The main Working Areas for designing in EICASLAB™

The Mission Area

EICAS Automazione S.p.A.
excellence and passion in automatic control design
Via Vincenzo Vela, 27 10128 Torino - ITALY (IT)
Tel. +39 011 56 23 798 +39 011 56 23 088
Fax +39 011 43 60 679
www.eicas.it

TABLE OF CONTENT

- General description of the Mission Area
- The Graphical Mission
- The ANSI C Mission
- The Elementary Missions

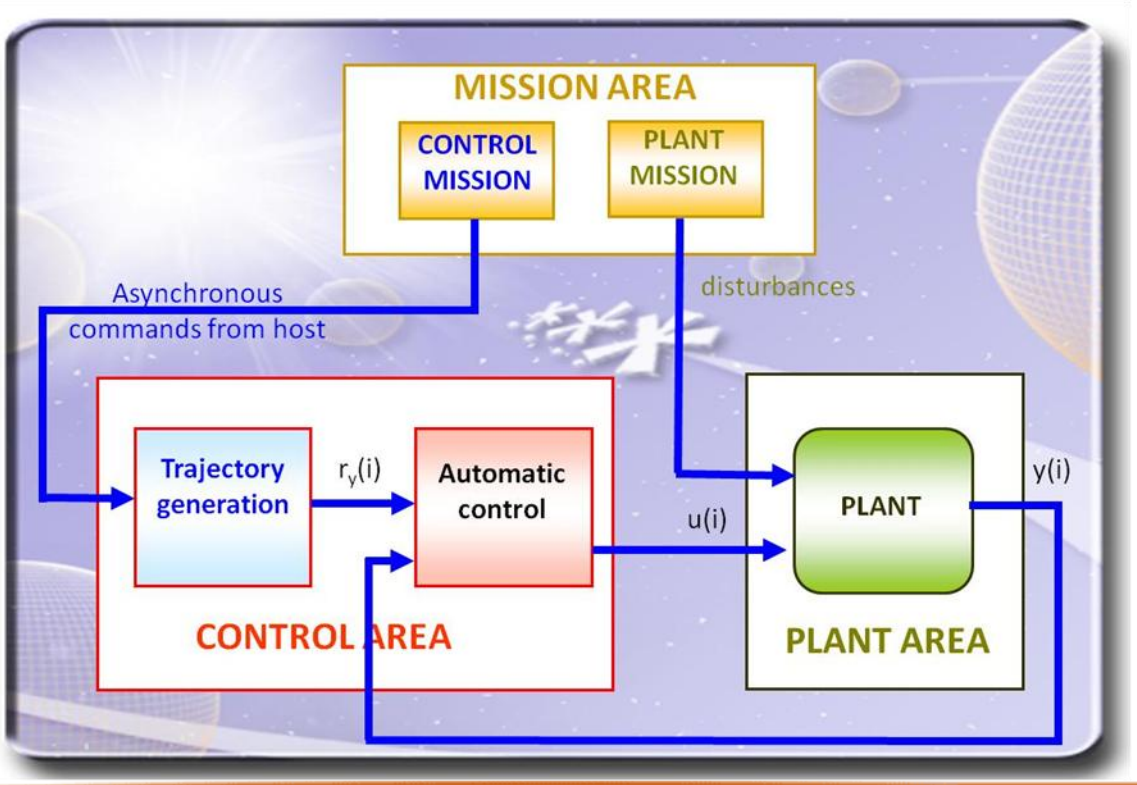
The three main Working Areas

EICASLAB™ has been conceived and developed as a professional software suite supporting the automatic control design and allows to develop and test embedded control system architectures at different hierarchical levels.

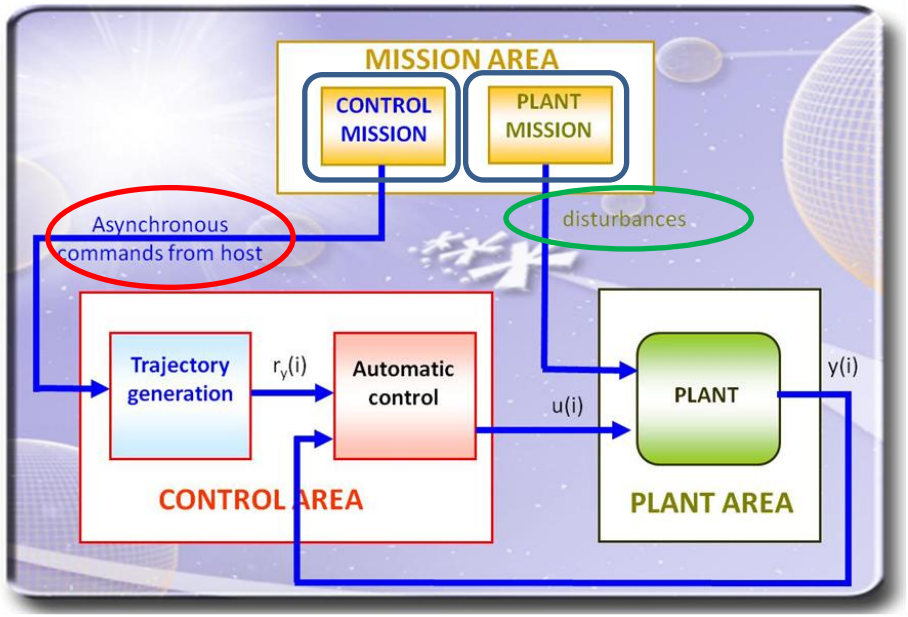
Three main Working Areas are available in EICASLAB:

- the ***Plant Area***,
- the ***Control Area***,
- the ***Mission Area***,

specifically devoted and customized to program the different parts of your project.



The Mission Area concept



The ***Mission Area*** is used to plan the simulated trials.

It is split in two sections, respectively, the ***Plant Mission*** and the ***Control Mission***.

The *Plant Mission* has to generate the disturbance acting on the plant during the simulated trials and to schedule any other event concerning the plant performance, like plant parameters variations.

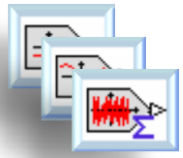
The Control Mission is devoted to generate the host command (which is an external references of high hierarchical level) to be sent to the plant control during the simulated trials.

The Mission Area design and implementation is a key task for the control system design and testing.
EICASLAB gives all the necessary features for designing and implementing the Plant Missions and the Control Missions.

Mission Categories in the Mission Area

The following categories of mission may be programmed in the Mission Area:

the **User Mission**:
it is a Mission entirely programmed by the user.



the **Elementary Missions**:
they are a set of pre-defined signals.

The Programming modes of the Mission Area

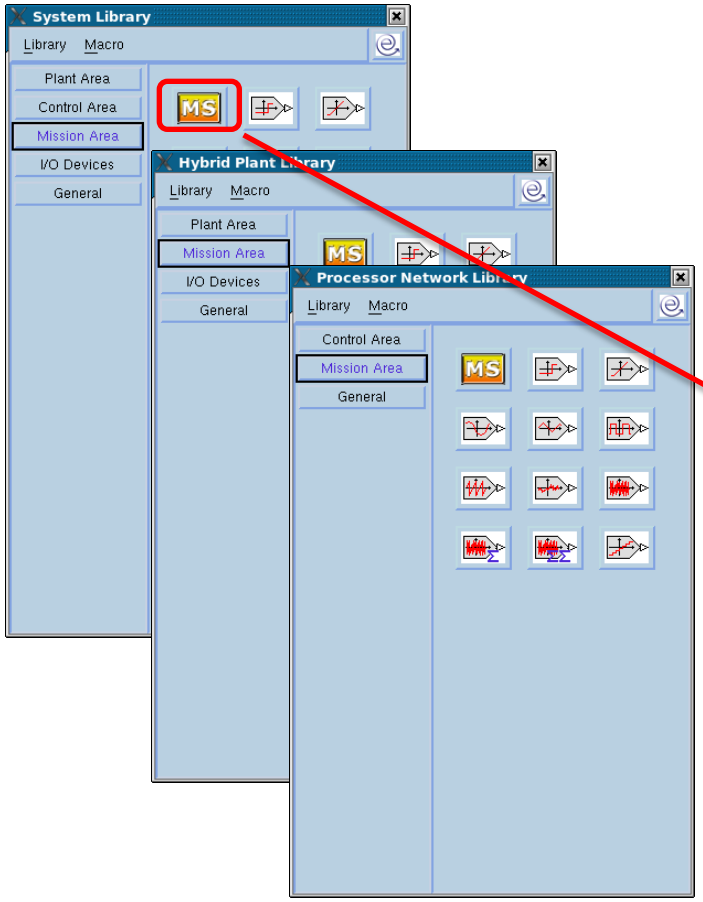
You can develop your Mission:

- **graphically** programming:
  you work on **graphical layouts** equipped with specific and oriented **libraries** that contain a set of suitable pre-defined blocks,

- programming with **ANSI C language**:
  EICASLAB allows an easy programming in ANSI C language
  by means of an open and customizable pre-organized structure
  that allows you to focus just on specific and crucial aspects of the system to be programmed.
  You have at disposal a set of template files and libraries,

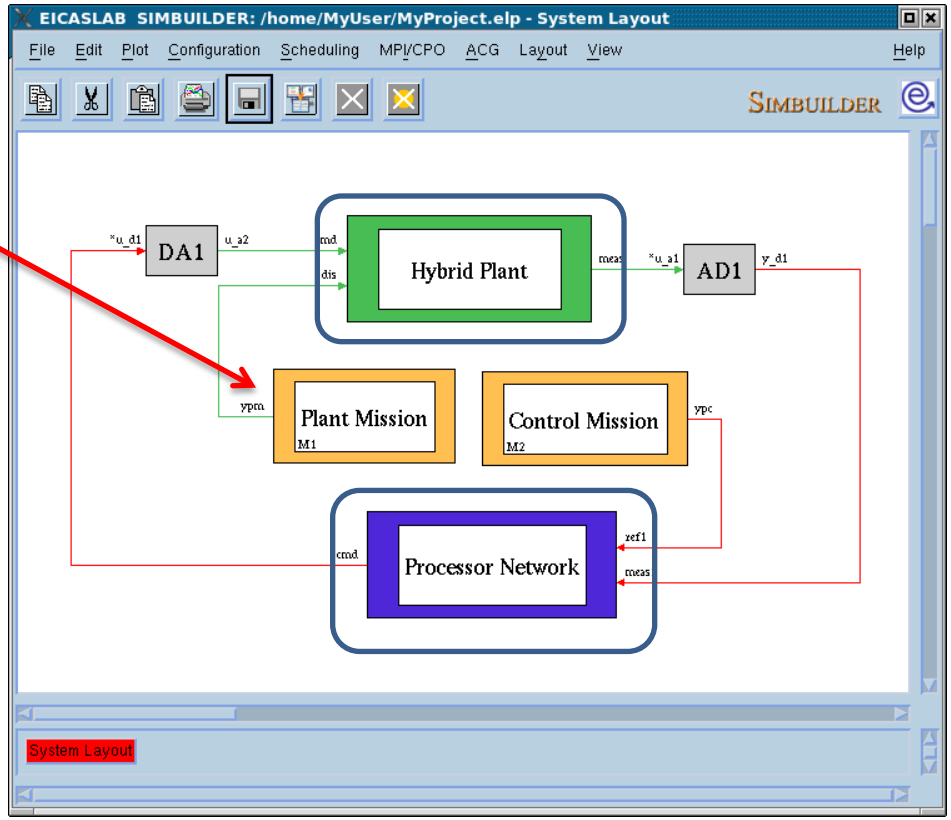
- Using the Elementary Missions.

The Mission Area library



The Mission block can be inserted in:

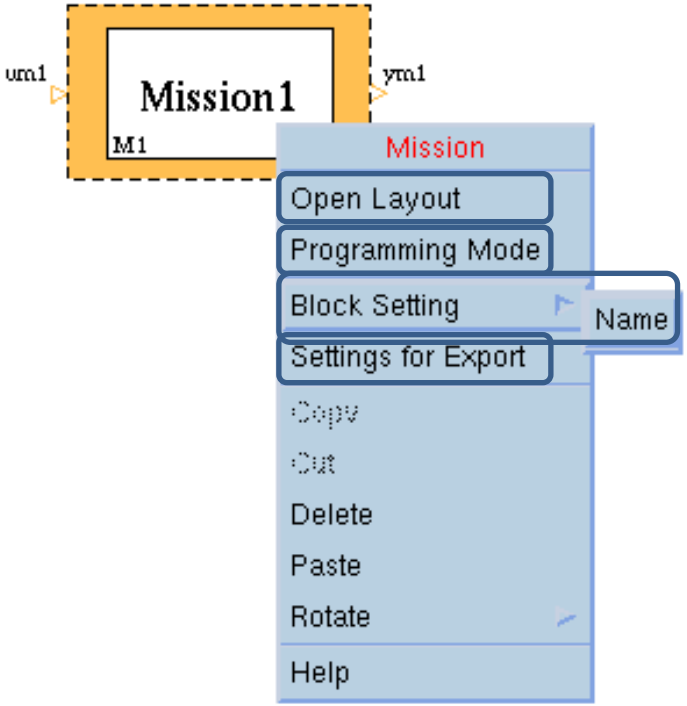
- the System Layout,
- the Hybrid Plant Layout (Plant Mission)
- the Processor Network layout (Control Mission)

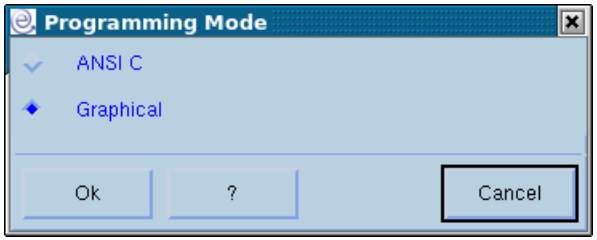


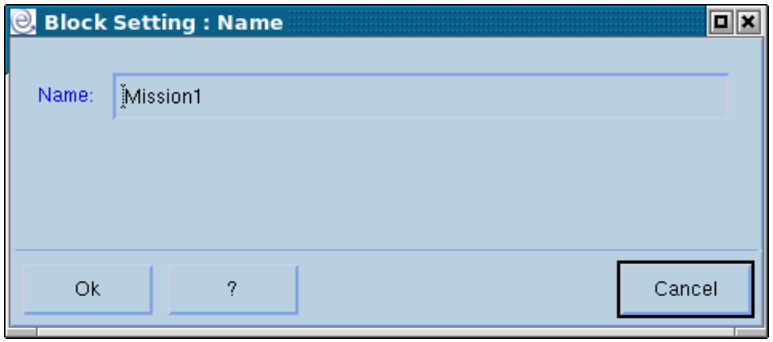
The User Mission

Associated popup menu

The User Mission is by default graphically programmed.

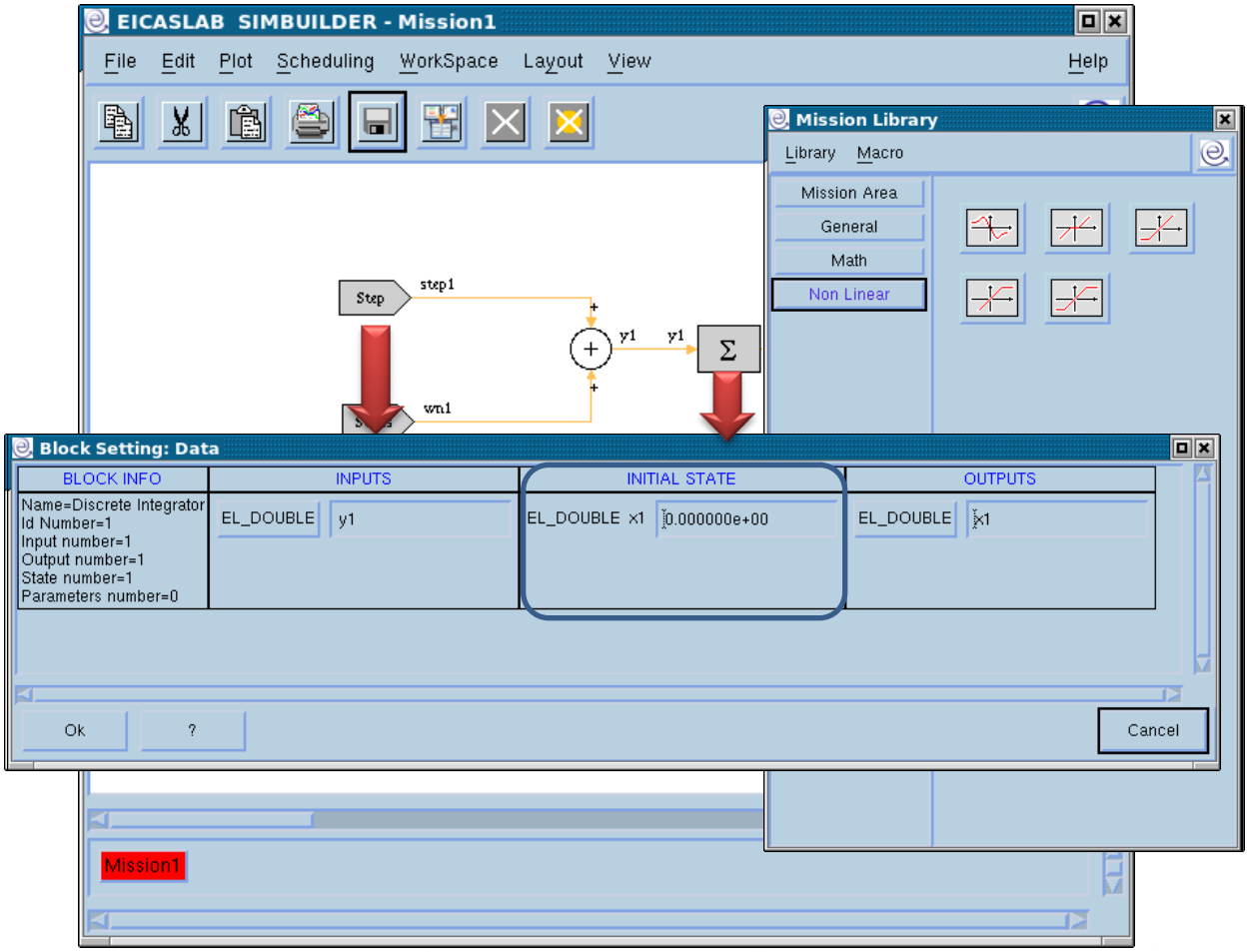






The Mission graphically programmed

The Mission Layout



The Mission Layout allows to graphically program the Mission.

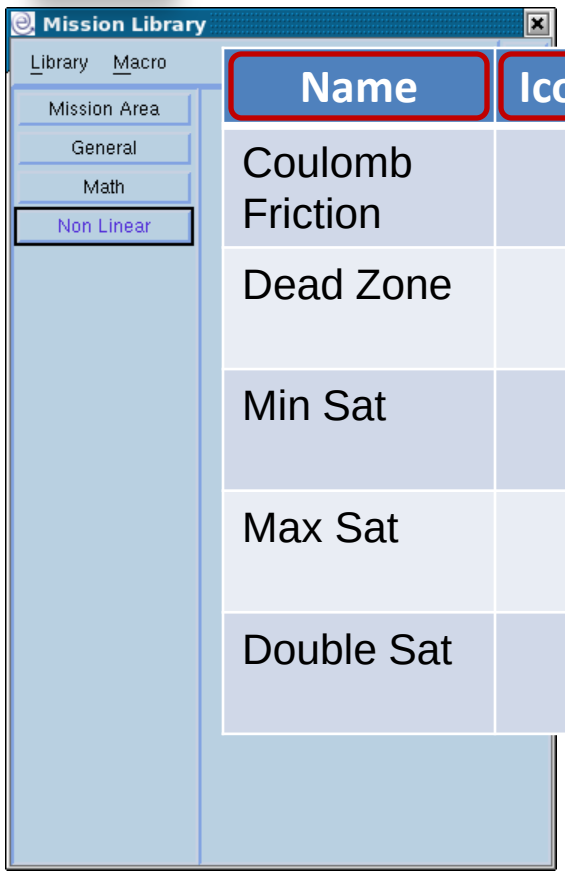
You can implement your Mission by using the blocks available in the Mission Library window,

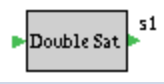
and by setting their:

- outputs,
- parameters,
- initial states (dynamic blocks).

The Mission graphically programmed

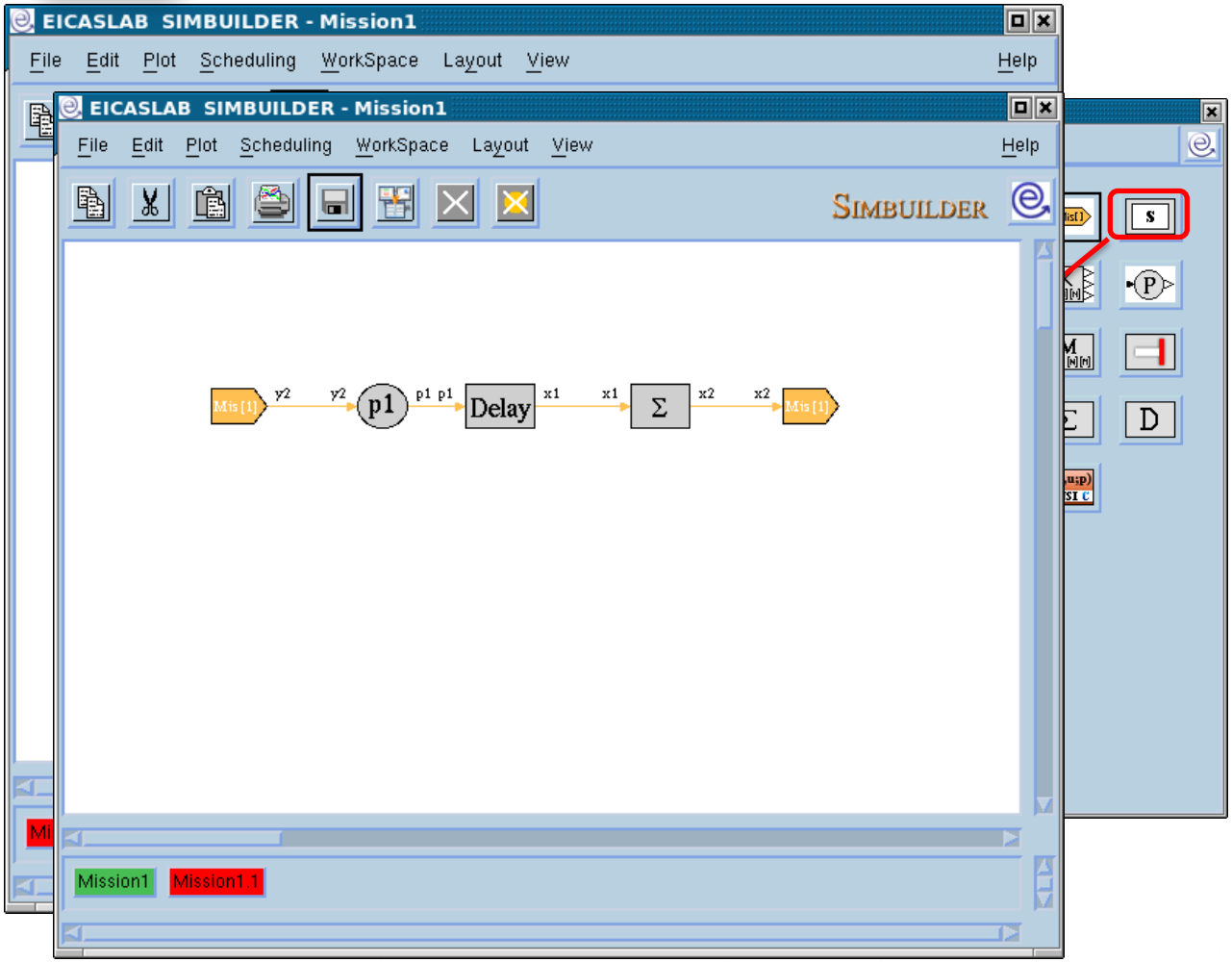
The non-linear library



Name	Icon in library	Block in the layout	Description
Coulomb Friction			Generate output according to a coulomb friction model
Dead Zone			Generate output according to a backlash model
Min Sat			Limit the lower value of a signal
Max Sat			Limit the upper value of a signal
Double Sat			Limit the range of a signal

The Mission graphically programmed

The subsystems

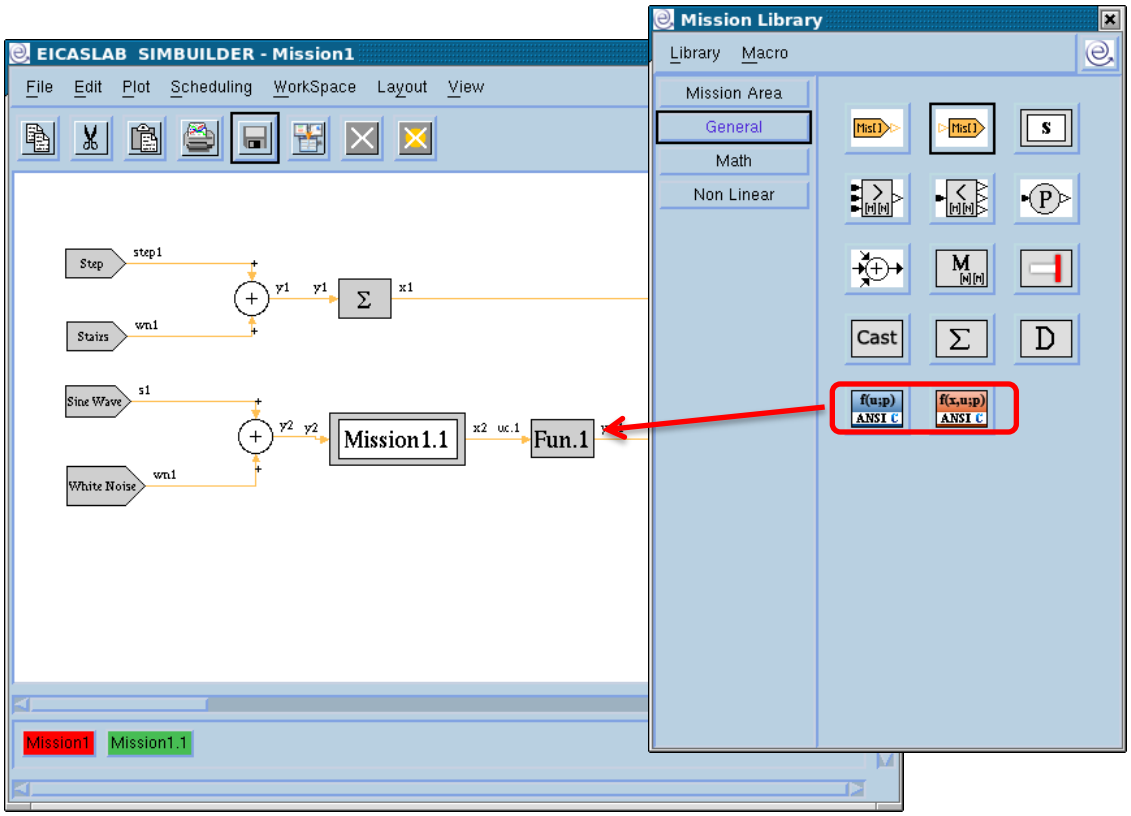


You can simplify the representation of your system by collecting parts of your block diagram in a block called **Subsystem**.

Double clicking on the subsystem opens the *Subsystem* layout, where you can use all the blocks available in the related library.
You can also create other subsystems in order to build a hierarchical block diagram.

The Mission graphically programmed

The ANSI C blocks



It is possible to use special blocks programmable in ANSI C language.

There are two types of blocks, allowing you to program in ANSI C language:

- static functions
  in this case the C block implements the function:
  y= f(u;par);

- dynamic functions
  in this case the C block implements the function:
  y= f(x,u;par);

(having indicated:
y: outputs, u inputs, x: states, par: parameters)

The Mission graphically programmed

The macros

The Mission library window is **customizable** with user blocks called **'macros'**.

The macros are created by the user in order to complete the library according to the user needs.

The macros can be programmed:

- **graphically** (working on the Graphical Macro layout) or
- **in ANSI C language**.

They are then available in the Mission Library window and can be used in the current project.

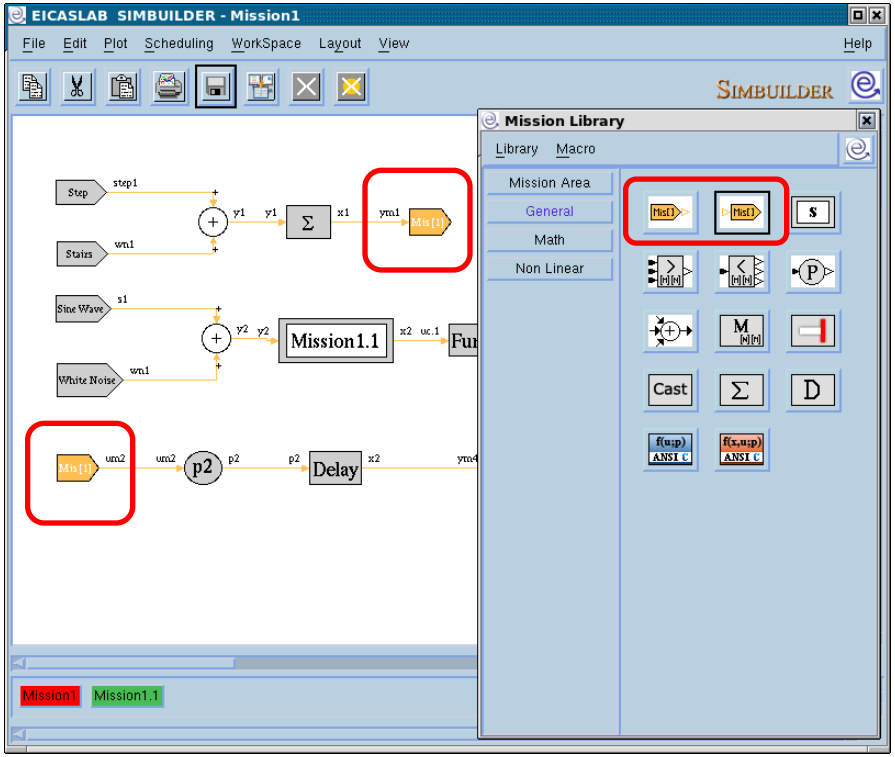
They can also be exported and then used in other projects.

The Mission graphically programmed

The Input/Output variables

In order to define the inputs and the outputs of a graphically programmed block:

Insert inside the graphical layout the input – outputs blocks.



Mis[] Mission Input

Mis[] Mission Output

The Mission programmed with ANSI C language

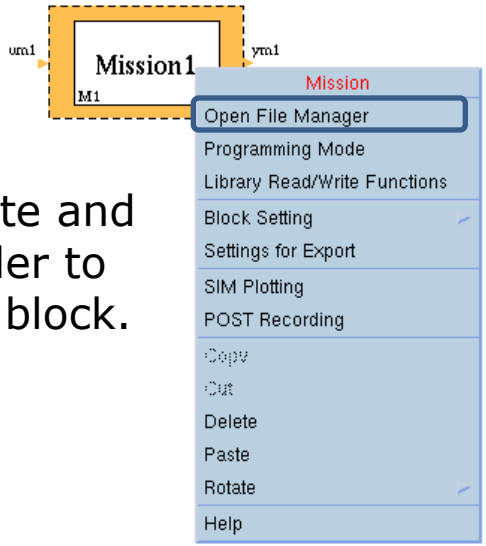
The Mission file manager

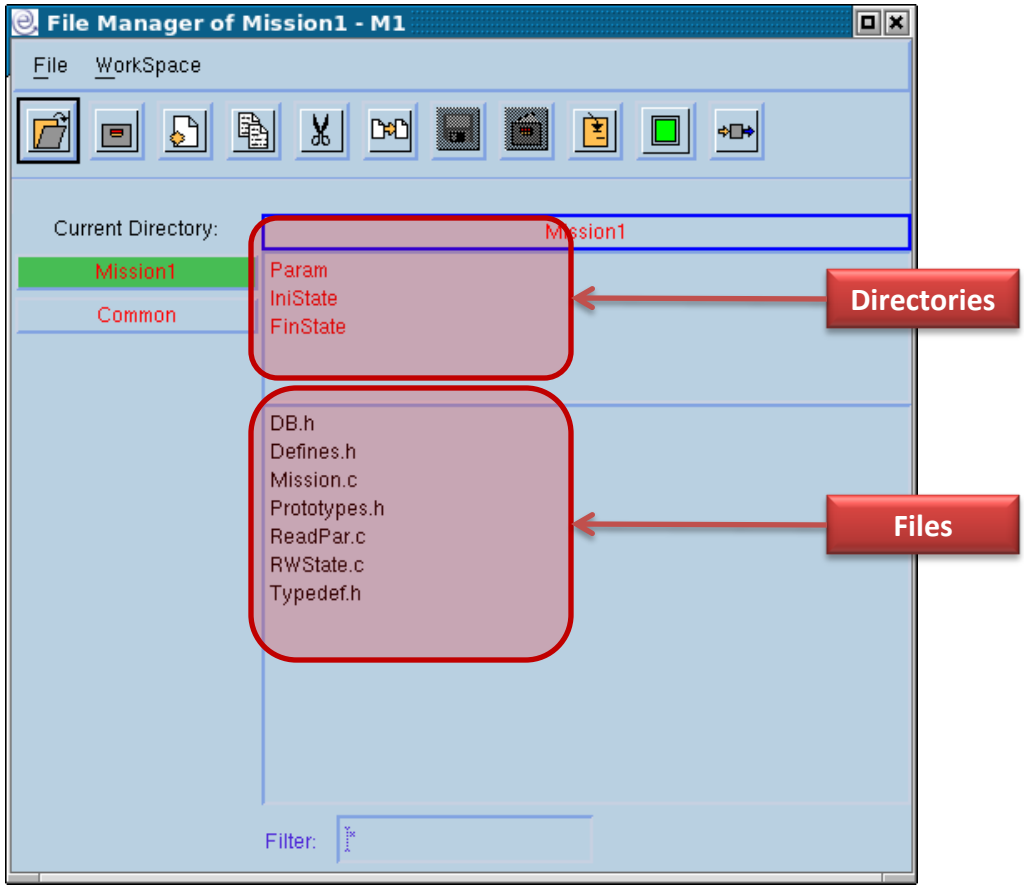
The User Mission programmed with ANSI C language has its own file manager through which it is possible to program the block.

EICASLAB provides a pre-organised structure: a set of template files subdivided in:

- data files,
- header files,
- ANSI C files,

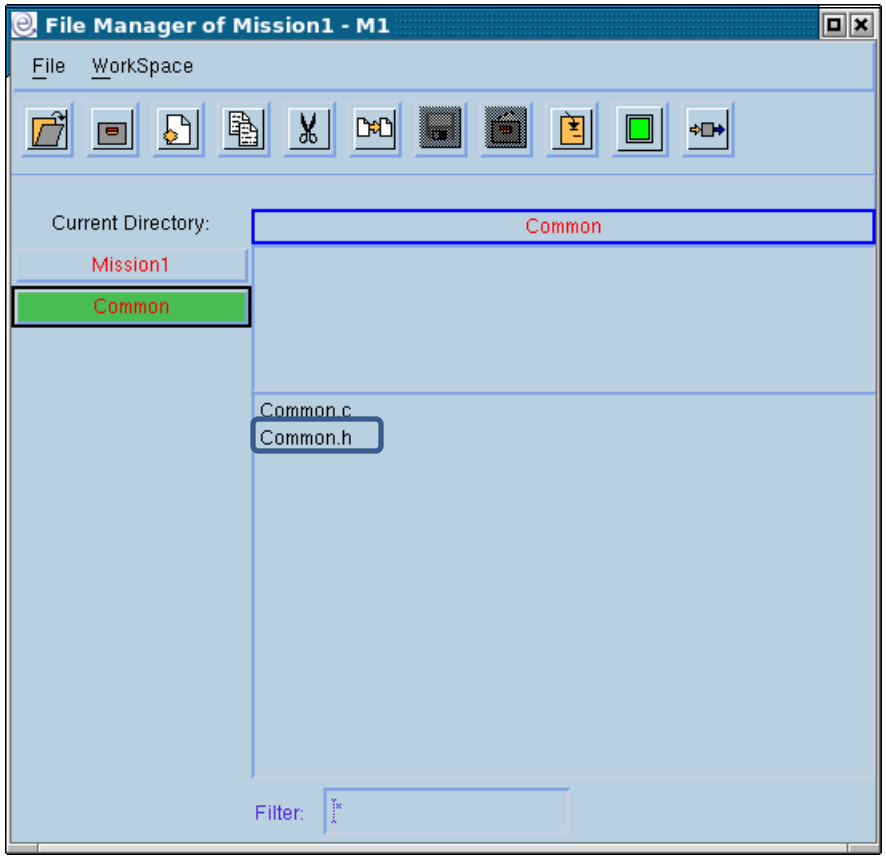
that you can write and customize in order to implement your block.





The Mission programmed with ANSI C language

The header files



Header files of the pre-organised structure that are written by the user.

Defines.h	Definition of user constants
Typedef.h	Definition of user structures
DB.h	Definition / declaration of user variables
Prototypes.h	Declaration of the function prototypes
Common.h	Available for all the blocks programmed in ANSI C

The Mission programmed with ANSI C language

Initialization functions

Name	Description	ANSI C File	Data File
M#_ReadPar	Parameter file reading	ReadPar.c	Mission.par
M#_ReadState	Initial state file reading	RWState.c	Mission.instate
M#_Ini	User initialisation function	Mission.c	---

The Mission programmed with ANSI C language

Execution functions

Name	Description	C File
M#_Exe	Computation of the next state of the Mission as a function of its current state and of its inputs	Mission.c
M#_Out	Computation of the outputs of the Mission	Mission.c

The Mission programmed with ANSI C language

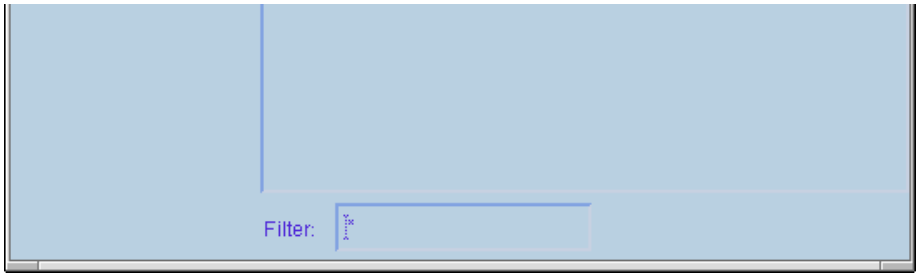
Final functions

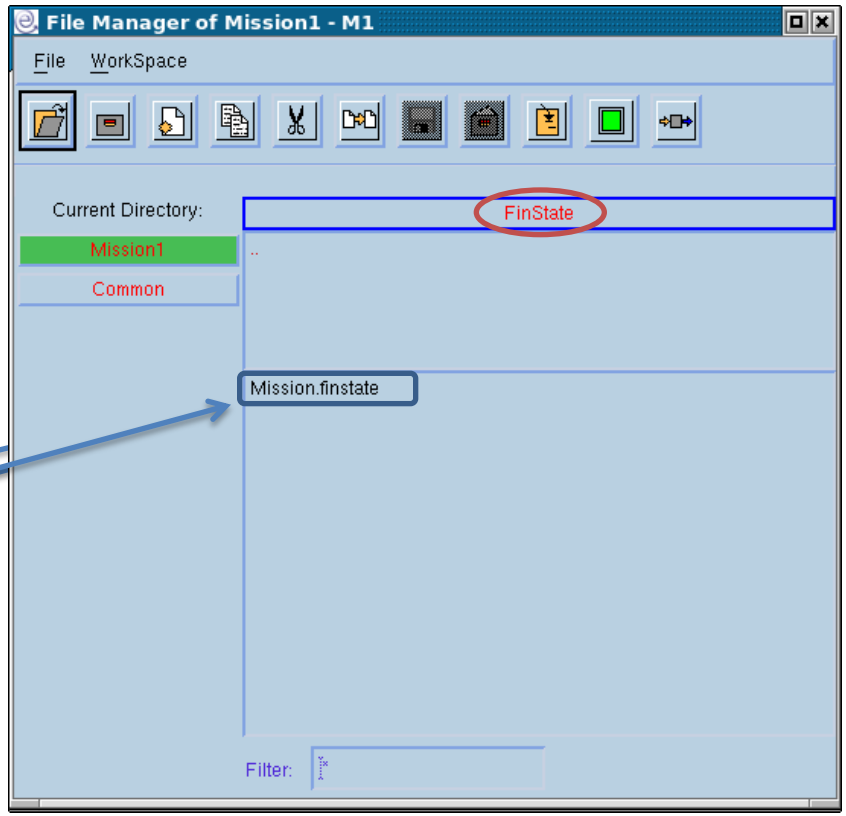
Name	Description	C File	Data File
M#_Fin	User final function	Mission.c	---
M#_WriteState	Final state file writing	RWState.c	Mission.finstate

The Mission programmed with ANSI C language

Data file management

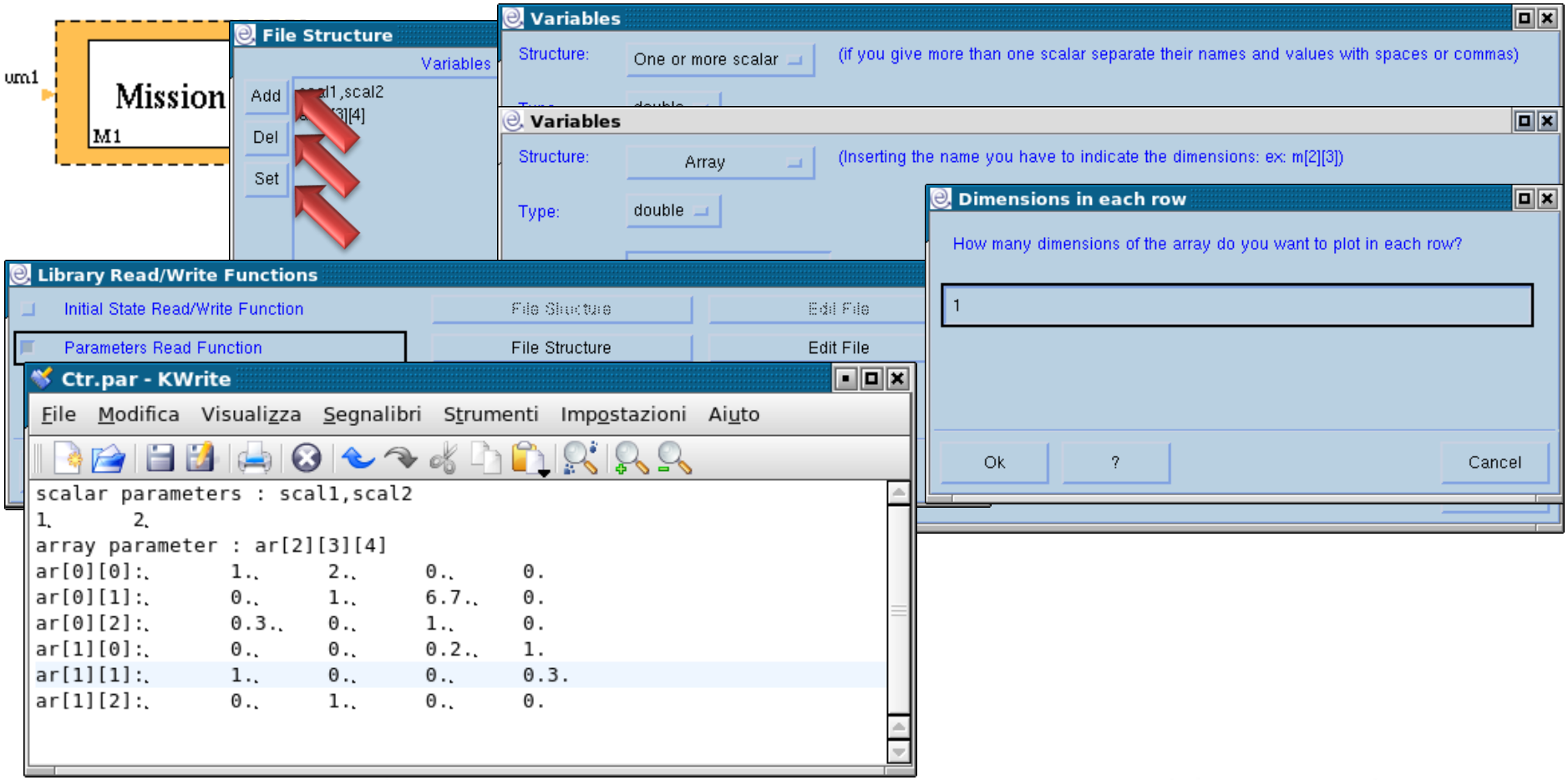
```
/***************************/
void    M1_ReadPar(FILE *fp)
/*
INPUTS:
fp      file pointer to the file Mission.par

OUTPUTS:
initial value of the parameters of the Mission1

OBJECTIVES:
The function can read the parameter set of the mission, from the file Mission.par

All the parameters should be defined in:
        DB.h            database of the Mission1 Module

SCHEDULE:
The function is called by the EICASLAB simulator nucleus,
once at the beginning of the simulation,
before the functions M1_ReadState and M1_Ini.
*/
{


return;
}
/***************************/
```





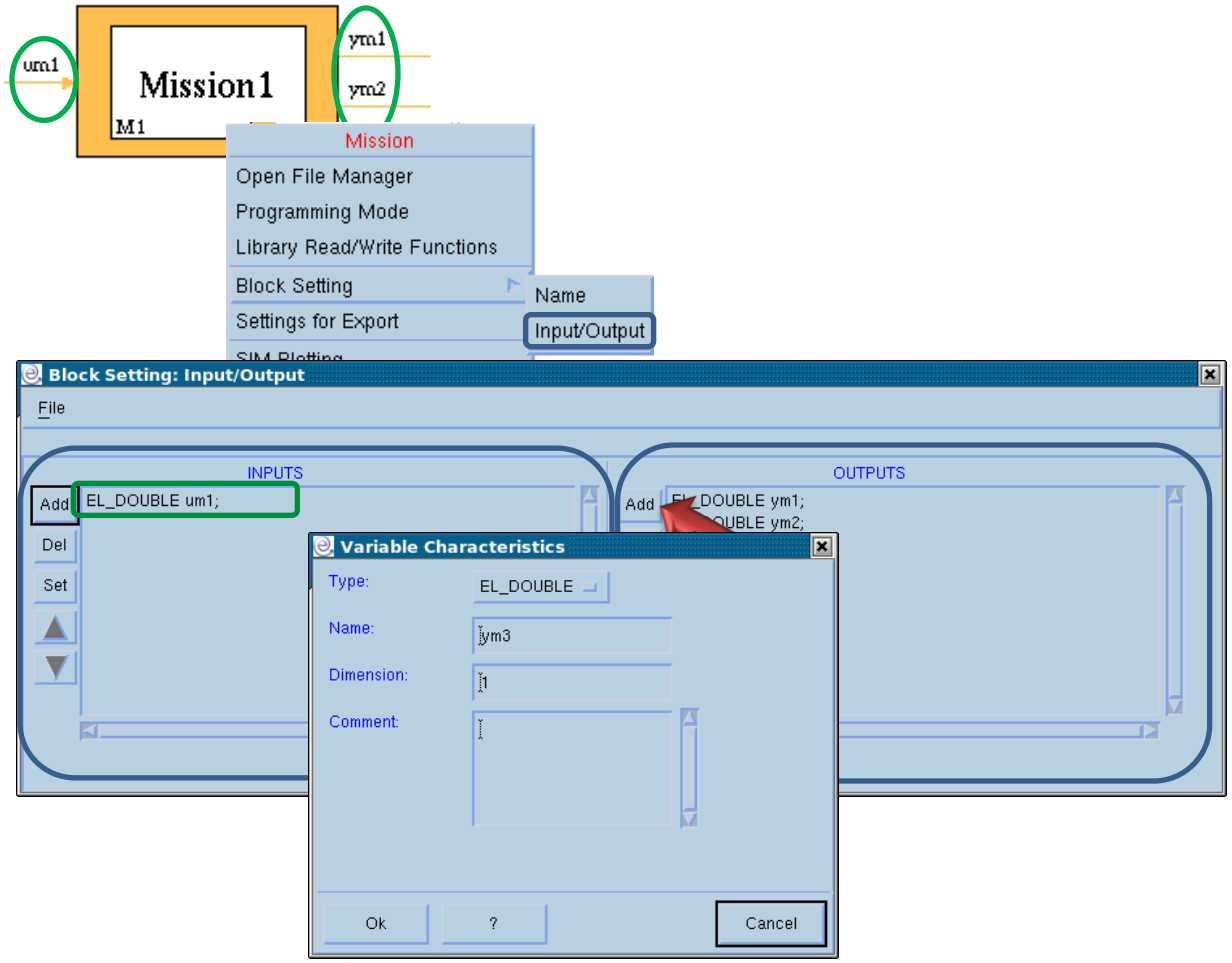
The Mission programmed with ANSI C language

The Library Read/Write Functions



The Mission programmed with ANSI C language

The Input/Output variables



The input/output variables of the block are defined by means of an appropriate window.

The input/output variables are ANSI C variables that can be used in any ANSI C function of the block.

The scheduling of the Mission functions

The Mission functions

The User Mission may be programmed through a set of activities (functions):

- **Graphical** Mission:
  all the functions are entirely created and managed by EICASLAB and depend on the graphical scheme of the Mission Layout and on the data (e.g. parameters, states) directly inserted by the user.

- Mission programmed in **ANSI C**:
  all the functions have a template provided by EICASLAB and are managed by the user.

The scheduling of the Mission functions

Functions categories

The functions belong to three main categories:

Mission Activities

- Initialization functions → called just once at the beginning of the simulation
- Execution functions → periodically scheduled
- Final functions → called just once at the end of the simulation

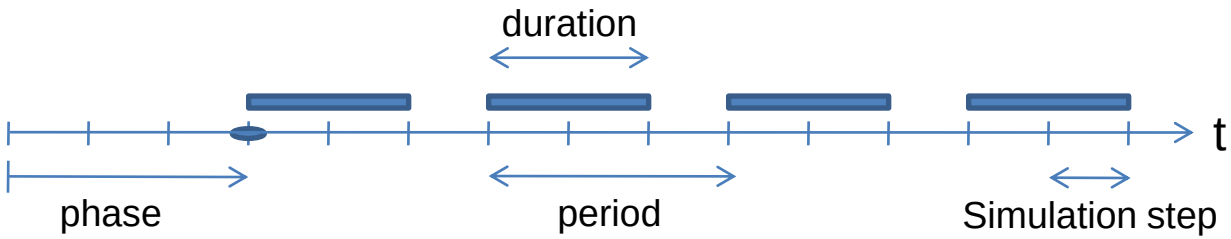
The scheduling of the Mission functions

Scheduling parameters

The user has to fix a **simulation step**,
which represents the time resolution applied in the simulation of the overall project.

The execution functions implement periodic activities characterized by the following scheduling parameters (expressed as a multiple of the simulation step):

- **Phase** time at which they are called for the first time,
- **Period** their sample time interval,
- **Duration** their execution time.



The scheduling of the Mission functions

Initialization functions

- **Graphical** Mission:
  → functions entirely created and managed by EICASLAB,
- Mission programmed in **ANSI C:**
  → functions created by EICASLAB (template) and managed by the user.

The initial functions are called just once at the beginning of the simulation, in the following order:

1) Parameter file reading,

2) Initial state file reading,

3) User initialisation function (Only when programmed in ANSI C language).

The scheduling of the Discrete Plant

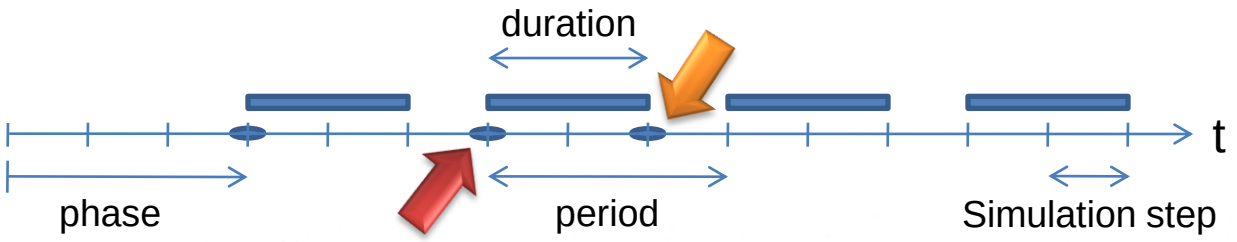
The execution functions

- **Graphical** Mission:
  → functions entirely created and managed by EICASLAB,
- Mission programmed in **ANSI C:**
  → functions created by EICASLAB (template) and managed by the user.

Exe function	Updating of the state of the Mission
Output function	Computation of the outputs of the Mission

To guarantee the correct scheduling of the Missionit is necessary to take into account its **duration**:

Exe function	called when the Mission is scheduled (considering its **phase** and **period**),
Output function	called with the same period of the *Exe function* but with a delay equal to the **duration** of the Mission in order to provide the outputs when they are expected



The scheduling of the Discrete Plant functions

Final functions

- **Graphical** Mission:
  → functions entirely created and managed by EICASLAB,
- Mission programmed in **ANSI C:**
  → functions created by EICASLAB (template) and managed by the user.

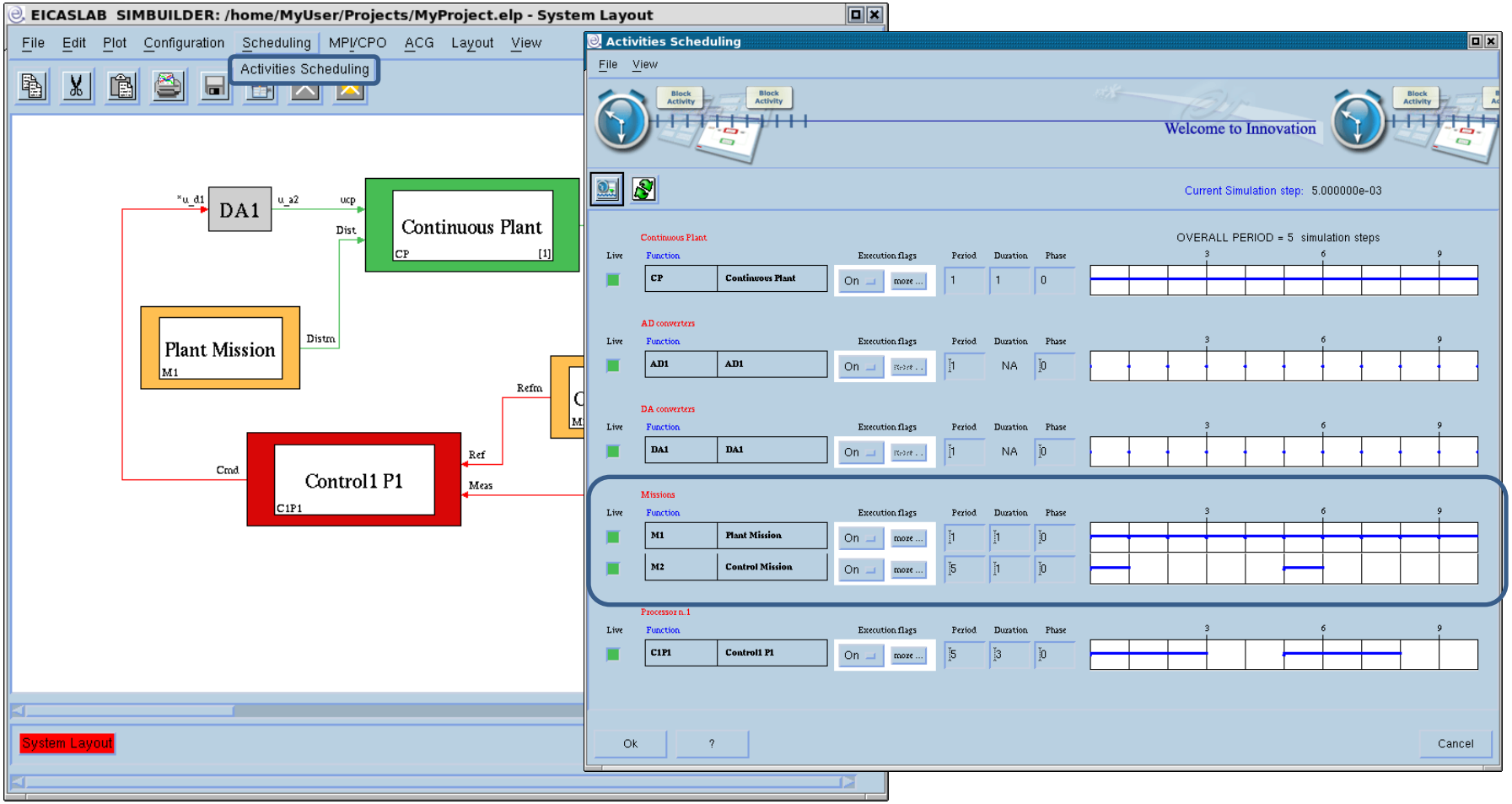
The final functions are called just once at the end of the simulation in the following order:

1) User final function (Only when programmed in ANSI C language),

2) Final state file writing.

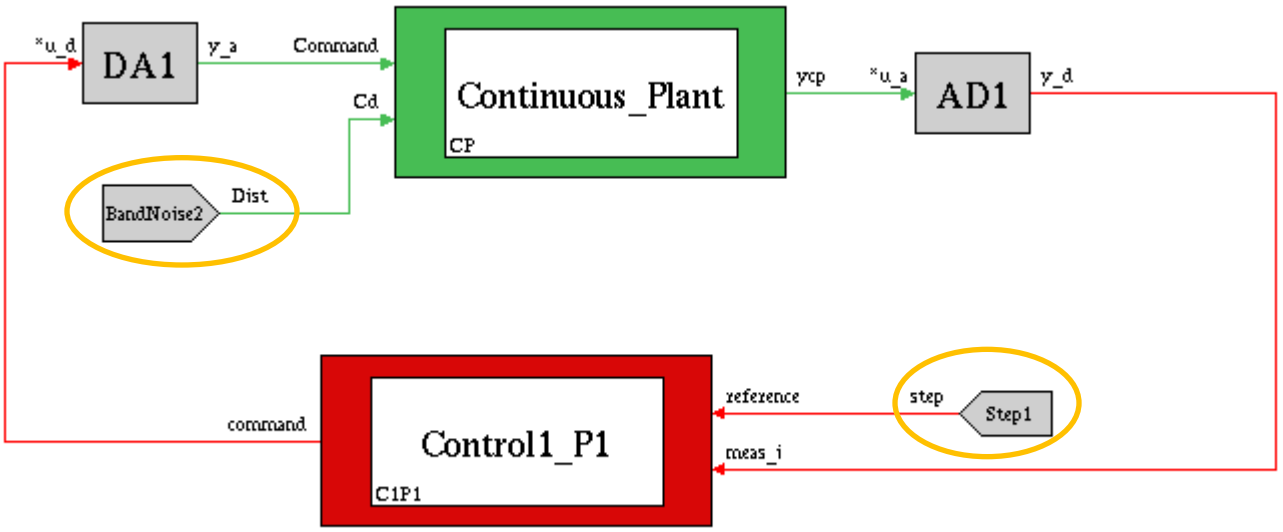
The scheduling of the Mission

How to set the scheduling



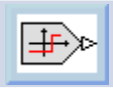
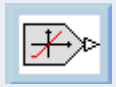
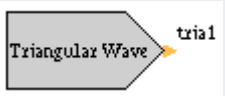
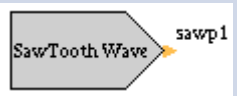
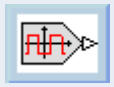
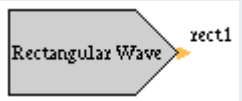
The Elementary Missions

The Elementary Missions are a set of pre-defined signals.
They are represented by blocks that do not have any inputs and have one output:



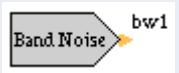
The Elementary Missions

Basic functions

Name	Icon in library	Block in the layout	Description
Step			Generates a step function
Ramp			Generates a constantly increasing or decreasing signal
Sin Wave			Generates a sine wave
Triangular Wave			Generates a triangular wave
SawTooth Wave			Generates a saw tooth wave
Rectangular Wave			Generates a rectangular wave
Stairs			Generates stair wave

The Elementary Missions

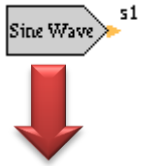
Noise functions

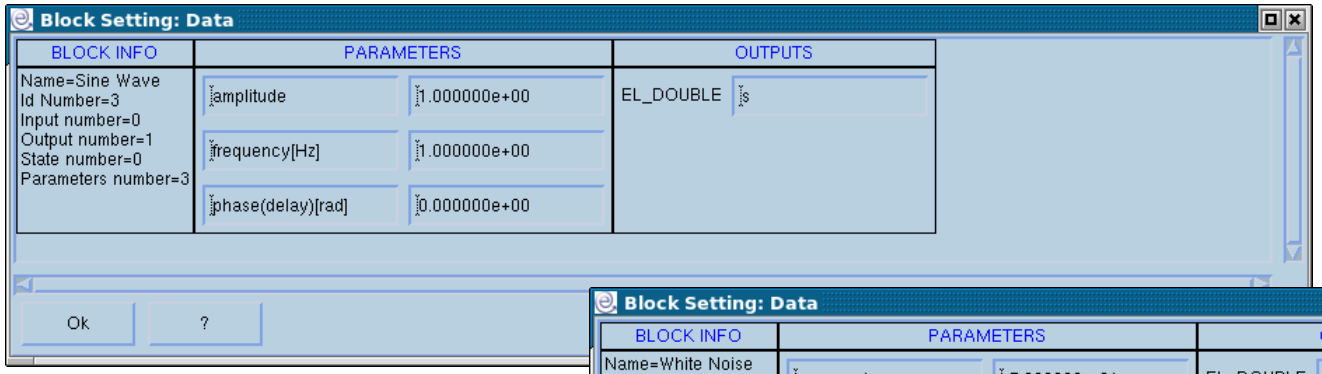
Name	Icon in library	Block in the layout	Description
White Noise			Generates a white noise
Band Noise			White noise (W) filtered by a discrete first order filter
IW Noise			Generates a simple summation of white noises (W).
I2W Noise			Generates a double summation of white noises (W).

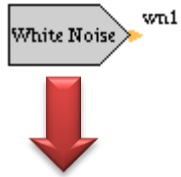
The Elementary Missions

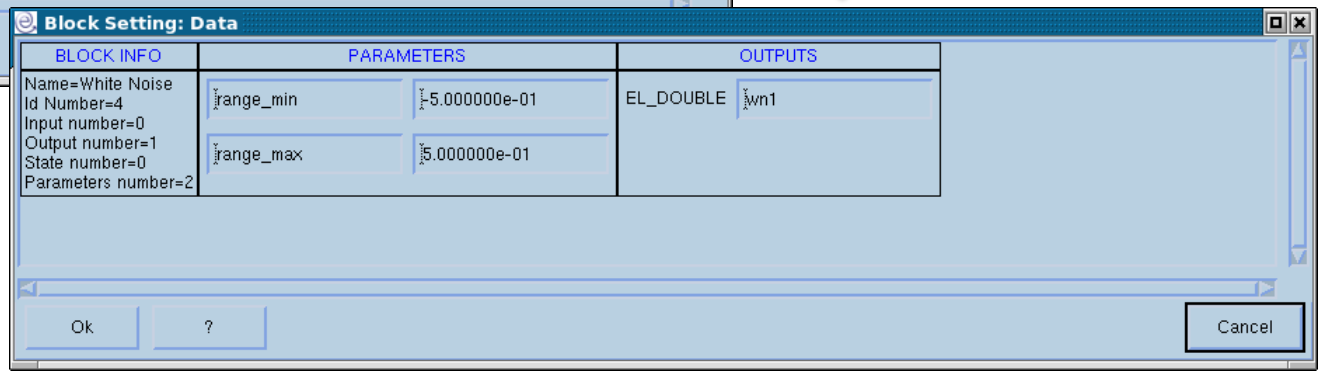
The parameters of the Elementary Missions

Any instance of any Elementary Mission has its own parameters.
By central clicking on the instance you can view and modify the parameters.



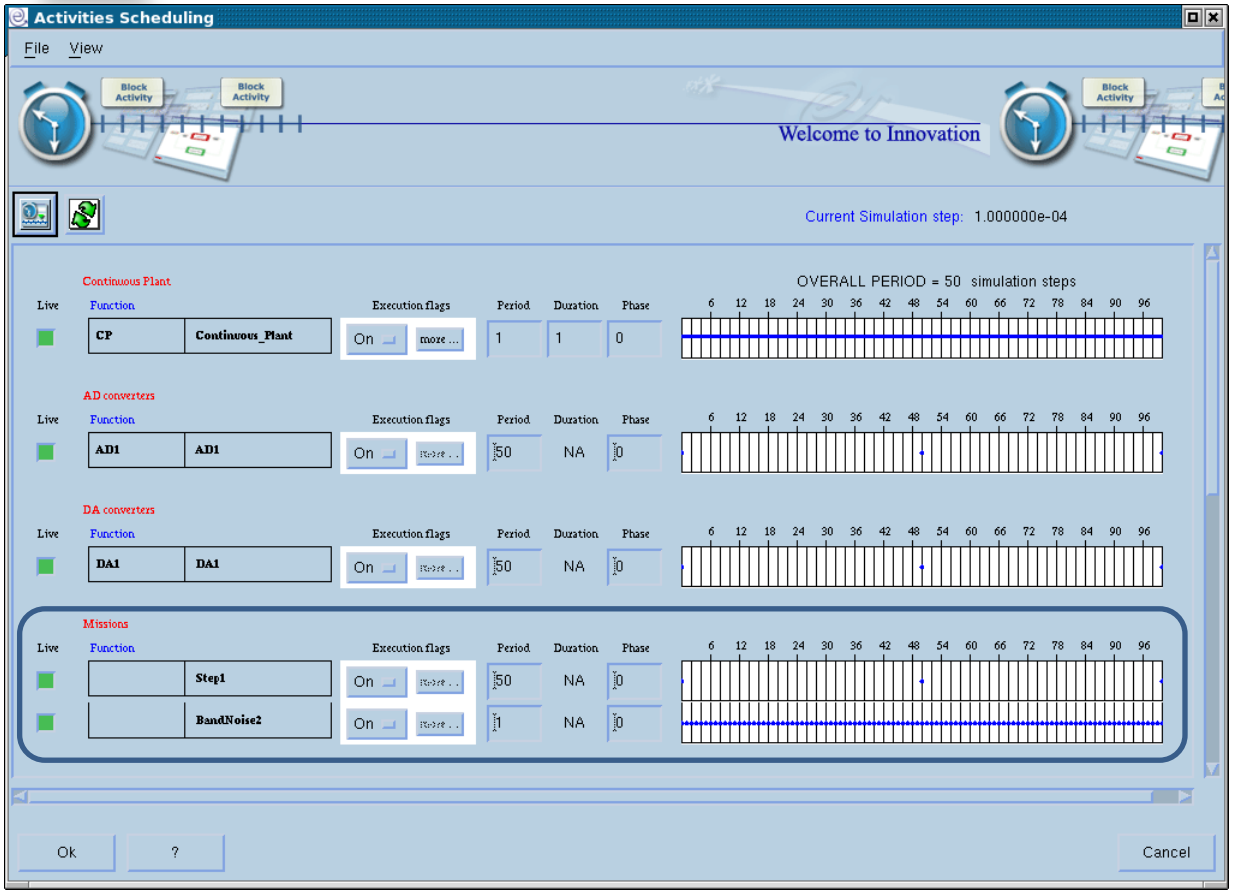






The Elementary Missions

The scheduling



The elementary missions provide a signal given by one EICASLAB function which is periodically scheduled.

EICASLAB™

The Professional Software Suite
for Automatic Control Design
and Forecasting

for Linux & for Windows

www.eicaslab.com

Welcome to Innovation