T300. It allows to indicate the port for FTP connection.

'**RTU Redundancy**' field must be selected for redundant configurations. The following picture shows an example of Saitel DR redundant RTU.

Figure 47 – Redundant Saitel DR RTU.



In this case, in Easergy Builder only one ITB should be define with two IP addresses, CPU A (primary) and CPU B (secondary).

Figure 48 – Configuring redundant CPUs.

Connection

IP CPU A: Transfer Timeout: ms

IP CPU B:

When the configuration is sent to the RTU, select '**CPU A**', '**CPU B**' or '**Both**'.

Figure 49 – Sending the configuration to redundant CPUs.

Connection

CPU A CPU B Both

Login

Password

Remember for this session

If option '**Both**' is selected, Easergy Builder will send the configuration in this order:

- Send configuration to CPU A.
- Send the configuration to CPU B.
- Ask to reboot CPU A.
- Ask to reboot CPU B.

If a communication problem is detected, the operation will stop.

2.3.2 Configuring a Default Local Acquisition


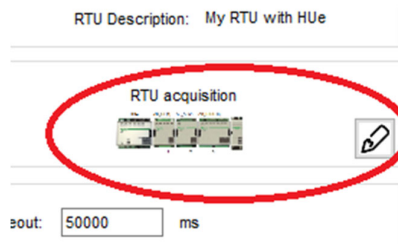
The default local acquisition can be configured when the RTU is being created or later, using button , next to the schematic picture.

Figure 50 – Local acquisition schematic.





Saitel DR Local Acquisition

The user needs understand some basic concepts about Saitel DR before configuring the acquisition:

- An **ITB** is a set of acquisition blocks connected to a CPU (HU)
- An **Acquisition Block** or **AB** is a Saitel DR input/output module.
- Each acquisition block is allocated to a unique address in the ITB, the **Node Number**; this number identifies both the module and its I/O points.
- The procedure **AAP (Automatic Addressing Procedure)** is performed by the operator every time an AB module is added, deleted, replaced or moved inside the ITB. It can be launched manually or automatically by setting switch #3 in the HU module to auto (please, refer to Saitel DR Platform User Manual).

Figure 51 – Saitel DR local acquisition.

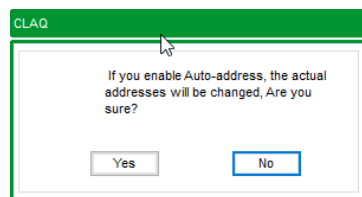


The number between parentheses next to each module name is the node number. Select an AB and use buttons   to change its physical position.

When '**Auto Address**' box is selected (by default), if the ITB is reordered, a module is added or deleted, all addresses are automatically recalculated matching their physical position in the rail. Address number 1 is assigned to the AB closest to the HU.

If '**Auto Address**' box is unselected, modules will retain the allocated address, ignoring any changes made. If it is selected again, the following message will appear:

Figure 52 – Confirmation for automatic addressing.



Select an AB (click on the AB image) and use button  to remove it.


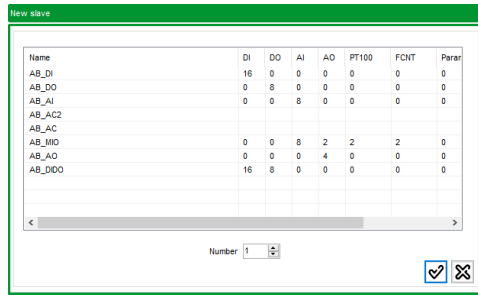
Use button  to add a new AB. Select the type of AB in the following window:

Figure 53 – Adding one (or several) AB.



If 'Auto Address' is selected, several AB can be added at one time. This window allows selecting the quantity of modules to be added. If 'Auto Address' is unselected, only one AB can be added at the same time and the address should be indicated.

Saitel DP Local Acquisition


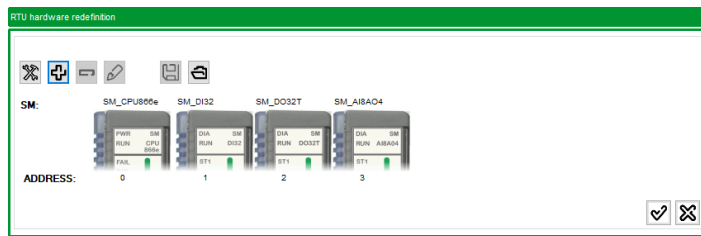
Pressing button  the composition of the RTU can be modified:

Figure 54 – Saitel DP local acquisition.





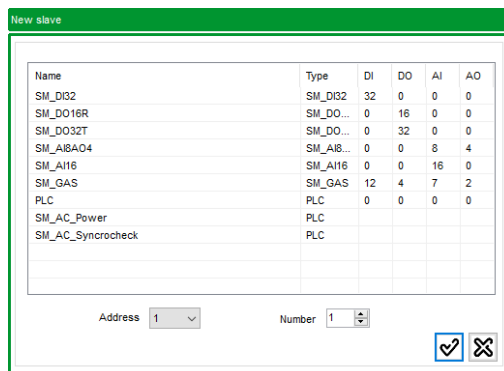

Select a module (click on the image) and use button  to remove it. Use button  to add a new I/O module. Select the type of module in the following window:


Figure 55 – Adding new I/O modules.



NOTICE

Legacy devices are shown in orange color.

The address assigned to the I/O module is shown under its image. It can be changed clicking button . The configuration of any module integrated in Saitel DP, including all the changes made to it can be stored as a template. Thus, this configuration can be repeated in this project or in any other project in the future.

Use button  to use a template, that is, an XLB file containing a pre-configured set of I/O modules


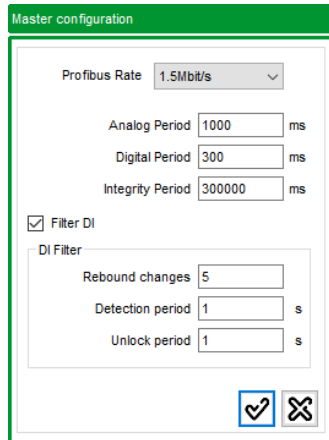
Press button  to configure Profibus configuration and acquisition strategy:

Figure 56 – Profibus configuration and acquisition parameters.



- **Profibus Rate:** CPU communication rate with I/O modules.
- **Analog and Digital Period:** Interval of time for the acquisition of analog and digital input points when they are configured to be updated periodically (ChgEvt of the point in local acquisition is set to 'N'). Default value for digital points is 1000ms (10ds) and for analog points is 300ms (3 ds). Both values can be changed in intervals of 100ms.
- **Integrity period:** When a digital point is set to be updated by event (ChgEvt of the point is set to 'Y'), this value indicates that if during this time no event occurred, the point is updated anyway. This determines the integrity of the point. Default value is 300 s and it can be changed in intervals of 100ms.
- **Filter DI:** This checkbox allows configuring the filtering parameters for digital inputs.
 - **De-bounce changes:** Number of changes necessary to activate the anti-debounce filter (default value, 5).
 - **Detection period:** Time window when the number of rebounds will be counted to activate the debouncing filter blocking the point. This time is expressed in seconds (default value, 1 s).
 - **Unlock period:** Time without changes in a blocked point in order that this point is unblocked. This time is expressed in seconds (default value, 1 s).

2.3.3 Configuring RTU Users

NOTICE

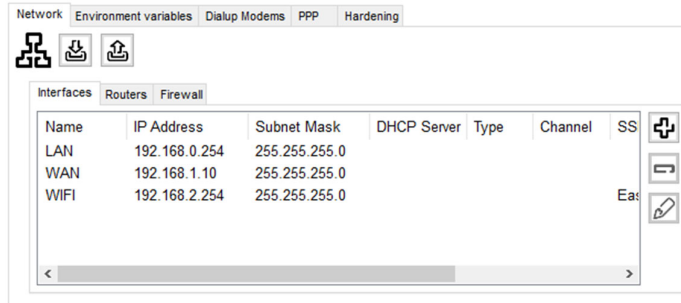
User management is only available for old Saitel CPU without cybersecurity brick: HU_B, HU_BI, HU_A, HU_AF, SM_CPU866 and SM_CPU866e V0).

More information about user management in previous versions of this manual.

2.3.4 Configuring Network

Select tab '**Networks**' in the Edition Area (see Figure 10).

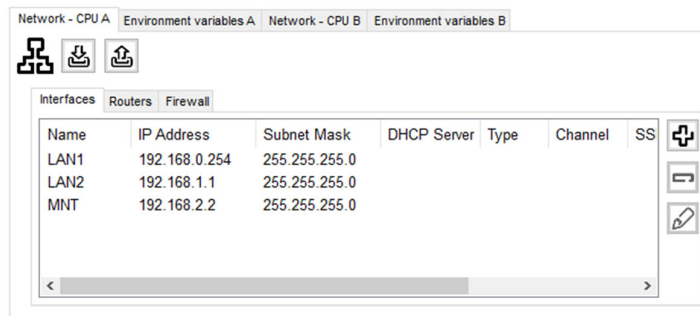
Figure 57 – Network configuration.





If the RTU has one CPU, only section '**Network**' will be shown.

If the RTU is redundant, sections '**Network –CPU A**', '**Network-CPU B**', '**Environment variables A**' and '**Environment variables B**' will be available. IP addresses for CPU A are included in section 'Network – CPU A', and IP address for CPU B are included in 'Network – CPU B'.

Figure 58 – Configuring networks with redundant CPUs.



Press button  in the Network tab to load in Easergy Builder the information about the networks defined in the RTU (CPU A or CPU B if it is a redundant RTU). Use button  to export the network information from Easergy Builder to RTU.

NOTICE

This button in the Network tab only transfers to the RTU the file netConfig.xml. Please, do not confuse this button with the button in the Toolbar next to the RTU tree (see paragraph 2.2.4 and 2.2.5).

Regarding to the networks, the following information should be configured:

- Interfaces. Configuration of the Ethernet ports of the CPU.
- Routers. Devices to allow accessing to other networks.
- Firewall. Access management to the CPU through each Ethernet port. This feature is only available for HUE, SM_CPU866e V1 and PowerLogic T300 v1.4 or later.

Interfaces

Select tab 'Interfaces' in section 'Network':

Figure 59 – Configuring 'Interfaces'.

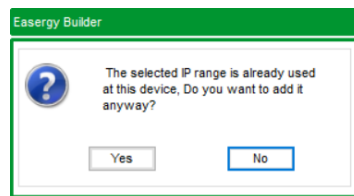
Name	IP Address	Subnet Mask	DHCP Server	Type	Channel	SS
LAN	192.168.0.254	255.255.255.0	Y			
WAN	192.168.1.10	255.255.255.0				
WIFI	192.168.2.254	255.255.255.0	Y			

The number of interfaces to define depending on the CPU type:

- **SM_CPU866e**: ETH1, ETH2, ETH3, ETH4, PRP1 (ETH1Ð2) and PRP2 (ETH3Ð4), BOND1 (ETH1Ð2) and BOND2 (ETH3Ð4). In addition, more than one IP can be assigned to the same port.
- **PowerLogic T300**: WAN, LAN and WIFI.
- **HUe**: MNT, LAN1, LAN2, PRP (LAN1 & LAN2), HSR (LAN1 & LAN2) and RSTP (LAN1 & LAN2). In addition, more than one IP can be assigned to the same port.

In case the IP addresses are in the same range, the following message will be shown:

Figure 60 – IP addresses in the same range.



PRP (Parallel Redundancy Protocol), **HSR** (Highly available Seamless Redundancy), **RSTP** (Rapid Spanning Tree Protocol) and **BOND** (Bonding) protocols allow using two physical ports as a unique logical port, with a same MAC address and IP.

NOTICE

If a PRP, HSR, RSTP or BOND interface is defined, the associated ETH or LAN ports could not be defined and vice versa.




Buttons ,  and  next to the interface list allow to add, remove and edit network interfaces.

Figure 61 – Configuring an Ethernet port.

Only the WAN port in PowerLogic T300 allows to use IPv4 and IPv6 for addressing.

Figure 62 – Using IPv6 for the WAN port (only PowerLogic T300).

To use IPv6 format for this port, '**IPv6**' field must be checked, and fields '**IPv6 Address**' and '**Subnet prefix length**' need to be filled. If the IP address is automatically assigned, mark the field '**Automatic**' and fill in the field '**Interface ID**'. If this value is automatically assigned too, mark the field '**Automatic ID**'.

Regarding DHCP feature, depending on the CPU type, define this interface as a DHCP client, DHCP Server or any.

- If '**DHCP Client**' is selected, it is not necessary fill the '**IP Address**' and '**Subnet Mask**' fields. These data are generated automatically by other DHCP server and they are assigned to the interface.
- If '**DHCP Server**' is selected, fill the fields 'IP Address' and 'Subnet Mask' with the IP address for this interface. On the other hand, an IP range should be set to be assigned to other DHCP clients on the network.

NOTICE

It is possible to define more than one configuration associated with the same physical port, except in case the box 'DHCP Client' is selected.

When a WIFI interface is being defined (only for PowerLogic T300), the configuration window is as follow:

Figure 63 – Configuring a WIFI network interface.

Where:

- **SSID:** Unique identifier for the wireless LAN. By default, this name is 'EasergyT300'.
- **Password and Confirm:** Necessary password that should be set in a device to it can be connected to the wireless LAN. It must be 6-32 characters in length.
- **Hidden SSID:** When this box is selected the SSID visibility is deactivated.
- **Timeout:** Maximum time to wait before deactivating the wireless LAN when any connection is established. This disconnection timeout does not apply when the WIFI was started trough the Local/Remote button.
- **Enabled:** Checking this box allow the use of the wireless LAN. If this box is unselected, the wireless LAN cannot be started any way.

Routers

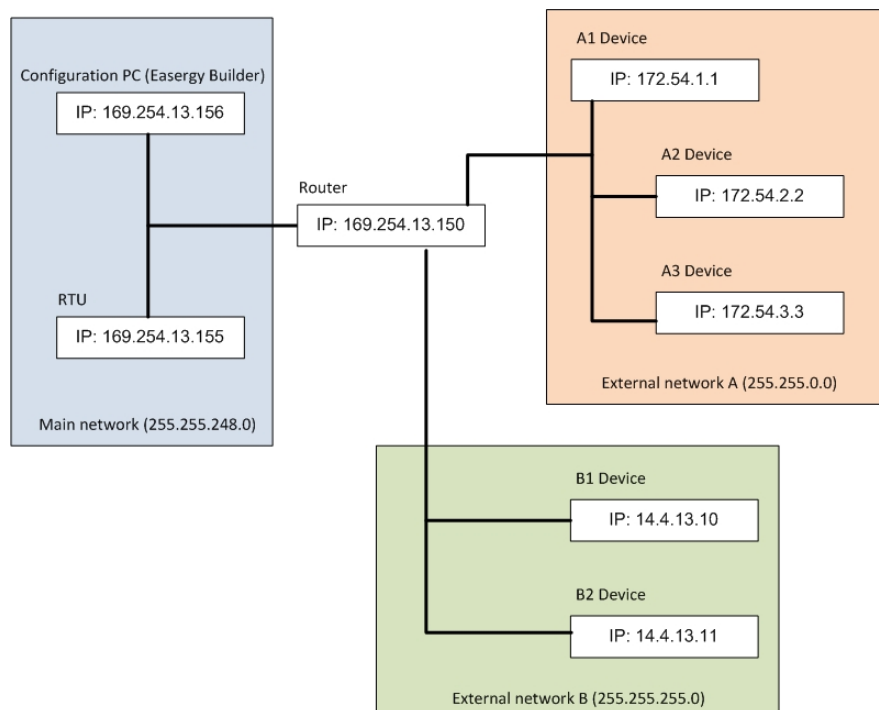
If there are Interfaces on different networks, a router should be defined to access them. Each one of this interfaces has to be defined in the label 'Routers' with the router IP that allows the access.

Figure 64 – Configuring Routers.



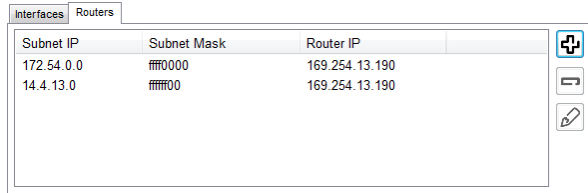
The following figure shows a possible situation where there are two external subnets:

Figure 65 – Example of external subnets.



The configuration is shown in the next figure:

Figure 66 – Routes configuration window




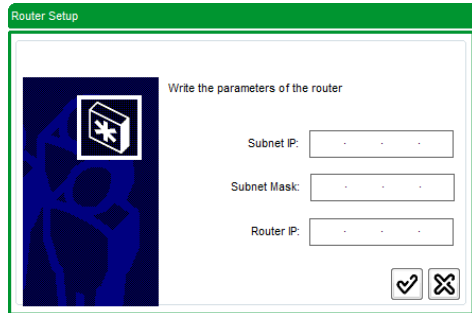
Use button  next to the routers list to add a new router:

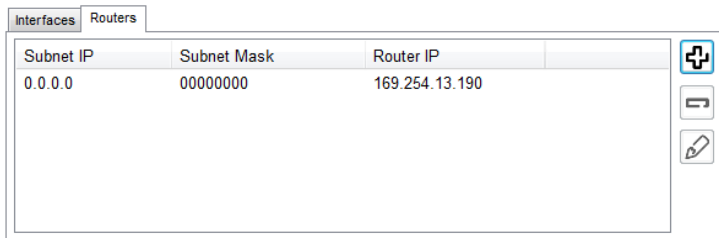
Figure 67 – A new Router for a new subnet.



- **Subnet IP** and **Subnet Mask**: Set of external IP addresses which are accessible through the router.
- **Router IP**: IP address of the router device in the main network.

It is possible to include a single record for an external network where the IP address as well as its mask are the default. Using this unique record allows the CPU access to any external device on a network connected to the router.

Figure 68 – Configuring an external network using the default IP and mask.



Firewall

This feature is only available for HUE, SM_CPU866e V1 and PowerLogic T300 v1.4 or later.

Figure 69 – Configuring a Firewall.



Available ports depend on the CPU type. Previous picture shows the Firewall window for a Saitel DR-HUE.

For each Ethernet port, the following rules could be configured:

- **White list:** A list of IP addresses authorized to access this interface. The mask must be 255.255.255.255.
- **Black list:** A list of IP addresses not authorized to access this interface. All try to access from any of these addresses will be rejected.
- **Blocking a port.** No device can connect using this port.

Firewall configuration is not required on the system. If a port is not configured, it will not have any restrictions or capabilities related to firewall.


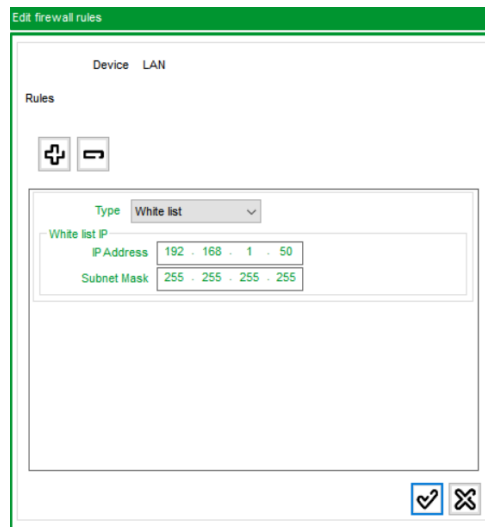

To configure a port, select it in the list and press button .

Figure 70 – Adding rules to the Firewall.



Button  allow adding all rules necessary to manage access to the port.

Select '**White List**' or '**Black List**' in the Type field and specify the set of addresses to include in the list, using the fields IP Address and IP Mask.

To block a port, in the Type field select '**Drop Port**' and indicate the port number and its type (TCP or UDP).

Firewall rules are strictly enforced in the following order:

- First, each port included in a Drop port rule are closed.
- Second, all addresses included in Black lists are rejected.
- Third, all addresses included in White lists are granted to access.

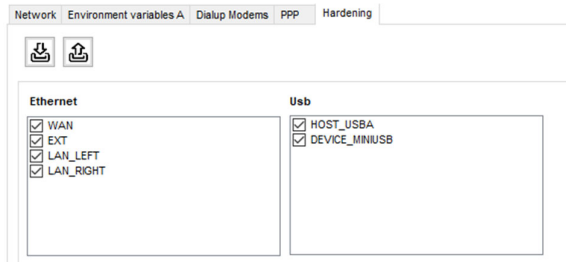
This ordering determines that when information arrives from an address or port, it will be allowed or rejected according to the order of rules. If the source IP or port is not affected by any rules, access to the port is allowed.

2.3.5 Hardening

This functionality is only available for PowerLogic T300. It consists of the ability to physically disable any port.

If in the Firewall tab any port can be blocked at the software level (for some functions), in the Hardening tab any port can be completely blocked.

Figure 71 – Hardening ports.



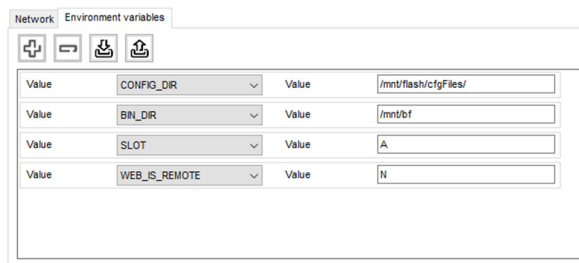
When a port is selected (default value), it means that it is available.


Unselect any port that must be disabled for connections.

2.3.6 Environment Variables

Use this tab to manage the environment variables for a CPU. For redundant systems, two tabs will be available, for CPU A and CPU B.

Figure 72 – Configuring Environment variables.



Environment variables can be added, removed or edited only if previously the variable were read from the RTU with button . While not, only this read button is available in this tab.

The environment variables are stored in the **main_cfg.xml** file. Depending on the type of CPU, the software baseline includes the followings variables:

- **CONFIG_DIR**: Path for configuration files (Default value: /mnt/flash/cfgFiles/)
- **WEB_DIR**: Path for Saitel webTool files (Default value: /mnt/bf/webFiles/)
- **BIN_DIR**: Alternative path for binary files. If the binary files cannot be located in the main path (/mnt/flash), then they will be looked for into the alternative path. (Default value: /mnt/bf).
- **SLOT**: Define the role of this CPU in a redundant system. If this CPU has to start as HOT, SLOT variable must be A, but this CPU is the secondary, it will start as STANBY and SLOT must be B. (Default value: A. This value must be changed to B in the secondary CPU).
- **MONITOR**: Path for the BLMon tool (Default value: /mnt/flash/BLMon/)
- **WEB_IS_REMOTE**: When this variable is set to N, it does not have effect. If it is set to S, in case that RTU is in LOCAL mode, commands through web server cannot be executed. (Default value: N)

2.3.7 Configuring Modems

This feature is only available for PowerLogic T300.

Dialup Modems

Only one slot can be set at the same time. When a modem is set in slot 1 or 2, the other is disabled.

Figure 73 – Configuring modems.

For one Slot, the following information must be configured:

- **Type of device:** Depending on the CPU type, select: K7 3G/2G, K7 4G o K7 4G US. The last is only available for PowerLogic T300 v1.2 and later.
- **Name:** Name of the connection.
- **Network:** Connection type: 3G, 4G, GPRS or AUTO.
- **PIN and Pin number:** Check PIN box if this code is necessary to initialize the connection. If during initialization, the PIN number is empty or wrong, an error message will be returned by the modem. In this case, the PIN initialization will be retried 3 times and stop if the issue is still present.

NOTICE



The PIN Number only can be configured for RTUs with profile v1.6 y previous. For profiles v1.7, and later, these fields must be configured using webApp.

- **Operator:** Only for K7 4G US modems, the network operator must be indicated: AT&T or VERIZON.

PPP

Select tab 'PPP' to configure a PPP connection:

Figure 74 – Configuring a PPP connection.

Use buttons  and  to add a new PPP connection:



- **Name:** PPP connection name.

- **Modem:** Name of the connection by modem used. This name must be defined in tab 'Dialup Modems'.
- **Timeout (in minutes):** If no data flow is detected during this time, a disconnection is made. (Value 0 means disabled).
- **Reboot modem after failed connection:** Mark this box if the modem must be rebooted after max connection retries.
- **APN name:** Access Point Name.
- **Daily disconnect and Hour:** If this box is selected a disconnection is made daily at indicated time in Hour.
- **Auth, Login and Password:** If authentication is required to connection, select **Auth** box and indicate a valid **Login** and **Password**.

NOTICE

Authorized user and password only can be configured for profiles v1.6 and previous. For v1.7 and later, these fields must be configured using webApp.

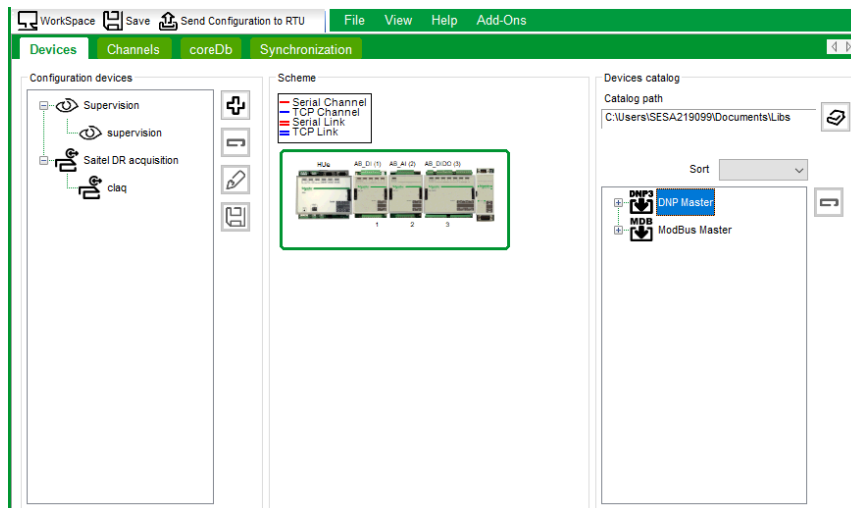
- **Ping:** Select this box to monitor the IP connection. The following data must be indicated:
 - **IP** address of the host to be ping
 - **Interval:** Interval repetition in minutes of the ping process.
 - **Attempts:** Number of successive pings.
 - **Timeout:** Timeout in seconds for ping request.

Use buttons  and  to read / send the information from / to the RTU.

2.4 Working with Configurations

The next figure shows an example of Easergy Builder when in Configuration mode.

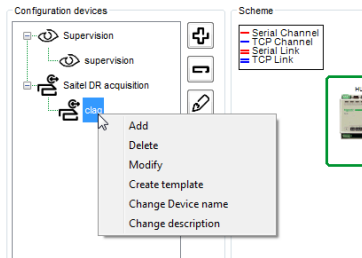
Figure 75 – Easergy Builder in Configuration mode.



More details about different zones in this mode in paragraph 1.3.2

Right-clicking on the configuration name, a contextual menu is displayed:

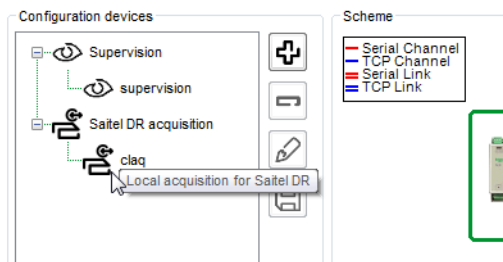
Figure 76 – Contextual menu for Configuration.



This menu allows:

- Add, remove or modify a configuration.
- Create a template with the information associated to this configuration.
- Change the name or description of the Device (description field allows 128 characters maximum. The description is shown when the mouse is located on the Device name as follow:

Figure 77 – Description of a Device.



2.4.1 Devices

The information exchange between the environment and coreDb is made through Devices. Each type of Device defines the rules that must be followed when designing the set of real-time points, and the relationship between them and with coreDb points.

By default, two devices are included in a configuration:

- Supervision
- Local acquisition. This Device depends on the type of RTU:
 - Saitel DR: claq Device.
 - Saitel DP: laq Device.
 - PowerLogic T300: lioc Device.


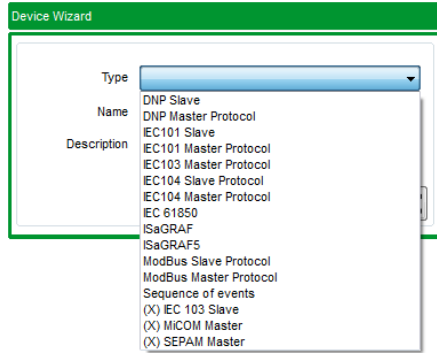
Each Device is associated with a Plugin that must be installed in Easergy Builder and a Controller that must be available in the RTU. To add a new Device, use button  in the toolbar of the tab Device, while in Configuration mode or right-clicking mouse button on the Device tree and select 'Add':

Figure 78 – Selecting the type of the new Device.

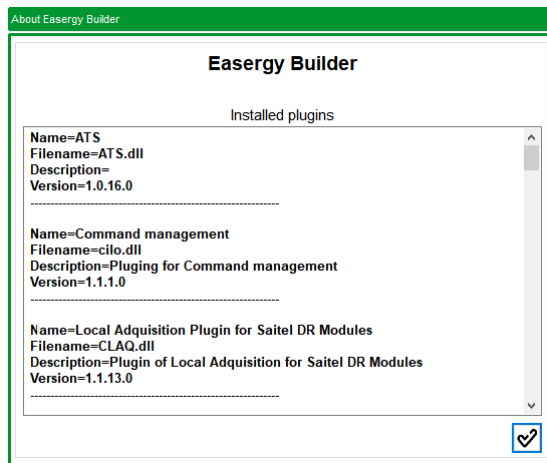


When a Device is available for a type of RTU, but its configuration plugin has not been installed, its name is shown with an (X) before the name. More information about available devices in Table 2.

This software implements the rules to follow and offers a friendly graphical interface for points configuration. Every Device controller into this list can be installed and uninstalled independently of Easergy Builder.

To know which Devices are installed in Easergy Builder and the plugin versions, please, select **Help → About** in the main toolbar.

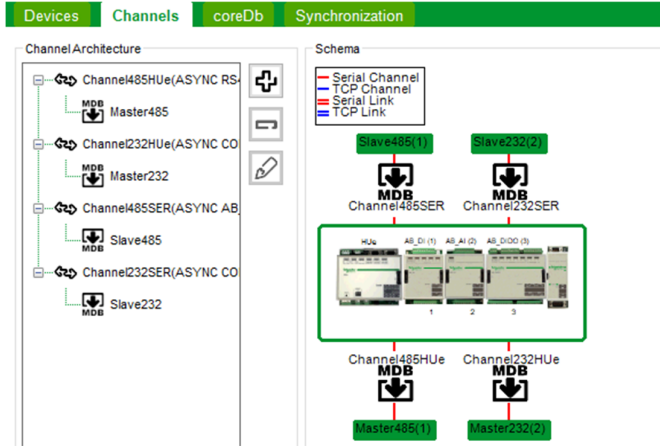
Figure 79 – Devices and plugins installed in Easergy Builder.



2.4.2 Communication Channels

The ports used to communicate with field devices are configured as communication channels. They can be configured in the Channels tab of the configuration mode of Easergy Builder. The number and type of these channels depends on the type of CPU and communication modules installed in the RTU.

Figure 80 – Configuring communication channels.



For example, this figure shows an RTU with a HUE and an AB_SER module. The 'Channel Architecture' zone displays all channels and links (channel associations) which are defined.

- Channel485Hue: RS-485 (HUE)
- Channel232Hue: COM1 (HUE)
- Channel485SER: COM1 (AB_SER)
- Channel232SER: COM2 (AB_SER)

Buttons in the toolbar allow:

- Add a new channel or link.
- Delete a channel or link.
- Modify the configuration of a channel or link

2.4.2.1 Adding a Channel


Use button  to create a new communication channel:

Figure 81 – Channel configuration parameters.

Where:

- **Name:** Identifying name.
- **Description:** Description.

- **Type:** Channel type depending on the implemented communication protocol. There are three types:
 - TCP
 - UDP
 - ASYNC (RS-232, RS-485 or RS-422 serial communication)
- **Encryption:** "SM2 digital certificate" can be selected.
- **Specific Parameters:** Reserved, this box must not be checked.

Other fields depend on the type of channel.

TCP Channel

Figure 82 – TCP channel configuration.

- **Mode:** Type of communication trigger supported by this channel:
 - CALLED: The RTU acts as a 'server', accepts incoming tries of connection.
 - CALLING: The RTU acts as a 'client', tries to connect a remote server.
 - BOTH: Both are applicable.
- **Local Port:** Local TCP port associated to input messages. This field has no effect when the specified mode is 'Calling'.
- **Remote Port:** TCP port (client) associated to output messages. This field has no effect when the specified mode is 'Called'.



NOTICE

Some of the reserved ports are 2404 for IEC-104, 20000 for DNP 3.0 and 502 for Modbus TCP.

- **Remote IP List:** Client-associated IP addresses from which communication requests are accepted through this channel. If the list is empty, it would accept requests from any known client.

On a channel type CALLED, this is the address that will be validated after accepting the connection.

On a channel type CALLING, it will attempt to establish connection to the addresses specified in this list. Only after an unsuccessful try or disconnection, the following remote IP address in the list (if any) is used to the next try.

New clients can be added to the list by entering its IP in the field over the list and pressing the  button. To remove a client from the list, select it and press the  button.

For CALLING and CALLED channels, the maximum Number of IPs is 4.

- **Connect timeout:** For CALLING channels. Timeout (in milliseconds) waiting an answer for a connection.
- **Reconnect time:** For CALLING channels. Minimum time to wait for connection retry.
- **Use local IP:** It is optional. If is checked, IP field specifies the only local IP address to be used in the TCP connection.

NOTICE

To avoid non-desired behavior, the following restrictions have been established:

- For a local logical port, none of the CALLED channels that are associated to him is allowed to have a remote IP as ANY (no IP configured in the list). ANY is allowed only if this channel is the only channel associated to the port.
- It is not allowed to configure CALLING type channels to which no remote IP is associated.
- For channels configured as BOTH, the two previous restrictions are applied.

UDP Channel

If selecting UDP, the following information can be specified:

Figure 83 – UDP channel configuration.

- **Remote IP List:** The list of remote IPs is optional and indicates the addresses from which UDP packets are accepted through the local port. Messages will be always transmitted to the IP and port of the last received UDP packets. Until the reception of the first UDP packet, UDP packets are sent to the first IP address of the list and the Remote Port.

New clients can be added to the list by entering its IP in the field over the list and pressing the button. To remove a client from the list, select it and press the button.

- **Local Port:** Port where to receive UDP packets from. Mandatory and no zero.
- **Remote Port:** Remote UDP port to send UDP packets to, until the reception of the first UDP packet.
- **Use local IP:** It is optional. If selected, IP specifies the only local IP address to be available to receive and send UDP packets.

ASYNC Channel

Figure 84 – ASYNC channel configuration.

- **Channel:** The physical port that will be used for this channel. The list of physical channels available depends on the CPU type:
 - SM_CPU866e: COM1, COM2, COM3, COM4 and SM_SERx-COMy. Only ports of the SM_SER configured will be available.
 - HUE: COM1, COM2, RS-485 and AB_SERx-COMy. Only ports of the AB_SER configured will be available.
 - PowerLogic T300: RS-485, K7 RS SLOT(1) and K7 RS SLOT(2). K7 RS SLOT 1 and 2 will be available or not depending on the configuration of the slots 1 and 2.
- **Baudrate:** To specify the communication speed. The speed ranges between 300 and 256000 bps.
- **Protocol:** It defines the asynchronous protocol which will be used. Depending on the CPU, the available protocols are RS-232, RS-485 and RS-422. RS-485 allows 2 or 4-wire communications.

NOTICE

When a 2-wire cable is used with RS-485 or RS-422, to avoid receiving the echo of the own transmissions, the field 'RTS Control' must be configured as 'Toggle' or 'Auto'.

- **Parity, Stop bit, Data bits:** To configure communication parameters.
- Section **RS-485 / RS422** (only available for PowerLogic T300):
 - Termination resistor. It allows activating the termination resistor of devices 'K7_RS_i' and 'RS-485' when the communication mode is 'RS485'.
 - Polarization. It allows activating the polarization of devices 'K7_RS_i' and 'RS-485' when the communication mode is 'RS-485'.
- **Modem control:** These parameters allow configuring the signals for modem control in the communication ports. Not all values are available for each CPU
 - **DTR control** (Data Terminal Ready): Flow control:
 - **Enable:** DTR at high logical level (1).
 - **Disable:** DTR at low logical level (0).
 - **Toggle** (only available for K7 modem): DTR control depends on the parameter 'DTR - RTS delay'.
 - **RTS control** (Request to Send): To configure the RTS output:
 - **Disable.** Disables the use of the RTS signal.
 - **Enable:** It enables the RTS signal and keeps it active.
 - **Auto:** The RTS signal timing will be managed automatically by the device driver. This value is recommended when RS-485 or RS-422 is used, except for SM_SER module.
 - **Toggle:** It allows defining timing for RTS signal according the time parameters (T_FRAME, T_RTS_W_CTS, T_CTS_W_TX y T_TX_E_RTS). See Figure 76.
 - **CD Control** (Carrier Detect):
 - **ENABLE:** CD at high logical level (1).
 - **DISABLE:** CD at low logical level (0).
 - **DSR Control** (Data set Ready):
 - **ENABLE:** DSR at high logical level (1).
 - **DISABLE:** DSR at low logical level (0).

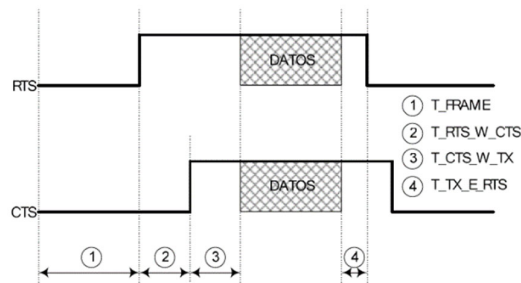
The time setup when DTR is configured as TOGGLE:

- **Delay before transmission (T_FRAME):** Elapsed time from the data transmission is ready and activation of the RTS.
- **DTR – RTS delay:** Timeout before establishing the RTS signal when DTR has been activated.
- **Timeout CTS (T_RTS_W_CTS):** Standby time from the activation of the RTS to the activation of the CTS. If the CTS value is zero, it means that the transmission will be done regardless of CTS value. For non-zero values, channel transmission is controlled by the CTS; if the time defined in this attribute elapses without a CTS activation, then the package pending to be sent is discarded.
- **RTS (or CTS) message delay (T_CTS_W_TX):** Elapsed time from the activation of the CTS to data transmission.
- **Message – RTS delay (T_TX_E_RTS):** time from the end of the data transmission to RTS deactivation.

NOTICE

For 2-wire RS-485 communications, it is very often necessary to set the RTS control to TOGGLE and define all times to 0ms. Thus, reception while transmitting is disabled (to avoid echo).

Figure 85 – Time parameters for modem control.



2.4.2.2 Adding a Link

Some devices support double channel. The functionality will be different for each protocol. A Link is an association of two channels, and it can be used to identify a double channel.


Use button  to create a new communication link:

Figure 86 – New link.

The screenshot shows the 'Channel Setup' dialog box. The 'Link' radio button is selected. The dialog contains the following fields and controls:



- Channel Selection:** Radio buttons for 'Channel' and 'Link' (selected).
- Name:** Text input field.
- Description:** Text input field.
- Channel 1:** Dropdown menu.
- Channel 2:** Dropdown menu.
- Mode:** Dropdown menu (set to 'AutoSwitch').
- Force Switch Time:** Text input field (set to '0 s').
- Buttons:** Checkmark and Close (X) buttons.

Where:

- **Name:** Link name.
- **Description:** Description.
- **Channel 1:** Name of the first channel associated with the link. Must match a correctly configured channel name.

- **Channel 2:** Name of the second channel associated with the link. It must match a properly configured channel name. If it is blank, the link is considered as simple (a single channel associated with it). For modules that support both links and channels, and the situation does not require channel duality, it is recommended to use a link with a single associated channel.
- **Mode:** There are two options:
 - **AutoSwitch:** In the slave, channel module is configured to switch channels automatically. The slave receives and transmits through a channel. At any moment, if it stops receiving through this channel, it switches to the other, which becomes active.
 - **SwitchByMaster:** In the master, this option defines the Device to control switching.
- **Force Switch Time:** A periodic switching between channels can be defined for the item 'Mode → SwitchByMaster'. Master protocols must switch the channel to verify state every TIME_FORCE_SWITCH seconds. When the Link has been set to AUTO_SWITCH mode, this value will be not considered. When SwitchByMaster has been defined, it is recommended to set this parameter to a non-zero value when the module is able to indicate (supervision points) the channel error state.

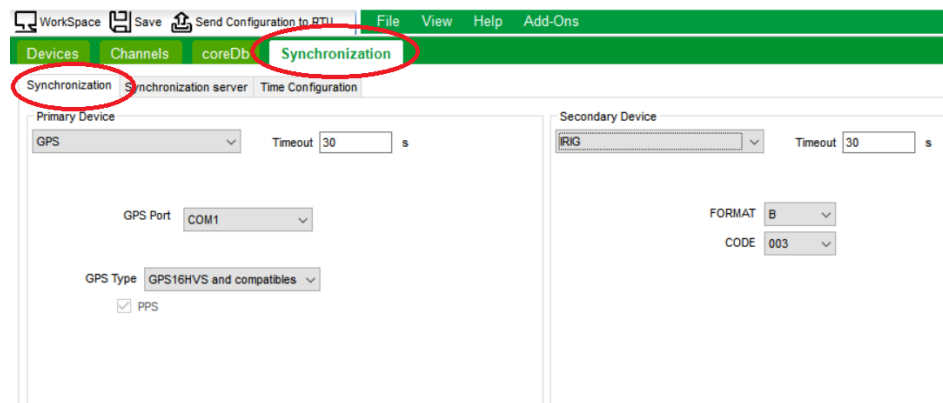
2.4.2.3 Deleting and Editing a Channel or Link

Select the channel or link in the tree and use button  to remove it. Select the channel or link in the tree and use button  to change the data associated to it.

2.4.3 Synchronization

The synchronization can be configured in the tab Synchronization of the configuration mode of Easergy Builder. This functionality offers a wide range of capabilities to synchronize the RTU.

Figure 87 – Synchronization configuration.



Two synchronization devices can be defined: Primary and Secondary.

NOTICE

With Saitel modules, the console tool always can be used as a default synchronization device (with the minimum priority). The following console command could be used:

- **thmShow:** Shows the status of the synchronization devices and the current time.
- **thmConsoleSetTime 'YY:MM:DD:HH:NN:SS':** Manual synchronization of the CPU.

Depending on the CPU type, the synchronization can be performed by:

- **Protocol:** Most telecontrol protocols allow slave devices to synchronize.
- **SNTP:** The synchronization module includes a SNTP client and server, which can be used to synchronize from a network SNTP clock or as a time reference for other modules.

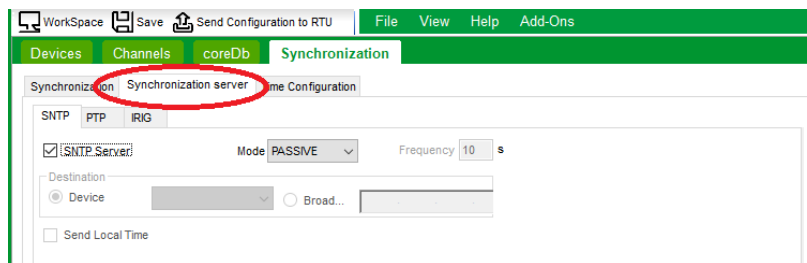
- **IRIG:** (Only available for HUE and SM_CPU866e). IRIG-B-compliant devices are used to synchronize. Compatible formats are: IRIG-B002, IRIG-B003, IRIG-B006 and IRIG-B007.
- **GPS:** Both Saitel DP and Saitel DR allow direct connection to a GPS for time synchronization): The following devices have been validated as GPS: GPS35, GPS16, TSU with protocol NMEA and TKR2 with protocol PTAREE.

For PowerLogic T300, since v1.3 GPS is supported. The only GPS supported is the one integrated in the K7 3G/2G, K7 4G or K7 4G US.

- **PTP** (Only available for SM_CPU866e and HUE). As indicated in the IEEE-1588 standard, a PTP master can synchronize other PTP devices (slaves) through one or several Ethernet interfaces.

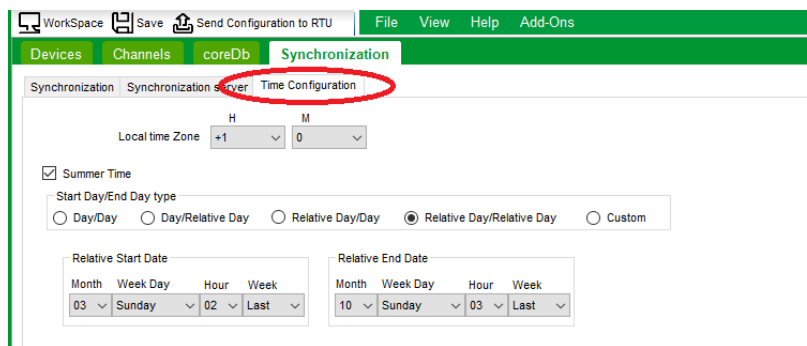
The RTU can be configured as a synchronization server, using SNTP, PTP or IRIG.

Figure 88 – Synchronization server.



The synchronization module allows the time zone and summer/winter (day light saving) calendars to be configured.

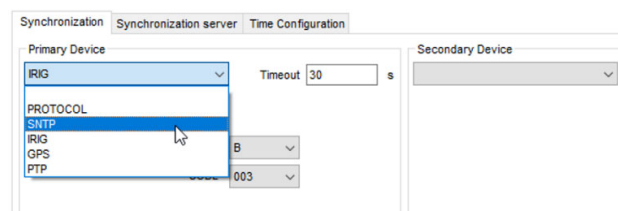
Figure 89 – Time configuration.



2.4.3.1 Configuring a Synchronization Device (as Source)

For Primary and Secondary Device, depending on the synchronization method selected a set of fields should be configured for each device (PROTOCOL, SNTP, IRIG, GPS or PTP).

Figure 90 – Configuring a synchronization device.



- **Timeout:** (Only visible when a type of device is selected) Time in seconds to mark the device as 'off-line' if no synchronization messages are received.

If Primary Device and Secondary Device are considered enabled (Primary Device is the online device synchronization) and Timeout time elapses without receiving synchronization from the

Primary Device, then the Secondary Device becomes the synchronization device. If secondary device also times out, then the active synchronization source will be the console.

PROTOCOL

When a telecontrol protocol is configured as synchronization source, a device source must be selected:

- **Device Sources:** It allows selecting the synchronization sources from the existing devices or from a point into the table analog or status. Several Devices can be selected as source.

SNTP

Please, add a Server (using 'Add Server' button) for each one of them

Figure 91 – Configuring synchronization with a SNTP server.

- Synchronization using SNTP can be in **active mode** (field Passive unselected) or **passive mode** (field Passive selected).
- **SNTP Server IP:** IP address of the SNTP server.
- **Period:** Time (in seconds) for a new interrogation to this server.

The SNTP servers list will be managed as a circular queue.

In **active** mode, of all configured servers, only the active SNTP server is interrogated. If a response is not received from the active server, the following servers in the list will be interrogated (managed as a circular queue) until a response is received from one of them. This new server will be marked as the active for the following SNTP requests

In **passive** mode, the RTU is synchronized by listening to SNTP messages sent by the configured servers. In this mode, the Frequency attribute has no meaning and it will not be necessary to set any value.

IRIG

Synchronization using IRIG-B is only available for HUE and SM_CPU866e.

Figure 92 – Configuring synchronization with IRIG.

The following formats are available (consult the standard IRIG STANDARD 200-04): IRIG-B002, IRIG-B003, IRIG-B006 and IRIG-B007. Information to be defined for this type of device includes:

- **FORMAT:** Selection of a format, supported by IRIGB, in which the time data will be managed. Only the format 'B' is available.
- **CODE:** The value selected in this field together with the previous item will complete the configuration of the IRIGB streaming. Select 002, 003, 006 or 007, which are the formats recognized by the IRIG-B 200-98 standard.

GPS

When a GPS is defined as synchronization source, the following information should be configured:

Figure 93 – Configuring synchronization with a GPS.

- **GPS Port:** Serial port used to connect the GPS. For PowerLogic T300 only K7_SLOT is available.
- **GPS Type:** It allows choosing a GPS from a list. The available devices are depending on the type of CPU. One of the two last options must be selected if synchronizing by a Schneider Electric TSU module; the rest of the options are available if synchronizing by a Schneider Electric MSAC module or directly from the GPS. Depending on the CPU, the following types of GPS are available:
 - GPS 16HVS and compatible devices. This is the adequate option when synchronizing through MSAC.
 - GPS TKR2.
 - TSU-G1 (IRIG-B).
 - TSU-G2 (GPS).
 - K7_GPS_1/ K7_GPS_2 (The only option for PowerLogic T300)
- **PPS:** It indicates if the PPS signal is taken from the GPS. If this field is selected, depending on the CPU, the pin reserved for PPS signal in the port must be wired. More information in the CPU user manual.

PTP (Precise Time Protocol)

The PTP synchronization protocol only is available for SM_CPU866e and HUE.

According to IEEE-1588 terminology, when the RTU is synchronized by an external PTP device, the RTU works in 'Ordinary Clock' mode (OC). If the RTU is a PTP server, then it works in 'Boundary clock' mode (BC).

NOTICE

The RTU can be configured a PTP client or a PTP server, but not both at the same time.

The following information is required:

Figure 94 – Configuring synchronization with a PTP device.

- **Port:** Select the physical port for synchronization with the PTP device. Only available ports for each CPU will be shown.
- **domainNumber:** Value of the 'domain' attribute of the local clock. The default value is 0.
- **Announce messages period:** Average time interval between 'Announce messages'. Value in seconds, as a base two logarithm. The default value is 1, that is, 2 seconds.
- **Max. Announce messages lost:** Maximum number of lost messages before considering a time-out in the reception. The default value is 3.
- **Delay req period:** Minimum allowed time interval between 'Delay_Req' messages. It is specified, in seconds, as a base two logarithm. The default value is 0, that is, 1 second.
- **Sync period:** Average time interval between synchronization messages. A shorter interval may improve the accuracy of the local clock. It is specified, in seconds, as a base two logarithm. The default value is 0, that is, 1 second.
- **Pdelay req period:** Minimum allowed time interval between 'Pdelay_Req' messages. It is specified, in seconds, as a base two logarithm. The default value is 0, that is, 1 second.
- **delay mode:** Delay determination mechanism. The possible values are:
 - E2E. End to end. This is the default value.
 - P2P. Peer-to-Peer.
 - AUTO. It starts as E2E and changes a P2P when a P2P delay request is received.
- **net transport:** network transport mechanism. Possible values are:
 - UDP_IPV4. UDP / IPv4 network transport. This is the default value. PTP messages are transported over UDP and IPv4.
 - IEEE_802_3. IEEE 802.3 network transport. PTP messages are transported directly over Ethernet frames.
- **vlan tagging:** It selects whether the PTP frames should include their VLAN tagging (IEEE 802.1Q). It can only be selected when 'Net transport' is IEEE 802.3.
- **ID:** ID of the vlan. From 0 to 4095. It is only available if vlan_tagging is selected.
- **Priority:** Priority of the vlan. From 0 to 7.

2.4.3.2 Configuring the RTU as a Synchronization Server

The RTU can be configured as a synchronization server and it could be used to synchronize other slave devices. Depending on the CPU, the server could be:

- SNTP
- PTP

- IRIG.

NOTICE

Depending on the type of CPU, tabs PTP and IRIG will be available or not.

SNTP Server

Select tab SNTP:

Figure 95 – Configuring the RTU as SNTP server.

Where:

- **SNTP Server:** Allows configuring the RTU as a SNTP server.
- **Mode:** Can be ACTIVE or PASSIVE. If Passive field is selected, the server only will answer when a request is received from a SNTP client. If ACTIVE mode is selected, the server sends (periodically) a synchronization broadcast message. In ACTIVE mode the server will answer too each request from a client.
- **Device or Broadcast IP:** Only available in ACTIVE mode. Ethernet port or IP address where the server will send the synchronization broadcast message.
- **Frequency:** Only available in ACTIVE mode. Time (in seconds) between broadcast messages.

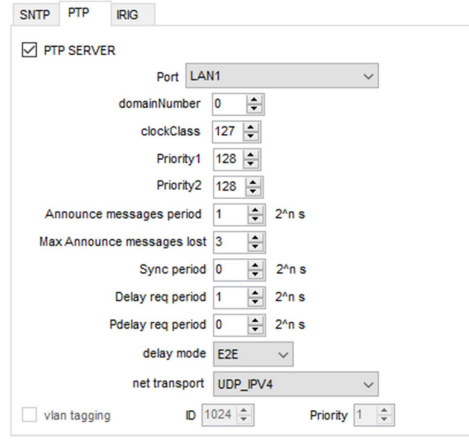
For an PowerLogic T300 CPU, select IPv6 in order to use this addressing protocol.

PTP Server

Only SM_CPU866e and HUE can be configured as a PTP Server.

Select tab PTP and configure the following parameters:

Figure 96 – Configuring the RTU as PTP server.



The information for each field is detailed in the previous paragraph. Only following specific parameters should be set in this window:

- **Clock Class:** Denotes the traceability of the time or frequency distributed by the grandmaster clock. Default value is 127.
- **Priority 1:** A configurable designation that a clock belongs to an ordered set of clocks from which a master is selected, through the 'Best Master Clock algorithm'. Values can range from 0 to 255 (lower values take preference). Default value is 128.
- **Priority 2:** A user configurable designation that provides finer grained ordering among otherwise equivalent clocks. Values can range from 0 to 255 (lower values take preference). Default value is 128.

When a SM_CPU866e is configured as PTP server, according to IEEE-1588, the PTP clock acts as a 'Boundary Clock' (BC).

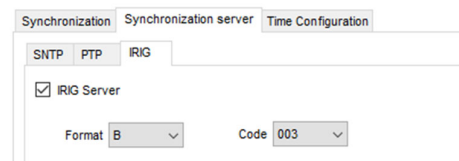
NOTICE

An RTU cannot be OC (Ordinary clock) and BC at the same time. If it is synchronized by a PTP device, it cannot be configured as PTP server.

IRIG Server

In order to the RTU be an IRIG server, please, select IRIG tab and configure the IRIG, check IRIG Server and select the format (only B is available), and the code used (IRIG-B002, IRIG-B003, IRIG-B006 and IRIG-B007).

Figure 97 – Configuring the RTU as IRIG server.



2.4.3.3 Time Configuration

The following information defines the time zone and daylight:

Figure 98 – Configuring time zone and daylight.

- **Local Time Zone:** It allows configuring the time zone. H and M indicates the difference (H: hours and M: minutes) in the geographic zone with the GMT.
- **Summer Time:** It enables the summer time option. This configuration can be defined (depending on each country) using relative or absolute dates. The initial and final date of the summer time should be configured. Example: For example, in Europe, summer time starts the last Sunday of March and finishes the last Sunday of October.

2.5 Configuring a Saitel RTU

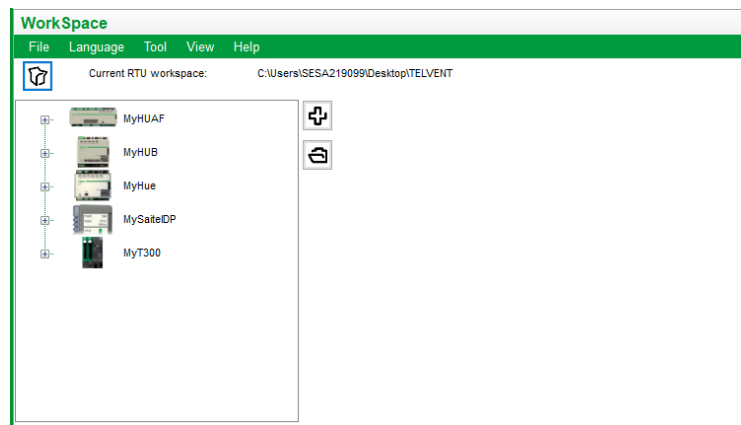
This section covers, step by step, how to configure a Saitel RTU. The configuration process for a Saitel DR RTU is similar to Saitel DP, so only one of them (Saitel DR) is included in the following examples

In this example, an ITB (Intelligent Terminal Block) composed of an HUE as CPU and two slave modules; AB_DI and AB_DO

2.5.1 Creating the RTU into a WorkSpace

Open Easergy Builder and make sure the right Workspace is loaded. This workspace show should show all the RTUs defined in it:

Figure 99 – Initial environment



Create an RTU →  and define the ITB:

Figure 100 – A new Saitel DR RTU using HUE

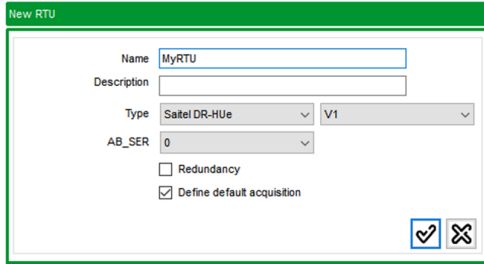
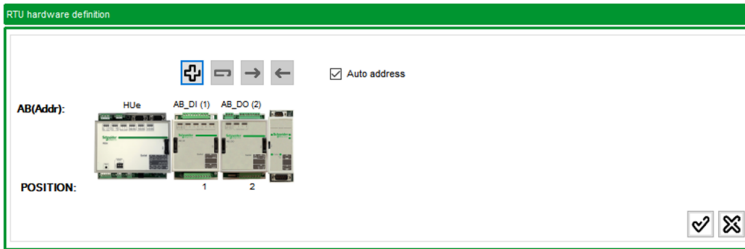


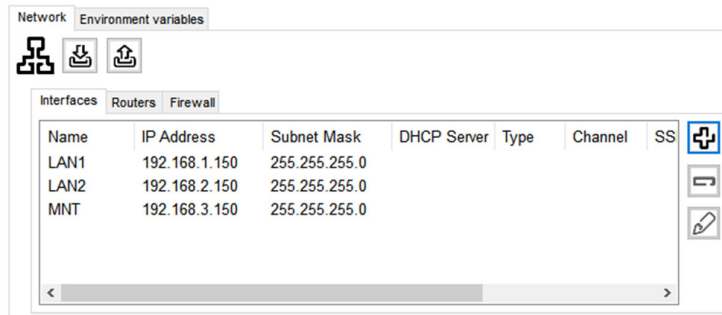
Figure 101 – Including slave modules



2.5.2 RTU Parameters

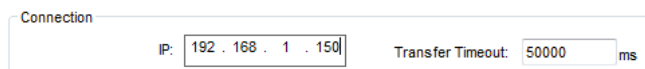
Configure each interface:

Figure 102 – Configuring interfaces



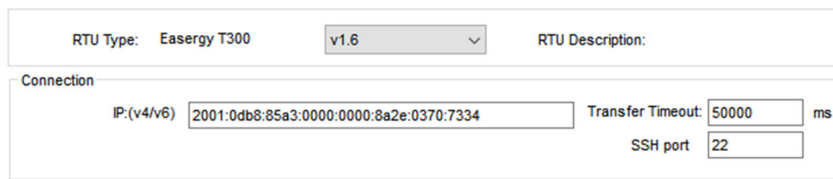
Define the connection of Easergy Builder to the RTU:

Figure 103 – Configuring the connection for Easergy Builder



All ethernet ports use only IPv4 except the WAN port in PowerLogic T300 RTUs. For this RTU type (only for v1.6 profile and later), the WAN port can use IPv4 and IPv6.

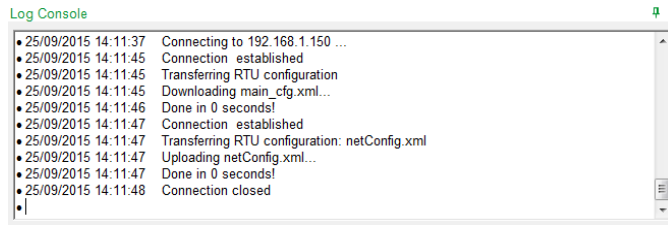
Figure 104 – Using IPv6 for connection with the RTU.



Transfer the general parameters to the RTU (only network parameters) →

Please, be sure that a ping command can be executed and the answer from the RTU is correct.

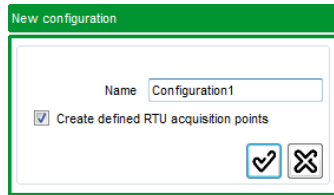
Figure 105 – Showing messages on the Log Console



2.5.3 Creating a Configuration

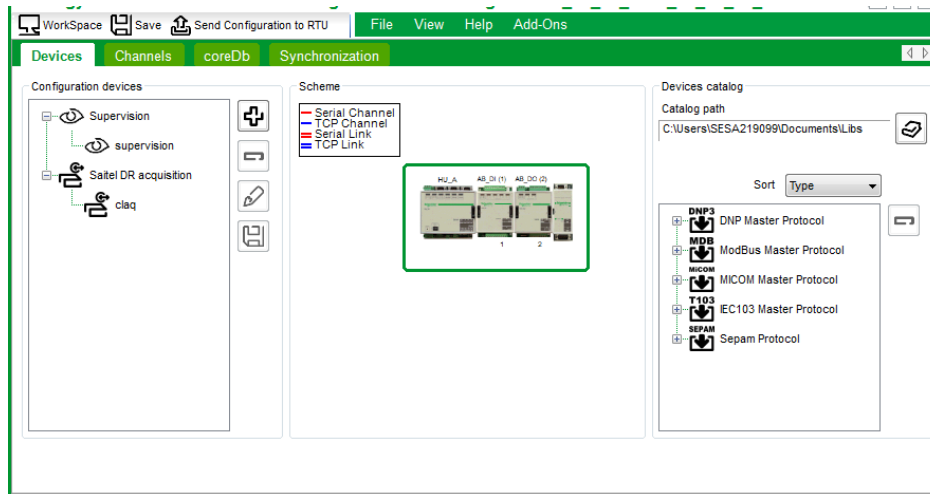
Create a Configuration →

Figure 106 – Creating a configuration



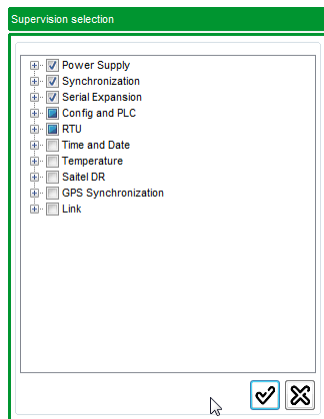
Double click on the configuration:

Figure 107 – Editing the new configuration



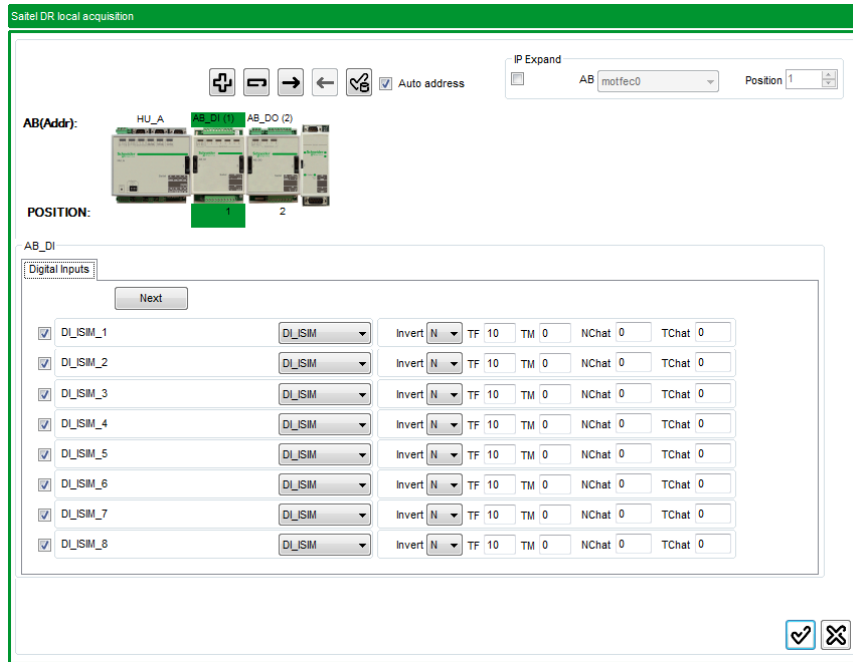
Double click on the supervision device to configure the supervision points:

Figure 108 – Selecting supervision points to be included in coreDb



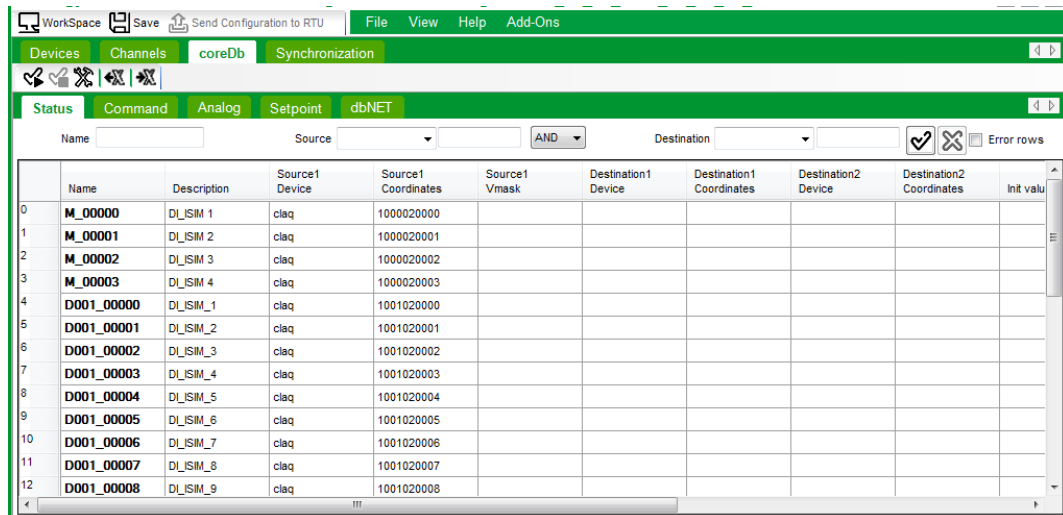
Double click on the claq device to configure local acquisition.

Figure 109 – Configuring local acquisition



Press button to insert in coreDb all points for this ITB. Selecting tab 'coreDb', all these points are shown for each table.

Figure 110 – Points defined in coreDb



Now, Channels, Synchronization and any other devices can be defined according the system architecture.

For more information about the configuration of the local acquisition in Saitel DR consult the manual Saitel DR Platform User Manual and the user manual for each I/O module. For more information about each Device, please, consult its manual.

Transfer the configuration files to the RTU → Send Configuration to RTU

2.6 Working with PowerLogic™ T300

Easergy Builder allows to create an RTU step by step, but this way is not recommended for this version. Import a configuration generated by the 'T300 Generator' tool or from a RTU configuration file (tar.gz) as explained in section 2.2.7 .

Using Easergy Builder, all these parameters or its configurations can be modified:

- Profile. There are two different types for 2.8 firmware version:
 - v02.08, without chan encryption.
 - v02.08(HU250-CN), based on v02.08 but with chan encryption.
 - v02.08.01+, without chan encryption.
 - v02.08.01+(HU250-CN), based on v02.08 but with chan encryption.
 - v02.08.02+, without wireless.
- Network and modem parameters
- Time synchronization parameters
- I/O local control
- Command control
- Formulas

From this initial configuration, new Devices can be added in Easergy Builder:

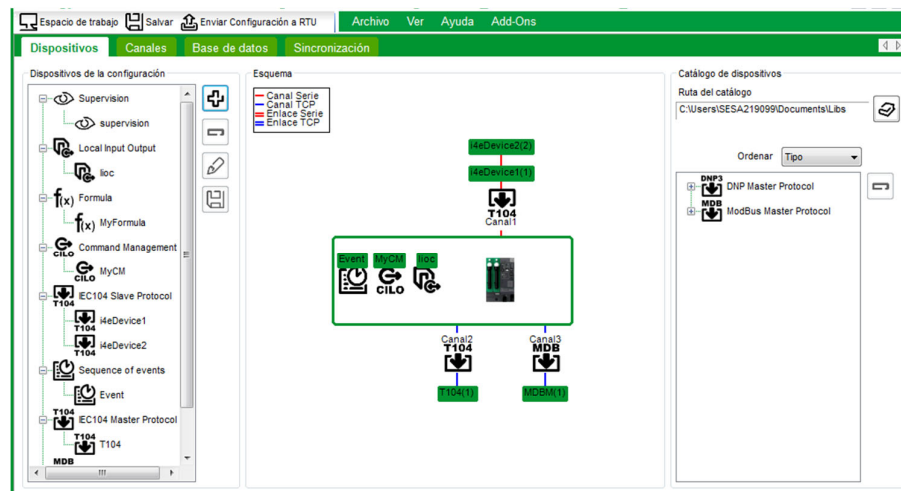
- Communication protocols (master and slave)
- Sequence of Events
- ISaGRAF®


All information about all these Devices in the user manual of each one.

2.6.1 Sending Configuration Files to the RTU

To modify the basic configuration that was imported from 'T300 Generator', select the Configuration in the tree and double click on it. The Easergy Builder Configuration mode is activated:

Figure 111 – Editing and sending a PowerLogic T300 configuration



Make all necessary changes and press button  Send Configuration to RTU .

The configuration can be sent too from the Workspace mode. From here, not only the configuration is sent (databases, channels, synchronization, ...), but also all changes in the RTU parameters are sent (network interfaces, modems, PPP and Port hardening).


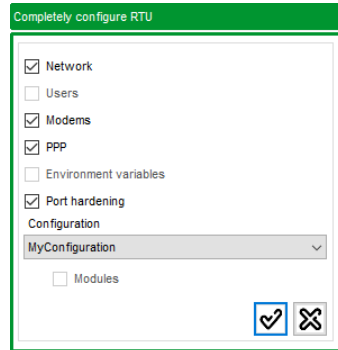
Next to the RTU tree, select the PowerLogic T300 RTU to be configured and press .

Figure 112 – Select data to be sent to the PowerLogic T300.



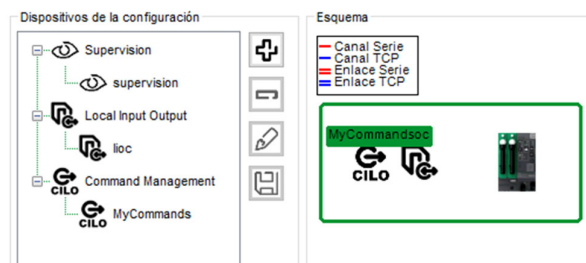
Select all data to be sent, including the selected Configuration if it is necessary.

More information about how to configure PowerLogic T300 in the PowerLogic T300 Devices user manual.

2.6.2 Cilo Configuration (Command Management)

This device is shown in the tree called 'Command Management'.

Figure 113 - Cilo Device in Easergy Builder.



If the device was not created in the tree, click on  and create a new 'Command Management' Device.

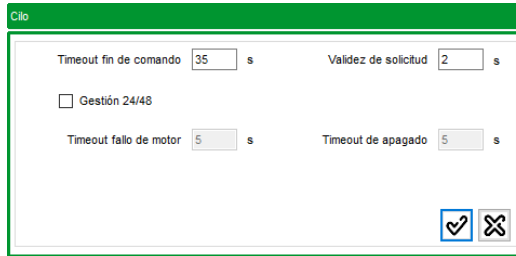
INFORMACIÓN

Only one Cilo device can be created in a configuration.

If the 'Command Management' device type is not available in Easergy Builder, execute the 'Cilo_SPlugin.msi' file to install it.

Double click on the new Device ('MyCommands' in figure above) and the configuration window will appear with the following parameters:

Figure 114 – Commands configuration.



The screenshot shows a configuration window titled 'Ciclo'. It contains four input fields for timeout values in seconds: 'Timeout fin de comando' (35 s), 'Validez de solicitud' (2 s), 'Timeout fallo de motor' (5 s), and 'Timeout de apagado' (5 s). There is an unchecked checkbox labeled 'Gestión 24/48'. At the bottom right, there are two icons: a checkmark and a close button (X).

- **End of command timeout** (in seconds): timeout to complete the command (from 1 to 40 seconds). Default value: 35 s.
- **Request validity** (in seconds): timeout to accept a command execution request (from 1 to 10 s). Default value: 2 s.
- **24/48 management** (in seconds): check this box to allow the PS50 power supply control.
- **Motor failure timeout** (in seconds): timeout to detect the change from 1 to 0 at the moment it detects motor failure (from 1 to 10 s). Default value: 5 s.
- **Power off timeout** (in seconds): timeout to send a power off command to the PS50 (from 1 to 10 s). Default value: 5 s.

More information about how to configure PowerLogic T300 devices, go to 'PowerLogic T300 Devices user manual'.

3 coreDb – Real-Time DataBase

Content

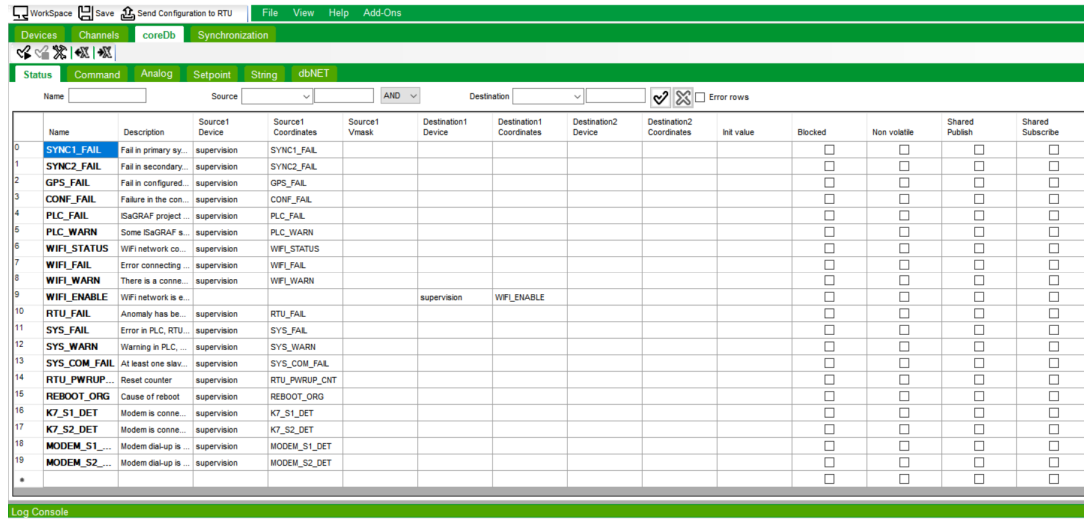
3	COREDB – REAL-TIME DATABASE	77
3.1	INTRODUCTION	79
3.2	BASIC OPERATIONS WITH REGISTERS	80
3.2.1	AUTOMATIC CREATION OF COREDB POINTS (LOCAL ACQUISITION).....	80
3.2.2	POINT MANAGEMENT.....	80
3.2.3	SEARCH.....	83
3.2.4	COREDB FIELD DESCRIPTION.....	83
3.2.5	SOURCE AND DESTINATION ALLOCATIONS.....	85
3.3	COREDB TOOLBAR	85
3.3.1	CHECKING INFORMATION IN COREDB	86
3.3.2	LOADING INFORMATION IN MEMORY	86
3.3.3	IMPORT / EXPORT COREDB INFORMATION	86
3.4	REAL-TIME INFORMATION OF THE POINT.....	87
3.4.1	QUALITY FLAGS.....	87
3.4.2	QUALITY FLAGS IN STATUS	88
3.4.3	QUALITY FLAGS IN ANALOG	89
3.5	SHARING COREDB INFORMATION WITH OTHERS RTU.....	90
3.6	CONFIGURING REDUNDANT RTUS.....	91
3.6.1	CONTROL.....	91
3.6.2	MODE	92
3.6.3	BUS	92
3.6.4	ADDITIONAL IPS	92

3.1 Introduction

This chapter describes in detail the real-time database of the BaseLine platform, as well as the administration tasks which can be performed in the database using Easergy Builder.

In the Configuration mode, select tab coreDb:

Figure 115 – Configuring coreDb in Easergy Builder.



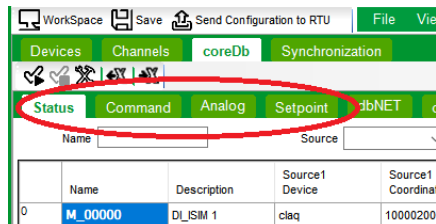
coreDb is the real-time database of the BaseLine Software Platform for RTUs. This database stores all the information associated to the points and the relationships with the I/O information managed by defined Devices. These relationships are implemented through the source and destination coordinates for each point.

The format of these coordinates depends on the associated source or destination Device.

All this information is arranged in different tables. Each table stores a different type of point (Status, Analog, Setpoint, String and Command tables)

Selecting a tab, the data points defined for each coreDb table are shown:

Figure 116 – coreDb main menu.



NOTICE

Avoid the use of the letter 'ñ' and vowels with accents in the field Name. This applies throughout the document.

3.2 Basic Operations with Registers

3.2.1 Automatic Creation of coreDb Points (Local Acquisition)

When a RTU or configuration is created, if field 'Create defined RTU acquisition points' is selected, default supervision points and I/O points will be generated in coreDb according to the I/O modules that were included.

NOTICE
This feature is only available for Saitel RTUs.


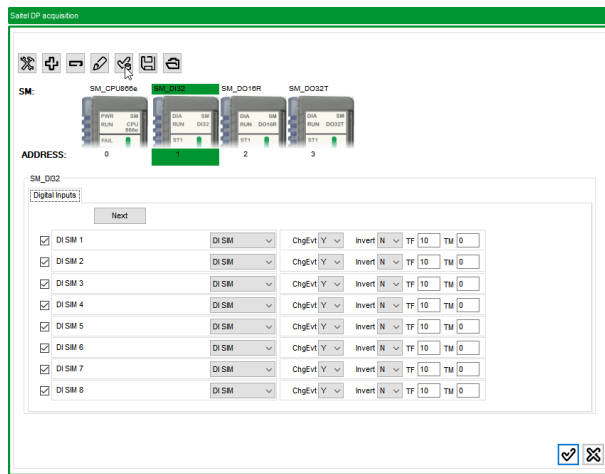
On the other hand, if a new module must be included in a configuration, use button  to generate automatically in coreDb its default supervision points and acquisition data points. The following picture shows an example for Saitel DP.

Figure 117 – Configuring local acquisition.

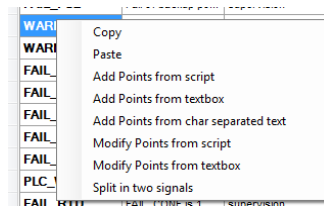


3.2.2 Point Management

There are several contextual menus available in the tool depending on where and how the user press right button of the mouse.

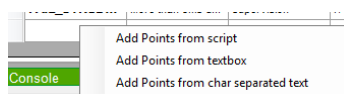
Right clicking on the field **Name** of a point, the following contextual menu is available:

Figure 118 – Contextual menu for a selected point.



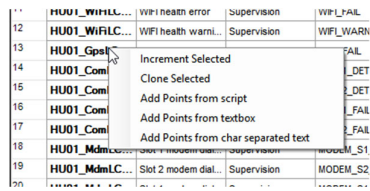
If the user right-clicks on the Name field of a blank row, the menu will be:

Figure 119 – Contextual menu for a new point.



If the user right-clicks on the Name field of a non-selected cell, the menu will be:

Figure 120 – Contextual menu for a new point.



Copy

The point selected is copied to the Microsoft Windows® clipboard.

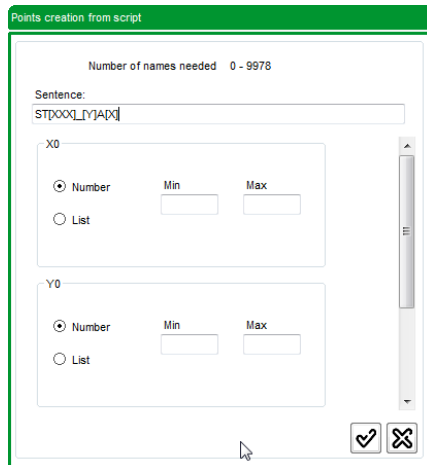
Paste

The content is copied from the Microsoft Windows® clipboard.

Add point from script

It allows adding a series of names with a common part and a variable one. Variables: [X], [Y] and [Z] in this priority order. Example: by entering ST[XXX][Y]A[X], first 'X' (X0 on Figure 110) is numeric from 1 to 2, 'Y' (Y0 on the Figure 110) is list type with values 'a, b and c', and second 'X' (X1 on the Figure 110) is numeric from 3 to 4. This example generates the following points: ST001_aA3, ST001_bA3, ST001_cA3, ST002_aA4, ST002_bA4 and ST002_cA4.

Figure 121 – Add point from script.

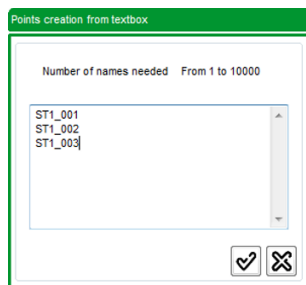


Use the scroll bar to access all the information of the variables. This area of definition of variables changes according to the script (field Sentence).

Add point from textbox

It allows entering the names in a text box.

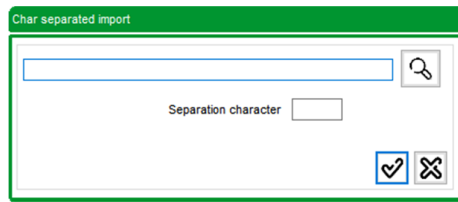
Figure 122 – Add point from text box.



Add point from char separated text

The file with the name to add is specified, separated by a character, which is entered in the **'Separation Character'** field.

Figure 123 – Add point from separated text.



Modify Points from script

All selected points will be modified using a series of names with a common part and a variable one. The number of the generated points by the script must be the same that selected points.

Modify Points from textbox

All selected points will be modified using a list of points indicated in a text box.

Split in two points

Duplicates the selected point. This option is used to create two simple points from one double. For example, this is useful when transferring points to ISaGRAF®.

Figure 124 – Transferring a double point in two single points to ISaGRAF®.

Name	Description	Source1 Device	Source1 Coordinates	Source1 Vmask	Destination1 Device	Destination1 Coordinates	Destination2 Device	Destination2 Coordinates	D
D001_00000	PROFL_STS 1	laq	2001020000		status	D001_00000_1	status	D001_00000_2	D
D001_00000_1		status	D001_00000	0x01	isa	D001_00000_1:B			
D001_00000_2		status	D001_00000	0x02	isa	D001_00000_2:B			

Clone selected

The content in this cell is copied on the selected cell.

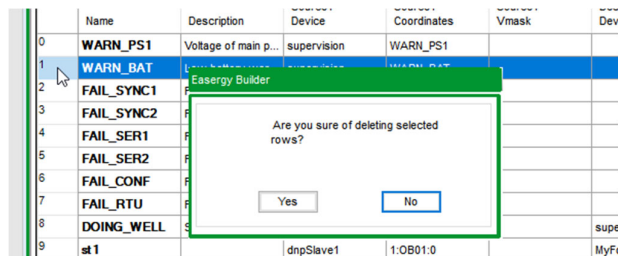
Incremented selected

The content in this cell is incremented and copied on the selected cell.

Remove a record

To remove a record in a table, select the record Number at right of the record to remove. When it is show in blue color, press 'Delete' key and select 'Yes' in the confirmation window:

Figure 125 – Delete a record in a table.

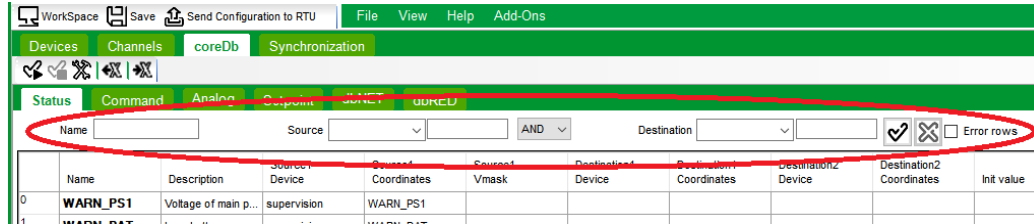


Note that the selected button by default in this confirmation window is 'No', pressing 'Enter' key, the record is not deleted.

3.2.3 Search

Each tab corresponding to coreDb has a set of buttons with a common functionality:

Figure 126 – Search bar of a coreDb table.



The information to be completed in this bar is:

- **Name:** Name of the point to search, if it is known and only one. If it is completed with a point name which does not exist, nothing is returned.
- **Source:** The drop-down box is used to select points which have a specific device as their source. The next field is to search a point by its source coordinate.
- **Destination:** The drop-down menu is used to select points that have a specific device as their destination. The next field is to search a point by its destination coordinate.
- **'AND' 'OR':** In the case of fill the Source and Destination fields, this drop-down menu is used to combine the other two filters to refine the search on the table shown.
- **Error rows:** By checking this box only points that are badly configured will be shown.

Press button to apply the defined search filter or to remove the filter and all records will be shown again.

3.2.4 coreDb Field Description

The following table shows the information associated to each point depending on the coreDb table. Each letter represents a table in coreDb, where **S:** Status, **A:** Analog, **C:** Command, **T:** String and **P:** Setpoints. A mark (✓) means the column is into the table.

Table 4 – coreDb field description.

Field	coreDb Table					Description
	S	A	C	T	P	
Init Value	✓	✓		✓		Initial value of the point.
Blocked	✓	✓				If it is checked, the value is allocated manually. All its sources are ignored.
Non Volatile	✓	✓	✓		✓	If it is checked, the point value is stored in NVRAM (non-volatile RAM) to avoid an information loss during a power-off and an RTU reboot. This box is ignored in the initial database load and the point will obtain the 'INIT VALUE'. (For PowerLogic T300, this feature is only available for v1.5 and later).
Shared Publish	✓	✓				The value of the point can be accessible for other RTUs connected to the control system. If this column is checked, the point is stored in the RTU database and will be accessible by the rest of the system. If this case, information in tab dbNET must be complain.

Field	coreDb Table					Description
	S	A	C		P	
Shared Subscribe	✓	✓				If it is checked, SOURCE columns in this point must be empty. The point is stored in an external database (in other RTU). In this external database, this point must be defined as publish.
Engineering units Convert		✓				It allows a unit conversion for the information source. If it is checked, the following four fields are available to define the conversion straight.
Engineering units Min raw		✓				Minimum value of the straight entrance (RAW). Only available if 'Engineering units Convert' is checked.
Engineering units Max raw		✓				Maximum value of the straight input (RAW). Only available if 'Engineering units Convert' is checked.
Engineering units Min Egu		✓				Minimum value of the straight output (EGU). Only available if 'Engineering units Convert' is checked and EGU value is Y. The range is -32768 to 32767.
Engineering units Max Egu		✓				Maximum value of the straight output (EGU). Only available if 'Engineering units Convert' is checked. and EGU value is Y. The range is -32768 to 32767.
Alarm Use		✓				It allows define up to four types of alarms to be set which are triggered when they cross a defined threshold value. These limits are defined with the following four fields (Lowest limit, Low limit, High limit, and Highest limit).
Alarm Lowest limit		✓				Lowest limit associated to this point. If it is crossed, an alarm is triggered, an event is generated and sent to all destinations and QF (quality bit) = 0x00000800 . This field is only available if 'Alarm Use' is checked.
Alarm Low limit		✓				Low limit associated to this point. If it is crossed, an alarm is triggered, an event is generated and sent to all destinations and QF (quality bit)= 0x00000400 . This field is only available if 'Alarm Use' is checked.
Alarm High limit		✓				High limit associated to this point. If it is crossed, an alarm is triggered, an event is generated and sent to all destinations and QF (quality bit) = 0x00000200 . This field is only available if 'Alarm Use' is checked.
Alarm Highest limit		✓				Highest limit associated to this point. If it is crossed, an alarm is triggered, an event is generated and sent to all destinations and QF (quality bit) = 0x00000100 . This field is only available if 'Alarm Use' is checked.
Destination Threshold		✓				For Analog points, it is possible to configure a change threshold for each destination. This threshold allows all destinations to update their value by events instead of by polling

NOTICE

- Engineering unit limits are only available if '**Engineering units Convert**' is selected.
- Alarm limits are only available if '**Alarm Use**' is selected.

3.2.5 Source and Destination Allocations

Each coreDb point must be associated with source(s) or destination(s).

Source(s) and destination (s) for a point are allocated by selecting coordinates from one or multiple Devices (which must be previously defined in Easergy Builder).

It should be noted that two points cannot have the same source or destination. However, it is possible to use a point declared as source or destination in coreDb but considering the followings factors.

NOTICE

By default, there are two Devices defined in order to use coreDb points as sources. They are called '**status**' and '**analog**' and are included in all Configurations by default.

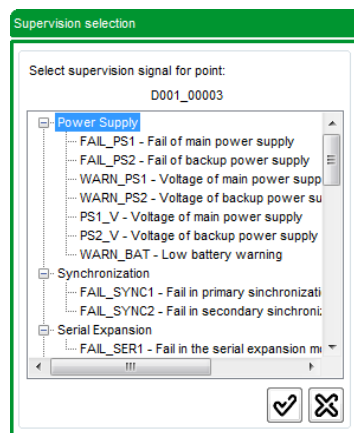
To use a coreDb point as source, this point must have the point(s) which will use the coreDb point as destination; the status or analog Device and the point defined as source will be the source of these destination points.

Following section illustrates this consideration.

Information points can be selected graphically. Right-click on the Source or Destination Device and select 'Select Device' or select the Device and right-click on the coordinate field and select '**Launch Point wizard**'. The displayed window will depend on the Device selected.

For example, the following window will appear for the supervision Device in a Saitel DP RTU.

Figure 127 – Selecting a supervision point as source.



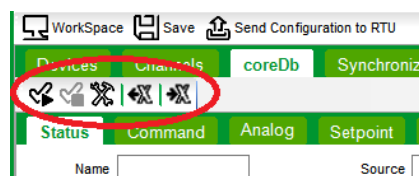
This window shows all points for the selected Device, which are not used yet. In previous windows, the point FAIL_RTU is not shown because it was used as source of other point.

For more information about how to assign Device points as destination and/or source of a coreDb points, please refer to the appropriate Device controller manual.

3.3 coreDb Toolbar

On the coreDb management area, the following tools are available:


Figure 128 – coreDb toolbar.



From the left to the right, these buttons allow:

- Check information in coreDb.
- Configure if the field '**Description**' is loaded or not on memory for each table.
- Import / Export coreDb information using Excel files.

3.3.1 Checking Information in coreDb

Press button  to check the information in coreDb before sending it to the CPU. Press button  to stop the current check.

3.3.2 Loading Information in Memory


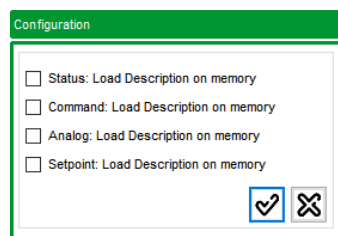
Button  allows configuring for each table if the field Description is loaded on memory or not.

Figure 129 – Use of the field **Description**.



3.3.3 Import / Export coreDb Information


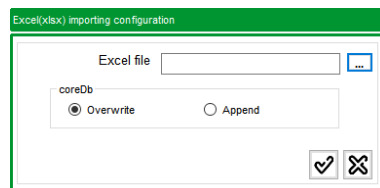
Button  allows importing databases from an Excel® file to the coreDb of the configuration. Press this button and select the file with the information to be imported.

Figure 130 – Using Excel® for data importing.



Select 'Overwrite' or 'Append' depending on the information in coreDb will be overwritten or expanded. If the chosen Excel file does not exist, the importing process is cancelled, and an error message appears.

NOTICE

To perform a successful data import, no empty rows can be found in the top or middle rows of a table.

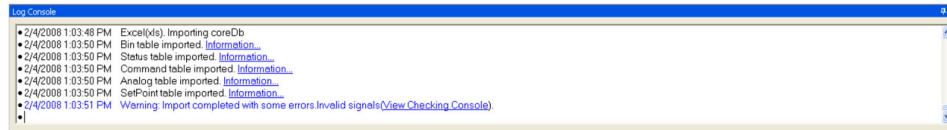
During the importing process, the tool performs pre-validations to incorporate the points to coreDb. The tool verifies that the point attributes comply with database rules:

- Field length
- Allowed characters.
- Valid value range.
- Valid allocation for source and destination coordinates.

In the Log console information messages about the process are shown.


An import is successful if all points in the Excel file have been seamlessly inserted in coreDb. Should there be an error due to the validation criteria, this point will not be inserted in the database and the console will display an error message. For example:

Figure 131 – Data importing process with error reporting.



Some cases where this happens include:

- If the source or destination is correct but the Device has not been defined in the configuration, the point is imported, leaving the field Device empty and completing the Coordinate field. The table record will display an error message.
- If the source or the destination is complete, but the coordinate is syntactically incorrect, the point will not be imported, and a message will appear in the console.
- If the source or destination coordinate is empty, the point will not be imported, and a message will appear in the console.
- If the source or destination Device is empty, the point will not be imported, and a message will appear in the console.

Button  allows exporting the information stored in coreDb to an Excel file. This file can be used to import data in other configurations, as explained in previous section.

3.4 Real-Time Information of the Point

The information about the coreDb point value in a time is associated to a Quality value according some circumstances. Following paragraph explain how the user could understand and use this information.

3.4.1 Quality Flags

Every information point in coreDb is associated to a set of bits which provide information about the point. These quality flags have been generated to avoid the loss of information, and we can find two types. This quality flags can be consult using the WEB tool (webTool or webApp depending on the CPU).

There are two types of quality information associated to a point:

- **Remote quality flags:** They reflect the value of the points sent by the information source which represents the point.
- **Local quality flags:** They reflect the value of the point considering only its treatment in the device.

The format of this quality flags or mask is: **0x[hexadecimal number]**. The meaning of these flags is explained in the following table:

Table 5 – Quality flags for coreDb points.

Local Quality Flags	
Value (hexadecimal)	Description
0x00000001	An overflow occurred.
0x00000002	An overflow or roll – over occurred in a counter.
0x00000004	The counter has been adjusted.

Value (hexadecimal)	Description
0x00000008	Excessive changes in a digital input.
0x00000010	Blocked point.
0x00000020	Manual point replacement.
0x00000040	The point has not been written to the database yet.
0x00000080	Invalid point.
0x00000100	The point value has exceeded Hi-Hi Alarm.
0x00000200	The point value has exceeded Hi Alarm.
0x00000400	The point value is lower than Lo Alarm.
0x00000800	The point value is lower than Lo-Lo Alarm.
0x00001000	Invalid time.
Quality Flags from the Device	
Value (hexadecimal)	Description
0x00010000	An overflow occurred.
0x00020000	An overflow or roll – over occurred in a counter.
0x00040000	The counter has been adjusted.
0x00080000	Excessive changes in a digital input.
0x00100000	Blocked point.
0x00200000	Manual point replacement.
0x00400000	The point has not been written to the database yet.
0x00800000	Invalid point.
0x10800000	Invalid time.

3.4.2 Quality Flags in Status

Apart from the basic settings shown in chapter 5.5, coreDb status data points support additional settings. To configure these, Easergy Builder has to be working in configuration mode and the 'Status' tab has to be selected inside the coreDb view.

NOTICE
Even though several masks can be defined to each source associated to the point, only the last association will be applied.

The following examples illustrate this functionality.

Example 1 - Using masks to create two simple points from one double

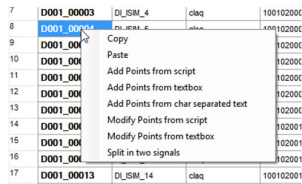
This is useful when transferring points to ISaGRAF®. The following example shows how to change from a double point from the local acquisition (Saitel DR local acquisition Device) to two simple ISaGRAF® points, using the mask.

Figure 132 – Mask to create two simple points from one double

	Status	Command	Analog	Setpoint	dbNET				
	Name	Source	AND	Destination					
0	Ddouble	claq	1003020000	status	DdobA	status	DdobB		
1	DdobA	status	Ddouble	0x01	Isagraf	DdobA:B			
2	DdobB	status	Ddouble	0x02	Isagraf	DdobB:B			

To do that, use 'Split in two signals option' on the contextual menu of a selected point:

Figure 133 – Split in two signals option



Example 2 – Using masks to create two simple Modbus points from one double IEC104 point

The mask is also applied to coreDb points to map the quality bits corresponding to another point in a database point (by applying a mask).

Figure 134 – Mask to create two simple Modbus points from one double IEC104 point

Status Command Analog Setpoint dbNET								
Name	Source	Source1 Device	Source1 Coordinates	Source1 Vmask	Destination1 Device	Destination1 Coordinates	Destination2 Device	Destination2 Coordinates
0	Ddouble104	IEC104master	15000:MDP		status	DdobMDBA	status	DdobMDBB
1	DdobMDBA	status	Ddouble104	0x01	MDBe	IS:1		
2	DdobMDBB	status	Ddouble104	0x02	MDBe	IS:2		

Example 3 – Using the mask to map quality bits

The suffix ':Q' in the mask indicates that it is applied to quality bits. The value is written in hexadecimal format. Since the points are composed of 32 bits, the mask has eight digits. In the mask, the value of the mapped bits is 1. If fewer digits are written, they will be the least significant and their value will be 0 ('0x00000001:Q' equals '0x0001:Q').

If all bits for the value are 0, then all are mapped, that is, '0x00000000:Q' equals '0xFFFFFFFF:Q'. If the mask is '0x00F0:Q', bits 4 to 7 are taken, as if there are no characters on the right.

Figure 135 – Quality bits mask

Status Command Analog Setpoint dbNET						
Name	Source	Source1 Device	Source1 Coordinates	Source1 Vmask	Destination1 Device	Destination1 Coordinates
0	Dinfo	IEC101master	01000:MSP		status	DinfoQ
1	DinfoQ	status	Dinfo	0x00000001:Q	Isagraf	DinfoQ.B

3.4.3 Quality Flags in Analog

For the information source (Source) it is also possible to define a mask. The mask is also applied to map the quality bits corresponding to another point in a database point (by applying a mask). The explanation for the points in Status table also applies here.

NOTICE	
To map the quality bits of an Analog point it is necessary to set up as destination of this point another of the same table. Though only a bit is mapped, it must be stored in a point of the Analog table.	

Example 1 – Using the Field Destination Threshold.

A data point corresponds with a temperature measurement. The destination Device is a slave protocol. The master protocol allows data to be sent by event and performs an integrity update on an hourly basis. The change threshold is adjusted to 10 (degrees, assuming that the engineering units are degrees). coreDb will generate events when the temperature is 0, 10, 20, 30, 40... degrees.

3.5 Sharing coreDb Information with others RTU

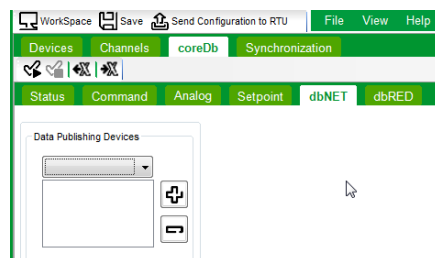
The tab dbNET in coreDb section allows multiple Saitel RTUs to share their databases within an IP network. dbNET operates on a broadcasting basis. This proprietary protocol has been optimized to let the RTUs within the same network share a limited number of database points.

The requirements to share data include:

- The RTU that **shares** data needs to define a coreDb point with a name and the '**SHARED PUBLISH**' checkbox needs to be enabled.
- The RTU that **acquires** the shared data point needs to define a coreDb point with the same name as the coreDb of the RTU that publishes the information. The '**SHARED SUBSCRIBE**' checkbox needs to be enabled. RTUs that 'subscribe' to a published coreDb point do not need to define a source for these points in their databases - the value will be acquired by the publishing RTU.
- Data points shared in both coreDb must have the **same name**.

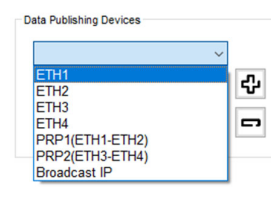
The configuration window (tab dbNET) for this functionality is shown below:

Figure 136 – Sharing data with others RTU.



The number of available ports to use as a channel to share data depends on the CPU that is being configured. A maximum of 5 devices can be configured to share information. For example:

Figure 137 – Possible 'Data Publishing Devices' for SM_CPU866e.



A Data Publishing Device shall be defined for each element publishing coreDb points. A 'Data Publishing Device' can be a network interface or the broadcast address of another network (Broadcast IP).

Considerations to be taken in account are:

- Only Status and Analog points can be shared.
- For each table Analog and Status, a maximum of 32 points can be published.
- For each table Analog and Status, a maximum of 128 points can be subscribed.
- When the network interface has got assigned several IP addresses, publishing is done only for the first IP in the list. For example, consider this situation:
 - The publisher RTU uses ETH1 which has got assigned the following IPs: 1.1.1.1:fffff00 (first) and 2.2.2.2:fffff00 (second). The subscriber RTU receives the information through ETH1 with IP 2.2.2.3:fffff00. This configuration **does not work**.
 - With the same configuration for the publisher RTU, if subscriber RTU use the address 1.1.1.3:fffff00, this configuration **is correct**.

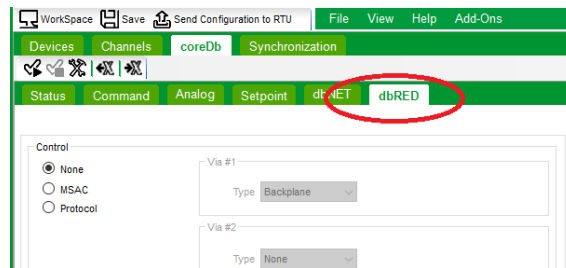
3.6 Configuring Redundant RTUs

NOTICE

Only available for SM_CPU866e and HUE.

By selecting tab **dbRED** in the coreDb configuration window, the user can configure the redundancy.

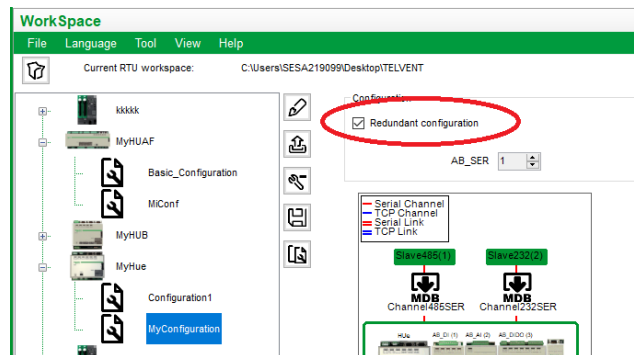
Figure 138 – Configuring a redundant RTU.



NOTICE

This tab is only available if this RTU was configured as redundant.

Figure 139 – Configuring a RTU with redundancy.



3.6.1 Control

The redundancy can be controlled using two different mechanisms:

- **MSAC** module. (Check field MSAC in Control zone). This is a legacy hardware that manages the switchover of CPUs as well as other supervisory features.
- **RCAP protocol** (Redundancy Control Asynchronous Protocol). In this case, there is a redundant switching channel between the CPUs, which is used to manage the switching operation using a Schneider Electric-proprietary protocol.

'Via #1' and 'Via #2' will be available when 'Protocol' is selected:

- **BACKPLANE**. Only available for Saitel DP and only when both CPUs are installed on the same backplane. Communications will be established by UDP over SLIP, over the backplane serial port.
- **NET** (by Ethernet). It is necessary configure the specific IP addresses used to exchange RCAP information between CPUs A and B.
- **SERIAL**. Two dedicated serial ports ('/tyCo/...') in each CPU will be used to carry the RCAP information.

3.6.2 Mode

Two different redundancy modes are available:

- **Cold:** There is no coreDb synchronization between the two CPUs. When a switchover occurs, the new ONLINE CPU will start using its own coreDb with default values.
- There is a high-speed communication channel (Ethernet or backplane) between the two CPUs, which is used to update the BACKUP CPU database with the ONLINE CPU database. When a switchover occurs, the new ONLINE CPU starts with updated values.

NOTICE

In Hot mode, database IDs must be identical, i.e., it is very important to use the SAME Easergy Builder Configuration in both CPUs.

In Hot mode, 'Via' allows to select if the replication is made by:

- **BACKPLANE:** Only available for Saitel DP.
- **NET** (by Ethernet): it is necessary to set the IP addresses used to synchronize coreDb between the CPUs A and B.

coreDb updates are synchronized by exception (only the variables that have changed), except the first time where the complete database is updated. The supervision point **DB_UPDATE** monitors the process.

3.6.3 Bus

Only available for Saitel DP.

The Bus field indicates if the CPUs share the same bus (I/O modules and Serial modules), regardless of whether they are in the same backplane or use RS-485 expansion. This is useful to detect unexpected behavior in dual redundant systems.

- **SHARED:** In this case, the bus of the STANDBY CPU is disabled.
- **DIFFERENT:** If checked, the bus is enabled even if the CPU is in STANDBY mode, so it can receive diagnostics from the modules.
- **NO_ACCESS:** No access to the bus regardless the CPU state (FAIL, STANDBY or ONLINE).

3.6.4 Additional IPs

This is a list of IP addresses associated to the CPU that is in ONLINE. These addresses are associated in a dynamic way, so that in a redundant system they allow to communicate always with the CPU that is active. Regarding virtual addresses, it is even possible to assign multiple IP addresses to each port.

NOTICE

If a static IP address and a virtual address are defined for the same device in the same subnet, a warning console message will be displayed to notify this abnormal situation (sup_redAddIPs: dev xxx ip x.x.x.x subnetMask xxxxxxxx).

This message is a warning from the operating system; nevertheless, it will not cause a system malfunction, since the configuration will operate properly.

4 Supervision Device

Content

4	SUPERVISION DEVICE	93
4.1	GENERAL DESCRIPTION	95
4.2	SUPERVISION POINTS FOR SAITEL DP	96
4.3	SUPERVISION POINTS FOR SAITEL DR.....	98
4.4	SUPERVISION POINTS FOR EASERGY T300	100

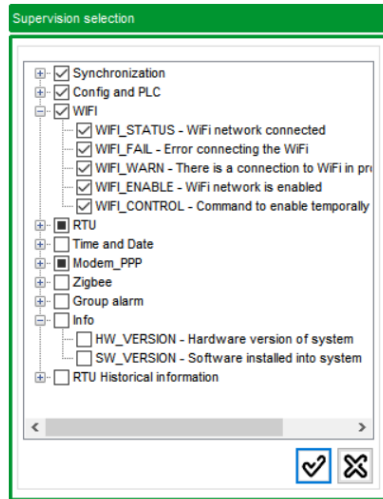
4.1 General Description

The Supervision Device is installed by default with Easergy Builder. This device is used to monitor the states of all CPU components, and to generate information which can be used by other RTU components.

The supervision device depends on the type of RTU. Each supervision point will be available or not depending on the CPU. The user only needs to select the correct points on the supervision window in order to know the information associated to this point.

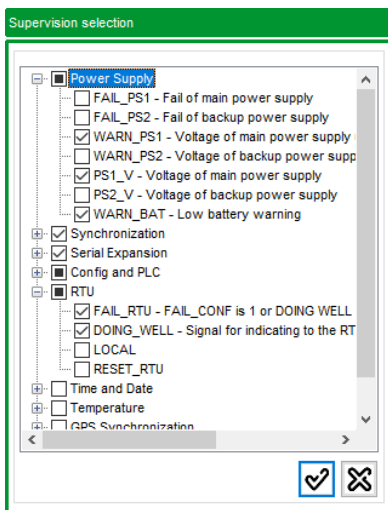
To access the supervision points, double click on the supervision Device in the tree:

Figure 140 – Supervision points for an PowerLogic T300 RTU.



In this window, the supervision points can be select or unselect depending on the RTU type.

Figure 141 – Supervision points for a SM_CPU866e CPU.



The supervision Device has two main functions:

- Monitor the information of supervision points
- Monitor the information to manage redundant systems.

The information generated by the supervision Device is supplemented with the control and diagnostic information generated in each Device.

NOTICE

For supervision device, coordinates match the name. For example, the coordinate associated to the FAIL_PS1 point is 'FAIL_PS1'.

More information about Device' coordinates in chapter 'coreDb' in this manual.

4.2 Supervision Points for Saitel DP

The following supervision point are available for SM_CPU866 and SM_CPU866e.

Table 6 – Supervision point for a Saitel DP RTU.

Point	Table	Type	Description
Power Supply			
FAIL_PS1	Status	Source	Active (1) indicates unavailability of the main power supply (SLOT 1 in the backplane).
FAIL_PS2	Status	Source	Active (1) indicates unavailability of the secondary power supply (SLOT 2 in the backplane).
WARN_PS1	Status	Source	Line PS1 voltage is below the warning level (5.2 V). PS1 is the main power line in the bus. It is associated to the PS in SLOT1
WARN_PS2	Status	Source	Line PS2 voltage is below the warning level (5.2 V). PS2 is the secondary power line in the bus. It is associated to the PS in SLOT2.
PS1_V	Analog	Source	Voltage of PS1. If this point in included in coreDb, the same points for redundant systems must not be included (PS1_V_A or PS1_V_B). If not, an error message is indicated when coreDb is checked.
PS2_V	Analog	Source	Voltage of PS2. If this point in included in coreDb, the same points for redundant systems must not be included (PS1_V_A or PS1_V_B). If not, an error message is indicated when coreDb is checked.
WARN_BAT	Status	Source	Allows knowing the system battery state.
Synchronization			
FAIL_SYNC1	Status	Source	Indicates that the main synchronization source is unavailable.
FAIL_SYNC2	Status	Source	Indicates that the secondary synchronization source is unavailable.
Serial Expansion			
FAIL_SER1 ... FAIL_SER8	Status	Source	Active, indicates if there is a failure in the correspondent module (FAIL_SER1 indicates failure in module 1, and so on). Only will be available the points corresponding to the communication modules installed in the RTU.
Configuration and PLC			
FAIL_CONF	Status	Source	Active (1), indicates the configuration is not correct.
FAIL_PLC	Status	Source	If ISaGRAF is used, it will indicate that there is no PLC program or that the program is stopped if FAIL_PLC is 1.
PLC_WARNING	Status	Source	If ISaGRAF is used, it will indicate that there are unmapped ISaGRAF signals in coreDb if PLC_WARNING is 1.
CPU_USAGE	Analog	Source	CPU usage (%).
MEM_USAGE	Analog	Source	RAM usage (%). Only available for SM_CPU866e.
RTU			
FAIL_RTU	Status	Source	If active (1), indicates that the RTU is in an anomalous state. If FAIL_RTU is 0, there is no configuration error (FAIL_CONF == 0), input signal (DOING_WELL == 1) and all tasks registered to the watchdog are responding. While (FAIL_RTU == 0), RTS and DTR pulses are generated. If any task is unresponsive (does not refresh watchdog timer), FAIL_RTU will be 1.

Point	Table	Type	Description
DOING_WELL	Status	Destination	This point indicates to the RTU that something external is running ok. If no source is defined for it, the initial value 1 should be assigned. Usually, this signal has ISaGRAF as origin, and indicates that system is working properly when PLC is working.
LOCAL	Status	Destination	If set to 1 and good quality (IV_LQF, IV_LQF, NT_LQF, NT_LQF are 0) coreDb will be in 'Local state'. If set to 0 and good quality (IV_LQF, IV_LQF, NT_LQF, NT_LQF are 0), coreDb will be in 'Remote state'. In other cases, coreDb will be in 'Unknown state'.
LOCAL:W	Status	Destination	This signal will show the state of system Saitel DP put by HMI.
RESET_RTU	Command	Destination	Command to reboot the RTU if the value is greater than 0.
Time and Date			
YEAR	Analog	Source	Current year.
MONTH	Analog	Source	Current month.
DAY	Analog	Source	Current day.
WDAY	Analog	Source	Current week day. (0: Sunday, 1: Monday ... 6: Saturday).
HOURL	Analog	Source	Current hour.
MINUTE	Analog	Source	Current minute.
SECOND	Analog	Source	Current second.
Temperature			
TEMP	Analog	Source	Current chip temperature.
Redundancy (Only available in redundant configurations)			
RED_VIA1_FAIL	Status	Source	Active (1) indicates unviability of the main communication line of the RCAP protocol.
RED_VIA2_FAIL	Status	Source	Active (1) indicates unviability of the secondary communication line of the RCAP protocol.
RED_I_STATE	Status	Source	Indicates redundancy state of the RTU where the supervision controller is installed. If RED_I_STATE is 1, the local node is ONLINE or STANDBY. If RED_I_STATE is 0, the local node state is FAIL.
RED_IT_FAIL	Status	Source	Indicates redundancy state of the other RTU (dual). If the other RTU is in FAIL state, this signal will be 1. If this signal is 0, then the other RTU is ONLINE or STANDBY.
COM_CTS	Status	Source	Indicates the state of the CTS pin of the serial port that communicates with MSAC. COM_CTS will be 0 and 1 if MSAC is sending 0 or 1. If COM_CTS is 2, the signal is being received by DTR.
DB_UPDATE	Status	Source	Active (1) indicates that a redundant system configured as 'Hot data', the database has been successfully updated.
NODE_A	Status	Source	Active (1) indicates that the current system is configured as node type A.
NODE_B	Status	Source	Active (1) indicates that the current system is configured as node type B.
ONLINE	Status	Source	Active (1) indicates that the current CPU is ONLINE in the redundant system.
GPS Synchronization			
FAIL_SYNCHW	Status	Source	Indicates GPS hardware malfunction if active (1). If the GPS hardware is working correctly, FAIL_SYNCHW will be 0.
FAIL_SYNCDES	Status	Source	If active (1) indicates that there is a deviation of 3 milliseconds.
Link			
LINK:MOTFEC0	Status	Source	Link in ETH1.
LINK:LNC0	Status	Source	Link in ETH2 for SM_CPU866 and SM_CPU866e.
LINK:LNC1	Status	Source	Link in ETH3.
LINK:LNC2	Status	Source	Link in ETH4.

For redundant configurations, all supervision points (except Time and Date) are available with the suffix '_A' and '_B', offering information about CPU A and CPU B respectively. Supervision points without suffix offer information about the CPU ONLINE.

For example:

- **PS1_V_A**: Main power supply voltage in CPU A.
- **PS1_V_B**: Main power supply voltage in CPU B.
- **PS1_V**: Main power supply voltage in CPU ONLINE.

4.3 Supervision Points for Saitel DR

Supervision points for Saitel DR are shown in table below.

Table 7 – Supervision point for a Saitel DR RTU.

Point	Table	Type	Description
Power Supply			
WARN_BAT	Status	Source	Allows knowing the system battery state.
Synchronization			
FAIL_SYNC1	Status	Source	Indicates that the main synchronization source is unavailable.
FAIL_SYNC2	Status	Source	Indicates that the secondary synchronization source is unavailable.
Serial Expansion			
FAIL_SER1 ... FAIL_SER4	Status	Source	Active, indicates if there is a failure in the correspondent module (FAIL_SER1 indicates failure in module 1, and so on). Only will be available the points corresponding to the communication modules installed in the RTU.
Configuration and PLC			
FAIL_CONF	Status	Source	Active (1) indicates configuration failure.
FAIL_PLC	Status	Source	If ISaGRAF is used, it will indicate that there is no PLC program or that the program is stopped if FAIL_PLC is 1.
PLC_WARNING	Status	Source	If ISaGRAF is used, it will indicate that there are unmapped ISaGRAF signals in coreDb if PLC_WARNING is 1.
CPU_USAGE	Analog	Source	CPU usage (%).
MEM_USAGE	Analog	Source	RAM usage (%).
RTU			
FAIL_RTU	Status	Source	If active (1), indicates that the RTU is in an anomalous state. If FAIL_RTU is 0, there is no configuration error (FAIL_CONF == 0), input signal (DOING_WELL == 1) and all tasks registered to the watchdog are responding. While (FAIL_RTU == 0), RTS and DTR pulses are generated. If any task is unresponsive (does not refresh watchdog timer), FAIL_RTU will be 1.
DOING_WELL	Status	Destination	This point indicates to the RTU that something external is running ok. If no source is defined for it, the initial value 1 should be assigned. Usually, this signal has ISaGRAF as origin, and indicates that system is working properly when PLC is working
COLD_RST_CNT	Status	Source	Cold reset counter (it currently counts all system reboots).
RESTART_RST_COUNTER	Command	Destination	Command to restart the system reboot counter.
RESET_RTU	Command	Destination	Command to reboot the RTU if the value is greater than 0.
Time and Date			
YEAR	Analog	Source	Current year.
MONTH	Analog	Source	Current month.
DAY	Analog	Source	Current day.
WDAY	Analog	Source	Current week day. (0: Sunday, 1: Monday ... 6: Saturday).

Point	Table	Type	Description
HOUR	Analog	Source	Current hour.
MINUTE	Analog	Source	Current minute.
SECOND	Analog	Source	Current second.
Redundancy (Only available for redundant configurations)			
RED_VIA1_FAIL	Status	Source	Active (1) indicates unviability of the main communication line of the RCAP protocol.
RED_VIA2_FAIL	Status	Source	Active (1) indicates unviability of the secondary communication line of the RCAP protocol.
RED_I_STATE	Status	Source	Indicates redundancy state of the RTU where the supervision controller is installed. If RED_I_STATE is 1, the local node is ONLINE or STANDBY. If RED_I_STATE is 0, the local node state is FAIL.
RED_IT_FAIL	Status	Source	Indicates redundancy state of the other RTU (dual). If the other RTU is in FAIL state, this signal will be 1. If this signal is 0, then the other RTU is ONLINE or STANDBY.
COM_CTS	Status	Source	Indicates the state of the CTS pin of the serial port that communicates with MSAC. COM_CTS will be 0 and 1 if MSAC is sending 0 or 1. If COM_CTS is 2, the signal is being received by DTR.
DB_UPDATE	Status	Source	Active (1) indicates that a redundant system configured as 'Hot data', the database has been successfully updated.
NODE_A	Status	Source	Active (1) indicates that the current system is configured as node type A.
NODE_B	Status	Source	Active (1) indicates that the current system is configured as node type B.
ONLINE	Status	Source	Active (1) indicates that the current CPU is ONLINE in the redundant system.
Saitel DR (Local Acquisition)			
POL_OK_ABDI	Status	Source	If its value is 1, the polarization of the digital inputs module is correct. If its value is 0, there is a malfunction in polarization or the correspondent point has not been mapped (polarization of the digital input is not watched).
LAQ_FAIL	Status	Source	Active (1) means that there is a malfunction in acquisition. This means that one of the acquisition modules is out of service or in a failure state.
LOCALREMOTE	Status	Source	This signal will show the value of the digital input 2 of the HU_A. If LOCALREMOTE is 1, module is in local mode. In local mode, commands are not activated in the digital output modules. If LOCALREMOTE is 0, RTU is in remote mode, which means normal RTU functioning (will also be in remote mode when this point is not mapped).
LOCALREMOTE:W	Status	Source	This signal will show the state of system Saitel DR put by HMI.
LOCALREMOTE:I	Status	Source	The digital input 2 value will be inverted. In this case, if LOCALREMOTE:I is 1, RTU is still in local mode, and if LOCALREMOTE:I is 0, RTU will still be in remote mode. The difference is in how to process the digital input 2. When the digital input 2 is 0, LOCALREMOTE:I will be 1 (local mode) and when digital input 2 is 1, LOCALREMOTE:I will be 0 (remote mode).
Temperature			
TEMP	Analog	Source	Current chip temperature.
GPS Synchronization			
FAIL_SYNCHW	Status	Source	Indicates GPS hardware malfunction if active (1). If the GPS hardware is working correctly, FAIL_SYNCHW will be 0.
FAIL_SYNCDESV	Status	Source	If active (1) indicates that there is a 3ms of deviation.
Link			

Point	Table	Type	Description
LINK:MOTFEC0	Status	Source	Link in ETH1.
LINK:MOTFEC1	Status	Source	Link in ETH2.
LINK:LAN1	Status	Source	Link in LAN1.
LINK:LAN2	Status	Source	Link in LAN2.
LINK:MNT	Status	Source	Link in MNT.

For redundant configurations, all supervision points (except Time and Date) are available with the suffix '_A' and '_B', offering information about CPU A and CPU B respectively. Supervision points without suffix offer information about the CPU ONLINE.

For example:

- **CPU_USAGE_A**: CPU usage in CPU A.
- **CPU_USAGE_B**: CPU usage in CPU B.
- **CPU_USAGE**: CPU usage in CPU ONLINE.

4.4 Supervision Points for PowerLogic T300

Supervision points for PowerLogic T300 are detailed in the T300 devices user manual.

5 SNMP Manager

Content

5	SNMP MANAGER	101
5.1	DESCRIPTION.....	103
5.2	CONFIGURING SNMP DEVICES	103
5.2.1	CONFIGURING SNMP AGENTS.....	103
5.2.2	CONFIGURING PERIODS.....	104
5.3	SNMP DEVICE IN COREDB	105
5.3.1	SNMP COORDINATE FORMAT.....	105
5.3.2	SNMP POINTS	106

5.1 Description

NOTICE

Please, consult paragraph 'Important information about Cybersecurity' at the beginning of this manual.

Simple Network Management Protocol is a method of interacting with networked devices. A network device runs a SNMP agent as a daemon process which answers requests from a SNMP manager. The agent provides Object Identifiers (OIDs), populating these values and making them available.

NOTICE

SNMP manager is compatible with HU250 and SM_CPU866e RTUs. HU250 is also compatible with SNMP agent functionality, this configuration is detailed in the T300 Devices User Manual (FTE-S857-EBD)

A SNMP manager can query the agents key-value pairs to get their information if allowed (SNMP OIDs can be read and/or written).

NOTICE

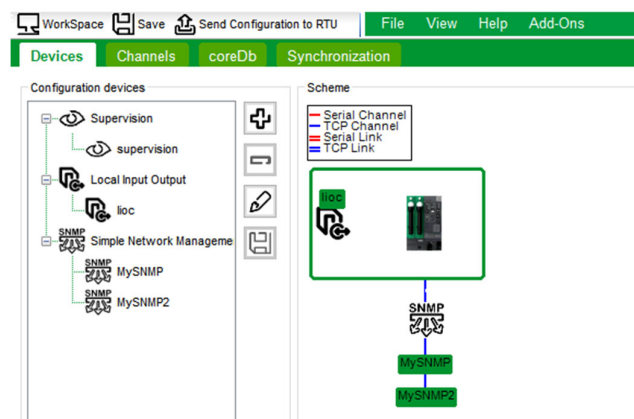
The maximum number of OIDs requested in a SNMP message is, currently, equal to 10.

HU250 or SM_CPU866e, if enabled, could host the SNMP manager, while another device, like a router, need to be installed as SNMP agent.


5.2 Configuring SNMP Devices

Each SNMP Device included in Configuration Devices left panel under the category of 'Simple Network Management Protocol Manager' is associated to an Agent.

Figure 142 – SNMP Devices in Easergy Builder



5.2.1 Configuring SNMP Agents

Press button  and create a new device type 'Simple Network Management Protocol Manager'.

NOTICE

The maximum number of agents for a SNMP Manager is 32.

If this Device type is not available in the Easergy Builder tool installed in the PC, please, install the plugin executing the file 'SNMPM_SPlugin.msi'.

The following configuration must be provided for each new SNMP Agent:

Figure 143 – Configuring a new SNMP Agent.

Where:

- **Name:** Name of the SNMP agent.
- **Version:** SNMP agent version. Available values: 1, 2C or 3. Default value: 3.
- **Authentication protocol:** Only available for Version 3. Authentication protocol to be used. Available protocols: 'MD5', 'SHA1', 'SHA224' and 'NONE'.
- **Community:** Only available for Versions 1 and 2C. UTF8 string with a minimum length of 1 character and 128 characters as maximum length. Mandatory attribute.
- **Port:** UDP port to connect with the agent.
- **Retries:** Number of retries when timeout to send a message expires. Default value: 2.
- **Timeout:** Waiting time (in milliseconds) to receive an answer before sending a new retry. Default value: 1000ms.
- **IPv6:** Mark this field if the IP used to communicate with SNMP agent is IPv6.
- **IP:** IP address of the agent. This address must be in IPv4 or IPv6 format depending on if IPv6 field is activated or not.
- **Interface:** Interface name (eth0, eth1, ...) to send requests. Mandatory if IP address is IPv6 type in Link-Local scope (not used in the rest of cases).

Double-clicking on an agent in the tree, all these previous values can be edited.

5.2.2 Configuring Periods

A period indicates the periodicity to request the value from a SNMP agent. When a period is defined in an agent window configuration, this period is associated to all other agents in the configuration.

NOTICE

The maximum number of periods which can be defined is 32.

The following values must be indicated to add a period:

- **Name:** Name of the period.
- **Period:** Value with the period duration in minutes. Values must be between 0.01 and 1440.99.

5.3 SNMP Device in CoreDb

5.3.1 SNMP Coordinate Format

Coordinates for SNMP points have the following format:

OID[:PER]

Where:

- **OID:** Object Identifier is a full qualified unique-key value in a SNMP agent. Its format is a sequence of points and integer numbers.
- **PER:** Mandatory field for source coordinates and invalid for destination ones. It contains a single period name. That period name is related with a value to indicate the periodicity to request the value from a SNMP agent OID.

NOTICE

The maximum number of coordinates per agent is 200.

Source coordinates examples, supposing P1, cicl2 and PERIOD3 are defined as period:

- .1.3.6.1.2.1.1.3.0:P1
- .1.3.6.1.2.1.2.1.0:cicl2
- .1.3.6.1.2.1.5.1.0:PERIOD3

Destination coordinates examples could be:

- .1.3.6.1.2.1.1.6.0
- .1.3.6.1.2.1.2.2.1.7.3

User can use the point wizard in coreDb in order to easily do the creation of point.

Figure 144 – Configuring a new OID with the SNMP Points Wizard

The screenshot shows the 'SNMP Points Wizard' dialog box. It has two tabs: 'Coordinates' and 'Supervision'. The 'Coordinates' tab is selected. Inside the dialog, there is an 'OID' text input field containing the value '.1.2.3.4' and a 'PER' dropdown menu with 'Period1' selected. Below these fields, there is a preview area showing the resulting coordinate '.1.2.3.4:Period1'. At the bottom right of the dialog, there are two buttons: a checkmark icon and a close icon.

For more information regarding how to use point wizards in coreDb, please refer to Section 3 for more details.

5.3.2 SNMP Points

In addition to SNMP points, some diagnostic data about SNMP agent status are available in coreDb through the following diagnostic points:

- **DIAG:SERV:** Its value indicates if a SNMP agent is in service (value 1). An agent is 'in service' if every OID defined as source or destination is valid and can be requested correctly. If any OID returns error or requests could not be sent, this agent is assumed to be 'out of service' (value 0).
- **DIAG:ONLINE:** This coordinate is equal to 1 when communication with an agent can be performed correctly. When a session to communicate with an agent cannot be created and open correctly, or the timeout expires, means agent is offline or could not be reached, so this point is set to 0.

Table 8 – SNMP points in coreDb.

Coordinate	Status	Analog	Command	Setpoint
OID:PER	Source	Source		
OID			Destination	Destination
DIAG:SERV	Source			
DIAG:ONLINE	Source			

6 Formula Device

Content

6	FORMULA DEVICE.....	107
6.1	GENERAL DESCRIPTION.....	109
6.2	USING FORMULAS.....	110
6.2.1	FORMULAS AS SOURCE.....	110
6.2.2	FORMULAS AS DESTINATION - TRIGGERS.....	111
6.3	MAKING FORMULAS.....	112

6.1 General Description

NOTICE

This Device is not available for HU_B nor HU_BI.

The plugin of Formula Device is integrated by default with the core of Easergy Builder. It allows configure a software developed to do calculation of expression depending on the value of its input variables.

The formula module goal for source coordinates is to calculate a value depending on its input variables. Its value is updated at each execution of the Controller entry (the time between each entry execution is set as the shortest for the RTU system, approximately 20ms depending on the platform).

At the first execution entry, the calculation and writing of results is mandatory for every valid expression. In the case of calculation, the value, quality flag and timestamp will depend on the initial values of coreDb points.

A formula includes any of the following elements:

- Functions: NOT, SPSTODPS, DPSTOSPS, TEMPO, OR, AND, SCALE, MIN, MAX, IF.
- Operators: +, -, *, /, <, >, ==.
- Name of coreDb points
- Name of variables defined in coreDb: FORM_PERIOD, FORM_CYCLETIME, or any other created by the user.
- Constant values
- Any combination of previous elements.

For example, a function with operator expressions as parameters, an operator expression with functions as parameter, a function or operator expression with coreDb point names or constant as parameters, etc.

In a division, if the second input is equal to zero, the calculation result will be equal to NAN value (Not a Number) and written once with this value and invalid quality flag in coreDb. If other expression uses this result on its calculation process, its result will be equal to NAN (0 at status or command), with invalid quality flag, while the precedence expression result is equal to NAN.

Formulas are introduced to coreDb as source coordinate of a point. When an expression is calculated, its value in database (value, quality flag and timestamp) is written at the coreDb point which this coordinate belongs to (as value, quality flag and timestamp, respectively, of this coreDb point).

The following picture shows an example of formula in Easergy Builder:

Figure 145 – Example of Formulas in coreDb.

Name	Description	Source1 Device	Source1 Coordinates
st1		formula	NOT(st2)
st2		formula	TEMPO(st3,10)
st3		formula	DPSTOSPS(st3)
st4		formula	DPSTOSPS(NOT(TEMPO(SPSTODPS(st4),21)))
st5		formula	st3
st6		formula	3.42
st7		formula	-99999901
st8		formula	-81
st9		formula	OR(st1,DPSTOSPS(st2),NOT(st3),TEMPO(st4,10))
st10		formula	st3-SPSTODPS(st1)
st11		formula	-2-(-4)

Formula Device also accepts destination coordinates of a point. They are used as triggers to execute source coordinates formula in a coreDb point designed by this destination coordinate name. More information about Triggers later in this chapter.

6.2 Using Formulas

6.2.1 Formulas as Source


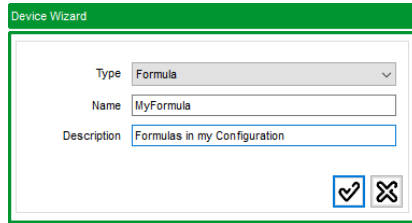
First, a Formula Device must be included in our configuration. Press button  next to the Devices tree and select Formula in the Type:

Figure 146 – Creating a Formula Device.



This Device MyFormula will be used in all following examples of this chapter.

To use Formulas in Easergy Builder, directly write the expression on the source coordinate field or use the wizard as follow:

- Create a signal in coreDb. Double-click on the field name in Status table and write the name of the new variable.

Figure 147 – coreDb signal creation.

7	FAIL_RTU	FAIL_CONF is 1 ...	supervision	FAIL_RT
8	DOING_WELL	Signal for indicati...		
9	VARIABLE_1			
*				

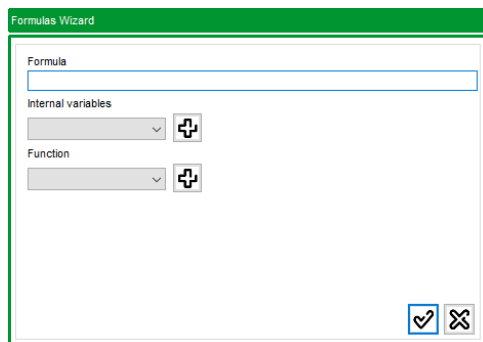
- Select the Formula Device (MyFormula) on the 'Source1 Device' field:

Figure 148 – Formula insert.

7	FAIL_RTU	FAIL_CONF is 1 ...	supervision	FAIL_RT
8	DOING_WELL	Signal for indicati...		
9	VARIABLE_1			
*				

- Select field 'Source1 Coordinates' and right-click. Select 'Launch Point wizard' in the menu.

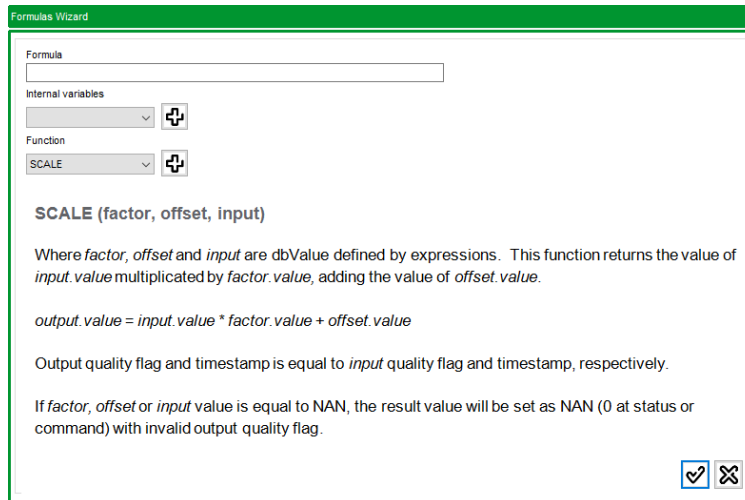
Figure 149 – Formula Point wizard.



This window allows using an easier way to write formulas.

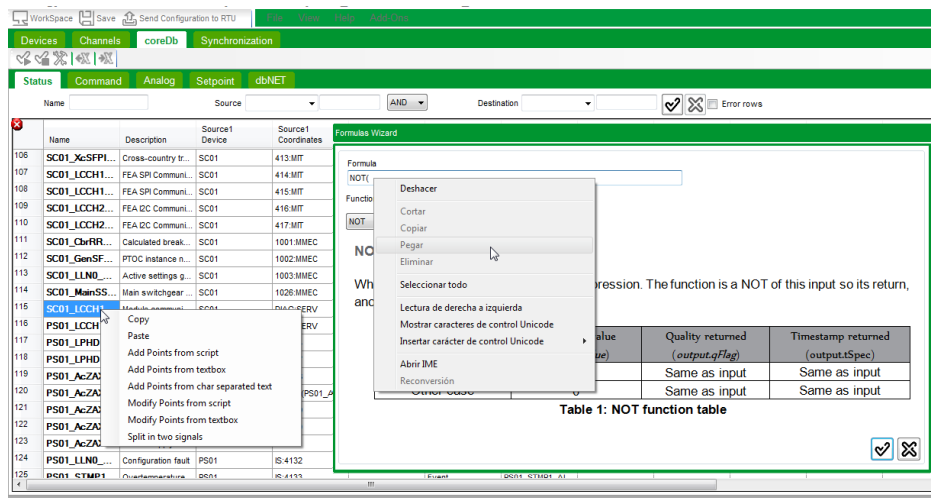
Select a Function and a help will be shown. For example:

Figure 150 – Formula Point wizard help.



When a function and/or Internal variable is selected in the field, press button to add it to the Formula. To use a coreDb point, copy its name from the corresponding tab and then paste it on the Formula field:

Figure 151 – coreDb formula interaction.



In order to complete the desired formula, write it directly on the field, add other function from the Function field or copy the name of others coreDb points.

Finally, press button or press Enter, and your formula is added as coordinate for the new coreDb signal. This coordinate will be use as a Trigger or an Expression depending on if is a Source coordinate or a Destination coordinate.

6.2.2 Formulas as Destination - Triggers

When a Formula is used as destination coordinate, it is called as triggers. A trigger definition must be formed by 2 or 3 field separated by colon (:).

The following figure shows some examples of triggers with source formulas waiting to be executed:

Figure 152 – Example of triggers definition.

st1		dnpSlave1	1:OB01:0		MyFormula	st5.EVENT
st2		dnpSlave1	1:OB10:2		MyFormula	st6.VAL:1
st3		dnpSlave1	2:OB03:1		MyFormula	st7.VAL:0
st4		dnpSlave1	1:OB12:2		MyFormula	st8.EVENT
st5		MyFormula	st3-SPSTODPS(st1)			
st6		MyFormula	2.3*st1+(-2)/NOT(st4)			
st7		MyFormula	IF(FORM_CYCLETIME>FORM_PERIOD, 1)			
st8		MyFormula	AND(st1,st10,MAX(2,FORM_CYCLETIME)<10)			

For each coreDb point:

- A trigger is executed when its coreDb point gets a defined value or when is changed and generates an event. When these requirements happen, the coreDb point with a formula as source device, designed in the destination formula coordinate, is executed.
- If the trigger is set as destination coordinates, it waits an event on their coreDb point to satisfy the trigger type (EVENT or VAL). When it is satisfied, the source formula, of the coreDb point designed on the trigger destination coordinate, will be calculated. We have two triggers type:
 - **EVENT**: This trigger waits for a new event on its point, to make the trigger first field formula source calculated.
 - **VAL**: This trigger waits for an event with a determined value to make the trigger first field formula source calculated. This value is set as third field of the destination coordinate name.

A source coordinate formula, which is being triggered, is never executed until the trigger is fired.

An event happens, for example, if:

- Status points: There is a change on the value, quality flag or a krunch data was received with the mode KDM_EVENT.
- Analog points: When is the first event received by this point, or if the difference between the new and the last analog values is higher than a determined value.
- Command or Setpoints: There is a change on the value, quality flag or timestamp.

6.3 Making Formulas

This section shows all expressions allowed to be calculated. The following terms can be used:

- **dbValue**: Is a structure which contains a value, quality flag and timestamp.
- **Expression**: Any of the defined functions, operations, coreDb point name or constant value.
- **Invalid quality flag**: Quality flag with the bit IV_LQF active. If a coreDb point has NT_LQF, NT_RQF, IV_RQF active, IV_LQF will make active internally by formula Controller.
- **Good quality flag**: Not Invalid Quality flag.
- **Controller start time**: Timestamp corresponding with the moment when the controller started.
- **NAN**: Not a Number. It is used when a result cannot be calculated.

The coordinates are fixed and are always mapped as status, command, analog or setpoint source and its coordinate name must follow the structure of the implemented triggers or expressions, described on the previous section

Examples of valid expressions (mapped as source coordinate names) are:

- **SCALE**(NOT(input2)+2,TEMPO(DPSTOSPS(input3),2*MIN(4-SPSTODPS(input5),input4)),input1)
- **MAX**(8*NOT(input1), (input2+input3)/2) * (TEMPO(input4, 10)+SPSTODPS(input5)) / 2
- **AND**(input1 > input2, **MAX**(1, FORM_CYCLETIME) == FORM_PERIOD/2, **IF**(input2, input1+1, 2))

If any of the points mapped for formula as source is wrong introduced, it will be set as *INVALID* and will not be considered during calculation process at entry, since this point will not be included in the list of formulas.

Examples of valid triggers (mapped as destination coordinate names) are:

- input1:**EVENT**
- input2:**VAL**:-2.34
- input3:**VAL**:1

Where *input2* and *input3* are coreDb point names defined on status, analog, command or setpoint tables. These coreDb points also have a formula source coordinate expression to be calculated. If not, the trigger will not be considered and deleted from memory.

#

NOT (input) → return: output

Where:

- **output** is a dbValue and **input** is a dbValue defined by expressions.

The function is a NOT of this input and it returns the value as output. It can return:

Input variable value (<i>input.value</i>)	Calculated value (<i>output.value</i>)	Quality returned (<i>output.qFlag</i>)	Timestamp returned (<i>output.tSpec</i>)
0	1	Same as input	Same as input
Other case	0	Same as input	Same as input

This function is implemented according to the following pseudo-code:

```
If (input.value != !previousResult)
  → output.value = !input.value
```

Output timestamp and quality flag values will be the same as input.

If input value is equal to NAN, the result value will be NAN (0 at status or command) with quality flag sets to invalid and timestamp equal to writing time.

Example

NOT (input1)

Given that *input1* is a coreDb point name, NOT function will return:

- 0 if *input1* is different to 0.
- 1 if *input1* is equal to 0.
- Quality flag will be always the same as *input1* quality flag.
- Output timestamp will be the same as *input1* timestamp.

TEMPO (**input**, **time**) → return: **output**

Where:

- **output** is a dbValue and **input**, **time** is dbValue defined by expressions.
- **time.value** indicates a number of seconds.

The function returns *input.value* if it moves to 1, or another value different to 0, for at least *time.value* seconds. After this time, the function returns this value of *input.value* as long as it is at this value. If *input.value* moves to 0, its returns 0 immediately:

Figure 153 – TEMPO function basic diagram.

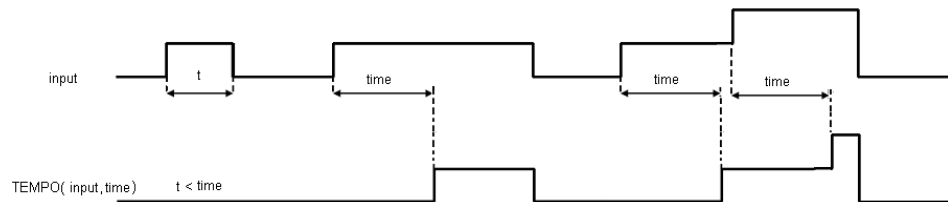


Figure 154 – TEMPO function with multiple changes diagram.

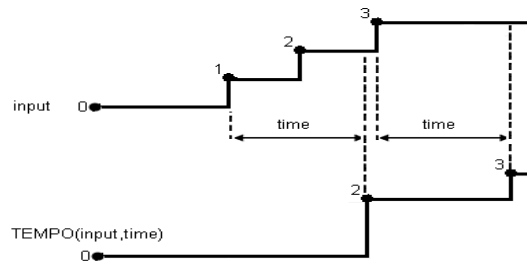
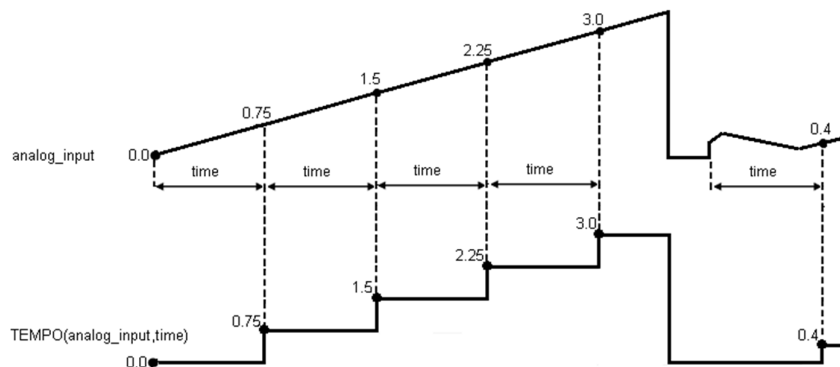


Figure 155 – TEMPO function with analog input diagram.



The returned quality is the same as *input.qFlag*.

The returned timestamp is the moment which tempo signal changes, so *time* seconds after *input* if its value is different to 0 or the moment in which *input* goes equal to 0, both in formula Controller local time.

The maximum *time* value allowed is the size of an unsigned long divided by 1000 (4294967295/1000 in our systems). If its value is negative, the expression will get the absolute value.

If *input* value or quality changes, when the value is different to 0, the TEMPO formula will wait *time* seconds to change both on return. If many changes are done on this time, the TEMPO counter will start with the first one and *time* seconds after, this formula will return with the last changes.

If *input* value is equal to 0, TEMPO formula will return any new change at quality flag immediately.

If input or time value is equal to NAN, the result value will be equal to NAN (0 at status or command) and its quality flag equal to IV_LQF.

This function is implemented according to the following pseudo-code:

```

If (input.value == 0 && (previousValue != input.value || previousQFlag != input.qFlag))
    → output.value = 0; output.qFlag = input.qFlag;
    → output.tSpec = LOCAL_NOW();
    → notWaitTime = 1;
Else if (input.value != 0 && previousResult!=input.value || previousQFlag!=input.qFlag) && notWaitTime)
    → cntTime = sysCnt1mlsg();
    → notWaitTime = 0
If (notWaitTime == 0)
    → if (sysCnt1mlsg() – cntTime >= time.value_ms)
        → output.value = input.value; output.qFlag = input.qFlag
        → output.tSpec = LOCAL_NOW();
        → notWaitTime = 1
  
```

Where:

- **notWaitTime** is a flag that indicates if the algorithm is not waiting (when it is equal to 1) the seconds indicated by *time.value* to change the *la_outputVal* to a value different to 0.
- **cntTime** stores the time when *input.value* changed to a value different to 0.
- **time.value_ms** is the value of *time.value* in milliseconds, converted to be compared with the system counter (*sysCnt1mlsg()*) which value is returned in ms.
- **LOCAL_NOW()** is a macro which returns the local time of the RTU at this moment.

Example

TEMPO (input1, 10)

Given that *input1* is a coreDb point name, TEMPO function will return:

- 0 if *input1* is equal to 0.
- A value equal to *input1.value* if it is different to 0, and this value has been written 10 seconds or more before.

SPSTODPS (input) → return: output

Where:

- **output** is a dbValue and **input** is a dbValue defined by expressions.

The function returns 1 if the input value last bit is 0, 2 if this bit is 1 and 3 if the input.qFlag is not GOOD.

Input value	Quality flag	Output value	Output quality flag	Output timestamp
0	GOOD	1	Same as input	Same as input
1	GOOD	2	Same as input	Same as input
X	IV_LQF, IV_RQF, NT_LQF or NT_RQF	3	Same as input	Same as input

The input value is always casted as long integer.

This function is implemented according to the following pseudo-code:

```

If (If (input.qFlag == IV || input.qFlag == NT)
    → output.value = 3
Else if (input.value & 1 == 0)
    → output.value = 1
Else if (input.value & 1 == 1)
    → output.value = 2

```

The quality and timestamp returned are the same as *input.qFlag* and *input.tStamp* respectively.

If input value is equal to NAN, the result value will be NAN (0 at status or command) with quality flag as invalid and timestamp equal to writing time.

Example

SPSTODPS (input1)

Given that *input1* is a coreDb point name, SPSTODPS function will return:

- 3 if input1 quality flag has IV_LQF, IV_RQF, NT_LQF or NT_RQF set to 1.
- 1 if input1 last bit is equal to 0.
- 2 if input1 last bit is equal to 1.
- Output quality flag will be always the same as input1 quality flag.
- Output timestamp will be the same as input1 timestamp.

DPSTOSPS (**input**) → return: **output**

Where:

- **output** is a dbValue and **input** is a dbValue defined by expressions.

Iqsw#yduledn#ydox# +Iqsw#ydox#	Fdfxawhg#ydox# +rxwsw#ydox#	Txdw #hwuqhg# +rxwsw#I@j#	Wp hvwdp s#hwuqhg# +rxwsw#Wshf#
0	0	<i>input.qFlag</i> IV_LQF	Same as input
1	0	<i>input.qFlag</i>	Same as input
2	1	<i>input.qFlag</i>	Same as input
3 or other	0	<i>input.qFlag</i> IV_LQF	Same as input

When an *output* returns with a quality flag equal to *input.qFlag* | IV_LQF, it means the same quality as input adding, and confirms the Invalid flag (IV_LQF) set up at output quality.

The returned timestamp is the same as *input.tSpec*.

This function is implemented according to the following pseudo-code:

```

If (input.value == 0)
    → output.value = 0 && output.qFlag = input.qFlag | IV_LQF
Else if (input.value == 1)
    → output.value = 0 && output.qFlag = input.qFlag
Else if (input.value == 2)
    → output.value = 1 && output.qFlag = input.qFlag
Else
    → output.value = 0 && output.qFlag = input.qFlag | IV_LQF

```

If input value is equal to NAN, the output value will be set as NAN (0 at status or command) with quality flag equals to invalid and timestamp equal to writing time.

Example

DPSTOSPS (input1)

Given that *input1* is a coreDb point name, DPSTOSPS function will return:

- 0 as output.value and input.qFlag | INVALID as output.qFlag if input1 value is equal to 0 or equal or greater than 3.
- 0 as output.value and input.qFlag as output.qFlag if input1 value is equal to 1.
- 1 as output.value and input.qFlag as output.qFlag if input1 value is equal to 2.
- Output timestamp will be the same as input1.
- Output timestamp will be the same as input2 timestamp when it got its minimum value.

OR (input_1, input_2, ..., input_N) → return: output

Where:

- **output** is a dbValue and **input_i** is a dbValue defined by an expression, where $\forall i \in \{1,2, \dots, N\}$.

The function is an OR of these inputs so its return, on another dbValue called as *output*, could be:

- 1 at *output* value if one or more of the input values are different to 0.
 - If one or more of these inputs, with value different to 0, has quality as GOOD, it returns the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input* with GOOD quality flag.
 - If every input with value 1 has quality different to GOOD, it returns the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input*.
- 0 at *output* value if all input values are 0.
 - If every input has quality as GOOD, it returns the output quality the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input*.
 - If one or more of the inputs have quality different to GOOD, it the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input* with quality flag different to GOOD.

This function is implemented according to the following pseudo-code:

```

output.qFlag = input_1.qFlag; output.tSpec = input_1.tSpec;
Foreach input_i where i ∈ {1,2, .. , N}
  If (input_i.value != 0) → output.value = 1
    → If(input_i.qFlag == GOOD)
      → if(output.qFlag != GOOD || input_i.tSpec > output.tSpec)
        → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
      → Else
        → if(output.qFlag != GOOD && input_i.tSpec > output.tSpec)
          → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
    Else if (output.value == 0)
      → If(input_i.qFlag != GOOD)
        → if(output.qFlag == GOOD || input_i.tSpec > output.tSpec)
          → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
        → Else
          → if(output.qFlag == GOOD && input_i.tSpec > output.tSpec)
            → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
  End foreach

```

Output timestamp will be equal to the timestamp value of the input which set the quality flag.

If any of the input values are equal to NAN, OR function result will be NAN (0 at status or command) with invalid as quality flag when none of the rest of inputs are equal to 1, or a value different to 0.

Example

OR (input1, 1, input2)

Given that *input1* and *input2* are coreDb point names and their values are equal to 0, OR function will return 1 as *output.value*. The quality flag and timestamp will come from the constant input equal to 1 (GOOD quality flag and Controller start time as timestamp):

- If *input1* value changes to a value different to 0, AND function will return 1 as *output.value*. The quality flag and timestamp will come from *input1* since is more recent than the constant input.
- If *input2* value also moves to a value different to 0, AND function will return 1 as *output.value*. The quality flag and timestamp will depend on:
 - If *input1* and *input2* quality flags are GOOD type, the output quality flag and timestamp come from the most recent of the inputs.
 - If *input1* or *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the input with not GOOD quality flag type.
 - If *input1* and *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the most recent quality flag.

AND (input_1, input_2, ..., input_N) → return: output

Where:

- **output** is a dbValue and **input_i** is a dbValue defined by an expression, where $\forall i \in \{1, 2, \dots, N\}$.

The function is an AND of these inputs so its return, on another dbValue called as output, could be:

- 0 at *output value* if one or more of the input values are equal to 0
 - If one or more of these inputs, with value equal to 0, has quality as GOOD, it returns the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input* with GOOD quality flag.
 - If every input with value 0 has quality different to GOOD, it returns the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input*.
- 1 at *output value* if all input values are different to 0
 - If every input has quality as GOOD, it returns the output quality the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input*
 - If one or more of the inputs have quality different to GOOD, it the output quality (*output.qFlag*) and timestamp (*output.tSpec*) of the most recent *input* with quality flag different to GOOD.

This function is implemented according to the following pseudo-code:

```

output.qFlag = input_1.qFlag; output.tSpec = input_1.tSpec;
Foreach input_i where i ∈ {1,2, .. , N}
  If (input_i.value == 0) → output.value = 0
    → If(input_i.qFlag == GOOD)
      → if(output.qFlag != GOOD || input_i.tSpec > output.tSpec)
        → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
      → Else
        → if(output.qFlag != GOOD && input_i.tSpec > output.tSpec)
          → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
    Else if (output.value == 1)
      → If(input_i.qFlag != GOOD)
        → if(output.qFlag == GOOD || input_i.tSpec > output.tSpec)
          → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
        → Else
          → if(output.qFlag == GOOD && input_i.tSpec > output.tSpec)
            → output.qFlag = input_i.qFlag; output.tSpec = input_i.qFlag;
  End foreach

```

Output timestamp will be equal to the timestamp value of the input which set the quality flag.

If any of the input values are equal to NAN, AND function result will be NAN (0 at status or command) with invalid as quality flag when none of them are equal to 0.

Example

AND (input1, 1, input2)

Given that *input1* and *input2* are coreDb point names and their values are equal to 0, AND function will return 0 as *output.value*. The quality flag and timestamp will depend on:

- If *input1* and *input2* quality flags are GOOD type, the output quality flag and timestamp come from the most recent of the inputs.
- If *input1* or *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the input with GOOD quality flag type.
- If *input1* and *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the most recent quality flag.

If *input1* value changes to a value different to 0, AND function will return 0 as *output.value*. The quality flag and timestamp will come from *input2*.

If *input2* value also moves to a value different to 0, AND function will return 1 as *output.value*. The quality flag and timestamp will depend on:

- If *input1* and *input2* quality flags are GOOD type, the output quality flag and timestamp come from the most recent of the inputs.
- If *input1* or *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the input with not GOOD quality flag type.
- If *input1* and *input2* quality flags are not GOOD type, the output quality flag and timestamp come from the most recent quality flag.

MIN (*numDays*, *input*) → return: *output*

Where:

- *output* is a dbValue, *numDays* and *input* are dbValue defined by expressions.

This function returns the minimum value of *input.value* during the number of days, indicated by *numDays.value* (between 0 and 30), at the end of this number of days. Once the new value is set, the function starts again.

Output timestamp and quality flag are set up equals to timestamp and quality flag from *input*, respectively, when the minimum value was read.

If the same minimum value is reached several times, the returned quality flag and timestamp are from the last one.

Each period starts and ends at midnight of the day. The writing is only done once the period ends.

If the number of days (*numDays.value*) is equal to 0, the function returns the value of the second parameter (*input*) each time it is updated.

If the number of days is greater than 30 or less than 0, the function will be updated with 0 as value and the output quality flag will be equal to invalid. The output timestamp, in this case, is equal to calculation moment.

If *numDays* or *input* values are equal to NAN, the result value will be equal to NAN (0 at status or command) with invalid output quality flag.

This function is implemented according to the following pseudo-code:

```

If(numDays.value == 0)
    → output = input
Else if(numDays.value > 0) && (la_fstNumDaysVal <= 30)
    → if(input.value <= previousResult)
        → output = input
    → Else if(fstVal)
        → output = input;
        → fstVal = 0;
    → if(acumTime < currentTSpec)
        → krunchData; startTSpec = currentTSpec;
        → fstVal = 1;
Else → output.value = 0; output.qFlag = IV_LQF; output.tSpec = LOCAL_NOW();

```

Example

MIN (3, *input1*)

Given that *input1* is a coreDb point names, MIN function will return the minimum value of *input1.value* for 3 days. Once the 3 days are reached, the days counter will start again. The output quality flag and timestamp will be equal to input quality and timestamp, respectively, when the minimum value was reached.

MAX (numDays, input) → return: output

Where:

- **output** is a dbValue, **numDays** and **input** are dbValue defined by expressions.

This function returns the maximum value of *input.value* during the number of days, indicated by *numDays.value* (between 0 and 30), at the end of this number of days. Once the new value is set, the function starts again.

Output timestamp and quality flag are set up equals to timestamp and quality flag from *input*, respectively, when the maximum value was read.

If the same maximum value is reached several times, the returned quality flag and timestamp are from the last one.

Each period starts and ends at midnight of the day. The writing is only done once the period ends.

If the number of days (*numDays.value*) is equal to 0, the function returns the value of the second parameter (*input*) each time it is updated.

If the number of days is greater than 30 or less than 0, the function will be updated with 0 as value and the output quality flag will be equal to invalid. The output timestamp, in this case, is equal to calculation local moment.

If *numDays* or *input* values are equal to NAN, the result value will be equal to NAN (0 at status or command) with invalid output quality flag.

This function is implemented according to the following pseudo-code:

```

If(numDays.value == 0)
  → output = input
Else if(numDays.value > 0) && (la_fstNumDaysVal <= 30)
  → if(input.value >= previousResult)
    → output = input
  → else if(fstVal)
    → output = input;
    → fstVal = 0;
  → if(acumTime < currentTSpec)
    → krunchData; startTSpec = currentTSpec;
    → fstVal = 1;
Else → output.value = 0; output.qFlag = IV_LQF; output.tSpec = LOCAL_NOW();
  
```

Example

MAX (15, input1)

Given that *input1* is a coreDb point names, MAX function will return the maximum value of *input1.value* for 15 days. Once the 15 days are reached, the days counter will start again. The output quality flag and timestamp will be equal to input quality and timestamp, respectively, when the maximum value was reached.

SCALE (**factor**, **offset**, **input**) → return: **output**

Where:

- **output** is a dbValue, **factor**, **offset** and **input** are dbValue defined by expressions.

This function returns the value of *input.value* multiplied by *factor.value*, adding the value of *offset.value*: $\text{output.value} = \text{input.value} * \text{factor.value} + \text{offset.value}$

Output quality flag and timestamp is equal to *input* quality flag and timestamp, respectively.

If *factor*, *offset* or *input* value is equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

This function is implemented according to the following pseudo-code:

```
output.value = input.value * factor.value + offset.value
output.qFlag = input.qFlag
output.tSpec = input.tSpec
```

Example

SCALE (2, 0.5, input1)

Given that *input1* is a coreDb point names, SCALE function will return the result of:

- $\text{output.value} = 2 * \text{input1.value} + 0.5$
- $\text{output.qFlag} = \text{input1.qFlag}$
- $\text{output.tSpec} = \text{input1.tSpec}$

IF (cond, ifVar) → return: output

Where:

- **output** is a dbValue, **cond** and **ifVar** are dbValue defined by expressions.

F r q g l w r q # y d o x h # + f r q g l y d o x h ,	U h w k u q h g # y d o x h # + r x w s x w l y d o x h ,	T x d o w # i h w k u q h g # + r x w s x w l t I a j , #	W l p h v w d p s # i h w k u q h g # + r x w s x w l w s h f , #
0	No change	No change	No change
Other case	Same as ifVar	Same as ifVar	Same as ifVar

If *cond* or *ifVar* value is equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

This function (with two parameters) only can be as used as main expression:

- IF(cond1, input1*input2) → VALID
- IF(NOT(cond1), input1) * input2 → INVALID (the main expression is the Multiplication Operation (*))
- SCALE(input2, IF(cond1, input3*input1), input4) → INVALID (the main expression is the SCALE function).
- This function is implemented according to the following pseudo-code:

```

If(cond.value != 0)
  → output = ifVar
  
```

Example

IF(1, input1)

Given that *input1* is a coreDb point names, IF with two parameters will always return the value, quality Flag and timestamp from *input1*.

IF (*cond*, *ifVar*, *elseVar*) → return: *output*

Where:

- *output* is a dbValue, *cond*, *ifVar* and *elseVar* are dbValues defined by expressions.

This function returns *ifVar* if the value of *cond* is different to 0. Otherwise, *elseVar* is returned as result:

<i>F r q g l w r q # y d o x h #</i> <i>+ f r q g y d o x h ,</i>	<i>U h w k u q h g # y d o x h #</i> <i>+ r x w s x w i y d o x h ,</i>	<i>T x d o w # u h w k u q h g #</i> <i>+ r x w s x w i t I o j , #</i>	<i>W p h v w d p s †</i> <i>u h w k u q h g #</i> <i>+ r x w s x w i w s h f , #</i>
0	Same as <i>elseVar</i>	Same as <i>elseVar</i>	Same as <i>elseVar</i>
Other case	Same as <i>ifVar</i>	Same as <i>ifVar</i>	Same as <i>ifVar</i>

If *cond*, *ifVar* or *elseVar* value is equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

This function (with three parameters) can be used as an expression as a sub-expression.

This function is implemented according to the following pseudo-code:

```

If(cond.value != 0)
  → output = ifVar
Else
  → output = elseVar

```

Example

IF(0, *input1*, *input2*)

Given that *input1* and *input2* are coreDb point names, IF with three parameters will always return the value, quality Flag and timestamp from *input2*.

Plus operation (**input1 + input2**) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValues defined by expressions.

This function returns the value of *input1.value* plus *input2.value*.

output.value = *input1.value* + *input2.value*

If both inputs are not constant values, the returned quality flag is *input1.qFlag* | *input2.qFlag*, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

This function is implemented according to the following pseudo-code:

```

output.value = input1.value + input2.value
If (input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec
  
```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag

Example

input1 + input2

Given that *input1* is a coreDb point name, this function will return the result of:

- output.value = input1.value + 3
- output.qFlag = input1.qFlag
- output.tSpec = input1.tSpec

Minus Operation (**input1 - input2**) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValues defined by expressions.

This function returns the value of *input1.value* minus *input2.value*.

output.value = *input1.value* - *input2.value*

If both inputs are not constant values, the returned quality flag is *input1.qFlag* | *input2.qFlag*, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

This function is implemented according to the following pseudo-code:

```

output.value = input1.value - input2.value
If(input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec
  
```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

Example

$-(input1 - input2 - MIN(4, input3))$

Given that *input1*, *input2* and *input3* are coreDb point names, the function will return, once each parameter gets its values:

- *output.value* = - (((*input1.value* - *input2.value*) - MIN(4,*input3*))). Calculating it from left to right.
- *output.qFlag* = *input1.qFlag* | *input2.qFlag* | MIN(4, *input3*).qFlag. In this case, the output quality flag is adding the quality flag values from each input, with type different to constant. The last one, is the returned value of MIN(4, *input3*) quality flag returned.
- *output.tSpec* will be equal to the most recent between *input1.tSpec*, *input2.tSpec* and the returned timestamp from MIN(4, *input3*) calculation.

Multiplication Operation ($input1 * input2$) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValues defined by expressions.

This function returns the value of *input1.value* plus *input2.value*.

$output.value = input1.value * input2.value$

If both inputs are not constant values, the returned quality flag is $input1.qFlag | input2.qFlag$, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

This function is implemented according to the following pseudo-code:

```

output.value = input1.value * input2.value
If (input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec

```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

Example

$(input1 + input2) * 4 + MIN(4, input3)$

Given that *input1*, *input2* and *input3* are coreDb point names, the function will return, once each parameter gets its values:

- Firstly, it calculates the value of $input1.value + input2.value$, then this result is multiplied by 4 and, finally, the value of calculate $MIN(4, input3)$ is added:
 $output.value = ((input1.value + input2.value) * 4) + MIN(4, input3)$
- Calculating it from left to right, having into account that multiplication has higher precedence than plus operation, but less than brackets.
 $output.qFlag = input1.qFlag | input2.qFlag | MIN(4, input3).qFlag$
- In this case, the output quality flag is adding the quality flag values from each input, with type different to constant. The last one, is the returned value of $MIN(4, input3)$ quality flag returned.
 $output.tSpec$ will be equal to the most recent between $input1.tSpec$, $input2.tSpec$ and the returned timestamp from $MIN(4, input3)$ calculation.

Division Operation ($input1/input2$) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValue defined by expressions.

This function returns the value of *input1.value* divided by *input2.value*.

$output.value = input1.value / input2.value$

If both inputs are not constant values, the returned quality flag is $input1.qFlag | input2.qFlag$, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

This function is implemented according to the following pseudo-code:

```

output.value = input1.value / input2.value
If (input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec
  
```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag. Besides, if the second input value is equal to 0, the returned value will be also NAN (0 at status or command) with invalid output quality flag.

Example

3 + input1 / NOT(input2)

Given that *input1* and *input2* coreDb point names, the function will return, once each parameter gets its values:

- Firstly, it calculates the value of $input1.value/NOT(input2)$, then 3 is added to this previous result.
 $output.value = (input1.value / NOT(input2)) + 3$
- Calculating it from left to right. Besides, division calculation has higher precedence than plus or minus operation.
 $output.qFlag = input1.qFlag | NOT(input2).qFlag$. In this case, the output quality flag is adding the quality flag values from each input, with type different to constant.
- The last one, is the returned value of $NOT(input2)$ quality flag returned. *output.tSpec* will be equal to the most recent between *input1.tSpec* and the returned timestamp from $NOT(input2)$ calculation.
- If $NOT(input2)$ result value is equal to 0 (so *input2.value* is different to 0), when the division operation is done, its result is equal to NaN (0 at status or command) and it is inherited to the complete formula.

Greater Conditional Operation (**input1>input2**) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValue defined by expressions.

This function returns 1 if the value of *input1* is greater than *input2.value*. Otherwise the returned value is 0.

If both inputs are not constant values, the returned quality flag is *input1.qFlag* | *input2.qFlag*, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

Conditional operators have the lowest precedence between the operators.

This function is implemented according to the following pseudo-code:

```

If(input1.value > input2.value)
    → output.value = 1;
Else
    → output.value = 0;
If (input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec
  
```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

Example

input1 > 2

Given that *input1* is coreDb point name, this function will return 1 if *input1.value* is greater than 2. If not, the returned value is equal to 0:

- output.qFlag = input1.qFlag
- output.tSpec = input1.tSpec

Less Conditional Operation ($input1 < input2$) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValue defined by expressions.

This function returns 1 if the value of *input1* is LESS than *input2.value*. Otherwise the returned value is 0.

If both inputs are not constant values, the returned quality flag is $input1.qFlag | input2.qFlag$, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

Conditional operators have the lowest precedence between the operators.

This function is implemented according to the following pseudo-code:

```

If(input1.value < input2.value)
    → output.value = 1;
Else
    → output.value = 0;
If (input1.type != CONST)
    → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
    → output.qFlag |= input2.qFlag
    → if(input2.tSpec > input1.tSpec)
        → output.tSpec = input2.tSpec
Else
    → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec
  
```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

Example

$(input1 + input2) * 4 < MIN(4, input3)$

Given that *input1*, *input2* and *input3* are coreDb point names, the function will return, once each parameter gets its values:

- Firstly, on one side, it calculates the value of $input1.value + input2.value$, and then this result is multiplied by 4. On another side, $MIN(4, input3)$ is calculated. Both sides are compared, and if the first one is the lowest, the *output.value* is equal to 1. If not, *output.value* is equal to 0.
- $output.qFlag = input1.qFlag | input2.qFlag | MIN(4, input3).qFlag$. In this case, the output quality flag is adding the quality flag values from each input, with type different to constant. The last one, is the returned value of $MIN(4, input3)$ quality flag returned.
- *output.tSpec* will be equal to the most recent between *input1.tSpec*, *input2.tSpec* and the returned timestamp from $MIN(4, input3)$ calculation

Equal Conditional Operation (**input1==input2**) → return: **output**

Where:

- **output** is a dbValue, **input1** and **input2** are dbValue defined by expressions.

This function returns 1 if the value of *input1* is equal to *input2.value*. Otherwise the returned value is 0.

If both inputs are not constant values, the returned quality flag is *input1.qFlag* | *input2.qFlag*, and the output timestamp is equal to the most recent of both inputs.

If only one of the inputs is not a constant value, the returned quality flag and timestamp are equal to this input values for quality and timestamp.

If both inputs are constants, the returned quality and timestamp are equal to the second input values for quality and timestamp.

Conditional operators have the lowest precedence between the operators.

This function is implemented according to the following pseudo-code:

```

If(input1.value == input2.value)
  → output.value = 1;
Else
  → output.value = 0;
If (input1.type != CONST)
  → output.qFlag = input1.qFlag && output.tSpec = input1.tSpec
If(input2.type != CONST) && input2.timestamp > input1.timestamp)
  → output.qFlag |= input2.qFlag
  → if(input2.tSpec > input1.tSpec)
    → output.tSpec = input2.tSpec
Else
  → output.qFlag = input2.qFlag && output.tSpec = input2.tSpec

```

If any of the input values are equal to NAN, the result value will be set as NAN (0 at status or command) with invalid output quality flag.

Example

input1 == 3

- Given that *input1* is coreDb point name, this function will return 1 if *input1.value* is equal to 3. If not, the returned value is equal to 0:
 - output.qFlag = input1.qFlag
 - output.tSpec = input1.tSpec

CoreDb Points

Formula Device also accepts a coreDb point name as a valid coordinate. In this case, the calculation function consists on copy the value of this coreDb point and refresh it when is needed.

The Device firstly look for the coreDb point name at *Status*, secondly at *Command*, followed by *Analog* and finally *Setpoint* type.

If the quality flag from the coreDb point name has the *Not Topical* or *Invalid* (local or remote), the *Invalid Local* quality flag will be set to 1.

If its value is equal to NAN, the value copied to formula variable is NAN (0 at status or command) with invalid quality flag and current local timestamp.

Variables

There are two special variables used in the Formula Device:

- FORM_PERIOD
- FORM_CYCLETIME

FORM_PERIOD

This variable name will get the value of the waiting time between all non-triggered expressions are calculated on a cycle and the next one.

NOTICE
In this version, this time is fix to 1 and cannot be changed in Easergy Builder. This functionality will be implemented in following versions.

Its quality flag is equal to 0 (GOOD) and its timestamp is the date of the Formula Device initialization.

FORM_CYCLETIME

This variable name gets the last duration of the Device entry calculation. Its unit is in milliseconds.

Its quality flag is equal to 0 (GOOD) and its timestamp is the local date of the end of the last formula calculation cycle time.

If any of these variable names are set up as coreDb point names, the Formula Device will get them as a normal coreDb point name variable.

Constant Values

Formula Device accepts a constant value (integer or float, positive or negative) as coordinate. In this case, the coordinate numeric value will be copied as its value.

Index of Figures

Figure 1 – Baseline Software Platform	11
Figure 2 – Relation between coreDb and other applications.....	12
Figure 3 – Saitel DR local acquisition plugin interface.	14
Figure 4 – coreDb interface.	15
Figure 5 – Display properties selection.	17
Figure 6 – Display configuration window.	17
Figure 7 – Display scale selection window.	18
Figure 8 – Startup of the Easergy Builder installation.	18
Figure 9 – Selecting plugins to be installed.	18
Figure 10 – Selecting plugins to be uninstalled.	20
Figure 11 – Easergy Builder startup window.	20
Figure 12 – Plugin not available.	21
Figure 13 – Easergy Builder in WorkSpace mode.....	22
Figure 14 – Workspace stored in folders.	23
Figure 15 – RTU tree.	24
Figure 16 – Information on the edition area for a HUE RTU.....	25
Figure 17 – Information on the edition area for an PowerLogic T300 RTU.....	25
Figure 18 – Information on the edition area for a Configuration.....	25
Figure 19 – Information on the Changes Console.....	26
Figure 20 – Information on the Log Console.	26
Figure 21 – Information on the RTU Sys Console.....	26
Figure 22 – Configuration mode.	27
Figure 23 – Device catalog.	28
Figure 24 – New RTU webUi.	29
Figure 25 – webUi enabling.	29
Figure 26 – webUI main configuration screen.	29
Figure 27 – webUI real time configuration.....	30
Figure 28 – Easergy Builder in WorkSpace mode (RTU parameters).	33
Figure 29 – Easergy Builder in WorkSpace mode (Configuration schematic).	33
Figure 30 – Creating a new RTU.....	34
Figure 31 – Removing an RTU.....	35
Figure 32 – Rebooting the RTU.....	35
Figure 33 – Error rebooting an RTU.....	36
Figure 34 – Importing the configuration from the RTU.....	36

Figure 35 – Selecting the information from an PowerLogic T300 RTU.....	36
Figure 36 – Selecting the information from a Saitel DR (HUe).....	36
Figure 37 – Importing only a type of information.	37
Figure 38 – Sending information to the RTU.	37
Figure 39 – Type of file to generate in the export of the RTU.	37
Figure 40 – Type of file to generate in the export of a Configuration.	38
Figure 41 – Importing information from a file.....	38
Figure 42 – sysLog imported from the RTU.	39
Figure 43 – New configuration for an RTU.....	39
Figure 44 – New configuration for an RTU.....	40
Figure 45 – RTU configuration parameters.	40
Figure 46 – Configuring the IP address.....	41
Figure 47 – Redundant Saitel DR RTU.	42
Figure 48 – Configuring redundant CPUs.	42
Figure 49 – Sending the configuration to redundant CPUs.....	42
Figure 50 – Local acquisition schematic.....	43
Figure 51 – Saitel DR local acquisition.....	43
Figure 52 – Confirmation for automatic addressing.....	43
Figure 53 – Adding one (or several) AB.	44
Figure 54 – Saitel DP local acquisition.....	44
Figure 55 – Adding new I/O modules.	44
Figure 56 – Profibus configuration and acquisition parameters.	45
Figure 57 – Network configuration.....	46
Figure 58 – Configuring networks with redundant CPUs.	46
Figure 59 – Configuring 'Interfaces'.....	47
Figure 60 – IP addresses in the same range.....	47
Figure 61 – Configuring an Ethernet port.	47
Figure 62 – Using IPv6 for the WAN port (only PowerLogic T300).....	48
Figure 63 – Configuring a WIFI network interface.....	48
Figure 64 – Configuring Routers.....	49
Figure 65 – Example of external subnets.	49
Figure 66 – Routes configuration window.....	50
Figure 67 – A new Router for a new subnet.....	50
Figure 68 – Configuring an external network using the default IP and mask.....	50
Figure 69 – Configuring a Firewall.....	50
Figure 70 – Adding rules to the Firewall.....	51

Figure 71 – Hardening ports.....	52
Figure 72 – Configuring Environment variables.....	52
Figure 73 – Configuring modems.....	53
Figure 74 – Configuring a PPP connection.....	53
Figure 75 – Easergy Builder in Configuration mode.....	54
Figure 76 – Contextual menu for Configuration.....	55
Figure 77 – Description of a Device.....	55
Figure 78 – Selecting the type of the new Device.....	56
Figure 79 – Devices and plugins installed in Easergy Builder.....	56
Figure 80 – Configuring communication channels.....	57
Figure 81 – Channel configuration parameters.....	57
Figure 82 – TCP channel configuration.....	58
Figure 83 – UDP channel configuration.....	59
Figure 84 – ASYNC channel configuration.....	59
Figure 85 – Time parameters for modem control.....	61
Figure 86 – New link.....	61
Figure 87 – Synchronization configuration.....	62
Figure 88 – Synchronization server.....	63
Figure 89 – Time configuration.....	63
Figure 90 – Configuring a synchronization device.....	63
Figure 91 – Configuring synchronization with a SNTP server.....	64
Figure 92 – Configuring synchronization with IRIG.....	64
Figure 93 – Configuring synchronization with a GPS.....	65
Figure 94 – Configuring synchronization with a PTP device.....	65
Figure 95 – Configuring the RTU as SNTP server.....	67
Figure 96 – Configuring the RTU as PTP server.....	69
Figure 97 – Configuring the RTU as IRIG server.....	69
Figure 98 – Configuring time zone and daylight.....	70
Figure 99 – Initial environment.....	70
Figure 100 – A new Saitel DR RTU using HUE.....	71
Figure 101 – Including slave modules.....	71
Figure 102 – Configuring interfaces.....	71
Figure 103 – Configuring the connection for Easergy Builder.....	71
Figure 104 – Using IPv6 for connection with the RTU.....	71
Figure 105 – Showing messages on the Log Console.....	72
Figure 106 – Creating a configuration.....	72

Figure 107 – Editing the new configuration	72
Figure 108 – Selecting supervision points to be included in coreDb.....	72
Figure 109 – Configuring local acquisition.....	73
Figure 110 – Points defined in coreDb	73
Figure 111 – Editing and sending an PowerLogic T300 configuration.....	74
Figure 112 – Select data to be sent to the PowerLogic T300.	75
Figure 113 - Cilo Device in Easergy Builder.....	75
Figure 114 – Commands configuration.	76
Figure 115 – Configuring coreDb in Easergy Builder.	79
Figure 116 – coreDb main menu.	79
Figure 117 – Configuring local acquisition.....	80
Figure 118 – Contextual menu for a selected point.....	80
Figure 119 – Contextual menu for a new point.....	80
Figure 120 – Contextual menu for a new point.....	81
Figure 121 – Add point from script.....	81
Figure 122 – Add point from text box.	81
Figure 123 – Add point from separated text.	82
Figure 124 – Transferring a double point in two single points to ISaGRAF®.....	82
Figure 125 – Delete a record in a table.	82
Figure 126 – Search bar of a coreDb table.	83
Figure 127 – Selecting a supervision point as source.	85
Figure 128 – coreDb toolbar.	85
Figure 129 – Use of the field Description	86
Figure 130 – Using Excel® for data importing.....	86
Figure 131 – Data importing process with error reporting.....	87
Figure 132 – Mask to create two simple points from one double	88
Figure 133 – Split in two signals option	89
Figure 134 – Mask to create two simple Modbus points from one double IEC104 point	89
Figure 135 – Quality bits mask	89
Figure 136 – Sharing data with others RTU.	90
Figure 137 – Possible 'Data Publishing Devices' for SM_CPU866e.....	90
Figure 138 – Configuring a redundant RTU.	91
Figure 139 – Configuring a RTU with redundancy.	91
Figure 140 – Supervision points for an PowerLogic T300 RTU.	95
Figure 141 – Supervision points for a SM_CPU866e CPU.	95
Figure 142 – SNMP Devices in Easergy Builder	103

Figure 143 – Configuring a new SNMP Agent.....	104
Figure 144 – Configuring a new OID with the SNMP Points Wizard.....	105
Figure 145 – Example of Formulas in coreDb.....	109
Figure 146 – Creating a Formula Device.....	110
Figure 147 – coreDb signal creation.....	110
Figure 148 – Formula insert.....	110
Figure 149 – Formula Point wizard.....	110
Figure 150 – Formula Point wizard help.....	111
Figure 151 – coreDb formula interaction.	111
Figure 152 – Example of triggers definition.	112
Figure 153 – TEMPO function basic diagram.....	115
Figure 154 – TEMPO function with multiple changes diagram.	115
Figure 155 – TEMPO function with analog input diagram.....	115

Index of Tables

Table 1 – Software in Easergy Builder	4
Table 2 – Symbols	6
Table 3 – Device compatibility for each platform.....	13
Table 4 – coreDb field description.	83
Table 5 – Quality flags for coreDb points.	87
Table 6 – Supervision point for a Saitel DP RTU.	96
Table 7 – Supervision point for a Saitel DR RTU.	98
Table 8 – SNMP points in coreDb.	106

Glossary

A

A: Ampere.

AAP: Automatic Addressing Procedure.

AB: Saitel DR Acquisition Block.

AB_AC: Direct measurements Acquisition Block.

AB_AI: Analog Inputs Acquisition Block.

AB_AO: Analog Outputs Acquisition Block.

AB_DI: Digital Inputs Acquisition Block.

AB_DIDO: Digital Inputs and Outputs Acquisition Blocks.

AB_DO: Digital Outputs Acquisition Blocks.

AB_MIO: Multiple Inputs and Outputs Acquisition Block.

AB_SER: Communication module for expansion.

AC: Alternate Current.

AI: Analog Input.

AO: Analog Output.

ATS: Automatic Transfer Switching.

AWG: American Wire Gauge.

B

Bps: Bits per second.

BC: Boundary clock.

C

°C: Celsius degree.

COM: Communication port.

CPU: Central Processing Unit.

CTS: Clear to Send.

D

DC: Direct Current.

DI: Digital Input.

DIN: Deutsches Institut für Normung.

DO: Digital Output.

DRAM: Dynamic Random-Access Memory.

E

EMC: ElectroMagnetic Compatibility.

EPROM: Erasable Programmable Read Only Memory

F

FTP: File Transfer Protocol.

G

g: Gram.

GPS: Global Positioning System.

H

HMI: Human-Machine Interface.

HU: Head Unit. Saitel DR CPU.

HU_A: Saitel DR Advanced Head Unit.

HU_AF: Saitel DR Advanced Head Unit with acquisition.

HU_B: Saitel DR Basic Head Unit.

HUe: Saitel DR High-Performance Head Unit.

Hz: Hertz.

I

IED: Intelligent Electronic Device.

I/O: Input / Output.

IRIG: Inter Range Instrumentation Group.

IRIG-B: Mode B of the standard IRIG.

ISO 9001: International standard for Quality Systems.

ITB: Intelligent Terminal Block.

K

KB: Kilobyte.

kHz: Kilohertz.

L

LAN: Local Area Network.

LED: Light Emitting Diode.

M

mA: Milliampere.

MHz: Megahertz.

MB: Megabyte.

Mbps: Megabits per second.

m: Meter.

mm: Millimeter.

ms: Millisecond.

N

N/A: Non-Application.

O

OC: Ordinary Clock.

P

PC: Personal Computer.

PPS: Pulses per Second.

PS: Power Supply.

PWR: Power.

R

RAM: Random Access Memory.

RS-232: Communication standard.

RS-485: Multipoint differential Bus.

RTDB: Real Time DataBase.

RTS: Request To Send.

RTU: Remote Terminal Unit.

Rx: Reception

S

s: Second.

SCADA: Supervisory Control And Data Acquisition.

SFTP: Secure File Transfer Protocol.

SNTP: Simple Network Time Protocol.

SRAM: Static Random-Access Memory.

SSH: Secure SHell.

T

TCP/IP: Transmission Control Protocol/Internet Protocol.

TU: Terminal Unit.

Tx: Transmission.

V

VAC: Volt of Alternate Current.

VDC: Volt of Direct Current.

W

W: Watt.

X

XU: Expansion Unit.

Printed in:

Thursday, March 16, 2023

Schneider Electric

C/ Charles Darwin s/n
Parque Científico y Tecnológico de la Cartuja
Seville, Spain

©2023 All rights reserved. The information contained in this document is confidential and is owned by Schneider Electric. It cannot be copied or distributed in any way, unless there is express written authorization by Schneider Electric. Although this information was verified at the time of publication, may be subject to change without notice.

SE-S856-MSS-4.0 03/2023