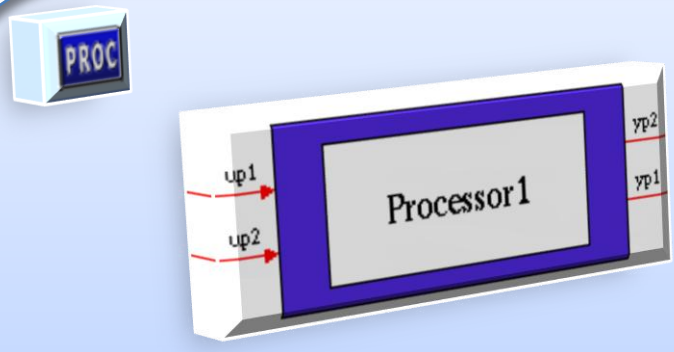
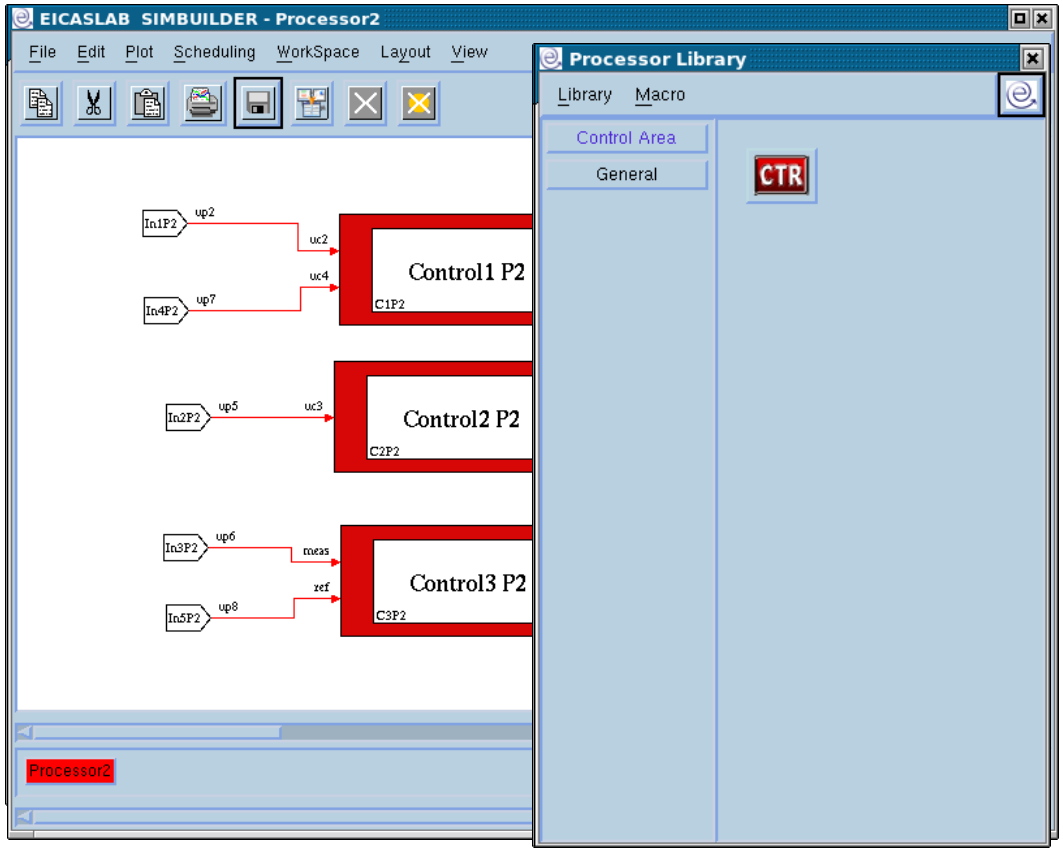
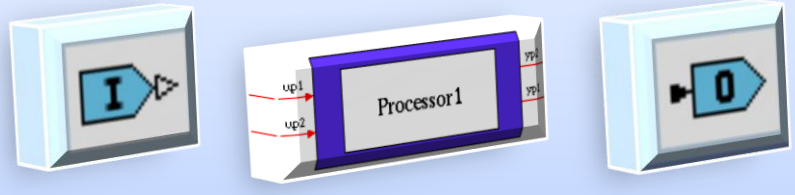


The Processor block in EICASLAB

The Processor Block



The Processor Block represents the final target in which your software will be installed.
It collects one or more schedulable control functions running on a single processor (mono or multicore).



Special Blocks inside the processor:

The Input/Output Blocks

The Processor Input/Output Blocks

Concept

The **Processor Inputs/Output Blocks** represent the hardware interface of the Processor.

The Processor Input Block represents the interrupt activities that receive and process with a suitable scheduling the inputs coming in the processor in which the block is inserted.

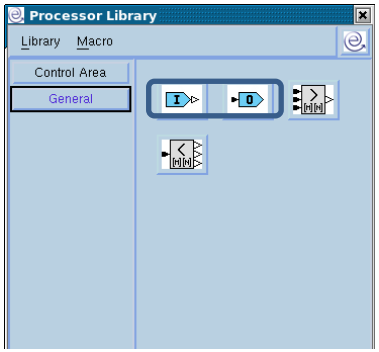
The Processor Output Block represents the interrupt activities that receive and process with a suitable scheduling the outputs of the processor in which the block is inserted.

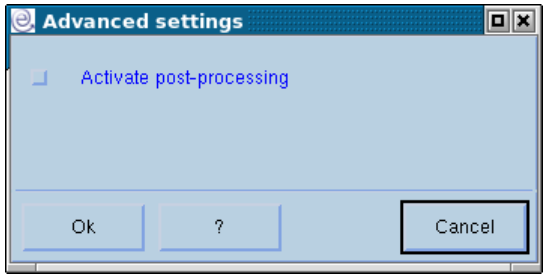
The following two programming options for the Processor Input/Output Blocks are available:

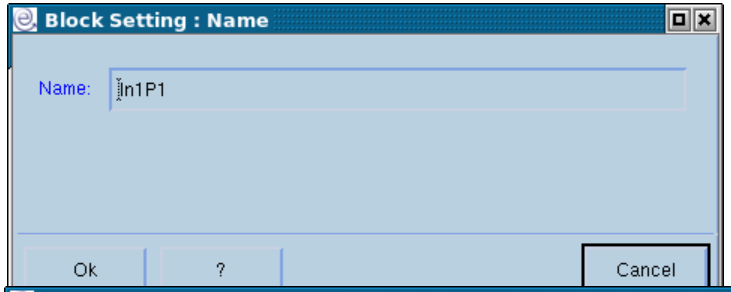
- default,
- post-processing.

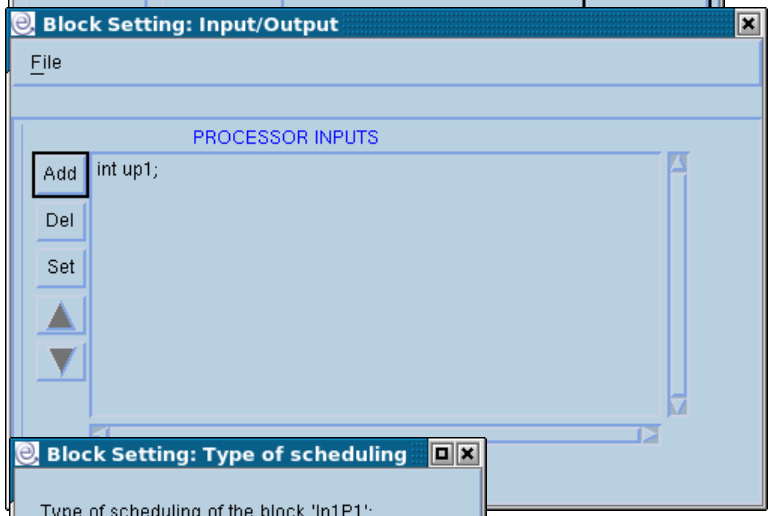
The Processor Input/Output Blocks

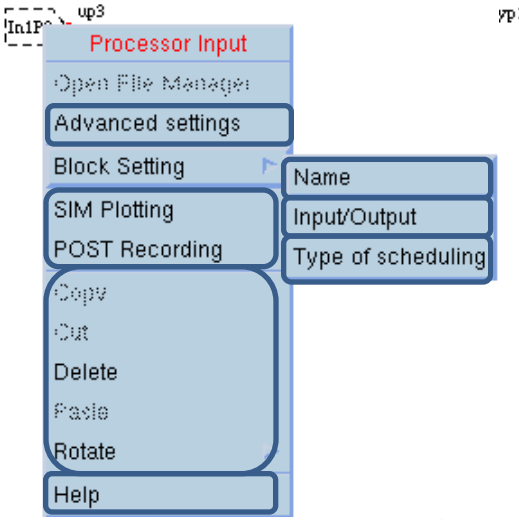
Associated popup menu

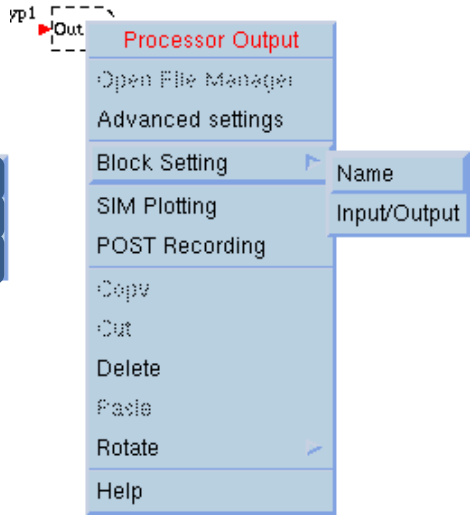


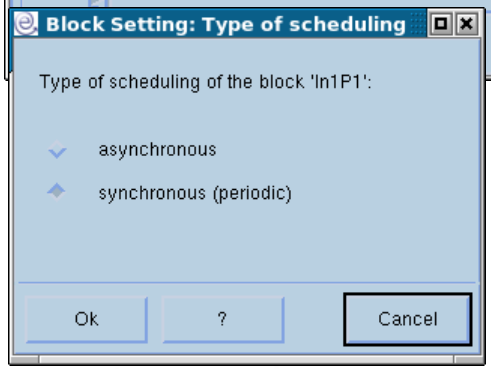


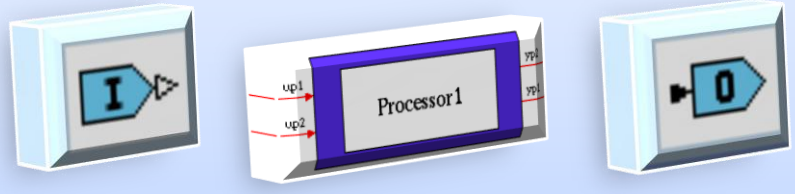












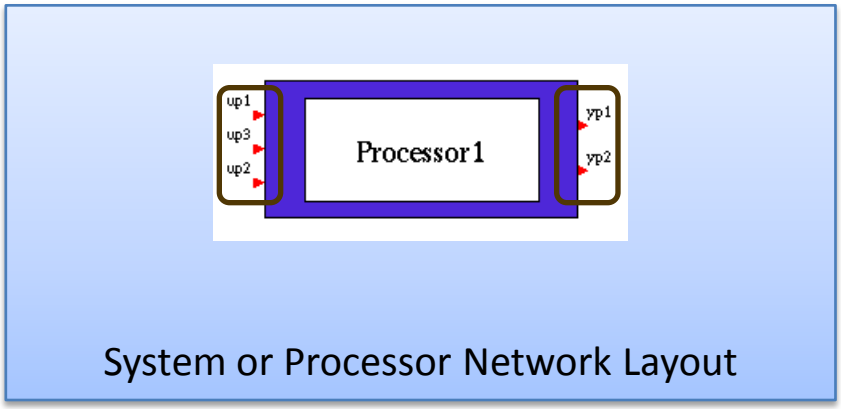
Special Blocks inside the processor:

Input/Output Blocks: default programming mode

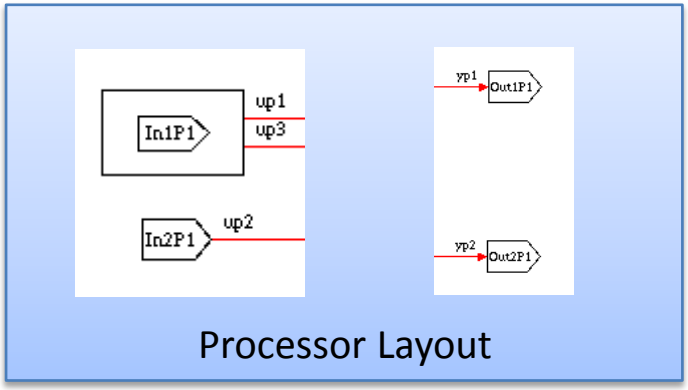
The Processor Input/Output Blocks: default programming mode

Inputs and Outputs of a Processor

You can set the Input and Output variables of a Processor by inserting, in the Processor Layout, the Processor Input/Output blocks and by setting their dimension.



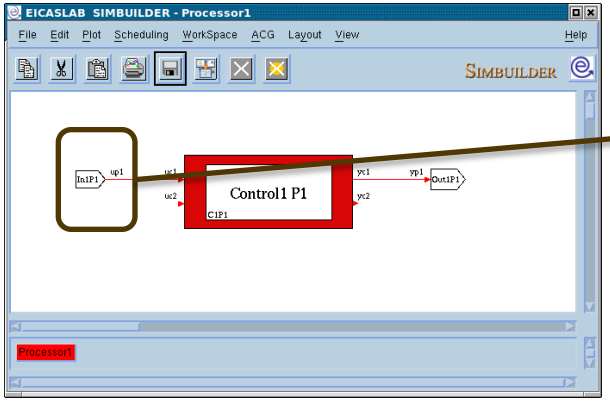
System or Processor Network Layout



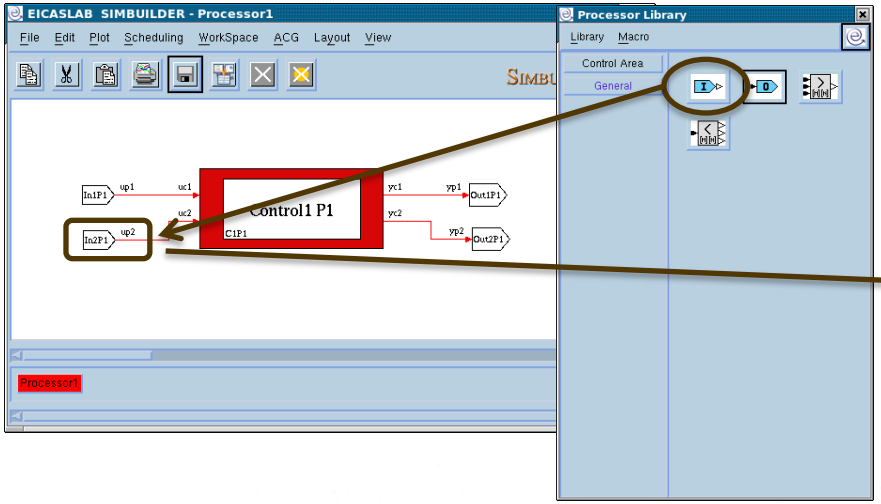
Processor Layout

The Processor Input/Output Blocks: default programming mode

Inserting a Processor Input Block in the Processor Layout



By inserting a Processor Input Block you introduce a new input in the Processor Block.

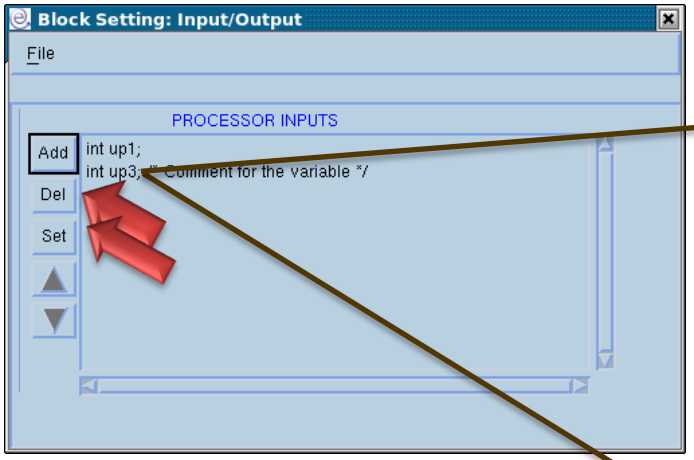


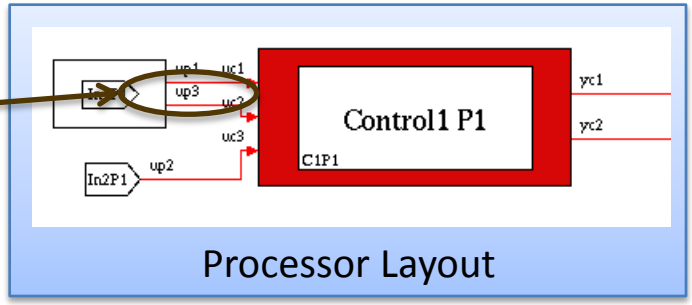
The Processor Input/Output Blocks: default programming mode

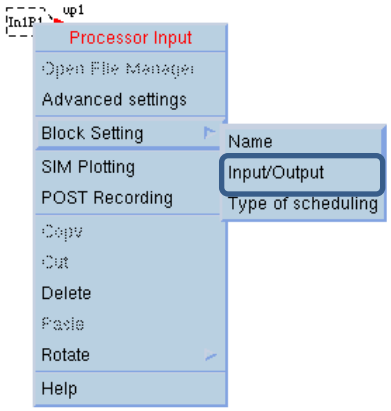
Multi-dimensional Processor Input Blocks

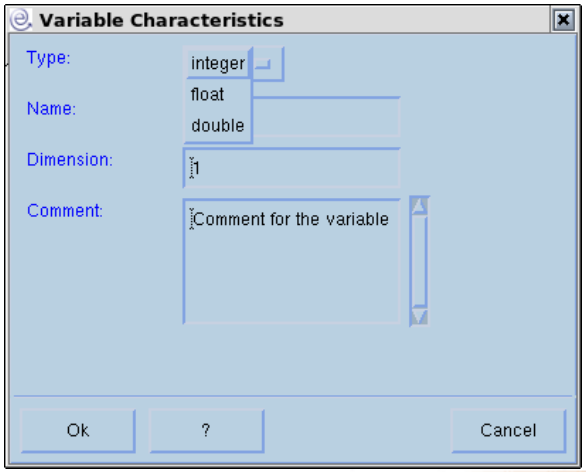
It is also possible to modify the dimension of a Processor Input/Output variables of each Processor Input:

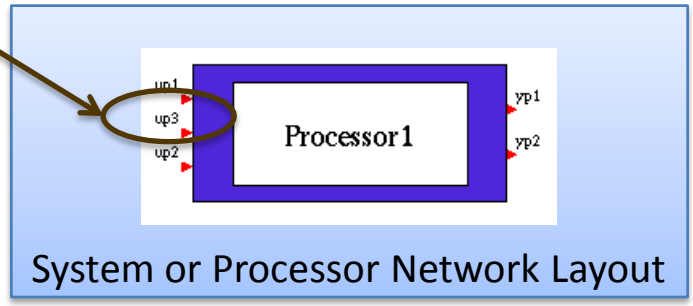
The input/output variables of the block are defined by means of a specific window.







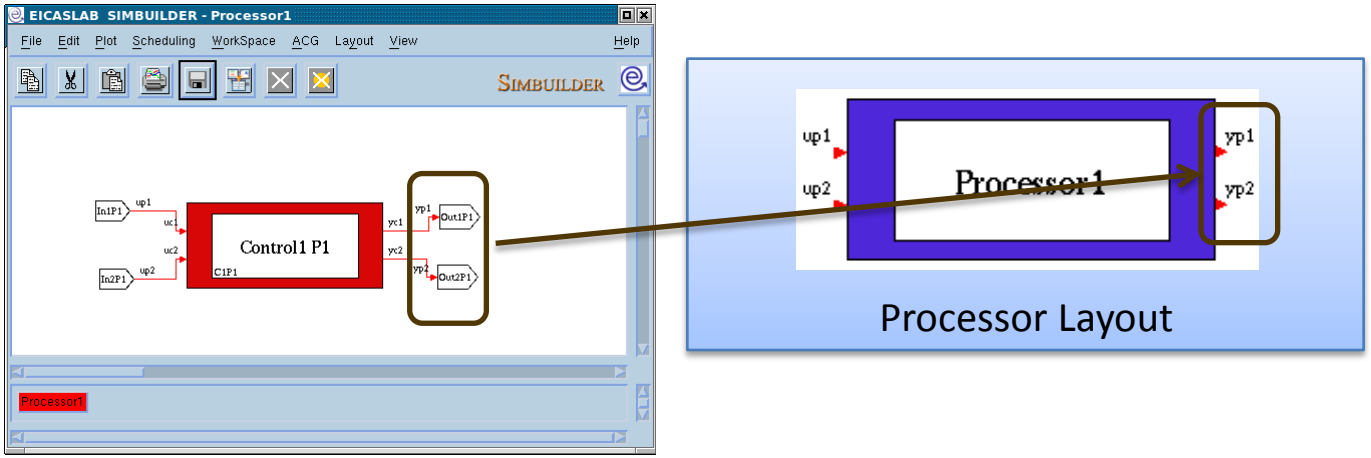




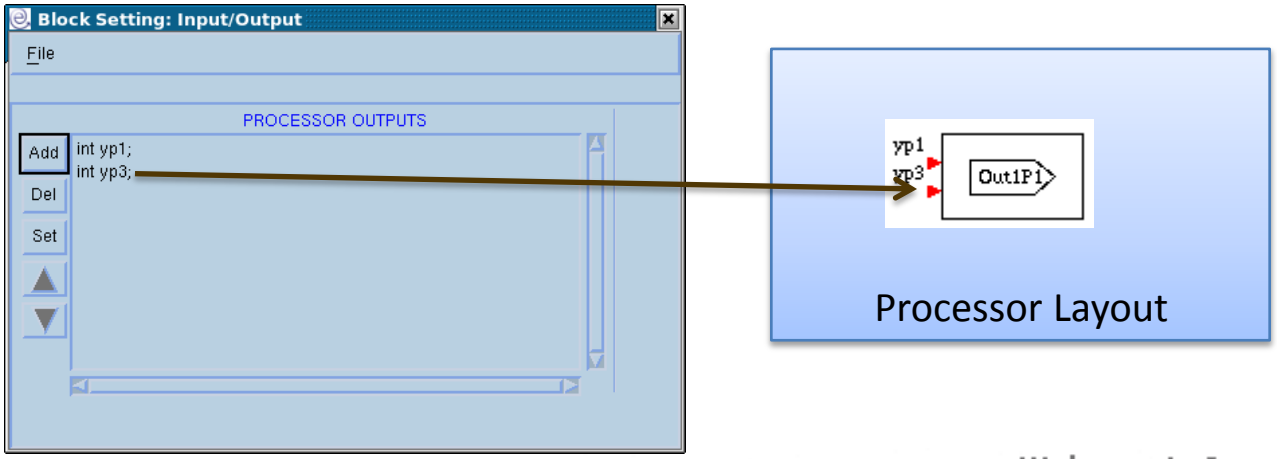
The Processor Input/Output Blocks: default programming mode

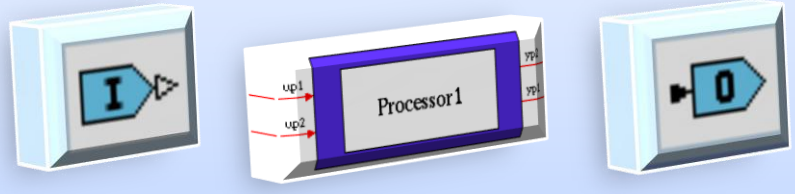
Setting of the Outputs of the Processor

By inserting a Processor Output Block, you introduce a new output in the Processor Block.



It is also possible to modify the Input/Output variables of each Processor Output Block:





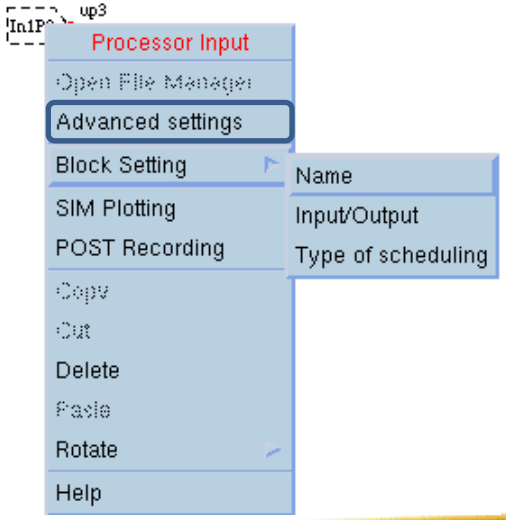
Special Blocks inside the processor:

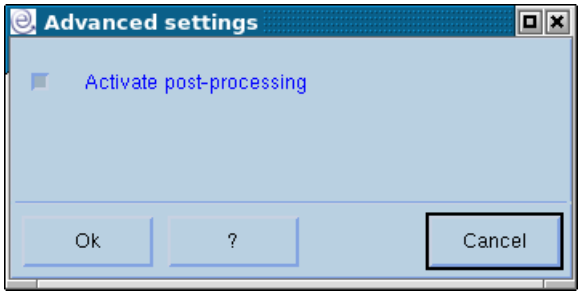
The Input/Output Blocks with post-processing

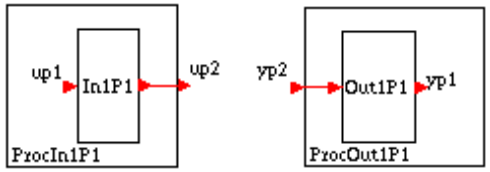
The Processor Input/Output Blocks with post-processing

The post-processing

When a Processor receives data, before making them available to its internal Control functions, it is often necessary to perform a processing of the data (e.g. to unpack them and to manipulate them).
It is also often necessary to process the output data of a Processor.
The post-processing programming option is then available for the Processor Input/Output Blocks.
A set of pre-defined ANSI C files, accessible through a file manager, allow you to perform the post-processing.



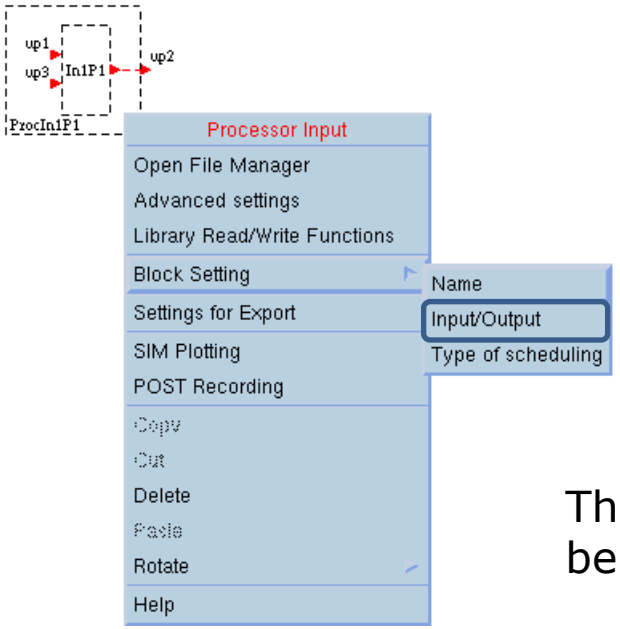


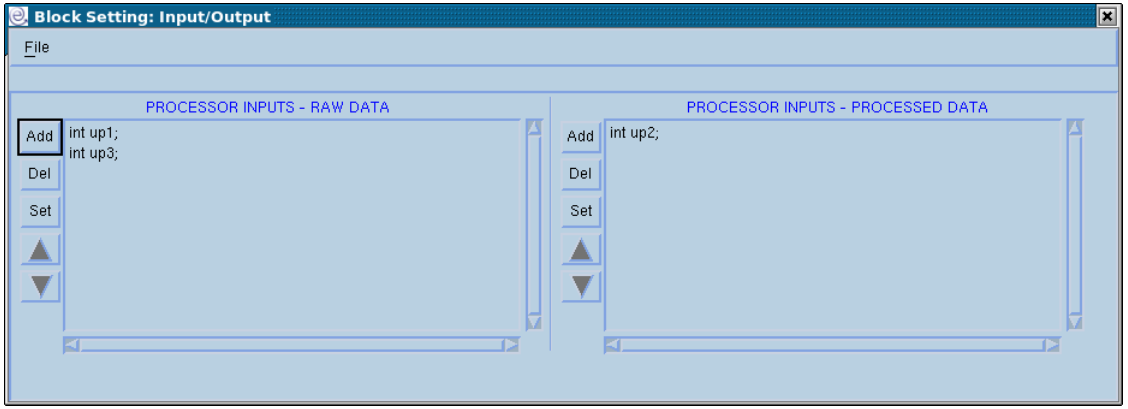


The Processor Input/Output Blocks with post-processing

The Input/Output window

The Processor Input Block with post-processing has at disposal a function for processing the 'raw' input data of the Processor.
By means of the Input/Output window you can define the raw data and the processed data.



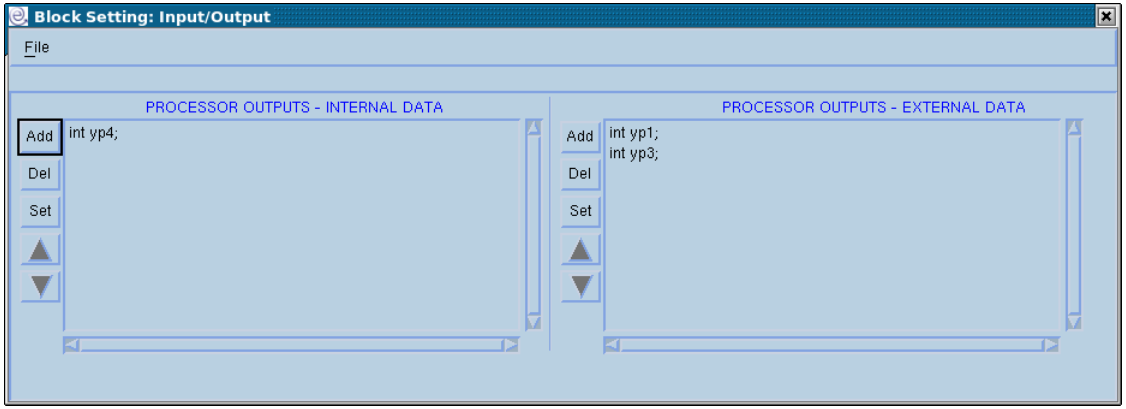


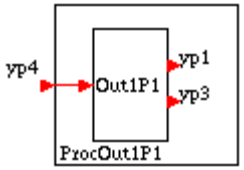
The input/output variables are ANSI C variables that can be used in any ANSI C function of the block.

The Processor Input/Output Blocks with post-processing

The Input/Output window

The Processor Output Block with post-processing has at disposal a function for processing its input data.
By means of the Input/Output window you can define the input data (the internal output data) and the processed data (the external output data).



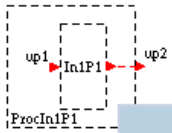


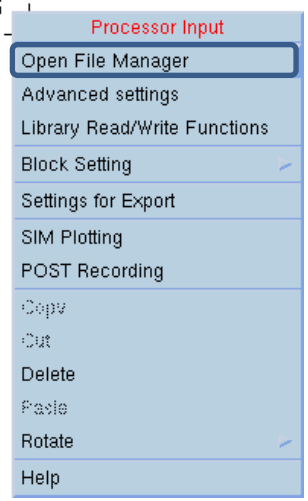
The input/output variables are ANSI C variables that can be used in any ANSI C function of the block.

The Processor Input/Output Blocks with post-processing

The file manager

The Processor Input/Output Blocks with post-processing have their own file manager through which it is possible to program the data processing in ANSI C language.

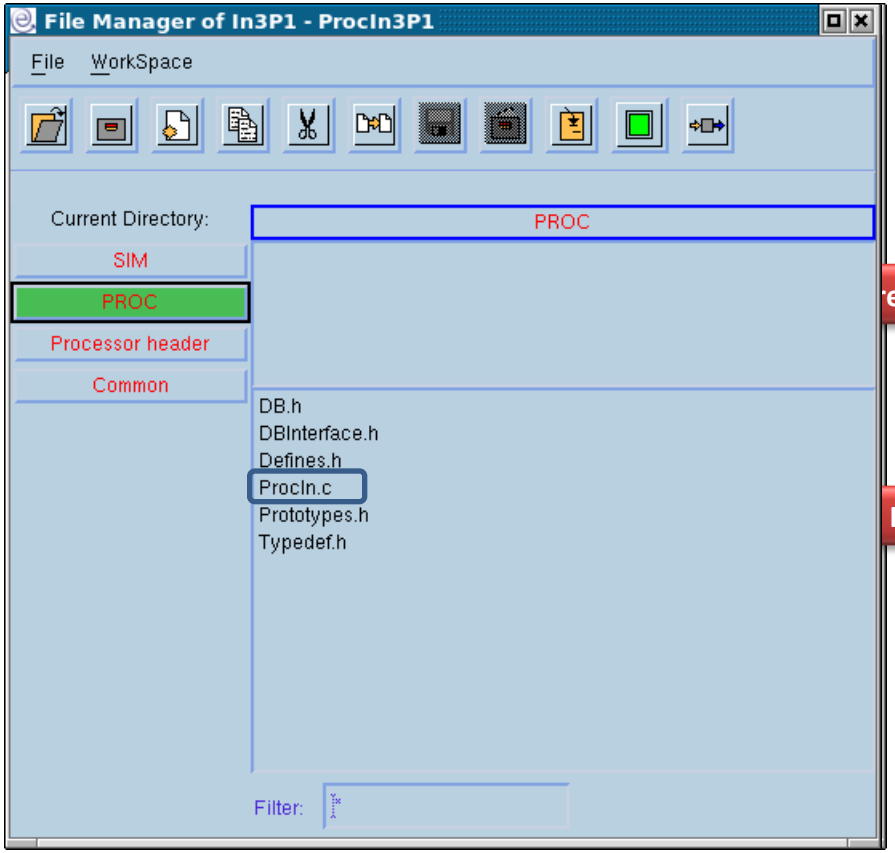




EICASLAB provides a pre-organized structure: a set of template files subdivided in:

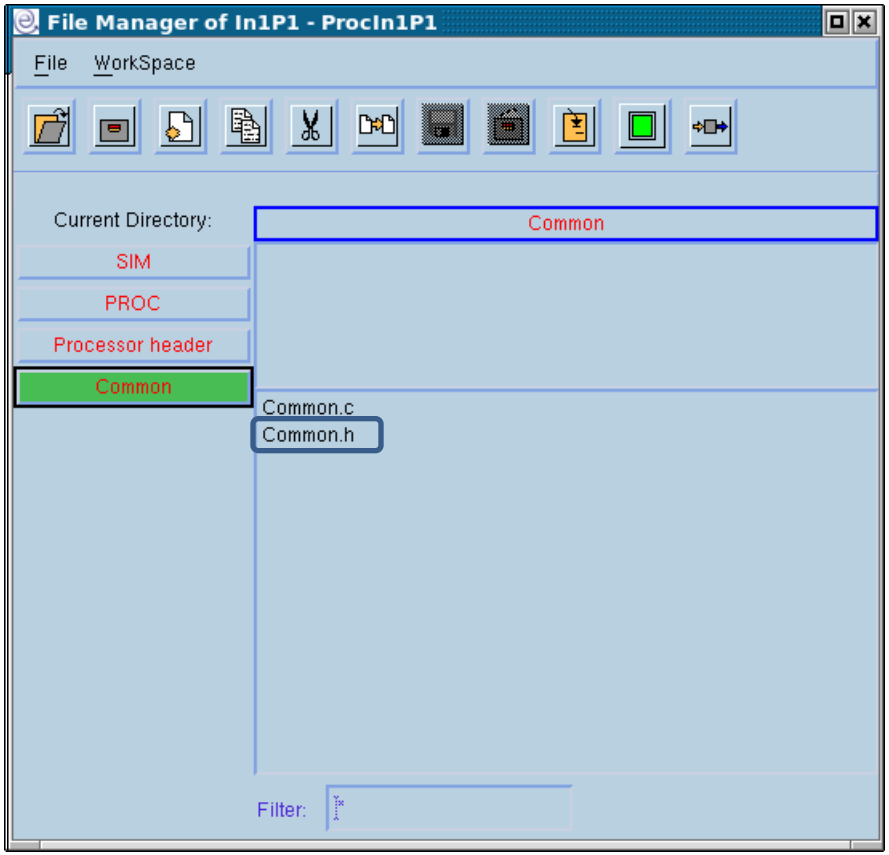
- data files,
- header files,
- ANSI C files,

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



The Processor Input/Output Blocks with post-processing

The header files



Header files of the pre-organized 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
DBP.h	Available for all the Controls belonging to the same Processor and programmed in ANSI C
Common.h	Available for all the blocks programmed in ANSI C

The Processor Input/Output Blocks with post-processing

Initialization functions

Name	Description	ANSI C File	Data File
ProcIn#P$_ReadPar	Parameter file reading	ReadPar.c	ProcIn.par
Proc In#P$_ReadState	Initial state file reading	RWState.c	ProcIn.inistate
ProcIn#P$_Ini	User initialisation function	ProcIn.c	---
ProcOut#P$_ReadPar	Parameter file reading	ReadPar.c	ProcOut.par
ProcOut#P$_ReadState	Initial state file reading	RWState.c	ProcOut.inistate
ProcOut#P$_Ini	User initialisation function	ProcIn.c	---

The Processor Input/Output Blocks with post-processing

Execution functions

Name	Description	ANSI C File
ProcIn#P$_Exe	Input post-processing	ProcIn.c
ProcOut#P$_Exe	Output post-processing	ProcOut.c

The Processor Input/Output Blocks with post-processing

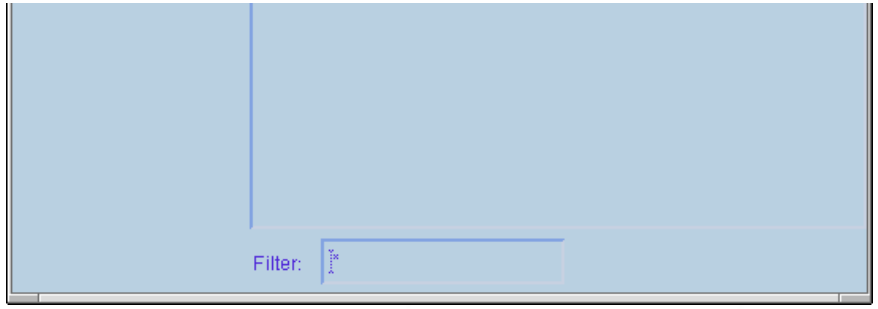
Final functions

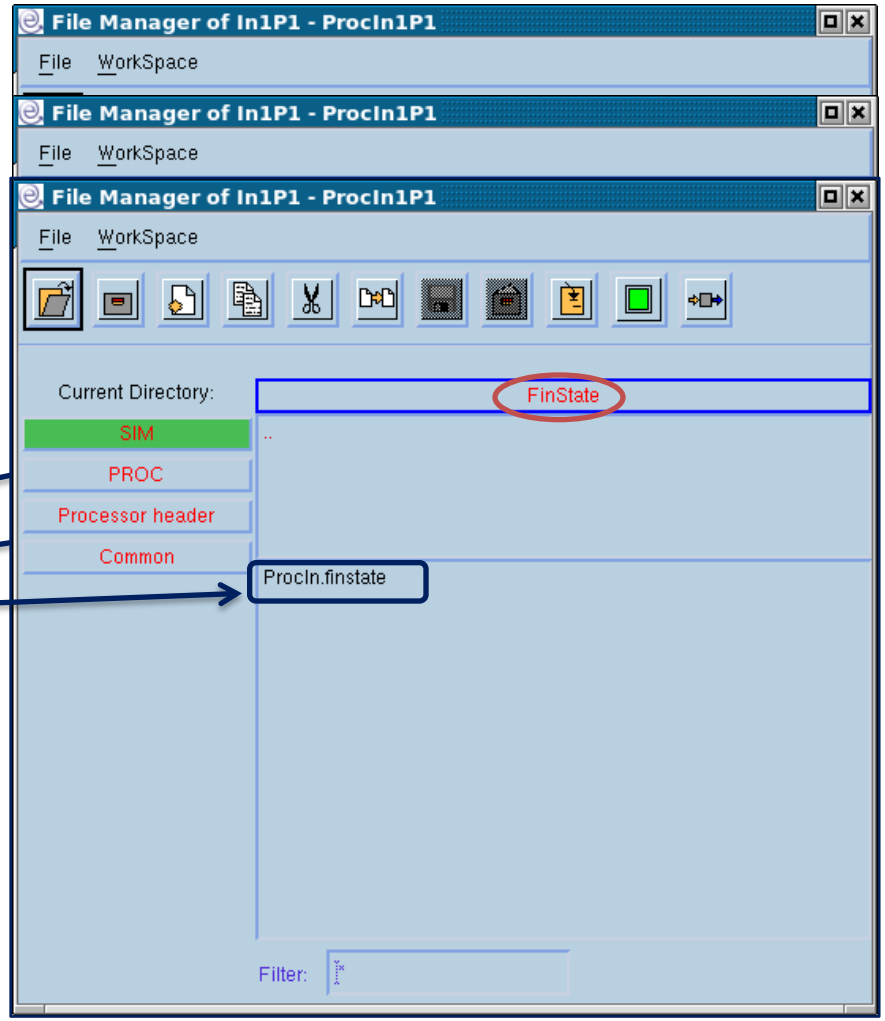
Name	Description	C File	Data File
ProcIn#P$_Fin	User final function	ProcIn.c	---
ProcIn#P$_WriteState	Final state file writing	RWState.c	ProcIn.finstate
ProcOut#P$_Fin	User final function	ProcOut.c	---
ProcOut#P$_WriteState	Final state file writing	RWState.c	ProcOut.finstate

The Processor Input/Output Blocks with post-processing

Data file management

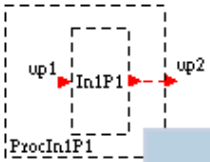
```
/****************************/
void    ProcIn1P1_ReadPar(FILE *fp)
/*
INPUTS:
fp      file pointer to the file Plant.par

OUTPUTS:
value of the ProcIn1P1 parameters

OBJECTIVES:
The function can read the parameters of the ProcIn1P1 module from the file ProcIn.par.

All the parameters should be defined in:
        DBInterface.h   interface database of the ProcIn1 P1 Module,
        DB.h            database of the ProcIn1 P1 Module,
        DBP.h           database of the Processor 1,
        Common.h        file shared with the other C block Modules

SCHEDULE:
The function is called  by the EICASLAB simulator nucleus,
once at the beginning of the simulation session.
*/
{


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

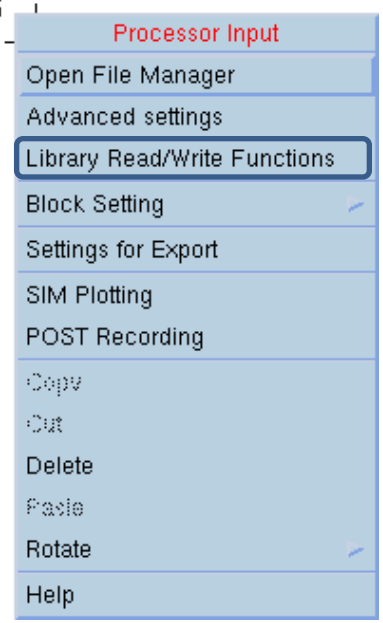


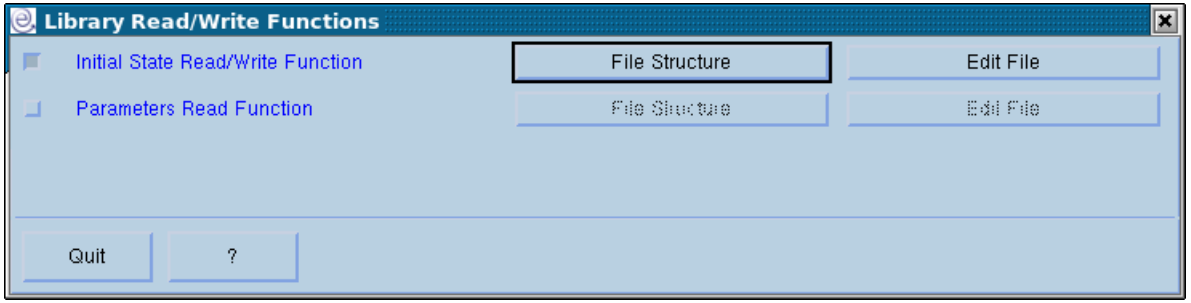


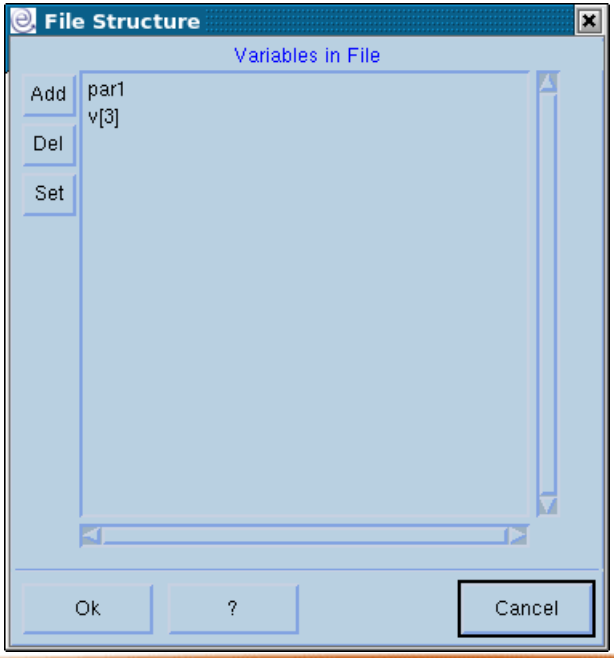
The Processor Input/Output Blocks with post-processing

The Library Read/Write Functions

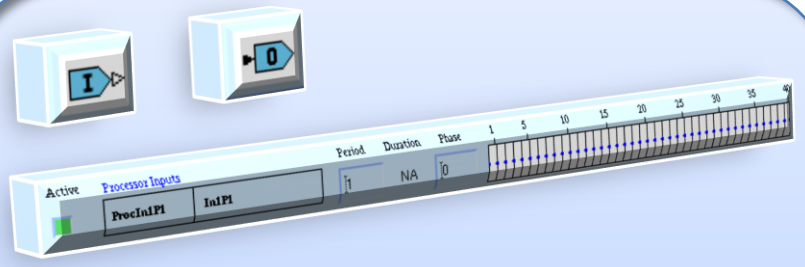








Welcome to Innovation



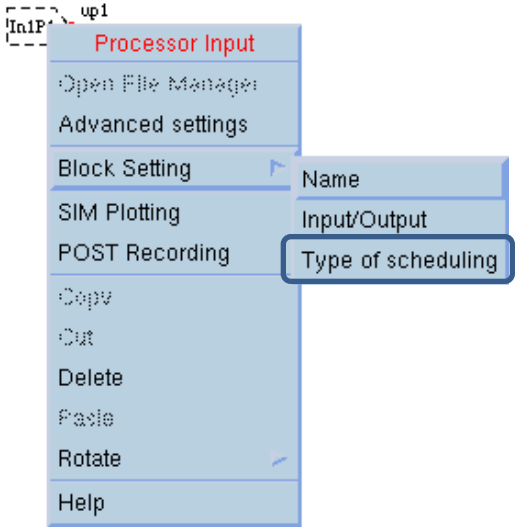
The scheduling of the Processor Input/Output

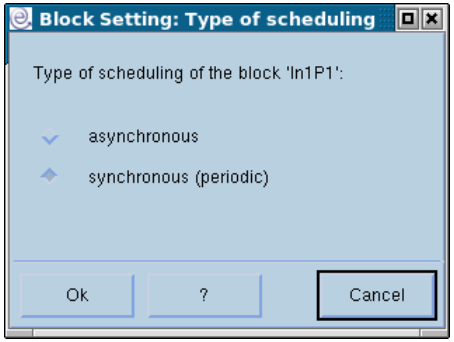
The scheduling of the Processor Input/Output Blocks

Synchronous and asynchronous Processor Inputs

The Processor Input Blocks represent the interrupt activities that receive and process the inputs coming in your processor.
Such inputs can be received:

- with a given periodicity (**synchronous** processor inputs),
- through an asyncrounous communication (**asynchronous** processor inputs).





The Processor Ouput Blocks are always **synchronous**.

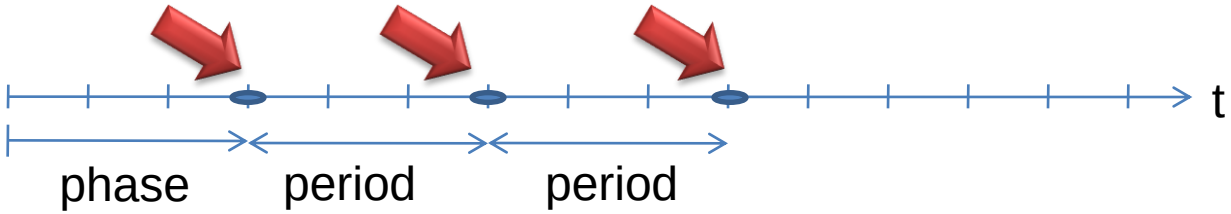
The scheduling of the Processor Input/Output Blocks

Synchronous Processor Inputs/Outputs

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

The scheduling of the synchronous Processor Input/Output Blocks is defined by 2 scheduling parameters:

- the **phase**, time at which they are called for the first time,
- the **period**, their sample time interval.



The scheduling of the Processor Input/Output Blocks

Function categories

The Processor Input/Output Blocks with post-processing may be programmed through a set of functions.

All the functions have a template provided by EICASLAB and are managed by the user.

The functions belong to three main categories:

Functions categories

- 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

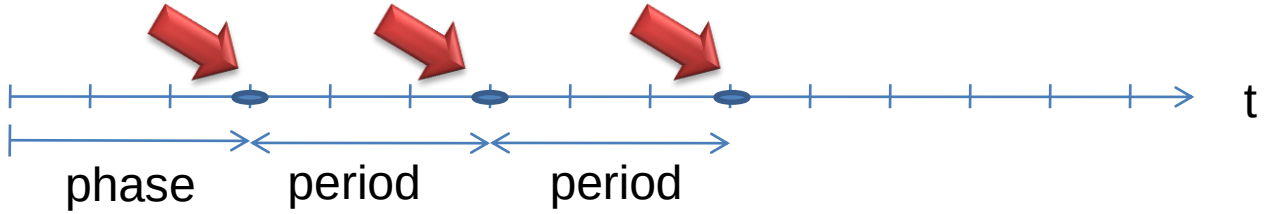
The scheduling of the Processor Input/Output Blocks

Function scheduling

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

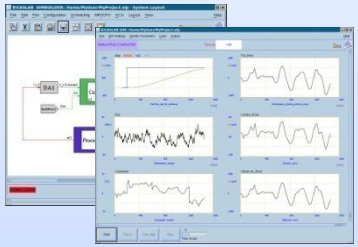
1) Parameter file reading,
2) Initial state file reading,
3) User initialisation function.

The *post-processing function is* called when the corresponding block is scheduled.



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

1) User final function,
2) Final state file writing.



Processor Input/Output scheduling in Modelling and Like Real-time Simulation phase

Processor Input/Output Blocks scheduling in Modelling and Like real-time Simulation phase

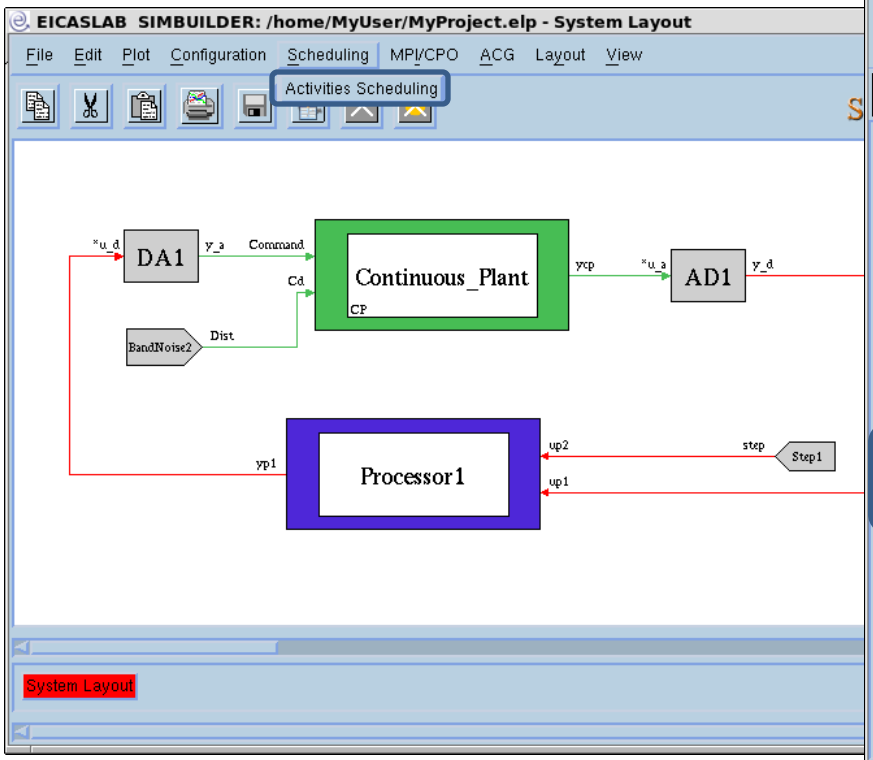
The SIM tool manages the scheduling of the Processor Input/Output Blocks by means of the **EICASLAB scheduler** that is **the core of the time scheduling algorithms and** allows to run like real-time simulations.

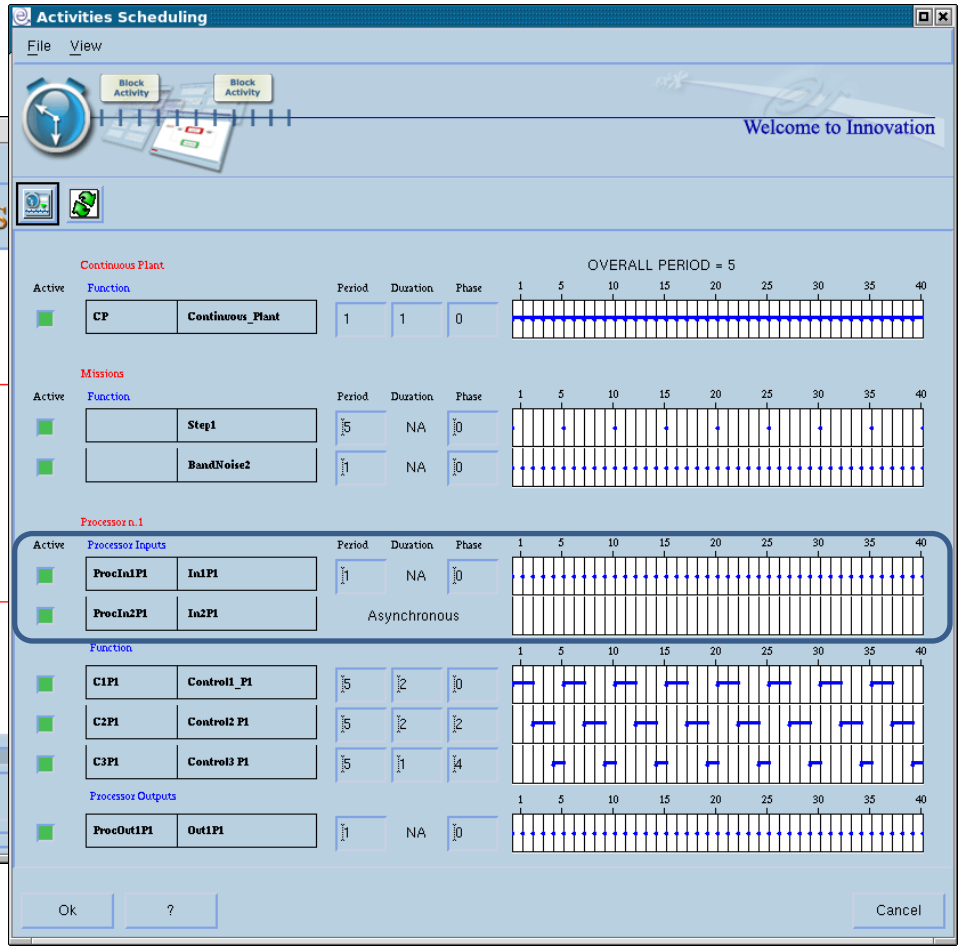
On the basis of the scheduling parameters

It defines the order in which the Processor Input/Output Blocks have to be executed

Processor Input/Output Blocks scheduling in Modelling and Like real-time Simulation phase

How to set the scheduling

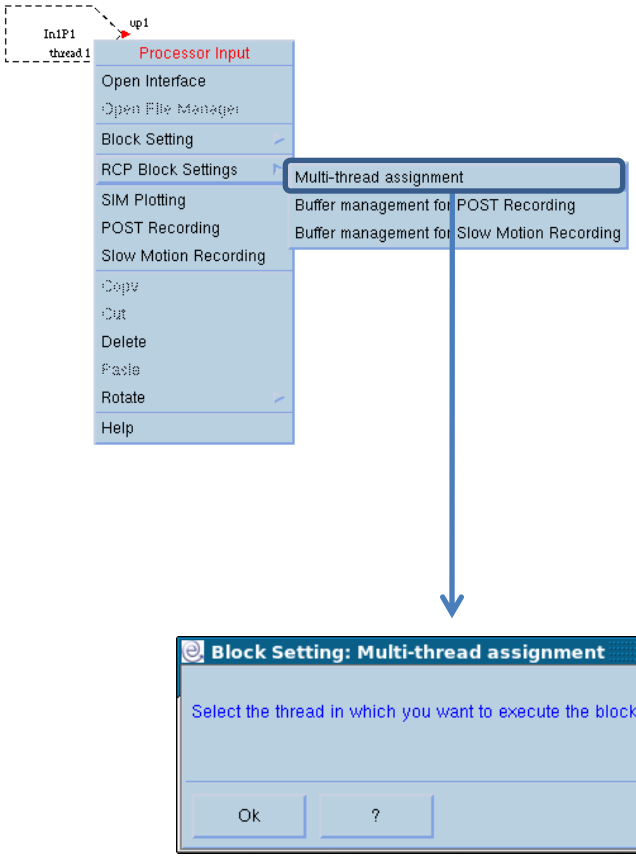




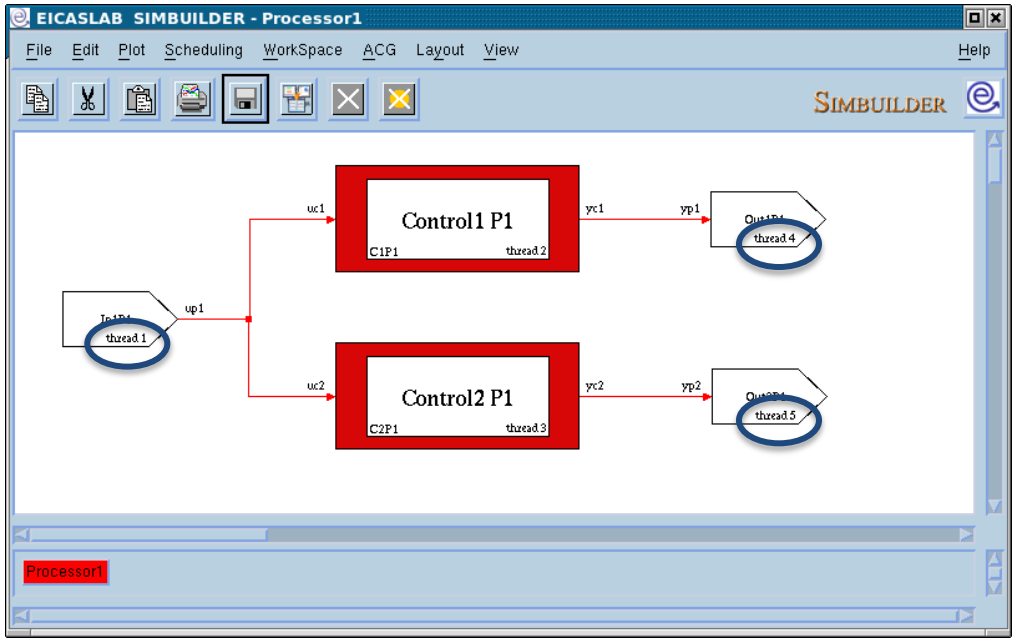
Processor Input/Output scheduling in Rapid Control Prototyping (RCP) phase

Processor Input/Output Blocks scheduling in Rapid Control Prototyping phase

Thread of the activities

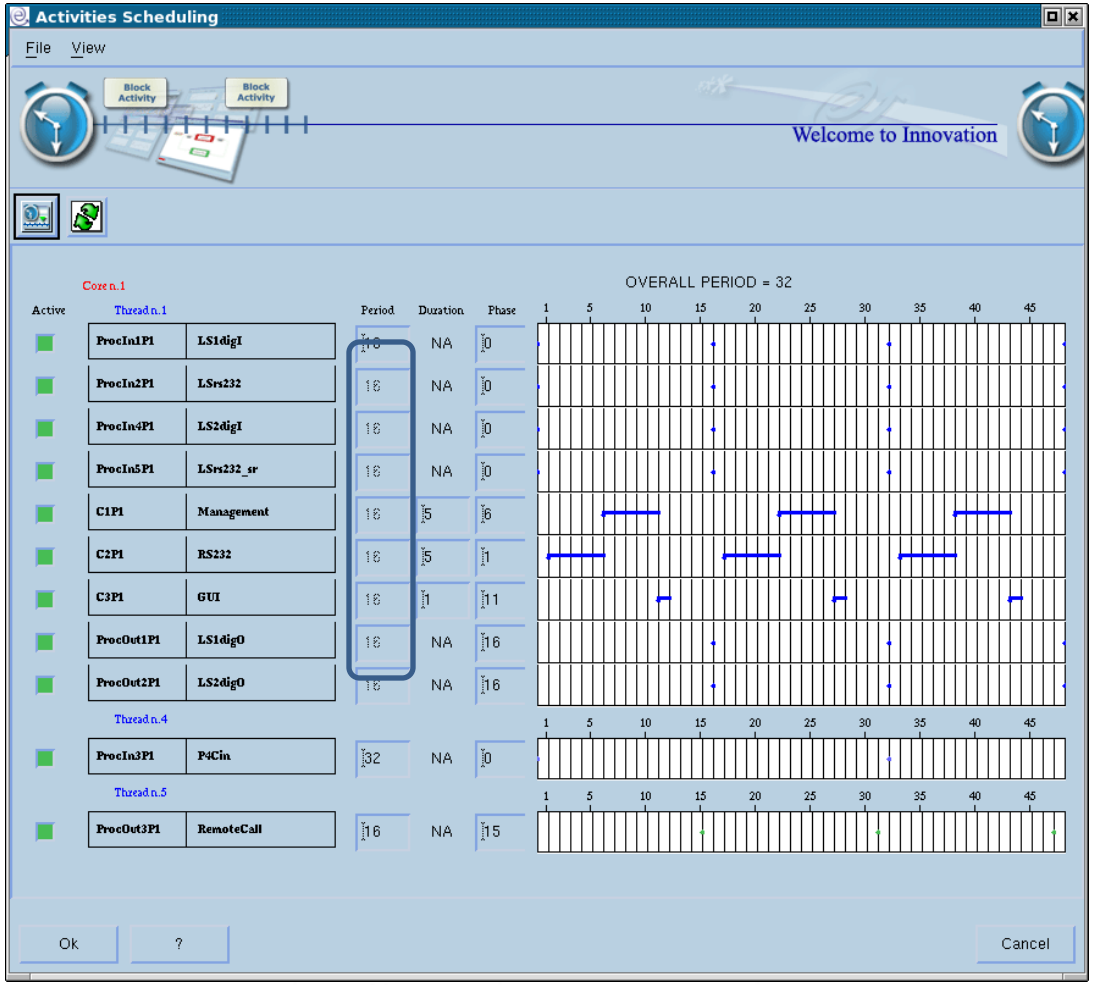


Every Control function and Processor Input/Output has to be associated to a thread.



Processor Input/Output Blocks scheduling in Rapid Control Prototyping phase

The scheduling window and the Scheduling constraints



The Processor Input/Output and the control functions are listed thread by thread.

- All the activities belonging to a same thread must have the same period and must not be overlapped,
- Each asynchronous Processor Input has its specific thread.

EICASLAB supports you and prevents you to make scheduling errors.