

SIMTSX

Introduction and Installation

Simulation Process

**Description of a SIMTSX Application Using Models
from the Library**

**Description of a SIMTSX Application Using new
Elements**

Validation of the SIMTSX Application

Validation of the PLC Program

SIMTSX Utilities

Examples

Appendices

A
B
C
D
E
F
G
H
I

Section	Page
1 Introduction and Installation	1/1
1.1 Introduction	1/1
2 Installation	2/1
2.1 Checking the Material Supplied	2/1
2.2 Required Configuration	2/1
2.3 Installing SIMTSX	2/2
2.4 Installing the Authorization Code	2/4
2.5 Accessing SIMTSX	2/6
2.6 SIMTSX Main Window	2/6
3 Connection to PLCs	3/1
3.1 Connections	3/1
3.1-1 Communication With a Micro/Premium	3/1
3.1-2 Communication With a Quantum	3/1

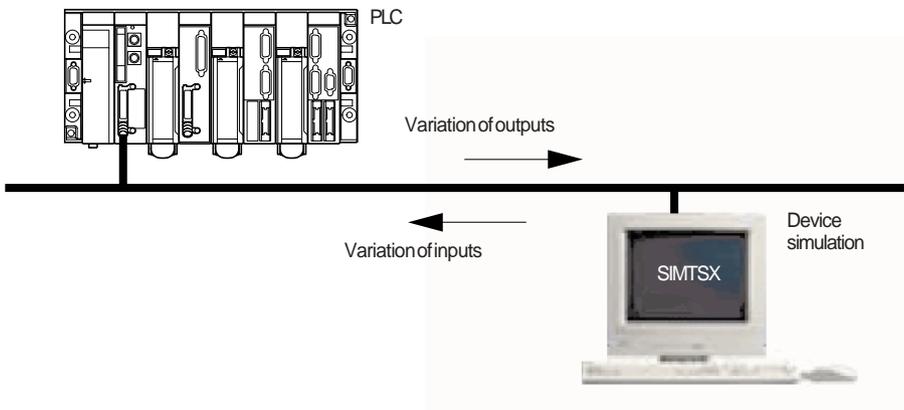
Section**Page**

1.1 Introduction

This document describes the installation and use of **SIMTSX** and **SIMTSX LITE** products with the references **TLX FCD SIM ●●●** and **TLX FCD SIM L●●●**.

SIMTSX is a software package which can be used to simulate the evolutions in an automated installation which is controlled by one or more Modicon TSX Micro, Premium or Quantum PLCs (**SIMTSX LITE** can only be used to simulate Micro PLCs).

When **SIMTSX** is connected to the PLC, it **replaces all the I/O and devices downstream of the PLC** (systems wired in the electrical cabinet, preactuators, actuators, sensors, physical installation measurements, product movement, etc) :



During simulation, the PLC program which is loaded in the PLC runs as if it is controlling the real installation.

The SIMTSX simulation platforms can then be used for :

- Complete high-speed debugging and validation of PLC programs and associated control-command functions :
 - control of devices in all possible operating modes, both normal and downgraded
 - supervision and man/machine interface functions in connection with the PLC(s)
 - communication tasks between PLCs or with higher data processing levels
- Acceptance testing of control-command with printing of the required documents
- Training for control and maintenance operators before taking charge of the equipment on site

2.1 Checking the Material Supplied

The SIMTSX software package comprises :

- a set of 3"1/2 SIMTSX software floppy disks
- a CD-ROM
- an authorization form which must be faxed off in order to receive a license number which is required for installation
- this manual

2.2 Required Configuration

SIMTSX software requires the following minimum hardware configuration :

- an IBM PC compatible computer with WINDOWS 95 or WINDOWS NT version 4.0 operating system
- a VGA screen (a high resolution screen is recommended)
- an 80486 DX or higher processor (a PENTIUM is recommended)
- 16 Mb of RAM (20 Mb recommended)
- a hard disk with 20 Mb of free space for installing SIMTSX
- a CD-ROM drive or a 3"1/2 floppy disk drive
- a mouse
- a parallel port for the printer
- a free COM serial port (COM1 to COM4) for connection to the PLC

Compatibility :

- **SIMTSX** is compatible with **PL7 Version 3.0** or later.
- **SIMTSX** is compatible with **CONCEPT and IEC Simulator version 2.1** (or later).
- **A SIMTSX application can only be connected to Micro or Premium (or PCX) processors with software version 3.0 or later.**
- **A SIMTSX application can only be connected to Quantum processors compatible with CONCEPT 2.1 or later.**
- EPSON and HP Laser Jet compatible printers are necessary for printing trend diagrams of movements. To print simulation application and model dossiers, use your usual printer.

2.3 Installing SIMTSX

Preliminary operations

SIMTSX can be installed from :

- a CD-ROM
- a set of floppy disks

Before installing SIMTSX software on the hard disk, it is advisable to :

- read the license certificate and warranty regarding restrictions on copying and installing the software
- make a copy of the original installation floppy disks and use these backup copies to work with, so as to avoid any accidental damage to the originals

Important:

The floppy disks (3" 1/2) are supplied write-protected. Do not change the position of the tab.

Installation procedure

The following operations must be carried out before installing SIMTSX software :

1. Close all current sessions (refer to the relevant WINDOWS documentation).
2. If the station has WINDOWS NT, open a session with administrator rights.

Installing SIMTSX software

- Select the "Start" menu, then "Run" from the WINDOWS 95 or NT desktop.
- Insert the first "SIMTSX" floppy disk in the drive, or insert the "SIMTSX" CD-ROM.
- Enter the command "x:setup" (where x is the drive identifier for the CD-ROM or floppy disks), then confirm with <Enter>.
- Follow the instructions (see operating modes).

Operating modes

Each installation procedure screen is standardized. It has a field for setting the parameters, and 3 buttons : <Back>, <Next> and <Cancel>.

The <Back> and <Next> options can be used to switch to the previous or next screen.

The <Cancel> option can be used to exit the installation procedure.



The first screen can be used to select the language used by **SIMTSX**. The language suggested by default is the language detected on the station.

The second screen can be used to specify the installation option, either <Standard> or <Personalized>.

These 2 options determine the installation procedure :

- Default :
 - SIMTSX is installed on the station, on the first disk which has sufficient space available, in the \SIMTSX directory. The application and model examples are also installed here. The UNI-TELWAY driver which provides a connection to the terminal port of a Micro or a Premium is installed in the \XWAYDRV\ directory.
 - The simulation modules for Micro and Premium are automatically installed if the station has a version of PL7 Micro, PL7 Junior or PL7 Pro.
 - The simulation EFB for CONCEPT required for online simulation on Quantum is automatically installed if the station has a version of CONCEPT (the CONCEPT location is detected automatically).
 - The MODBUS driver for connection to a Quantum is installed automatically.
- Customize :
 - A screen enables you to select the components you wish to install from a list :
 - SIMTSX
 - PL7 additions
 - CONCEPT additions
 - UNI-TELWAY driver
 and to customize their installation directories.

2.4 Installing the Authorization Code

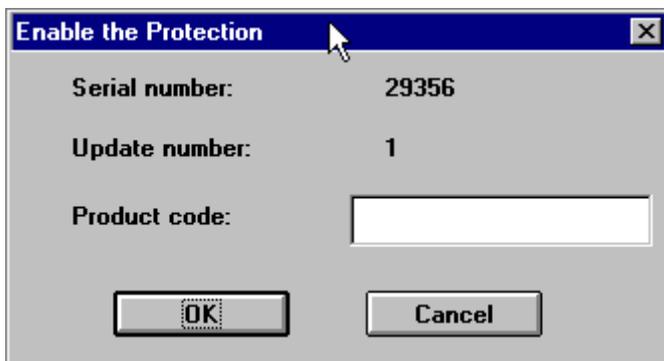
Simulation mode (offline and online) requires an authorization code. This code is entered using the menu.

A dialog box is displayed with the following commands :

<**Installation**> : this command is launched the first time by default. It indicates the serial number and the update number.

Enter these numbers on the authorization form and fax it off in order to receive your authorization code.

<**Enable**> : this command can be used to enter your authorization number (in the "Product code" field).



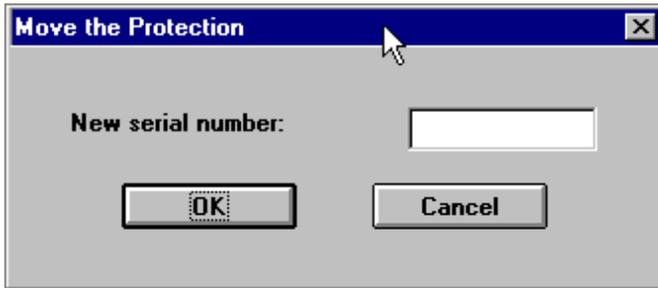
Important :

Never move or delete the **SIMTSX** directory after enabling the code, or you may lose the authorization code on the station. If you have to move **SIMTSX** to another station, use the <**Move**> command.

Important :

Keep the fax you sent and the returned copy with your license.

<Move> : this command can be used to move your authorization code from one station equipped with **SIMTSX** to another station.



1. Install the software in the usual way on the station which is to receive **SIMTSX**.
2. On the station which already has **SIMTSX** (station 1), click on the <Move> button. A dialog box is displayed.
3. On station 2, click on <Enable>, and a serial number is displayed. This code must be entered on station 1.
4. On station 1, enter the number in the <New serial number> box. You will be given a code.
5. On station 2, enter the code given by station 1 in the <Product code> box : your **SIMTSX** access right is then transferred to station 2.

Uninstalling SIMTSX

The **SIMTSX** software can be uninstalled as follows :

- Select the "Start" menu, then "Settings/Control Panel" from the WINDOWS 95 or NT desktop.
- In the Control Panel select the icon "Add/Remove Programs".
- Select "**SIMTSX**" from the list of installed components, then click on the "Add/Remove" button; after confirmation, this will remove the software from the station. Applications developed using "SIMTSX" are not removed from the station.

Notes

1. The uninstall function does not remove the authorization code from the station. This means that if you want to re-install **SIMTSX**, you must select the same directory as that used for the first installation, so that you can retrieve the authorization code.
2. If the UNI-TELWAY driver has been installed by **SIMTSX**, the uninstall function will remove shared XWAY components from the station. It may then be necessary to re-install the drivers for use by other software.

2.5 Accessing SIMTSX

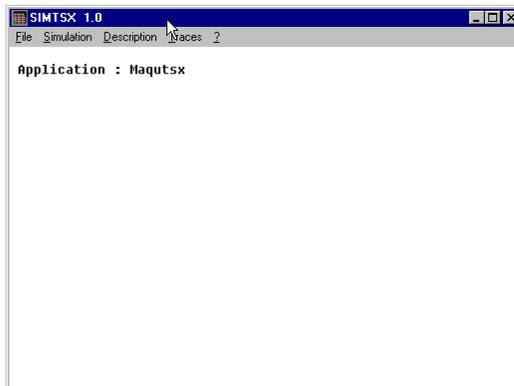
The access procedure is exactly the same as that used for all other programs with shortcuts in the WINDOWS 95 or WINDOWS NT desktop fast path.

SIMTSX is accessed by opening the corresponding window. To do this :

1. Open the Start menu on the desktop.
2. Go to Programs.
3. In the submenu, move the mouse onto the SIMTSX item.
4. From the dropdown menu, select the SIMTSX item.

2.6 SIMTSX Main Window

The main window for SIMTSX provides access to a range of software functions.



The menu bar on the window consists of the following 5 items :

- **File**

- Application :

- This accesses the list of existing SIMTSX applications.

- Chart :

- This item can be accessed once an application has been selected.

- This accesses the list of existing Grafcet charts.

- Exit :

- This is used to exit SIMTSX.

- **Simulation**

- This button activates a menu which is used to launch the simulation in either "offline" or "online" mode.

- **Description**

- This is used to access the Description editors.

- **Traces**

- This accesses the representation in the form of a trend diagram of the evolution of variables during simulation.

- **?**

- This accesses the Help files for SIMTSX.

- This also accesses the authorization code for **SIMTSX**.

3.1 Connections

The connections to PLCs are exactly the same as those used by PL7 and CONCEPT.

3.1-1 Communication With a Micro/Premium

The **SIMTSX** product includes as standard the tools required for connection to a Micro or Premium PLC via the terminal port. The PLC/**SIMTSX** connection is via the RS232 port. For a PC with a serial port, an RS232/RS485 converter, for example cable TSX PCU 1030, is used to connect the PC to the Micro/Premium.

SIMTSX can be connected to PLCs via the Fipway, Ethernet TCP/IP and Ethway networks if the relevant physical supports are installed. Please refer to the installation manuals for these media.

SIMTSX can also communicate with PCXs using the ISA-Way driver.

3.1-2 Communication With a Quantum

To connect SIMTSX to a Quantum via Modbus Plus, please use one of the following kits for Windows95/NT4.0:

- **AM-SA85-030** ISA bus - single channel
- **AM-SA85-032** ISA bus - dual channel
- **416NHM21230** PCMCIA type II
- **416NHM21233** PCMCIA type III

It is also possible to connect to the Quantum via an Ethernet link, i.e. using the Ethernet/Modbus service. To do this, you will need a PC format Ethernet card/RJ45 connector, connected via twisted pair link to a 140 NOE 211 00 module linked to the Quantum PLC. For the Modbus link, the PLC/**SIMTSX** connection is via the RS232 port on the PC; a standard Modbus cable (for example, cable 490 NAA 27101) can be used to connect the PC to the integrated Modbus port on the Quantum.

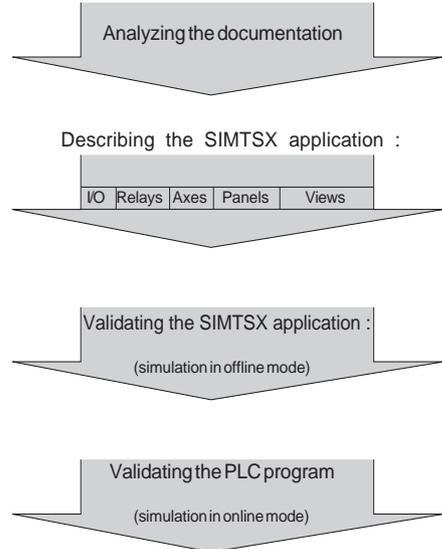
Section	Page
1 Simulation Process	1/1
1.1 Global Description and Simulation Process	1/1
1.2 Detailed Process	1/2
1.2-1 Analyzing the Documentation	1/2
1.2-2 Describing the SIMTSX Application (see parts C and D)	1/3
1.2-3 Validating the SIMTSX Application (Simulation in Offline Mode)	1/4
1.2-4 Validating the PLC Program (Simulation in Online Mode)	1/5
1.3 Main Window of the Description Interface	1/6

Section**Page**

1.1 Global Description and Simulation Process

The use of SIMTSX can be described as follows :

- **Analyzing the documentation :**
 - design dossiers
 - electrical and pneumatic diagrams
 - trend diagrams for exchanges, installation plans
- **Describing the SIMTSX application :**
 - model library
 - physical measurements
 - electrical part
- **Validating the SIMTSX application :**
 - reflecting the mechanical and electrical design of the installation
- **Validating the PLC program :**
 - control-command
 - inter-PLC dialog
 - supervision, etc.

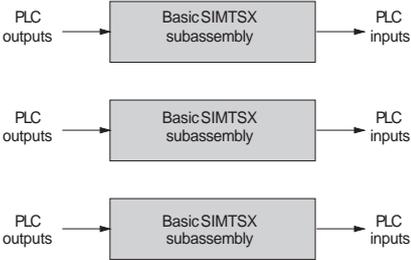


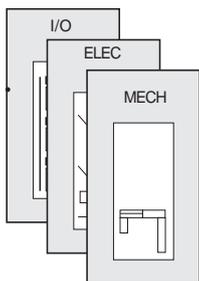
As well as debugging and validating the PLC program, the SIMTSX process can also be used, as required, for acceptance testing of the automated systems, and with the training of operators before they take control of real installations.

1.2 Detailed Process

The process suggested here is designed to assist the user in developing a SIMTSX application, and then using it as effectively as possible to test PLC programs. To do this, it is useful to perform a number of operations in chronological order. These operations are described in this part, and the flow chart below refers to the relevant sections.

1.2-1 Analyzing the Documentation

Which evolutions do you want to describe?	What are the PLC/SIMTSX exchanges?
	<p>Using the documents produced by the installation designers, identify all the possible basic movements of the installation which will be described in SIMTSX :</p> <ul style="list-style-type: none"> - mechanical movements : cylinders, plates, valves, etc. - movement of parts : parts on belts, conveyors, etc. - changes to certain values : level within a vat, temperature, pressure, flow rate, etc. - and, by extension, dialogs between the PLC and the subassemblies viewed as a "black box" : robots, soldering units, etc. <p>(The "movements" correspond to the transcription of data exchange trend diagrams).</p>



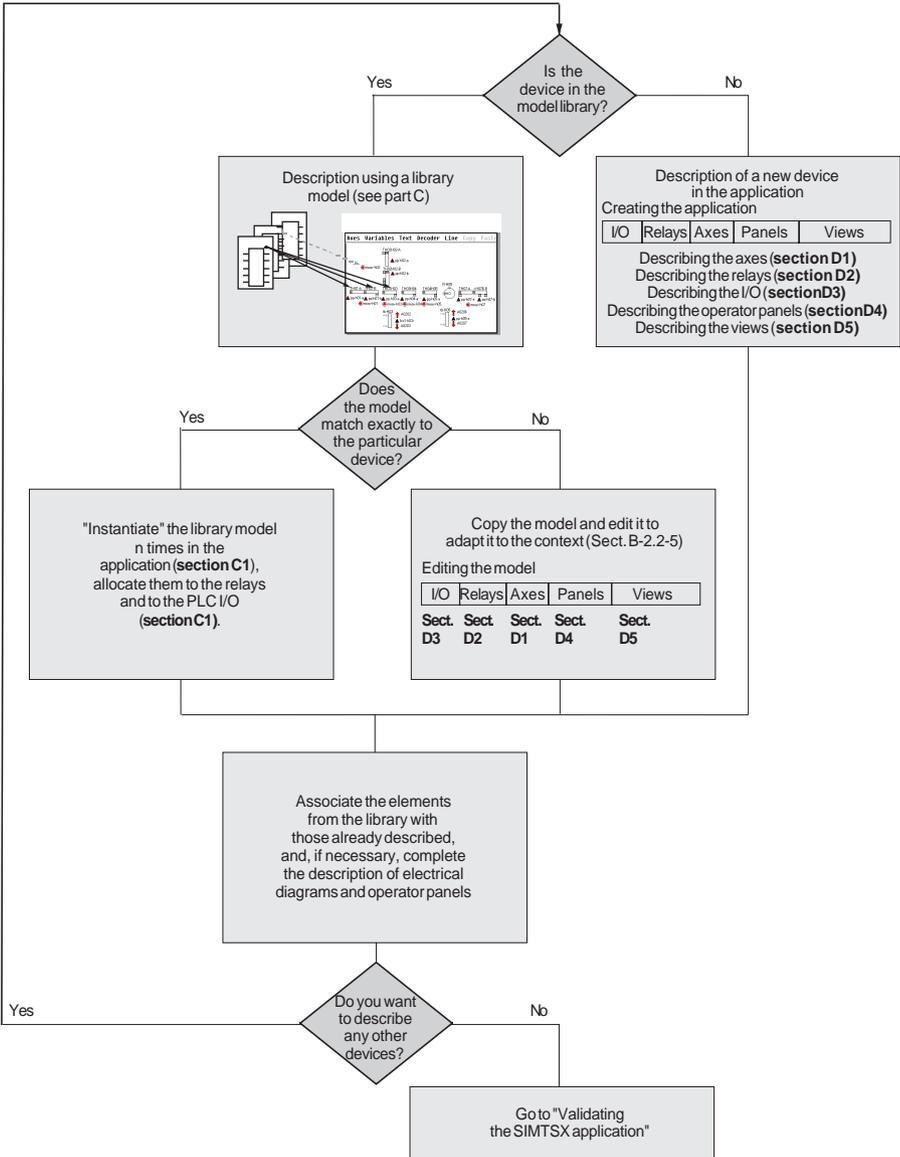
The documents to analyze are :

- the plans of the installation
- the mechanical design dossiers
- the flow charts (pneumatic, hydraulic)
- the electrical diagrams, with control panels, relaying, feed units and PLC I/O
- if necessary, a functional description providing an explanation of the installation

It is possible that one or more basic devices will correspond to standard devices or to devices already described in previous applications (for example, valves, materials handling systems). The SIMTSX descriptions for these devices have already been placed in the library in the form of a model which can be retrieved directly (see part C).

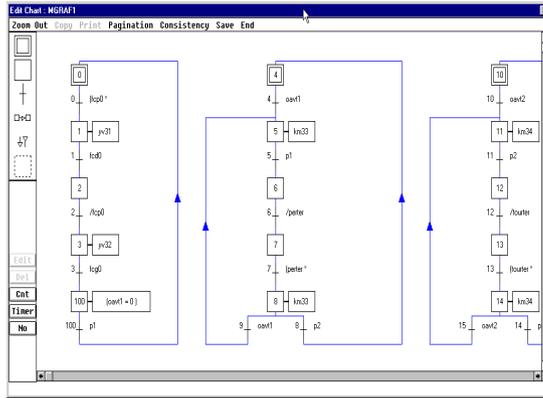
If this is not the case, the devices should be described using the basic SIMTSX tools : axes, movements, sensors, relays, etc. (see part D).

1.2-2 Describing the SIMTSX Application (see parts C and D)



It is highly advisable to validate the global behavior of the description, which also enables the overall design to be checked in collaboration with the mechanical designer. To do this, a "Chart" function is used to describe the sequence of movements (functional description) and to automatically animate the application, while remaining in offline mode (not connected to the PLC).

1.2-3 Validating the SIMTSX Application (Simulation in Offline Mode) (see part G)



Describing the animation chart (see section E2)

Here the operation of the installation is reproduced (for example, machining cycle of a machine) using the documents supplied by the mechanical designer: mechanical Grafset chart, trend diagram of movements, or the literal description).

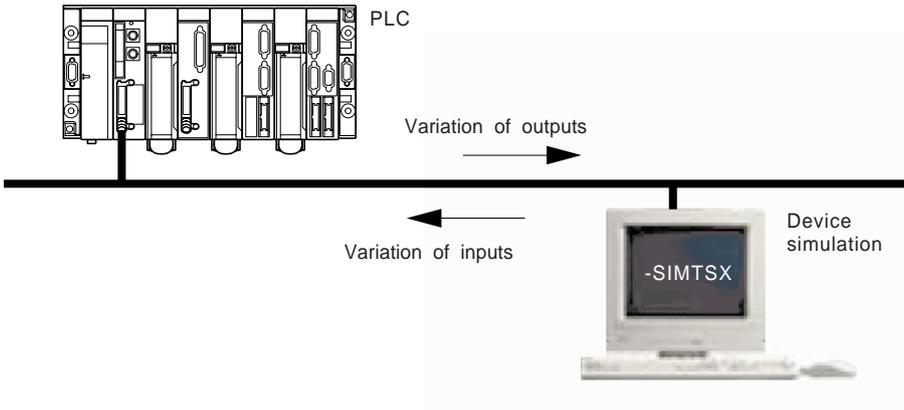
Since this is a question of substituting the PLC program in sequencing the evolutions in the application, the step actions can be on the electrical contacts, preactuators or actuators of the application, while the transition conditions can retrieve sensors, operator panel buttons or any other variable.

Load the application and the chart, to simulate in offline mode in the simulation environment (see section E4)

- Use the control tools to animate the simulation : **panels (section E4.2-2), external variables (section E4.2-1), contexts (section E4.2-3), scenarios (section E4.8).**
- Use the SIMTSX operating modes, in particular **step-by-step** mode, to advance operation of the application progressively (**section E4.1**).
- Use the display tools (**views, pages section E4.3**) and the analysis tools (**examination section E4.5, activity, evolutions, traps section E4.6**) to debug the model.
- Access the **Description (section E4.11)** function for any "online" modification which does not alter the current mechanical description.
- Use the **Trace (section E5)** function to save any changes made to the application and to represent these in the form of a trend diagram, so that the behavior of the SIMTSX application can be validated definitively (if possible in collaboration with the mechanical designer).

At this stage, the SIMTSX application obtained must correspond exactly to the real machine currently being developed. It can then be used to initiate debugging of the PLC program as soon as the coding and test phases have been completed by the control systems engineer.

1.2-4 Validating the PLC Program (Simulation in Online Mode) (see part F)



Prepare the simulation platform (see section F1.2-1)

Configure the SIMTSX module in the PL7 application (see section F1.3)

Load the SIMTSX application to simulate in online mode in the simulation environment (see section E4)

- Use the control tools to animate the simulation :
 - **panels** (section E4.2-2)
 - **external variables** (section E4.2-1)
 - **context** (section E4.2-3)
 - **scenarios** (section E4.8)
- Use the SIMTSX operating modes, in particular **step-by-step** mode, to advance operation of the model progressively (section E4.1).
- Use the display tools (**views, pages** section E4.3) and the analysis tools (**examination** section E4.5, **activity, evolutions, traps** section E4.6) to analyze the current status of the automated system.
- Follow the evolution of the I/O during simulation (section F1.5). Access the various functions in the Tools menu to display a historic of the evolutions of I/O or of the graphs traced for the analog variables. SIMTSX can also be used to manage channel and module faults on Micro and Premium.
- Use the **Fault** (section E4.7) and **Force** (section E4.4) tools to reproduce any possible installation breakdowns or malfunctions.
- The **Description** (section E4.11) function can still be accessed for any "online" modifications of the application which do not alter the current mechanical description.

- Confirm acceptance of the PLC program once all functions have been tested and validated. Use the **Trace (section E5)** function as required to edit the trend diagrams in the main operating modes.

Note :

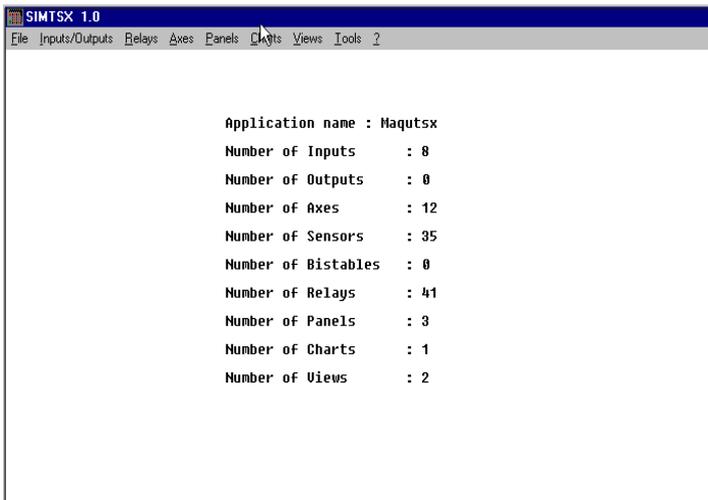
If the architecture of the automated system is distributed (several PLCs sharing the control-command of the application), online simulation can be carried out in two steps :

- single-PLC simulation, one PLC at a time
- multi-PLC simulation, with SIMTSX connected via a network to the entire architecture

When creating a SIMTSX project, many tools are available for documenting and saving the application (Part G, section 3), as well as for transforming an interesting or reusable part of the application into a model, and thus adding to the model library for future projects (Part G, section 1).

1.3 Main Window of the Description Interface

The main window of the description interface comprises two zones.



The top band accesses the description interfaces of the various parts of the installation. The central zone contains information about the application or model which is being edited. The role of each of these elements is explained below.

The File menu provides access to the options Application, Model, Print, Simulation and Exit.

- items for accessing description editors :

Application	is used to create, load and delete dossiers. When there is no application loaded in SIMTSX , the message "No application selected" appears at the top left of the window.
Models	see section 1 of parts C and G.
I/O	see section 3 of part D.
Relays	see section 2 of part D.
Axes	see section 1 of part D.
Panels	see section 4 of part D.
Views	see section 5 of part D.
Charts	see section 1 of part E.

- accessing other functions

Help	launches the online Help function for the SIMTSX description
Tools	
Printing the application	See section 3 of part G.
Saving to a floppy disk	See section 2 of part G.
Loading from a floppy disk	See section 2 of part G.
Copying an application	See section 2 of part G.
Transferring an application to a model	See section 1 of part G.
Tracing instantiations	See section 1.2 of part C.
Simulation	See part D.
Exit	Exits the SIMTSX description tool.

Section	Page
1 Describing a SIMTSX Application	1/1
1.1 Introduction	1/1
1.1-1 Incorporating a Model	1/1
1.1-2 Resolving Clashes	1/3
1.1-3 Parameter Values	1/4
1.1-4 Identifying the Inputs	1/4
1.1-5 Undefined Variables	1/4
1.1-6 Accessing the Model Description	1/5
1.2 Tracing Instantiations	1/6

Section

Page

1.1 Introduction

An application can be made up of a number of similar parts, where the same subassembly may be common to several different applications. It can therefore be useful to be able to reuse these "standard models" easily, thus avoiding laborious re-description and the possible introduction of errors.

Also, saving any particular information relating to the description of elements can be useful. For these reasons, the concept of a "model" has been introduced in the SIMTSX description interface.

The use of this function involves two aspects :

- **the definition of models** archived in a library (explained in section 1 of part G)
- **the incorporation of a model** in the description of an application

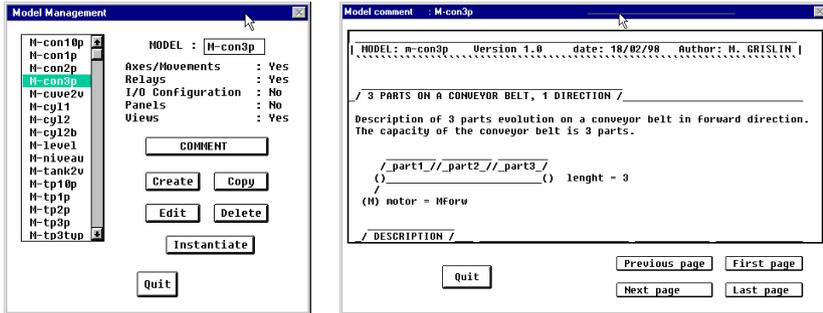
1.1-1 Incorporating a Model

Incorporating a model involves importing a model which has already been described and simulated in an application. The relevant model can be simply selected in the library provided in SIMTSX (select File > Model).

SIMTSX is supplied complete with a library of representative models, see section 1 of part H.

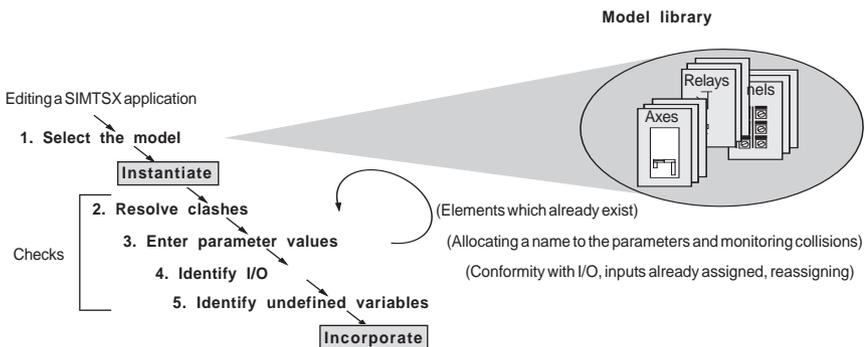
A model can be incorporated at any time when editing an application. However, if the model includes inputs or if some undefined variables in the model need to be identified with outputs, **the application I/O must have already been configured.**

A general comment is associated with the model as text which can be accessed using the **Comment** button in the Model Management window. Comments can also be added to the description variables to ensure the clarity of the model during use.



To incorporate a model in an application, first select the required model in the "Model Management" window, then click on the **Instantiate** button.

Before the model can be included in the application, a number of operations must be performed using various specific windows.



Note

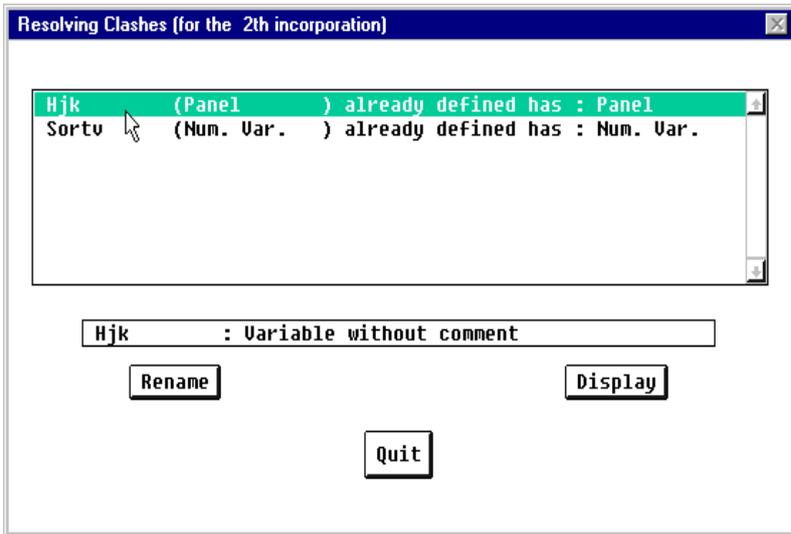
The process can be interrupted. It can be restarted using the **Adapt** button in the bottom left-hand corner of the main window for incorporating a model. As soon as the model can be included in the machine, this button is replaced by the **Incorporate** button.

1.1-2 Resolving Clashes

"Clashes" correspond to incompatibilities between the model and the application into which it is to be incorporated. These incompatibilities are due to the existence of defined variables with the same name in both the model and the application.

If there are any clashes, the list of affected variables appears in the Resolving Clashes window. These variables must be renamed before the incorporation process can continue.

For example :

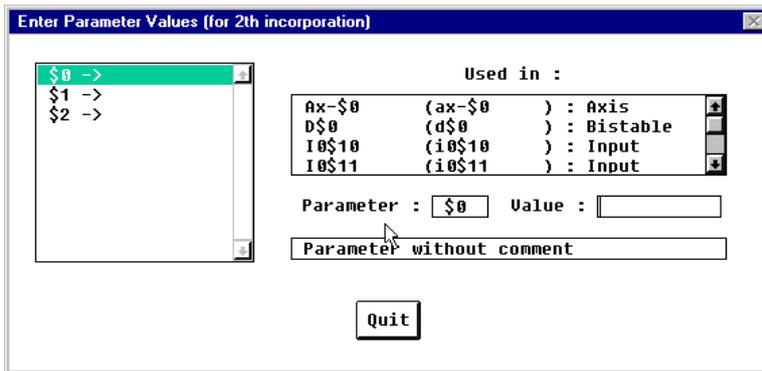


1.1-3 Parameter Values

If parameters have been used in the definition of the model to be incorporated, they must be given a value so that the model can be "instantiated".

The parameters are displayed in the left part of the window. Selecting a parameter displays the elements of the model in which it is used.

The value to be assigned to a parameter is entered in an editor and **confirmed** (the confirmation runs the automatic clash check). Only values which do not cause clashes are permitted.



1.1-4 Identifying the Inputs

If the inputs are defined in the model and their addresses do not conform to the application configuration or have already been assigned, these inputs must be allocated addresses which are both compatible and available.

1.1-5 Undefined Variables

A model can include "undefined variables", representing conditions which are external to the model.

Since a model is designed to be incorporated in the application, these undefined variables may correspond to existing variables.

The "Identify" button in the window for accessing undefined variables in the model is used to select the variable in the application with which one of these variables is to be identified. The "Rename" button is used to change the name of an undefined variable in the model.

1.1-6 Accessing the Model Description

The **Relays**, **Panels** and **Movements** buttons can be used to access the various components of the model to be incorporated. This means that the various elements in the model can be displayed and the name of the model variables can be changed, provided that they have not been parameterized.

- Special case of operator panels

An operator panel is made up of elements acting on the state of the variables (pushbuttons, switches) and indicator lamps and displays reflecting the state of the variables.

The definition of an operator panel is consistent if the variables associated with pushbuttons and switches are used in the relays or inputs, and if the variables associated with indicator lamps and displays are outputs or relays.

For an operator panel to be consistent, it must be accompanied by the description of a relay and/or input logic equations. The variables used in the elements of this operator panel must be defined and used in the relay and/or input logic equations.

If, however, the model only comprises an operator panel, it is definitely "inconsistent".

This consistency can be checked in the **Panels** window during the incorporation of a model using the **Check** button. If it is not consistent, the variables concerned are shown.

1.2 Tracing Instantiations

A model can be instantiated several times in an application. A trace is maintained of the instantiations of models in an application. This trace is accessed via the **Tools** menu of the description interface.

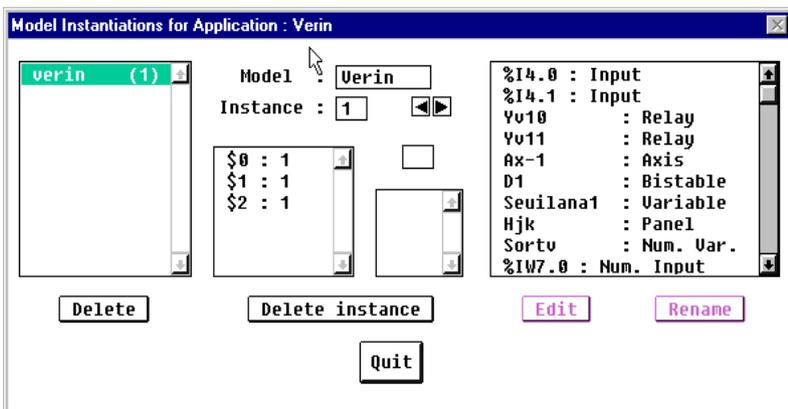
The names of the various models incorporated are listed along with the number of corresponding instances. If one of the model names is selected in the list on the left, the value given to the parameters for the first instance of this model is shown. The application variables corresponding to this instance are displayed on the right.

If several instances exist for the selected model, they can be displayed using the arrows to the right of the number of the instance displayed in the central part of the window. The values given to the parameters and the corresponding variables are shown.

A particular instance of a model can be accessed by selecting the values which have been given to the model parameters during incorporation in the application. To do this, click, in the central part of the window, on the parameter for which you wish to select the value. All the values taken by this parameter are then displayed, including values which have been selected for the previous parameters. If one of these values is selected, the first instance for which the corresponding parameter has this value is displayed.

When an instance is selected, the corresponding variables are displayed on the right-hand side of the window. These variables can be edited and their name can also be changed. If one of these variables is edited, the **Previous** and **Next** buttons in the corresponding edit window are used to move through all the application variables originating from the model variable. This makes it easier to correct the instances of an application at a later date.

The **Delete** button is used to remove all instances of this model from the application description. The **Delete Instance** button is used to remove the selected instance only from the application.



Section	Page
1 Describing the Axes	1/1
1.1 Main Axis Description Window	1/1
1.2 Creating and Editing Axes	1/2
1.3 Axis Characteristics	1/3
1.4 Associations Up and Downstream of the Axis	1/5
1.5 Standard Axis	1/5
1.6 Sampled Axes	1/6
1.7 Creating and Editing Sensors	1/6
1.7-1 Standard Configuration Suggested by the Interface	1/7
1.7-2 Describing a Sensor by Evolutions	1/8
1.7-3 Checking the Sensor Description : Display	1/9
1.8 Creating and Editing Movements	1/10
1.8-1 Taking Account of Variable Speed in Movements	1/11
1.9 Creating and Editing Bistables	1/13
1.10 Creating and Editing Variables	1/14
1.10-1 Boolean Variables	1/14
1.10-2 Numeric Variables : Definition and Evaluation	1/15
1.10-3 Taking Account of Numeric Variables in Logic Expressions	1/19
1.10-4 Links With the I/O	1/19
1.11 Accessing "Undefined Variables"	1/20
1.12 Checking the Axis Description	1/21

Section	Page
2 Describing the Relays	2/1
2.1 Introduction	2/1
2.2 Creating a Relay	2/2
2.3 Editing Relays	2/2
2.4 Taking Account of Time Delay Contacts in Relay Equations	2/3
3 Describing the I/O	3/1
3.1 Introduction	3/1
3.2 Configuring I/O Modules	3/1
3.2-1 Operating Modes for the I/O Configurator	3/1
3.2-2 Configuring Micro PLCs	3/6
3.2-3 Configuring Premium PLCs	3/12
3.2-4 Configuring Quantum PLCs	3/18
3.2-5 Multi-PLC Configuration	3/22
3.3 Editing the I/O	3/23
3.4 Editing Numeric I/O	3/24
3.5 Editing Words	3/25

Section	Page
4 Describing the Operator Panels	4/1
4.1 Introduction	4/1
4.2 Creating and Editing Operator Panels	4/1
4.2-1 Creating an Element	4/2
4.2-2 Modifying an Element	4/2
4.2-3 Deleting and Restoring an Element	4/2
4.3 Defining Operator Panel Elements	4/2
4.3-1 Pushbuttons	4/3
4.3-2 Latching Pushbuttons	4/4
4.3-3 Indicator Lamps	4/4
4.3-4 Illuminated Pushbuttons	4/5
4.3-5 Switch	4/5
4.3-6 2-position Switches	4/6
4.3-7 3-position Switches	4/7
4.3-8 N-position Switches	4/8
4.3-9 Thumbwheels	4/9
4.3-10 Hexadecimal Displays (or "Digits")	4/10
4.3-11 Text	4/10
4.3-12 Separator	4/10
5 Describing the Views	5/1
5.1 Introduction	5/1
5.2 Creating and Editing Views	5/1
5.2-1 Axes	5/2
5.2-2 Variables	5/4
5.2-3 Text	5/5
5.2-4 Decoders	5/6
5.2-5 Lines	5/6

Section	Page
5.3 Editing Aids for Views	5/7
5.3-1 Copy/Paste	5/7
5.3-2 Moving Elements	5/7
5.3-3 Printing Views	5/7

1.1 Main Axis Description Window

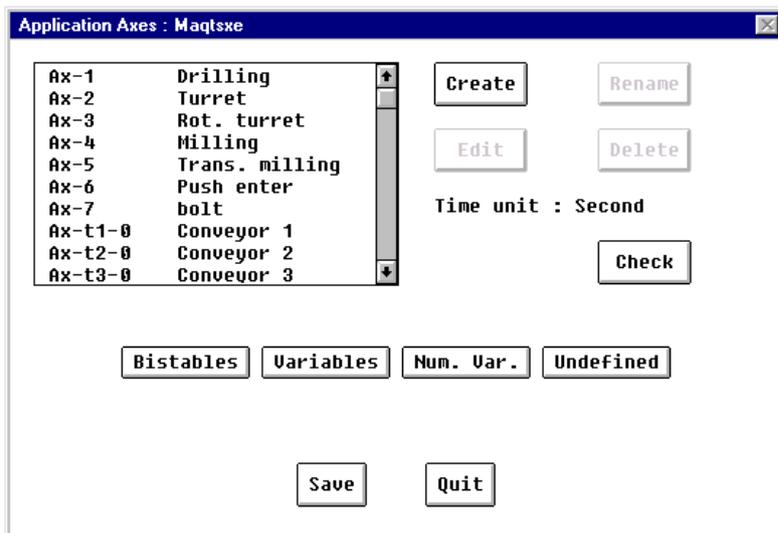
This window lists the axes which comprise the description. It has buttons for creating a new axis, and editing, renaming or deleting existing axes. The time unit used for configuration can be modified at this stage.

The default unit is seconds, but this can be replaced by 1/100th of a minute by clicking on the text **second** which appears in the window. The selection of the unit is important as it affects the representation of the simulation traces (see section 5, part E).

The **Check** button is used to signal any anomalies in the description.

The description variables and bistables (Boolean and numeric) can be associated with more than one axis. Buttons are used to gain access to the list of bistables and variables used in all the axes.

The **Undefined** button accesses the list of variables used in the description which have not been defined.



1.2 Creating and Editing Axes

A mechanical element of the installation can be described using an "axis".

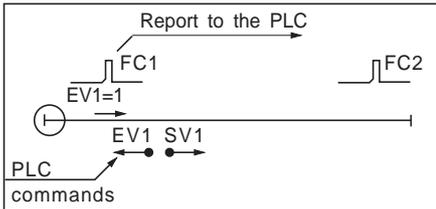
An axis represents a physical evolution of the mechanical element concerned, such as a cylinder, a belt or conveyor (evolution of parts), a vat (evolution of level), a valve (evolution of the opening position and therefore of the flow rate), etc.

Sensors located on the axis are used to represent exchanges with the PLC.

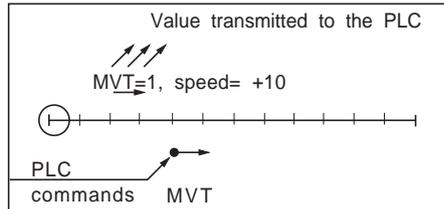
There are two types of axis :

- the standard axis, which is linked to sensors so that it can notify the PLC of the position of the actuator at certain well-defined points (characterized by its sensors)
- the sampled axis, which follows a set step so that it can provide the PLC with the position of a moving part or the value of a physical measurement continuously.

Standard axis



Sampled axis



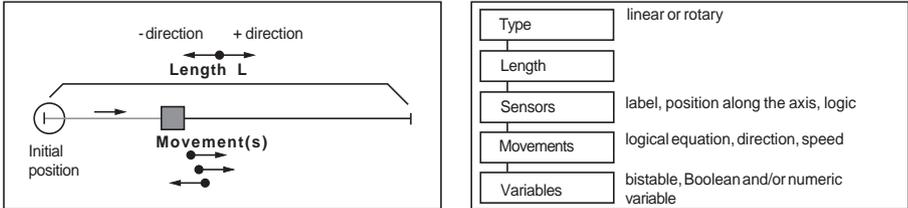
The creation of an axis starts with the definition of the name used to identify it.

An axis is named using the format **ax-<no>**, where <no> is the number of the axis incremented according to the description. This number can be modified by the user, and can be linked to a string of up to 7 characters.

The axis configuration window can be used to enter the individual axis characteristics and to access the variables associated with the axis.

1.3 Axis Characteristics

In addition to the axis label (string of up to 16 characters) the individual axis characteristics, entered directly in the edit window, are as follows :



Axis type

By default, the axis is linear, that is, evolutions are limited between a start and an end limit. This corresponds to the majority of actuators (cylinders).

An axis can be rotary, i.e. the end limit blends into another start limit (turret with various tools, transfer of parts, etc).

Length

This is a strictly positive number, expressed in the unit selected for the entire application.

Cam width

By default, this width is zero. In practice this cam is hardly used : the **activation width** associated with each of the axis sensors is often preferred.

Initial position along the axis

The default position is at the start of the axis, but it can be at the end of the axis or determined as a function of one of the axis sensors.

Sensors

The sensors correspond to the physical sensors installed on the mechanical element represented by the axis. The data from these sensors is returned to the PLC.

Movements

The movements are equations which manage the physical evolution logic of the mechanical element represented by the axis. Movements are generally functions of PLC outputs, via the electrical part, and an evolution speed corresponds to the dynamics of the mechanical element concerned.

Variables

Boolean or numeric variables can be associated to axes in order to manage the system evolutions.

Type

Length

Cam

Initial position

Sampling

Up/downstream associations

Management of axis evolutions, etc

The **OK** button is used to accept the data which has just been described.

The **Cancel** button is used to return to the previous set of values which were accepted.

The **Previous** and **Next** buttons can be used to edit the previous and next axes which have already been defined.

1.4 Associations Up and Downstream of the Axis

The sensors and movements are defined in relation to an axis : the association with this axis is thus immediate. For other variables, the association with the axes is made according to the two following principles :

- The upstream association consists of associating with the axis all the description variables used in defining the axis movements, except for the relays forming part of the general power supply (see section 3 of this part). When applied recursively, this association is used to pass on to the control system outputs.
- The downstream association consists of associating with the axis the description variables in which its sensors are used. As with the upstream association, it is applied recursively to the inputs.

The **Upstream** and **Downstream** buttons are used to display these variables in the window for editing an axis. If one of these variables is selected using the mouse, it can be edited.

This means that the description becomes a real network, representing the entire electrical and mechanical part of the installation which is being modeled.

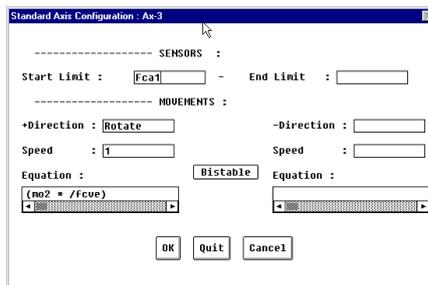
The buttons in the lower right-hand part of the screen open the windows for editing the sensors, movements, numeric variables, bistables and variables connected to the axis.

1.5 Standard Axis

The description of an axis can require the use of a certain number of windows, in order to define the associated variables. This is why the concept of a "standard axis" has been introduced.

The **Standard** button in the lower right-hand part of the window is used to open a page describing a typical axis, such as a cylinder.

A standard axis includes a maximum of two sensors, which must be positioned at the start and end of the axis. A maximum of two movements (positive and negative) are associated with it. The **Bistable** button is used to complete this description (see the bistable section).



1.6 Sampled Axes

A sampled axis, as compared with a standard axis, has an additional characteristic, its "step", which is defined by a numeric value. This corresponds to the desired precision for the physical measurement associated with the axis.

The "Sampled" box must be checked in order to obtain a sampled axis. The axis "step", i.e. the distance between 2 successive movements of the axis (excluding the real sensors) must be entered.

If the axis "ax-i" is sampled, the variable "ax-i" takes the value of the current number of steps in this axis and can be used as a numeric measurement for modeling. In particular, it can be converted into code.

On this basis, the modeling of an encoder over 10 bits can be performed by means of an axis with a length of 1023 and step of 1. Code conversion will be performed, if necessary, by means of a numeric variable, as will be seen later.

Apart from the number of steps of a sampled axis, two other data items, which are Boolean, are available and can be used in the combined description variables and sensors equations : they are "+ax-i" and "-ax-i". They are true momentarily when the number of steps in the axis is incremented and decremented respectively.

1.7 Creating and Editing Sensors

The **Sensors** button in the Axis Configuration window displays the main menu for editing the sensors associated with the axis. At this level it is possible, as for all the variables linked to an axis, to create, edit, rename and delete sensors.

The windows for creating and editing sensors are similar. The differences are as follows :

- **In the create window**, the mnemonic of the created sensor must be entered. The **Undefined** button is used to select the name of the sensor from the list of variables in the description which have not yet been used.
- **In the edit window**, the mnemonic of the sensor cannot be modified : this can only be done using the **Rename** function.

In this mode, you can switch from one sensor to another without exiting the edit window.

Two different ways of describing a sensor are offered in the description interface.

1.7-1 Standard Configuration Suggested by the Interface

This is the default option. It consists of describing the sensor using the four following pieces of information :

The position

This can be the start or the end of the axis, or a numeric value. In this latter case, the user should ensure that the start or end axis options are no longer selected, which can be done by confirming the value entered by pressing <Enter>.

The logic type

1. Positive logic : the sensor is active (its logic state is 1) when the cam acts upon it.
2. Negative logic : the state of the sensor is the opposite of the previous case.
3. Latch : the sensor changes to the active state from its position and remains so until the end of the axis.
4. Release : operation is the exact opposite of latch.

The activation width

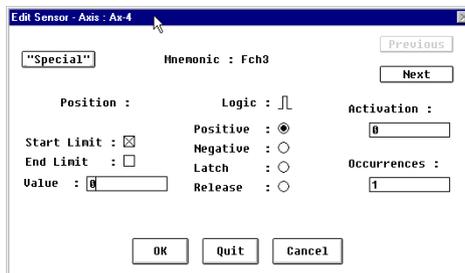
This information is the length over which the sensor is active (positive logic) or inactive (negative logic). By default, the sensor "inherits" the width of the cam associated with the axis.

The number of occurrences

This is 1 by default, and is used to repeat the sensor several times along the axis at regular intervals. This can be useful, for example, in the case of a rotary axis activating a sensor via cams distributed at regular intervals. This function is only active if the type of logic is positive or negative.

Note :

The activation width is different from the concept of the "cam" attached to the axis. If a linear axis has a cam which is not zero, its length is reduced accordingly and the position of each of its sensors is advanced, as it is considered that the movement of this cam is "internal" to the axis.



The **Previous** and **Next** buttons can be used to edit the previous and next axes which have already been defined.

The **OK**, **Quit** and **Cancel** buttons can be used to exit the creation and modification windows.

1.7-2 Describing a Sensor by Evolutions

The "standard" way of describing a sensor may be unsuitable in certain cases. This is particularly true when an axis sensor is activated by cams which are of variable widths and/or placed at irregular intervals. This problem can be avoided by describing a sensor as required.

By clicking on the **Special** button in the top left-hand corner of the Edit Sensor window the content of this window can be modified.

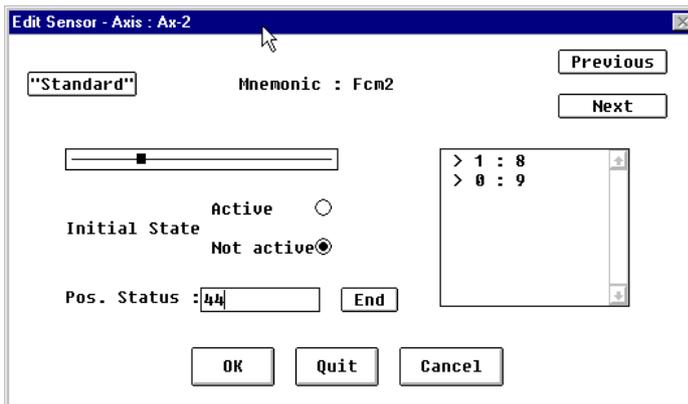
If the data has been entered in "standard" mode, it is displayed here in the form of a trend diagram on the one hand, and a series of logic evolutions of the sensor on the other hand, in a scroll-down list on the right-hand side of the window. The initial state is indicated via a 2-position selector switch.

The sensor configuration consists of giving the series of evolutions affecting the sensor starting from an initial state. These evolutions are entered via the editor at the bottom of the window.

Pressing the <Enter> key for confirmation adds the evolution position to those displayed in the scroll-down list, if the value entered is a positive number less than or equal to the length of the axis. The trend diagram representing the sensor evolutions is simultaneously updated.

The **End** button to the right of the editor is used to display the length of the axis with which the sensor is associated. This is used to set an evolution at the end of an axis after confirmation, or simply to recall this length.

An evolution is removed by selecting it in the scroll-down list with the mouse. The trend diagram is updated automatically.

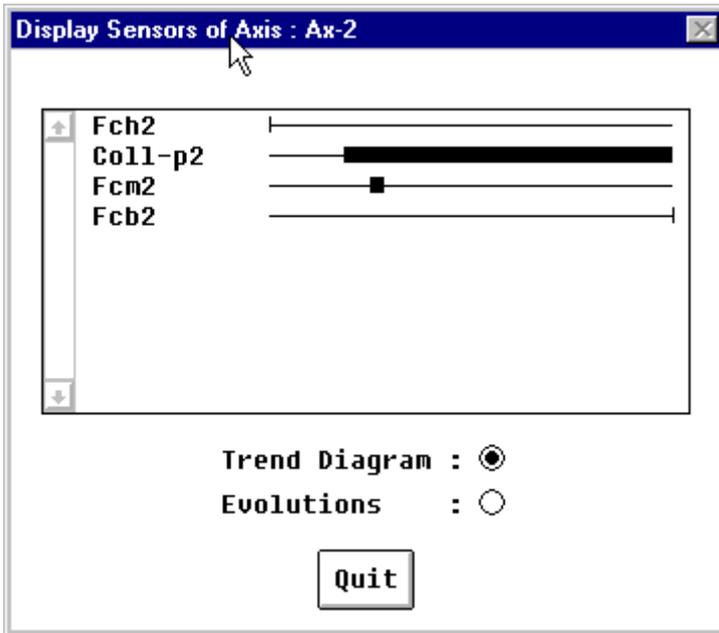


1.7-3 Checking the Sensor Description : Display

To check if the sensor configuration created corresponds to the desired configuration, a Display function is available.

The **Display** button in the sensor edit menu is used to display a window showing a trend diagram of the evolutions of the axis sensors.

To gain a more precise view of the positions at which these sensors change state and of the order in which they occur, the user can also display the succession of evolutions of these sensors all the way along the axis.



Note :

At this level of representation in the form of a trend diagram, it is possible to edit the sensors by clicking on their name using the mouse. You can also correct an incorrect configuration.

1.8 Creating and Editing Movements

The **Movements** button in the axis configuration window displays the main menu for editing movements associated with an axis. It is possible to create, edit, rename and delete movements.

Editing a movement consists of declaring :

The mnemonic

This can be entered in the editor or selected from the mnemonics of the terminal block outputs (via the **Outputs** button).

In this latter case, clicking on one of these outputs creates the movement associated with its mnemonic and the address of the output for the equation.

The direction of the movement

A movement with a positive direction causes movement towards a higher position on the axis (towards the end of the axis). A movement with a negative direction causes movement towards the start of the axis.

The speed

This determines the duration of the movement of the axis between 2 sensors. The evolution speed can be a constant (value) or a numeric variable.

Note : An "infinite" speed leads to a zero movement duration from one sensor to another.

The logic equation

This determines the state of the movement. It is a logic function of description variables, relays, sensors, outputs or numeric comparisons.

The screenshot shows a dialog box titled "Edit Movement - Axis : Ax-2". The dialog contains the following elements:

- Mnemonic : Turret-dow
- Direction: Positive Negative
- Speed : 2
- Equation : $(km17 * ml2 * (/coll-p2 + coll-p2 * mb2))$
- Buttons: Previous, Next, Terms, Select Var., Dictionary, OK, Quit, Cancel

The **Terms** button provides information on the terms used in the active equation.

The **Edit Help** button opens a window in which the variables which are likely to appear in an equation can be selected. They can be transferred to the editors for entering these equations.

The **Dictionary** button accesses a user list which can be used to store, then Copy/Paste those equations which are used most often. This is in fact a "notepad" which can be accessed at the level of any of the equations in the description environment.

The **Copy/Paste** function is activated using the mouse. Select the equation to be copied by holding down the left mouse button, release the button to paste, then double-click on the position where you want to paste. The equation is stored until a new selection is made. **This function is available throughout the SIMTSX editor.**

The **Previous** and **Next** buttons can be used to edit the equations for the previous and next movements which have already been defined for the axis. Since these equations generally use the same terms, this avoids retyping an equation in full : it is often simply necessary to add a few variables to gain the new movement equation.

The **OK**, **Quit** and **Cancel** buttons can be used to exit the creation and modification windows.

1.8-1 Taking Account of Variable Speed in Movements

The speed of a movement can be the value of a numeric variable. If the variable has a negative value, the direction of the movement is the opposite of that defined during configuration.

For example :

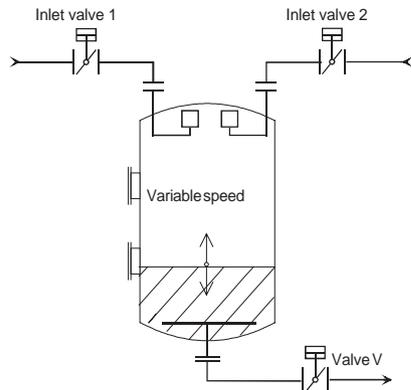
Let us take the example of a vat which is filled by two "discrete" control valves and emptied by a third.

To describe the change in level of this vat, 5 movements - or 7 if the inflows have different values - must be defined and activated depending on the possible opening combinations of the valves.

Moreover, if the valves are proportional valves, the concept of movement at constant speed is too limiting.

The example of the vat mentioned above can be processed more simply using a variable speed, in the following way :

Suppose that valves valve1 and valve2 give inflows which cause a variation in the level of the vat with speeds of 10 and 5 respectively, while the outflow via valve3 causes the level of the vat to fall at a speed of 12.



From a default value of 0, the assignment table below can be used to calculate the variation speed resulting from the level of the vat, by algebraic summation as a function of the Boolean conditions representing the opening of the valves :

Speed

Assignments	Conditions	Explanation
0		the speed is 0 if valves 1, 2 and 3 are closed
+ 10	< valve1	the speed is plus 10 if valve 1 is opened
+ 5	< valve2	the speed is plus 5 if valve 2 is opened
- 12	< valve3	the speed is minus 12 if valve 3 is opened

The activation condition - or equation - of the filling/emptying movement is thus the logical sum of the opening of the valves, i.e. : valve1 + valve2 + valve3.

When the axis end limit is reached - the vat is full or empty - movement is deactivated. However, the variable which describes the speed can retain a non-zero value as long as one of the valves remains open.

On the contrary, if the variable speed of a movement is canceled, movement stops.

Since the speed of a movement can be varied, it is possible to do without the equation which defines the activation condition, required for movements at constant speed. This equation may however still be necessary for describing, for example, an energy condition (useless for this example of a vat, as the actuator is gravity).

1.9 Creating and Editing Bistables

The SIMTSX bistable description window contains the following elements :

The mnemonic

This is the name of the bistable. This name is used in equations of the elements downstream of the bistable, notably movements controlled by the bistable.

The latch term

This is the name of a description element upstream of the bistable. The bistable takes the state "1" on the rising edge of the latch term when the condition is verified. It keeps this value as long as the release has not been triggered, even if the latch term returns to state "0".

The release term

This is the name of a description element upstream of the bistable. The bistable takes the state "0" on the rising edge of the release term when the condition is verified. It keeps this value until a new latch has been triggered, even if the release term returns to state "0".

The condition

This is the name of a description element upstream of the bistable. The bistable cannot change state if the condition has not been verified. Generally, the condition is a term representing the power supply.

The screenshot shows a window titled "Create Bistable" with a close button in the top right corner. Inside the window, there are four rows of labels followed by input fields:

- Mnemonic :
- Latch :
- Release :
- Condition :

Below these fields is a button labeled "Select Var". At the bottom of the window are two buttons: "OK" and "Quit".

The **Undefined** button displays a window containing the description variables. This selection is limited to the undefined variables upstream of the axis, used for example in the axis movement equation.

The **Edit Help** button opens a window in which the variables which will be used in the latch, the release or the condition can be selected.

The **OK** and **Quit** buttons can be used to exit the creation and modification windows.

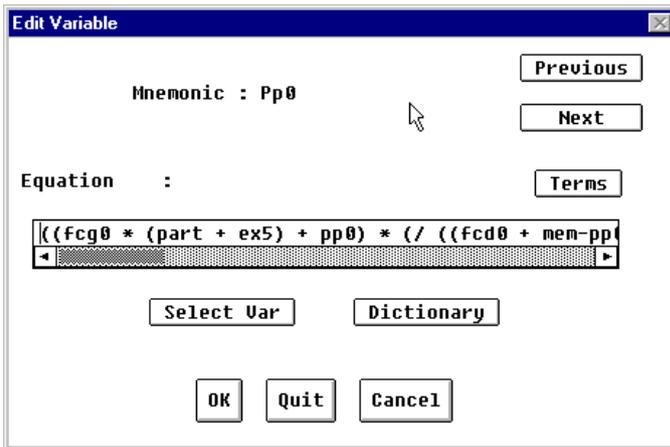
1.10 Creating and Editing Variables

1.10-1 Boolean Variables

The **Variables** button in the Edit Axis window displays the main menu for creating, editing, changing the name of and deleting Boolean variables associated with the axis.

The variables are associated with logic equations which are generally used to manage internal description evolutions (similar to the internal bit of the PLC program). The following elements may be used in the equations :

- sensors
- operator panel variables
- external variables
- relays
- outputs
- other variables



The **Terms** button provides information on the terms used in the active equation.

The **Edit Help** button opens a window in which the variables which are likely to appear in an equation can be selected. They can be transferred to the editors for entering these equations.

The **Dictionary** button accesses a user list which can be used to store, then Copy/Paste those equations which are used most often. This is in fact a "notepad" which can be accessed at the level of any of the equations in the description environment.

The **Copy/Paste** function is activated using the mouse. Select the equation to be copied by holding down the left mouse button, release the button to paste, then double-click on the position where you want to paste. The equation is stored until a new selection is made. **This function is available throughout the SIMTSX editor.**

The **Previous** and **Next** buttons can be used to edit the equations for the previous and next movements which have already been defined for the axis. Since these equations generally use the same terms, this avoids retyping an equation in full : it is often simply necessary to add a few variables to gain the new movement equation.

The **OK**, **Quit** and **Cancel** buttons can be used to exit the creation and modification windows.

1.10-2 Numeric Variables : Definition and Evaluation

Numeric variables can be used in the modeling of applications with significant analog features, which often means that the architecture of the automated system includes specialized modules such as analog I/O modules, axis control modules, fast counters, speed controllers, etc.

The Edit Axis window provides access to the configuration of numeric variables (via the **Numeric** button).

The value of a numeric variable is defined by an "assignments table", a series of assignment lines which may be conditioned. The assignment can be a constant, a numeric variable or a position on the sampled axis, or an output register.

The assignment can also be an elementary process which requires an "operator" and an "operand" :

- code conversion of another numeric measurement
- incrementation or decrementation when an expression changes to "1"
- arithmetic operation on a constant or another numeric variable : addition, subtraction, multiplication, division
- mathematical expression as a function of a constant or other numeric variable : cos, sin, exp, sqrt, log, abs
- logic expression : shift left or right (shl, shr), logic and/or

Opposite the assignment, a relative Boolean condition can be inserted :

- event on a Boolean variable (rising or falling edge)
- state of a Boolean variable
- equality or comparison test for numeric variables
- logic expression of several Boolean or numeric variables

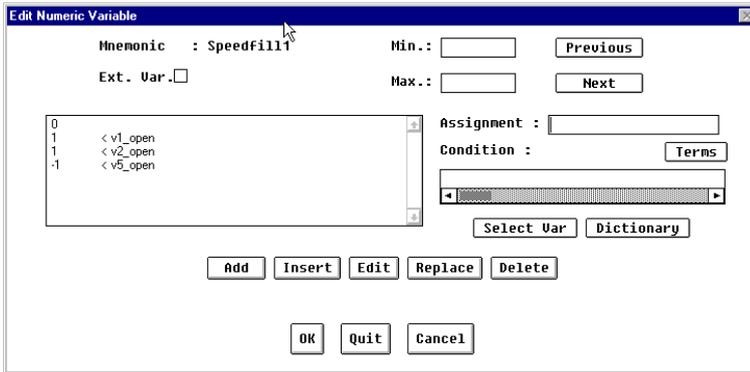
A numeric variable is defined by :

- its mnemonic (10 characters)
- a minimum and maximum value which can be assigned to it to determine its variation range
- its value, which is updated by a conditioned assignment table

The **Add** and **Insert** commands can be used to complete the assignment table using the contents of the Assignment and Condition fields.

The **Edit**, **Replace** and **Delete** commands relate to the line selected in the assignment table.

Unlike external Boolean variables which appear implicitly when an undefined term is entered into a logic expression, external numeric variables must be defined explicitly. This is done by checking the **Ext. Var.** box in the configuration window.



The **Terms** button provides information on the terms used in the active equation.

The **Edit Help** button opens a window in which the variables which are likely to appear in an equation can be selected. They can be transferred to the editors for entering these equations.

The **Dictionary** button accesses a user list which can be used to store, then Copy/Paste those equations which are used most often. This is in fact a "notepad" which can be accessed at the level of any of the equations in the description environment.

The **Copy/Paste** function is activated using the mouse. Select the equation to be copied by holding down the left mouse button, release the button to paste, then double-click on the position where you want to paste. The equation is stored until a new selection is made. **This function is available throughout the SIMTSX editor.**

The **Previous** and **Next** buttons can be used to edit the equations for the previous and next movements which have already been defined for the axis. Since these equations generally use the same terms, this avoids retyping an equation in full : it is often simply necessary to add a few variables to gain the new movement equation.

The **OK**, **Quit** and **Cancel** buttons can be used to exit the creation and modification windows.

Details of assignments	Example
<ul style="list-style-type: none"> · Simple assignment (<i>no space before the assignment</i>) <ul style="list-style-type: none"> Output register Output table <output>[n] (n is an integer from 1 to 32) Constant as an integer Constant expressed in hexadecimal Other numeric variable %QW03,1 %Q03,1[5] · Coding <ul style="list-style-type: none"> btd <numeric variable> dtb <numeric variable> btg <numeric variable> gtb <numeric variable> bdr <numeric variable> · Increment, decrement <ul style="list-style-type: none"> +, - (<i>there must be a space after the operator</i>) · Arithmetic operations <ul style="list-style-type: none"> + <numeric variable> - <numeric variable> * <numeric variable> / <numeric variable> (<i>there must be a space after the operator</i>) · Reading the "number of steps" for a sampled axis <ul style="list-style-type: none"> ax-i · Mathematical functions <ul style="list-style-type: none"> cos, sin, exp, sqrt, log, abs <numeric variable> (<i>there must be a space after the operator</i>) · "Logic" functions <ul style="list-style-type: none"> shl n shift left (n is an integer from 1 to 15) shr n shift right and n logic AND or n logic OR xor n exclusive OR (<i>there must be a space after the function</i>) 	<p>13 H00FE Level</p> <p>Binary → decimal conversion Decimal → binary conversion Binary → gray conversion Gray → binary conversion Binary → BDR+3 code conversion</p> <p>+ position - HFF02 * scale / 100</p> <p>shr 3</p>

During simulation, a numeric variable is evaluated by executing sequentially on each simulation "cycle" assignments whose condition has been verified. At the end of a cycle the value of the variable will be equal to the result of the last assignment made. In the case of incrementation or decrementation, the operation is performed when the condition changes to its true state.

Let us suppose that we wish to define a variable Y which continuously verifies the following equation:

$$Y = a.x^2 + bx + c$$

where X is a numeric variable.

The assignment table for Y will be as follows :

Assignments	Conditions	<i>Explanation</i>
a		$Y = a$
* x		$Y = \text{previous value} * x \text{ or } a*x$
+ b		$Y = \text{previous value} + b \text{ or } (a*x)+b$
* x		$Y = \text{previous value} * x \text{ or } ((a*x)+b)*x$
+ c		$Y = \text{previous value} + c \text{ or } (((a*x)+b)*x)+c \text{ to or } y = a.x^2 + bx + c$

All of the numeric variables present in the modeling of an application are processed according to the same principles as Boolean variables : an evaluation is made whenever one of the terms in the variable assignment table changes state, or value if another numeric variable is concerned.

All the variables in which this term is used are evaluated in a "synchronously", i.e. new values are only assigned when all the variables concerned have been evaluated. This ensures that the result of the evaluation does not depend on the order in which these variables have been described.

This makes it possible, for example, to model a shift register containing numeric values. Let us suppose that each position in the register is represented by a numeric variable "Ri". To be able to shift to the "right" or to the "left" when the variables "shr" and "shl" respectively change to "1", the assignment table for Ri should be written as follows :

Ri:

Assignments	Conditions	<i>Explanation</i>
Ri-1	< +shr	<i>The variable Ri will be Ri-1 when the variable shr appears</i>
Ri+1	< +shl	<i>It will be Ri+1 when the variable shl appears</i>

1.10-3 Taking Account of Numeric Variables in Logic Expressions

Numeric variables and sampled axes can be used in Boolean description variables and in movement equations in two ways :

- Either by means of comparisons with other numeric variables or constants, for example : level > 10
- Or by bit extraction : "varnum,n" is true if the nth bit (counting from 0) of the binary coding of the value of variable "varnum" is "1".

1.10-4 Links With the I/O

SIMTSX exchanges numeric data via the %IW%QW, %IW%QW registers of the PLC. Communication with the PLC is as follows :

- To send a numeric value to the simulator, simply assign this value to a PLC output register, and, if necessary, perform rescaling. For example, the speed of a movement can be read from the PLC by the following assignment table :

Assignment	Conditions
0	
+ %QW1,2	< valve1
+ 5	< valve2
- 12	< valve3

- Reciprocally, a numeric value is transmitted to the PLC by copying it to an input register, for example IW20,0 = position, position being a numeric variable calculated by **SIMTSX**.

An encoder feeding back a Gray code (or "reflected binary code") value to a PLC is modeled in the following way : the number of steps for a sampled axis is converted by means of ad hoc code conversion into a numeric variable, ie. "val-gray". The value of the val-gray variable is transmitted to the appropriate register in the PLC.

For example :

monitored axis (sampled) :	<i>ax-1 ; corresponds to the axis step</i>
numeric variable :	<i>val-gray</i>
assignment table :	<i>gray ax-1</i>
input register :	<i>%IW5,8</i>
assignment :	<i>val-gray</i>

1.11 Accessing "Undefined Variables"

These variables represent "modeling limits" corresponding to external conditions such as the power supply, the presence of parts entering the machine or sensors signaling the opening of a door.

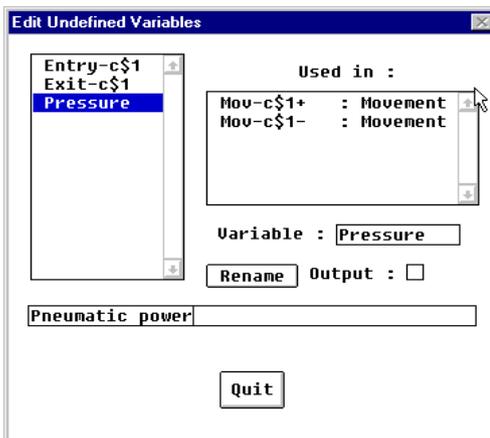
In the simulation environment, the state of these variables will be determined by an external action.

During description, variables can only be "temporarily undefined". For example, if the description starts with the definition of the I/O (see section 5 of this part), the sensors used in the input logic equations are "undefined variables" as the axes activating these sensors have not yet been described.

When a modeling entity is created, the list of existing undefined variables can be accessed to name the new element.

However, the existence of "undefined variables" in the description can also result in a data entry error in the definition of an element. In order to solve this problem, the **Undefined** button in the first window of the axis configuration interface is used to open a window giving the list of these variables used in the description.

Selecting one of these variables displays the elements of the description in which it is used. If one of these elements is selected using the mouse, the edit window for that type of element appears and can be used to make a correction, thus removing the "undefined variable" which is not required.



External numeric variables

Unlike Boolean external variables which appear implicitly when an undefined term is introduced into a logic expression, external numeric variables must be defined explicitly by the user, insofar as any numeric variable whose assignment table is empty becomes an external numeric variable. The user has the opportunity - during simulation - to introduce a value to this variable at any moment.

D

1.12 Checking the Axis Description

The **Check** button in the main Edit Axis window is used to activate a consistency check of the modeling.

The following anomalies are signaled :

- axes without sensors
- linear axes which do not have at least one movement for each direction
- rotary axes without a movement
- unused variables

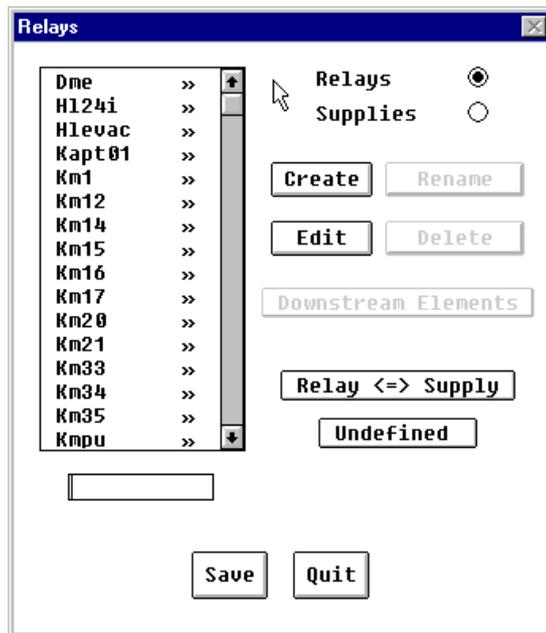
D

2.1 Introduction

In SIMTSX, the description of relays consists of transcribing the electrical diagrams available in the documents and used in the application to be simulated in the form of Boolean expressions.

A distinction is made in the description interface between the "Supply" and the "Relays". The terms used in the supply correspond to the general application functions such as electrical power distribution, powering up the various subsets, etc. and are not linked directly to the actuators. The "Relays", however, are used directly in the control of movements.

This distinction does not affect the operation of the simulator, but is important when associating the description variables with the application axes.



2.2 Creating a Relay

To create a relay, a name is required (maximum 10 characters).

The relay to be described may have already been used to define an element in the model (for example another relay). In this case, an undefined variable exists with the name of the relay which is to be created : the **Undefined** button can be used to find this name. The name of the relay to be created can be the mnemonic of an output described in the I/O. The **Mnemonics** button accesses the list of outputs with their mnemonic. The user can then select the name of the relay to be created.

It is possible to associate a time delay with a relay, by checking the **Delayed** box in the window for creating these elements.

The duration of the time delay must then be specified : its value is entered via an editor. The unit is that which was selected by the user for the application configuration in the axis creation window.

By default, the relay is an on-delay, which means that the time delay contact appears some time after the relay logic equation has been checked. By clicking on the text **on-delay** in the creation window, the relay can be changed to an off-delay, which means that the time delay contact will be released just after the relay logic equation becomes "false".

Note

Time delays are not taken into account in the relay itself, but only in the fax-<relay> and dax-<relay> variables which are associated with it.

The creation of a relay is accepted once the **OK** button has been pressed. The creation window stays on the screen. If the name of the relay is modified, this creates a new relay.

The screenshot shows a dialog box titled "Modify Relay". It has a blue title bar with a close button. The main area contains the following elements:

- Name**: A text field containing "Dne". To its right are "Previous" and "Next" buttons.
- Equation**: A text field containing "(pw-r * qsg * qf2)". Below it is a scrollable area.
- Below the equation field are three buttons: "Terms", "Select Var", and "Dictionary".
- Delayed**: A checkbox that is currently unchecked.
- To the right of the checkbox are "On-delay" and "Duration" fields.
- At the bottom of the dialog are three buttons: "OK", "Quit", and "Cancel".

2.3 Editing Relays

In edit mode, it is possible to modify the characteristics of the relay, including the equation and the time delay which may be associated with it.

2.4 Taking Account of Time Delay Contacts in Relay Equations

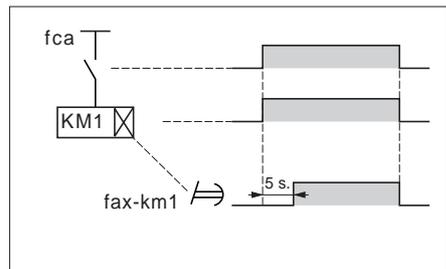
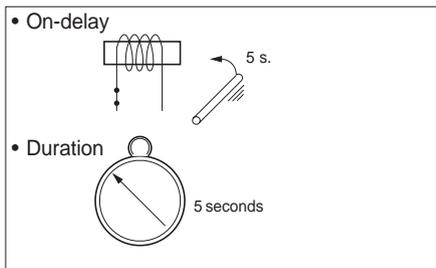
Time delays can be associated with relays. This is only useful if it is possible to use the corresponding time delay contacts in the relay equation.

For **on-delay** relays, the true state of time delay contact is fax-<relay> where "<relay>" is the name of the relay with which the time delay is associated.

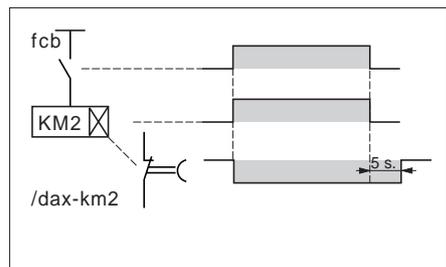
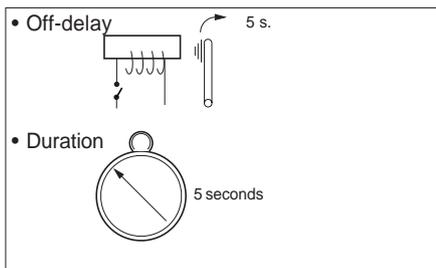
The active state of the time delay contact of an **off-delay** relay named <relay> is /dax-<relay>.

This comes from the internal representation of time delays. They are represented by axes named ax-<relay>. fax-<relay> and dax-<relay> are variables, automatically associated with these axes, which change to the "true" state as soon as the axes arrive at the end and start limits respectively.

On-delay



Off-delay



Example

Take an on-delay relay Km1, with a duration of 5. As soon as the equation associated with this relay is checked :

- km1 becomes true immediately.
- Fax-km1 becomes true after a time delay of 5 units (i.e. after the simulation time has advanced by 5 units).

The input linked to a time delay relay will change as described, depending on the input entered in the equation :

- <relay> it concerns the image of the relay
- fax-<relay> it concerns the image of the relay after the on-delay has elapsed
- dax-<relay> it concerns the image of the relay after the off-delay has elapsed

D

3.1 Introduction

In SIMTSX, the I/O configurator is used to describe the interface between the devices (sensors/actuators, physical measurements) and a program executed in a PLC.

The I/O configurator comprises the description of in-rack and remote I/O modules.

Note 1

The description of I/O in the SIMTSX configurator must correspond to the configuration of the PLC program. In particular, the slots must be identical for the simulated modules.

Note 2

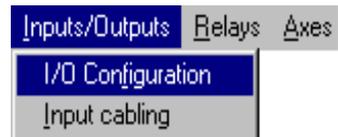
For the simulation of Micro and Premium PLCs, a theoretical **simulation module** must be configured in the PLC configuration in a **slot which must be free**.

3.2 Configuring I/O Modules

3.2-1 Operating Modes for the I/O Configurator

Accessing the SIMTSX configuration interface

The I/O configuration interface is accessed using the "I/O Configuration" command in the "Inputs/Outputs" menu.



Modes and principles of entry

Whatever the type of target PLC, the I/O is always configured using the same principles :

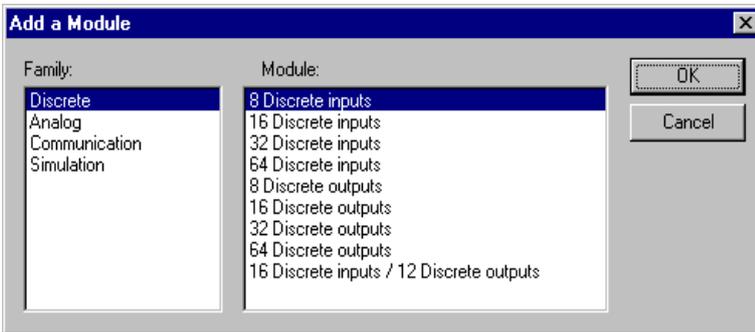
- One or more tabs detail the various types of I/O which can be configured (in-rack, remote) :
- 
- A green box moves dynamically over the various slots at the same time as the mouse pointer. This means that the configurable slots can be labeled easily.
 - A left click on a slot gives a **red box**. The **Edit** menu commands or the shortcut menu (right click) commands can then be used for this slot.

- A **Configuration** toolbar can be used to select the various tools available for entering the configuration :



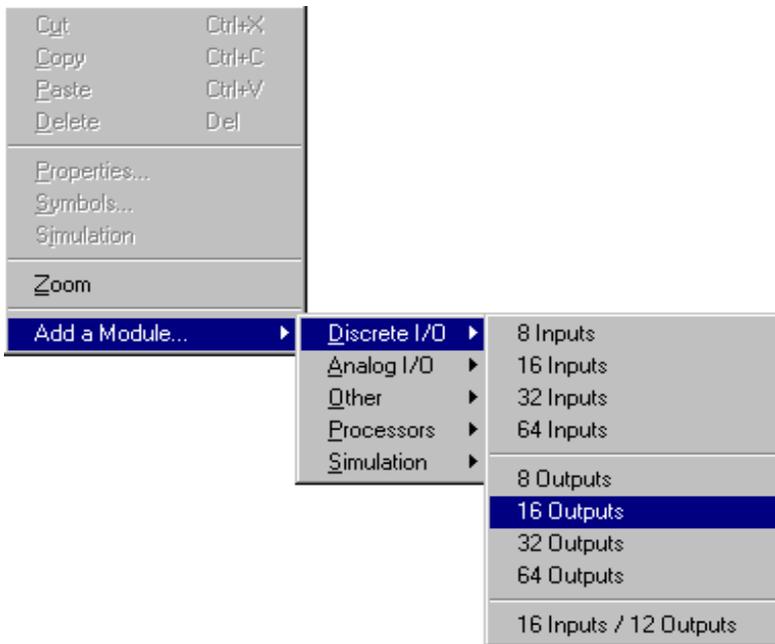
The first icon accesses the configuration tool and is used to position the I/O modules. The second icon is used to delete I/O modules. The third icon is used to select and deselect zoom mode.

- To add an I/O module, double-click on an empty slot. A selection list is displayed :



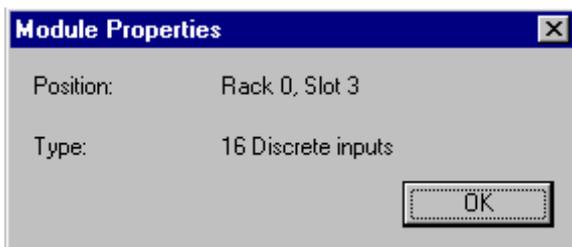
Use the **Family** field to select the type of module and the **Module** field to select the desired modularity (for example, 8 Discrete inputs). Confirm the selection with OK or by double-clicking on the selected module. The module is displayed in the configuration.

- I/O modules can also be added using the "**Add a Module...**" command in the shortcut menu (right click with the mouse on a slot). The various families and modules are then displayed in a dropdown menu :



- The **Zoom** command is used to maximize or minimize the field used for configurable slots.
- **Modules** can be **moved or copied** in a single action using the mouse (Drag & Drop) : Click on the module to be copied or moved with the left mouse button and keep the button depressed. A red box will appear around the module. Move the mouse to enable the move or copy action. To move the module, position the mouse on an empty slot and release the button. To copy the module, hold down the Control key and release the mouse on an empty slot.
- The same process can be used to **move or copy several modules** : Select the group of modules using the mouse by clicking on the top left of the zone to be copied or moved and keeping the button depressed while moving the mouse to create a box down to the bottom right of the last module to be copied or moved. All the selected slots appear in red. To move this group, position the mouse on an empty slot and ensure there is enough space on the right for all the selected slots. If the operation is possible, a green box is displayed around the target slots. To copy the group, follow the same procedure while holding down the Control key.
- The standard commands in the **Edit** menu and the shortcut menu can be used to **cut, copy, paste or delete** modules in the configuration. To do this, select one or more modules (red box) and use the command for the desired operation.

-
- The **Properties** command in the **Edit** menu and the shortcut menu can be used to display information about the selected module :



When configuring a Micro or Premium PLC, an additional **Simulation** command is available in the **Edit** menu or shortcut menu. It can be used to indicate to the simulator whether the module is simulated or not. By default, all modules are simulated, which means that the inputs are sent from the simulator to the PLC and the outputs are fed back from the PLC to the simulator. To stop the exchange of inputs for a module, simply delete the **Simulation** option for that module. To feed back the outputs for a module actually present in the PLC to the simulator, simply delete the **Simulation** option for that module.

Editing symbols and I/O comments

Whatever the type of target PLC, symbols and comments associated with the I/O are always edited using the same principles.

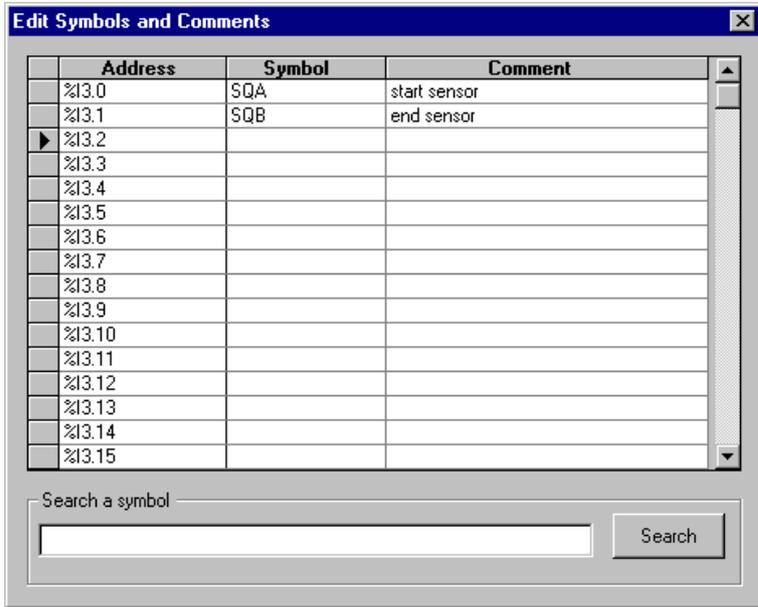
- The **Symbols** command in the **Edit** menu or the shortcut menu accesses a dialog box which can be used to enter symbols and comments.
- **Double-clicking** on a module also opens this dialog box.

The entry grid depends on the type of I/O. For example, for a Premium PLC the I/O symbols for Bus X are available if the dialog box is opened for a Bus X module.

When the entry grid is opened, the cursor is automatically positioned on the first address of the selected module.

To confirm entry of a symbol or comment, you must change line when the entry is completed.

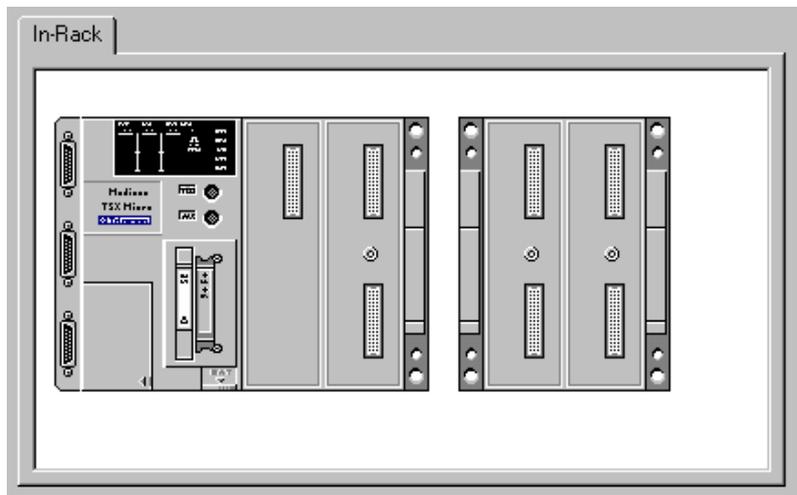
It is possible to search for a character string in the list of symbols and comments. Enter the character string you wish to find in the **Search a symbol** field. Each time the **Search** button is pressed, the cursor moves to the next occurrence of the character string and the associated symbol or comment is selected.



3.2-2 Configuring Micro PLCs

In-rack I/O

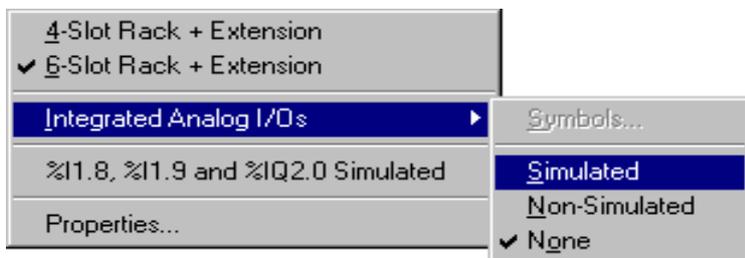
In-rack I/O for a Micro PLC are configured using the following screen :



Select the number of slots (4 or 6) on the basic rack for the target PLC : open the shortcut menu for the power supply unit/CPU or the properties window and select the desired type of rack.

Select the type of integrated analog I/O depending on the target PLC : open the shortcut menu for the power supply unit/CPU or the properties window and select the desired type in the **Integrated Analog I/O** submenu.

Select/deselect the simulation of I/O channels associated with the target power supply unit/CPU : open the shortcut menu for the power supply unit/CPU or the properties window and select/deselect the **%I1.8 %I1.9 and %Q2.0 Simulated** command.



Complete the configuration using the list of available modules for the various families :

- Discrete family
 - 8 inputs
 - 12 inputs
 - 32 inputs
 - 4 outputs
 - 8 outputs
 - 32 outputs
 - 16 inputs/12 outputs
 - 32 inputs/32 outputs
- Analog family
 - 4 inputs
 - 8 inputs
 - 2 outputs
 - 4 outputs
- Communication family
 - AS-i module
 - Extension link module

I/O on AS-i

It is only possible to enter an I/O module for configuring channels on AS-i in slot number 4 of the Micro PLC.

Select slot 4 with the mouse. Using the **Add a Module** dialog box or the shortcut menu, select the AS-Interface module from the **Communication** family.

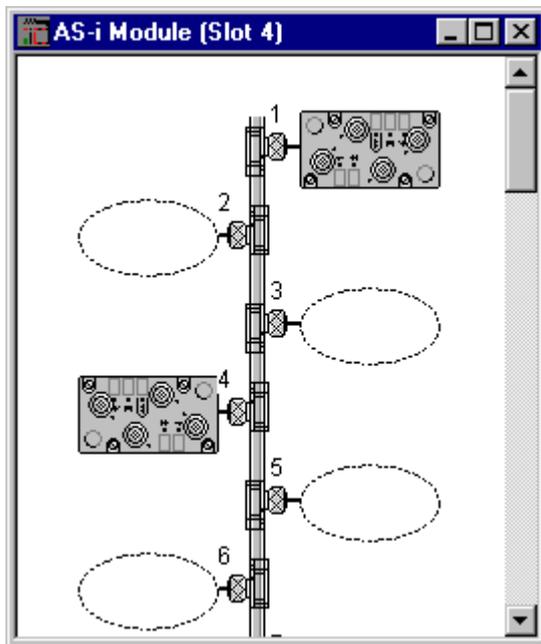
A dialog box which can be used to position the AS-i slaves opens automatically.

Double-click on a slot marked by a dotted ellipse to add an AS-i slave and generate 4 input channels and 4 output channels.

Double-click on a configured element to access the window for entering symbols and comments.

Click on the cross in the top right-hand corner to close the AS-i slave entry window.

To return to the slave entry window, double-click on the AS-i module or use the **Properties** command in the **Edit** menu or shortcut menu.



I/O on Extension link (Nano PLC)

It is only possible to enter an I/O module for configuring an extension link for Nano PLCs in slot number 4 of the Micro PLC.

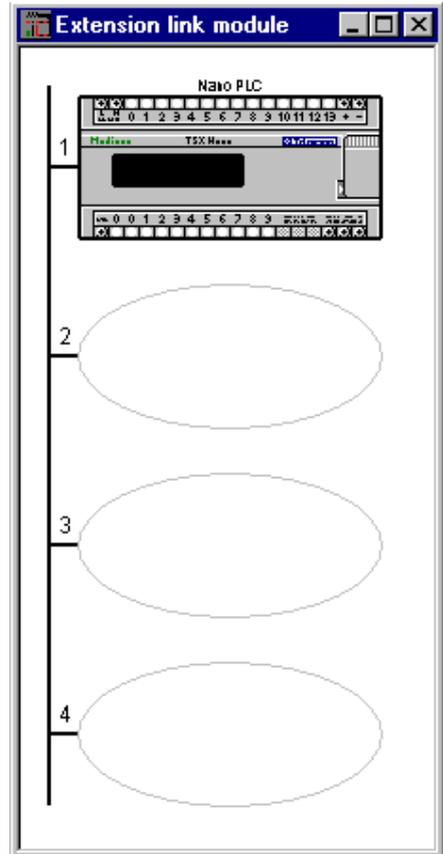
Select slot 4 with the mouse. Using the **Add a Module** dialog box or the shortcut menu, select the extension link module from the **Communication** family.

A dialog box which can be used to position the Nano PLCs opens automatically.

Double-click on a slot marked by a grayed-out ellipse to add a Nano PLC and generate 14 input channels and 10 output channels. Double-click on a configured element to access the window for entering symbols and comments.

Click on the cross in the top right-hand corner to close the extension link entry window.

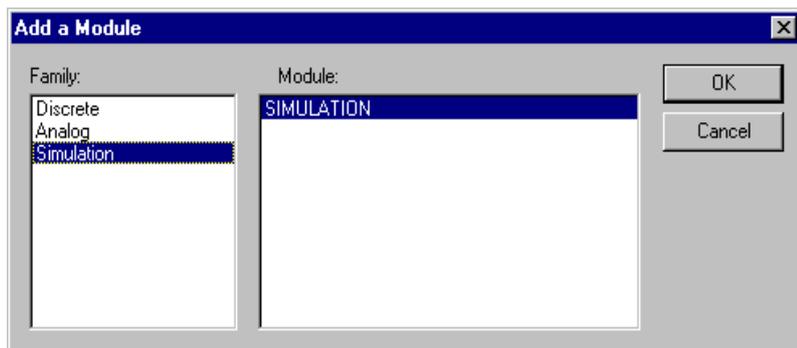
To return to the Nano PLC entry window, double-click on the extension link module or use the **Properties** command in the **Edit** menu or shortcut menu.



Simulation module

To perform a simulation with a Micro PLC, a theoretical simulation module must be placed in the configuration on the simulator side and the PLC side. **This module must be positioned in a slot which is available in both the I/O configurator of the simulator and the I/O configurator of the PLC programming software.**

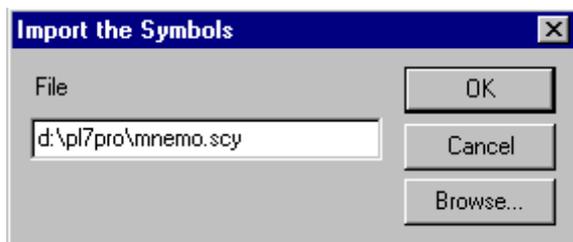
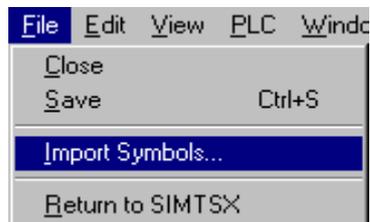
This module is available in the **Simulation** family of the **Add a Module** dialog box or from the shortcut menu.



Importing symbols and comments from PL7

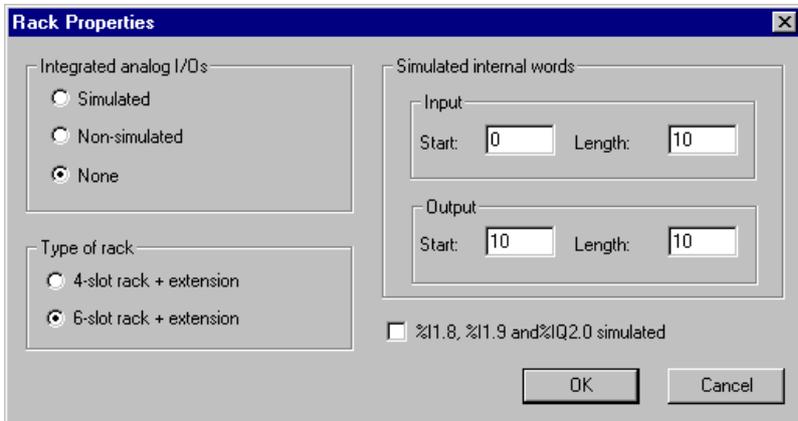
To reuse symbols and comments from PL7 and thus avoid entering them twice in the I/O configurator, an import function is available in the **File** menu.

In the dialog box, enter the path for the .SCY file generated by PL7 or use the Browse command to select it, and then press OK. The symbols and comments associated with modules found in the configurator are imported.



Words

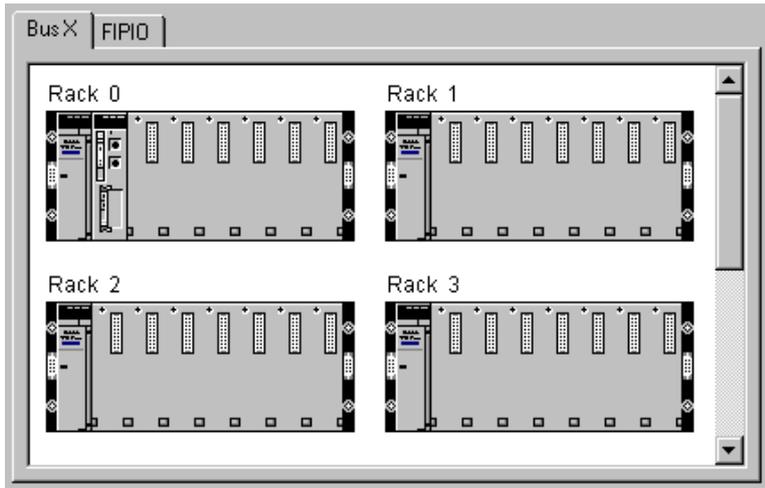
During simulation, it is possible to exchange %MW between the simulator and the PLC. To do this, simply fill in the fields given in the **Properties** dialog box of the power supply unit/CPU.



The internal words simulated as inputs are sent to the PLC by the simulator and the internal words simulated as outputs are fed back from the PLC to the simulator.

3.2-3 Configuring Premium PLCs

I/O for a Premium PLC are configured using the following screen :



Select the number of slots (4, 6, 8 or 12) for each rack used depending on the target PLC : open the shortcut menu for the power supply module or the properties window and select the desired type of rack.

Select the type of power supply module (single or double) : open the shortcut menu for the power supply module or the properties window and select/deselect **Double-Slot Power Supply**.

Select the type of processor (single/double slot, with/without integrated FIP module) : open the shortcut menu for the processor or the properties window and select the desired options. In the case of a processor with an integrated FIP module, a Fipio tab appears for entering the remote I/O.

I/O on Bus X

Complete the configuration on Bus X using the list of available modules for the various families :

- Discrete family
 - 8 inputs
 - 12 inputs
 - 32 inputs
 - 64 inputs
 - 8 outputs
 - 16 outputs
 - 32 outputs
 - 64 outputs
 - 16 inputs/12 outputs
- Analog family
 - 4 inputs
 - 8 inputs
 - 16 inputs
 - 4 outputs
 - 8 outputs
- Communication family
 - AS-i module
 - Interbus-S module

I/O on AS-i bus

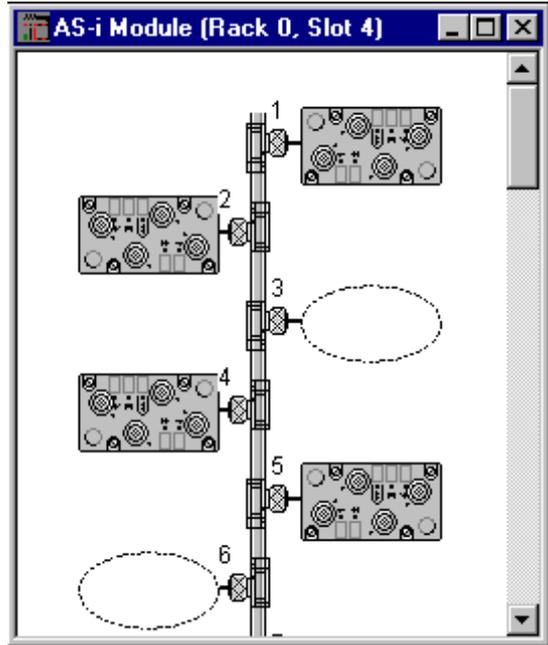
It is possible to enter an I/O module for configuring channels on AS-i in any slot of the Premium PLC. However, the total number of AS-i modules must not exceed eight.

Select a slot with the mouse. Using the **Add a Module** dialog box or the shortcut menu, select the **AS-Interface** module from the **Communication** family. A dialog box which can be used to position the AS-i slaves opens automatically. Double-click on a slot marked by a dotted ellipse to add an AS-i slave and generate 4 input channels and 4 output channels.

Double-click on a configured element to access the window for entering symbols and comments.

Click on the cross in the top right-hand corner to close the AS-i slave entry window.

To return to the slave entry window, double-click on the AS-i module or use the Properties command in the Edit menu or shortcut menu.



I/O on Interbus-S

It is possible to enter an I/O module for configuring an Interbus-S network in any slot of the Premium PLC. For one configuration, only one module can be used.

Select a slot with the mouse. Using the **Add a Module** dialog box or the shortcut menu, select the **Interbus-S** module from the **Communication** family. Adding this module generates 140 analog inputs and 140 analog outputs.

I/O on Fipio

On Fipio, the available slots are represented by grayed out TBX modules. The configuration entry principle is identical to that used for Bus X I/O.

Complete the configuration on Fipio using the list of available modules for the various families :

- TBX discrete family
 - 16 inputs
 - 16 outputs
 - 8 inputs/8 outputs
 - 8 inputs/8 mixed
- TBX analog family
 - 4 inputs
 - 2 outputs
 - 6 inputs/2 outputs
- Communication family
 - FSD profile
 - FED profile
 - FRD profile
 - Analog Momentum, 4 inputs
 - Analog Momentum, 4 outputs
 - Analog Momentum, 8 inputs
 - Analog Momentum, 4 inputs/2 outputs
 - Analog Momentum, 16 inputs
 - Discrete Momentum, 16 inputs/16 outputs
 - Discrete Momentum, 16 inputs/8 outputs
 - Discrete Momentum, 10 inputs/8 outputs
 - Discrete Momentum, 16 inputs
 - Discrete Momentum, 32 inputs
 - Discrete Momentum, 8 outputs
 - Discrete Momentum, 16 outputs
 - Discrete Momentum, 32 outputs
 - CCX 17
 - ATV 16

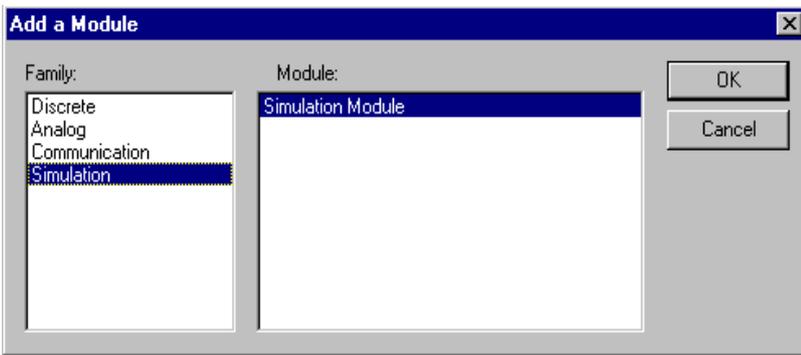
If the module that you want to simulate is not in the above list but is integrated in one of the standard profiles (FED, FSD or FRD) then you can replace it with the corresponding profile in the configurator. To find out which profile your module corresponds to, simply check in PL7 the number of channels (%I, %Q, %IW, %QW) it generates. A FED profile has 32%IW and 32%QW, a FSD profile has 8%IW and 8%QW and a FRD profile has 32%I and 32%Q.

Simulation module

To perform a simulation with a Premium PLC, a theoretical simulation module must be placed in the configuration on the simulator side and the PLC side (on Bus X or on Fipio). This module must be positioned in a slot which is available in both the I/O configurator of the simulator and the I/O configurator of the PLC programming software.

If there is at least one input or one output to be simulated on Fipio, then the simulation module must be positioned on Fipio after the last configured point. If this is not the case, it must be positioned on a Bus X slot.

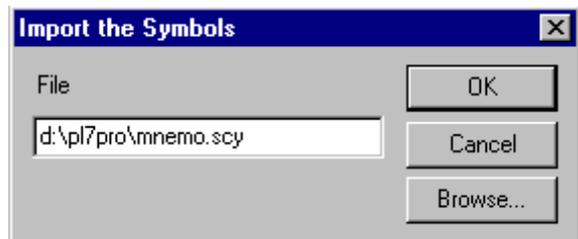
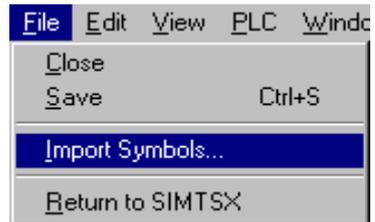
This module is available in the **Simulation** family of the **Add a Module** dialog box or from the shortcut menu.



Importing symbols and comments from PL7

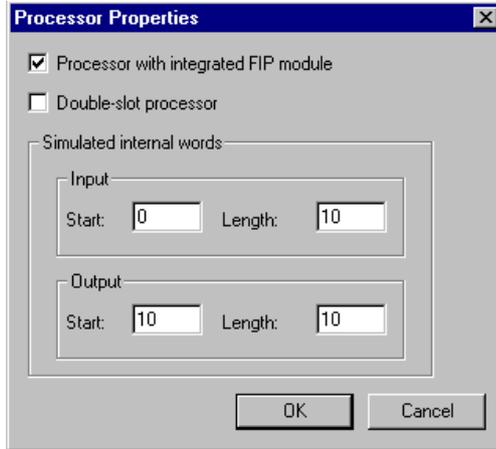
To reuse symbols and comments from PL7 and thus avoid entering them twice in the I/O configurator, an **import** function is available in the **File** menu.

In the dialog box, enter the path for the .SCY file generated by PL7 or use the **Browse** command to select it, and then press OK. The symbols and comments associated with modules found in the configurator are imported.



Words

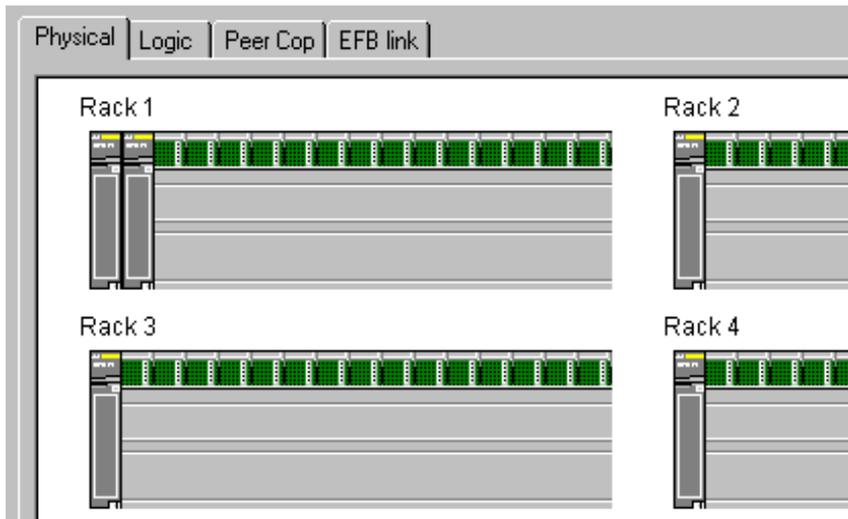
During simulation, it is possible to exchange %MW between the simulator and the PLC. To do this, simply fill in the fields given in the **Properties** dialog box of the processor.



The internal words simulated as inputs are sent to the PLC by the simulator and the internal words simulated as outputs are fed back from the PLC to the simulator.

3.2-4 Configuring Quantum PLCs

I/O for a Quantum PLC are configured using the following screen :



There are four tabs for this type of PLC :

- The first tab is used for physical configuration of the I/O modules to be simulated.
- The second tab (Logic) is used to adjust the parameters for addressing channels and the parameters for the working registers required for copying inputs to the PLC.
- The third tab (Peer Cop) is used to add, using a simple editor, registers 0x, 1x, 3x and 4x to be simulated. These do not correspond to the physical tab.
- The fourth tab (EFB link) indicates the number of SIMTSX EFBs which should be added at the start of the PLC program under CONCEPT, as well as the values for various parameters for each SIMTSX EFB.

Physical configuration

Complete the physical configuration using the list of available modules for the various families :

- Discrete family
 - 16 inputs
 - 24 inputs
 - 32 inputs
 - 8 outputs
 - 16 outputs
 - 32 outputs
 - 8 inputs/8 outputs
 - 16 inputs/8 outputs
 - 16 inputs/16 outputs
- Analog family
 - 1 input
 - 2 inputs
 - 8 inputs + 1 fault bit
 - 8 inputs + 2 fault bits
 - 1 output
 - 2 outputs
 - 4 outputs
 - 4 inputs/2 outputs + 1 fault bit
- Other family
 - 107 analog inputs
 - 6 analog inputs/6 analog outputs
 - 12 analog inputs/12 analog outputs
 - 12 analog inputs/13 analog outputs
 - 139 analog inputs/128 analog outputs
 - Interbus-S (267 analog inputs/264 analog outputs)

Logic configuration

The logic configuration accesses the addresses of configured I/O channels. These addresses are set automatically by the configurator each time a module is added in the physical configuration. They can be modified manually so that they correspond with the channels which are actually configured in the PLC :

Physical Logic Peer Cop EFB link							
	Slot	Start I	End I	Start O	End O	Start 4x	End 4x
▶	Rack:01 Empl.:04	100001	100016			400200	400201
	Rack:01 Empl.:05			000001	000008		
	Rack:01 Empl.:07	300001	300009			400500	400508
	Rack:01 Empl.:08			400001	400004		

The logic configuration can also be used to set the parameters required for the simulation to run smoothly. During simulation on a Quantum PLC, a certain number of 4x registers must be reserved for each adjacent field of discrete or analog inputs, so that the inputs can be written to the PLC.

The configurator automatically reserves these registers with default values. These values can be modified manually so that they do not overlap with the analog output fields.

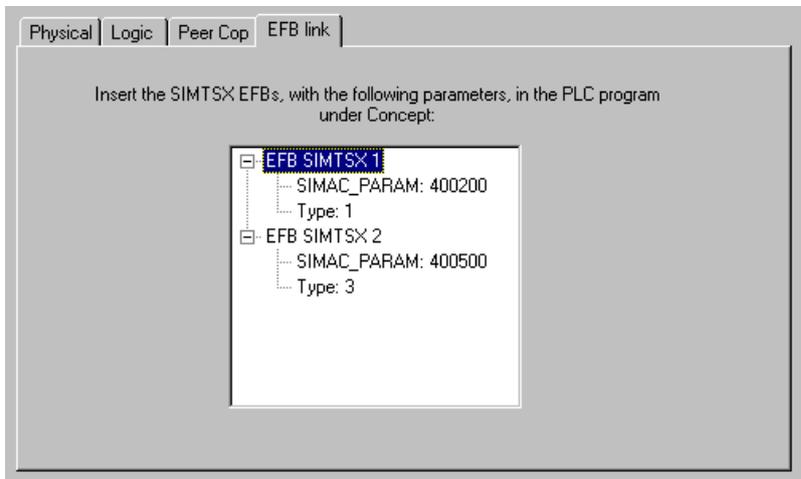
It is important to take note of the 4x registers corresponding to the start of each contiguous field. They are used to provide information to the simulation EFBs which are added in the PLC program. As many SIMTSX simulation EFBs must be added in the PLC program as there are adjacent discrete and analog input fields. For example, for the configuration described in the diagram for the next section, 2 simulation EFBs must be added. The values 400100 and 400200 are used to provide information to these EFBs (see online mode section).

Configuring with the editor

Using the editor in the Peer Cop tab, it is possible to add registers 0x, 1x, 3x and 4x to be simulated in SIMTSX. To add registers 0x and 4x, simply enter the start address and the number of registers to be added and click on the >> button. To add registers 1x and 3x, complete the Start and Length fields, and the Start Copy field which concerns the field for the working registers used to copy inputs to the PLC. This field must be selected in a free area of the 4x registers.

EFB link

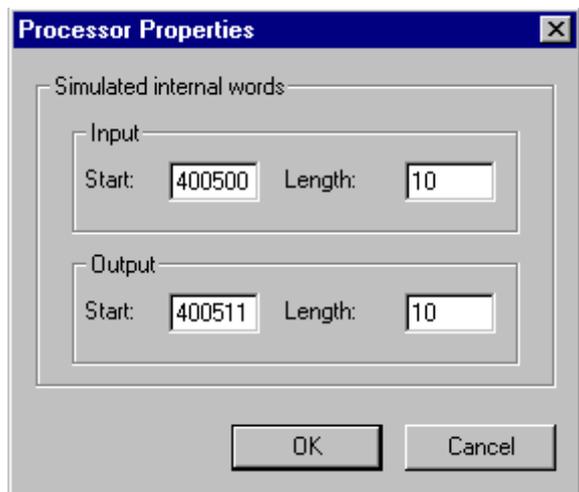
In this tab, a tree structure summarizes the number of SIMTSX EFBS which must be added at the start of the PLC program according to the actual logic and physical configuration. For the above example, SIMTSX indicates two SIMTSX EFBS in the tree structure, each with their corresponding parameters :



Words

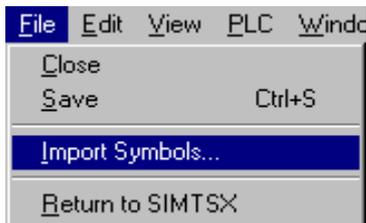
During simulation, it is possible to exchange 4x registers between the simulator and the PLC. To do this, simply fill in the fields given in the **Processor Properties** dialog box.

The internal words simulated as inputs are sent to the PLC by the simulator and the internal words simulated as outputs are fed back from the PLC to the simulator.

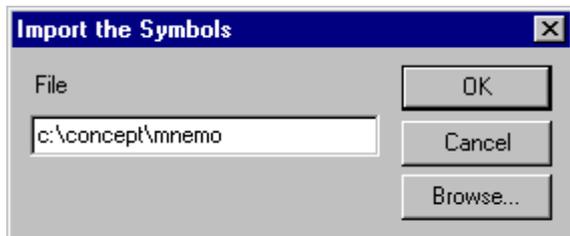


Importing symbols from CONCEPT

To reuse symbols and comments from CONCEPT and thus avoid entering them twice, an import function is available in the **File** menu.



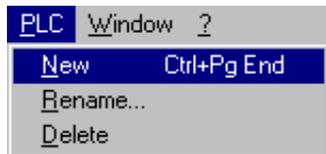
In the dialog box, enter the path for the file generated by CONCEPT or use the **Browse** command to select it, and then press OK. The symbols and comments associated with modules found in the configurator are imported.



3.2-5 Multi-PLC Configuration

To perform a multi-PLC configuration :

- Create a new application.
- Open the I/O configurator and select the first type of PLC to be configured from the list, then proceed with configuration of its I/O modules.
- Select the **New** command from the **PLC** menu to add a new PLC.



- Select the second PLC to be configured from the list.
- Etc., up to a maximum of 8 PLCs.

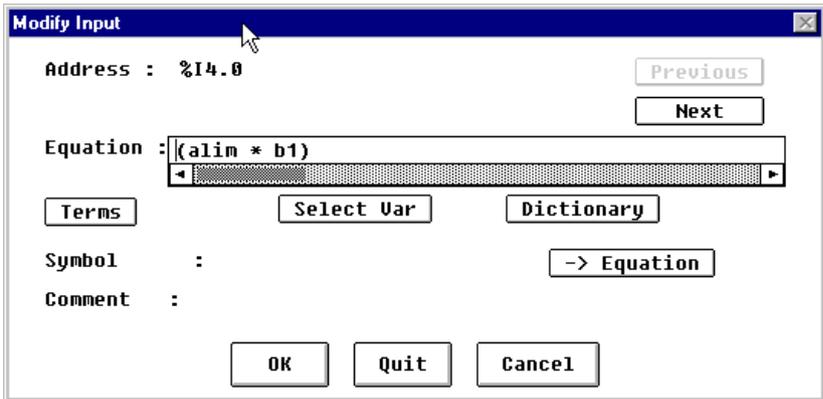
It should be noted that the I/O channels generated are preceded in all the editors by a number corresponding to the number of the PLC. This number increases automatically with each new PLC.

3.3 Editing the I/O

Configuring the I/O defines the I/O addresses.

This edit function is also used to define the input equations (**Input cabling**). A Help function is provided to help users when entering these equations, via the **Select Var** button.

This button opens a window in which the variables which are likely to appear in an input equation can be selected.



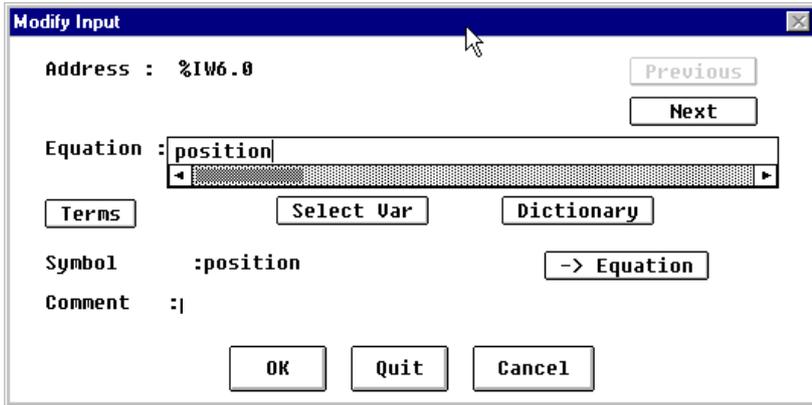
It is not possible to use an output in an input equation. Such a loop would require that an intermediate variable be defined. This can be a relay from the relay interface or a moving part description variable.

3.4 Editing Numeric I/O

Editing Numeric I/O can be used to determine the addresses for I/O and %IW, %QW registers.

For numeric registers, an edit window can be accessed where an assignment for the input register (the name of a numeric variable) can be defined.

D

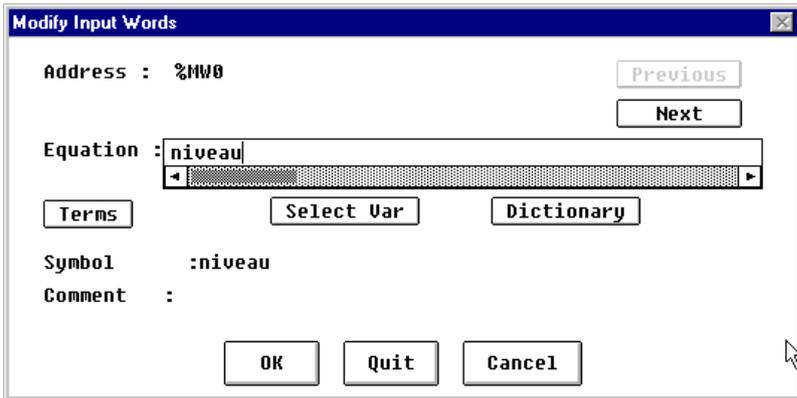


3.5 Editing Words

The exchange word table configuration determines the list of input and output words, which can then be edited.

An equation containing a term (the name of a numeric variable) can be associated with each word written by the simulator.

This can be changed by checking **Floating** in the double word edit window.



A word or double word can be referenced by a numeric variable.

Example

- Speed = %MW1 (speed is a numeric variable).
- The speed is copied into a speed of movement along an axis.
- As the axis is sampled, a monitoring variable can be associated with it, which can be copied to a word or double word : %MW10 = position.

D

4.1 Introduction

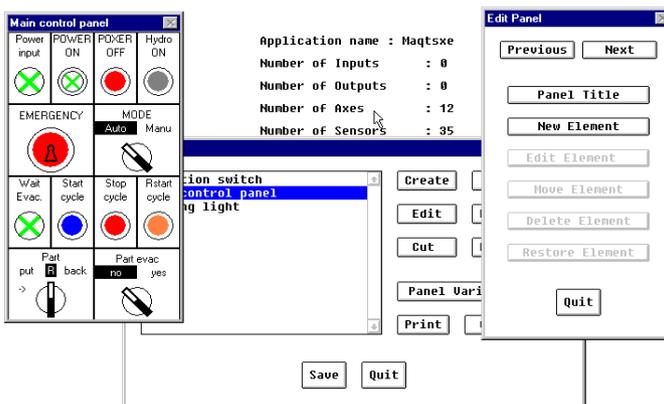
SIMTSX is used to graphically represent machine operator panels (pushbuttons, indicator lamps, etc).

Variables are associated with the pushbuttons and switches. These variables are activated according to the actions performed on these elements. They are used in the input logic equations or in the relays.

The indicator lamps and displays are controlled according to the outputs or relays and, if required, according to the application description variables.

4.2 Creating and Editing Operator Panels

To create an operator panel, it must first be given a title, which may be changed later. In the editing stage, the operator panel appears in the top left-hand corner of the screen. All the control buttons corresponding to the edit functions are on the right-hand side of the screen.



4.2-1 Creating an Element

Selection of the type of element to be created opens the corresponding definition window (see section 4.3. of this part, "Defining the operator panel elements").

Once the data entered has been confirmed, the new element is positioned in relation to an element which already exists in the operator panel selected using the mouse : this reference element then changes to reverse video.

4.2-2 Modifying an Element

Once an element has been selected from the operator panel being edited, the **Edit Element** button is used to call up the definition window specific to this type of element. The associated labels and the variable(s) can also be modified in this way.

4.2-3 Deleting and Restoring an Element

An operator panel element, previously selected using the mouse, can be deleted using the **Delete** button. The **Restore Element** button is used to cancel the last deletion made. The restored element must be repositioned in the operator panel.

It is possible to change the position of an element in an operator panel. The operator panel is first redisplayed without the selected element which must be repositioned in relation to a reference element.

4.3 Defining Operator Panel Elements

This section describes each of the individual elements in more detail.

4.3-1 Pushbuttons

A pushbutton is identified by a title made up of 2 superposed strings, each of 6 characters.

The color field is, in fact, a letter - the initial letter of the color of the button. The pushbutton will then appear in this color.

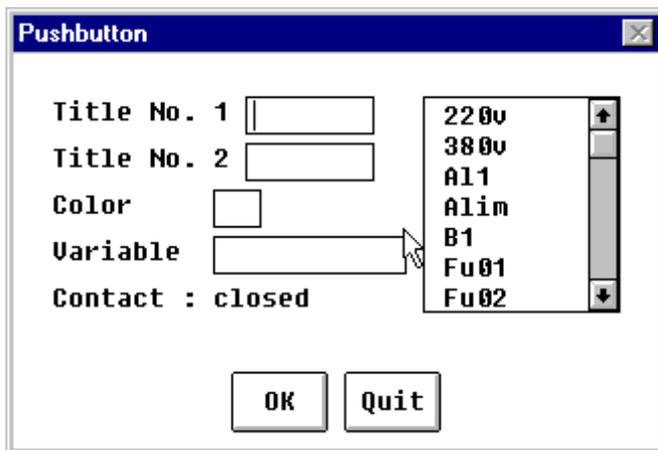
The user can select the color of the pushbutton from the following list :

- R for red
- B for blue
- V for green
- J for yellow
- N for black
- O for orange

The variable activated by the button is entered using an editor. Since this variable may have already been used in an input logic equation or in a relay, it is likely to be found in the list of "undefined" variables for the description : this list is therefore displayed in the window for editing the pushbuttons.

By default, the button contact is normally open : the logic state of the associated variable changes to "true" when the button is pressed. This operation is reversed by clicking on the text "closed" : this changes to an open contact.

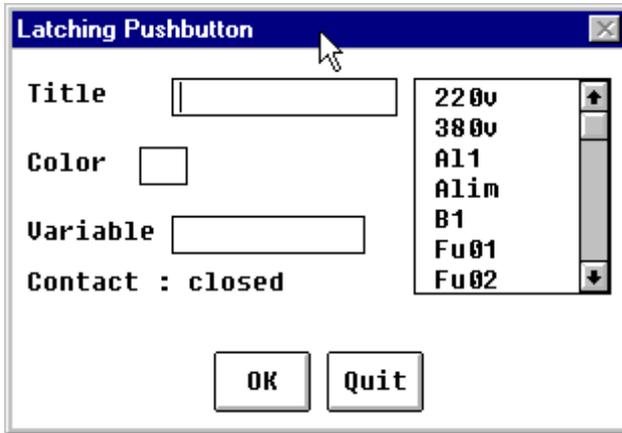
The "wide" pushbutton operates in the same way as a standard pushbutton. Since it is twice the width, the title strings can be a maximum of 12 characters.



4.3-2 Latching Pushbuttons

These buttons are used to represent elements such as "mushroom head" emergency stop buttons, which remain depressed and sometimes require a key to release them. These buttons have a title made up of a single string of up to 12 characters.

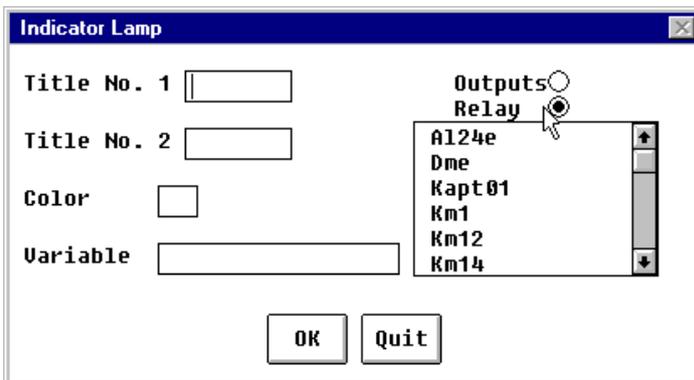
As with the simple pushbuttons, the contact can be normally open or normally closed.



4.3-3 Indicator Lamps

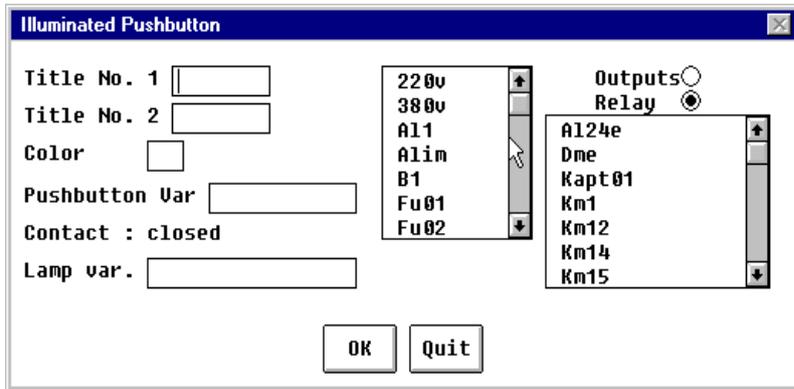
In the same way as pushbuttons, the title identifying these elements is made up of two strings of 6 characters.

Since the variable which lights up the indicator lamp is usually an output or a relay, these entities can be accessed. In the same way as for the pushbuttons, the "wide" indicator lamps are twice as wide in the operator panels, and their title strings can be up to 12 characters.



4.3-4 Illuminated Pushbuttons

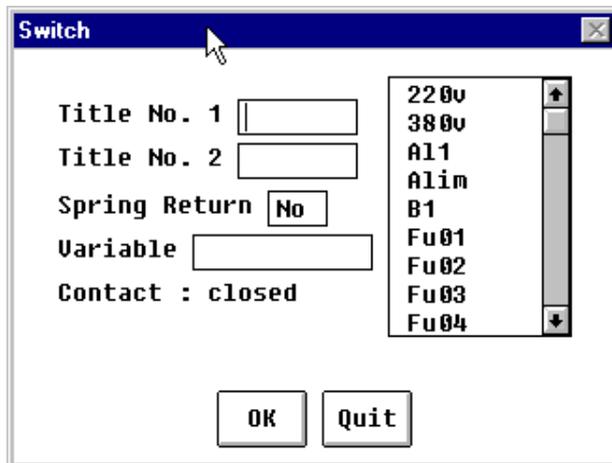
These elements combine the "pushbutton" and "indicator lamp" functions. Thus, their configuration involves both these elements.



4.3-5 Switch

This element takes up the same space as a simple pushbutton, and its title is also made up of two strings of 6 characters.

It operates in a similar way to the 2-position switch. It can also have a spring return function.



4.3-6 2-position Switches

Switches occupy the space required by two "standard" pushbuttons in operator panels. They have a 12-character title and each position is identified by a label of up to 5 characters. If the "Spring Return" function is selected, the switch will return to the off position, to the left, as soon as it is released. This is indicated by an arrow in the graphic representation of the element.

By default, the associated variable is activated when the switch is in the right-hand position. This can be reversed by replacing "normally open contact" with "normally closed contact", in the same way as for the pushbuttons.

The variable can be selected from the list of "undefined variables" for the description.

2-Position Switch

Title

Left Label

Right Label

Spring Return

Variable

Contact : closed

- 220v
- 380v
- A11
- Alim
- B1
- Fu01
- Fu02
- Fu03
- Fu04

OK Quit

4.3-7 3-position Switches

In these switches, two variables known as "right" and "left" can be activated. By default, the neutral position, where neither of these variables is at logic state 1, is when the switch is in the central position. This position can be changed using the selector switch provided for this purpose. If the neutral position is the central position, the right and left positions can spring return to the center.

3-Position Switch

Title

Left Label

Right Label

Left Spring Return

Right Spring Return

Left Variable

Right Variable

Neutral Position

Left

Center

Right

220v

380v

A11

Alim

B1

Fu01

4.3-8 N-position Switches

A certain number of variables are associated with each position of this switch. These variables will be at logic state "1" if the position is selected.

By default, this switch has 2 positions. A new position can be added : this will be $n+1$, if n is the number of switch positions.

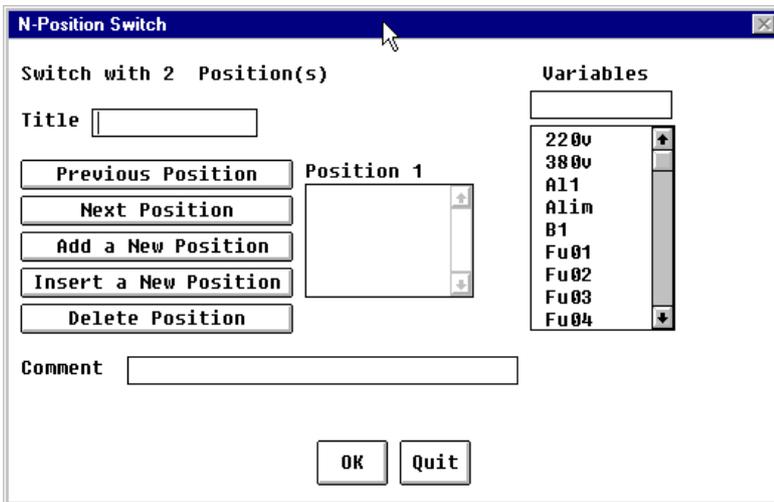
The insertion corresponds to a shift in the existing positions, from that which has been edited.

The edited position is changed using the "Previous Position" and "Next Position" buttons. It is possible to delete the current position.

The variables associated with the current position appear in a scroll-down list.

To add a variable to the current position, you can either enter it in the editor in the top right-hand corner of the N-position switch configuration window, or select it from the list of "undefined" variables which is displayed below the editor.

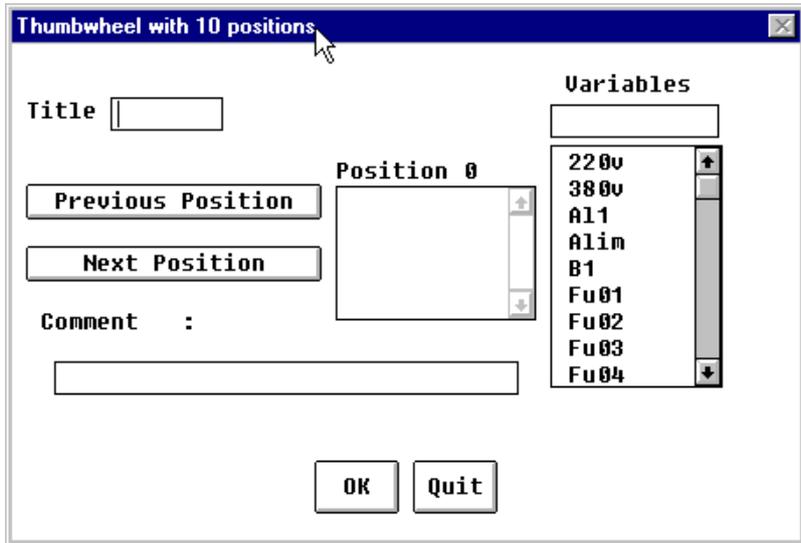
A variable is removed from the current position by clicking on it in the scroll-down list, under the text "Position < i >". A comment can be associated with each of the switch positions. These comments will be displayed, in the operator panel, to the right of the switch.



4.3-9 Thumbwheels

These elements are configured in a similar way to the N-position switches. The number of positions is set here at 10 or 16, and these positions are numbered from 0 to 9 or from 0 to 16.

The thumbwheel takes up a slot half the size of the N-position switch, and its title is restricted to 6 characters.



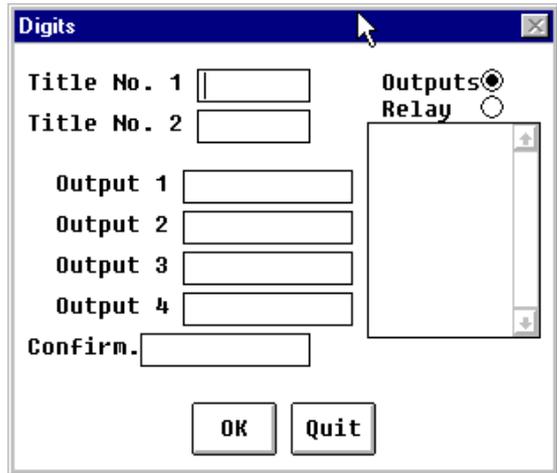
4.3-10 Hexadecimal Displays (or "Digits")

These elements make it possible to display a hexadecimal value in an operator panel.

The value displayed is given by 4 bits which can be outputs or relays.

Validation is optional. If it is defined, the state of the 4 bits is only displayed if the validation variable is at "1".

The display can be validated on a rising or falling edge of a variable. In this case, it must be prefixed by "+" or "-".



4.3-11 Text

This is used to insert one or two lines of comments in an operator panel.

4.3-12 Separator

The separator is used to insert "blanks" in the operator panel. These empty spaces are the same size as a "standard" pushbutton.

5.1 Introduction

The views associated with an application are used to follow the evolution of the variables and the various movements. They are made up of a background supporting various graphic elements relating to the application variables.

5.2 Creating and Editing Views

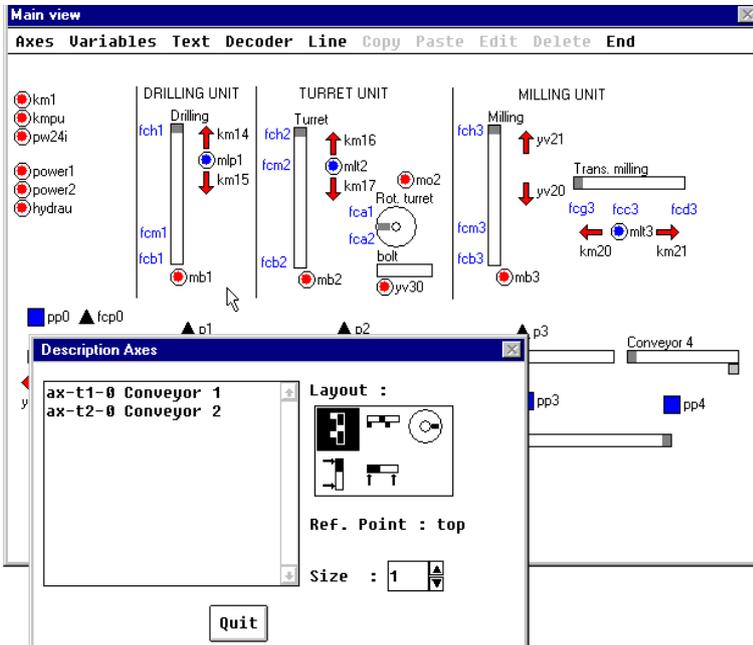
To create a view, it must first be given a title, which may be changed later. A special case is the blank title. It appears at the top of the list of views described under the heading "Background". This view is displayed by default in the simulation environment. In edit mode, the buttons for selecting the type of graphic element (axes, variables, texts, decoders) appear at the top of the screen, along with the buttons for editing and deleting an element from the view.

5.2-1 Axes

The axis to be represented is selected from the application axes. The axis characteristics (layout, reference point, size) are then specified.

- The layout is selected from the corresponding icons, and can be vertical, horizontal or circular. The position in the axis can be represented by a cam moving along the axis or by a bargraph (for example, for the evolution of a level). In this latter case, the sensors are represented by arrows.
- Selecting the reference point of the axis (to the left/right or above/below) determines the "0" of the coordinates. For example, a horizontal axis whose reference point is selected to the right and initial position is 0 will have its cam drawn to the right.
- The size of the axis is fixed by a proportionality coefficient between 0.5 and 5. By default, it is set to 1.

When the selections relating to the axis representation have been made, it is possible to insert the axis in the view using the mouse by clicking on the location in which it is to be placed. The list then disappears.



The last axis to be created (or selected) can be moved using the mouse or the arrow keys.
Its size can be increased and decreased using the "+" and "-" keys respectively.
It can be deleted by pressing the **Delete** button.

Representing the sensors associated with the axis

To avoid overloading the representation of the axis in the view, only the sensors used in the logic equations for the inputs in the I/O configuration or the relays are shown. Sensors which are only used in the description variables on the axis movements are considered to be fictitious and thus are not represented.

If the view is described before the I/O are configured, the axis sensors will not appear in the representation. This is particularly relevant if you wish to perform an offline simulation using a "mechanical Grafcet chart".

In this case, in simulation, the axis sensors used in the Grafcet chart transition conditions will appear in the view : these sensors are actually used as implicit inputs.

5.2-2 Variables

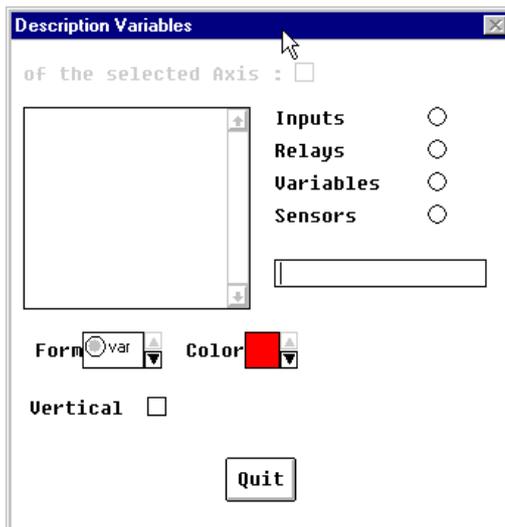
An application variable can be displayed in the view which, in simulation mode, will reflect its current state. To add a variable, first select the **Variables** button in the **Description Variables** window.

D

Variable selection	8 possible layouts		6 colors
Inputs	 km10	Radio button	Red
Outputs	 km10	Triangle	Blue
Relays	 km10	Square	Green
Boolean variables	km10	Displayed or not	Yellow
Numeric variables	 km10  km10	Up/down arrow	Black
Sensors	  km10 km10	Right/left arrow	Orange

The right-hand side of the window is used to select the type of variable from the inputs or outputs, the relays, the variables for the moving part and the sensors. This selection is made easily by selecting the variable in the list on the left using the mouse, choosing the type of display and the color (clicking in the colored box stops the colors scrolling and selects the color displayed), then by selecting the point on the screen at which this variable is to appear (name + icon).

The variable to be displayed can also be selected by entering its mnemonic in the field provided for this purpose.



Numeric variables are selected in the same way, then displayed in the view. The current value is displayed in a box (absolute number, or hexadecimal number if the Hex. box is checked).

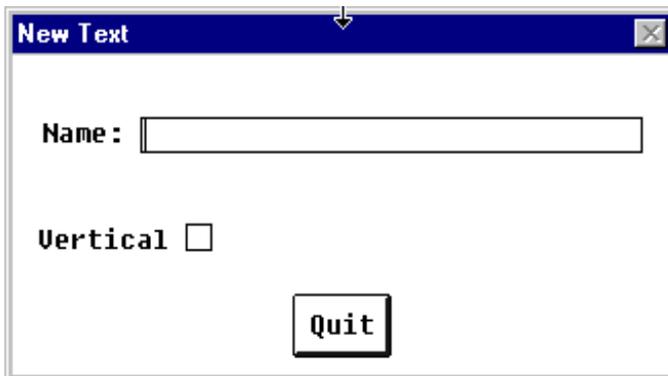
1234 **varana**

If an axis represented in the view is selected using the mouse, the variables are restricted to those which appear in the associations up and downstream of the axis in question. To return to all the application variables, use the right mouse button to click on the axis representation (or any element in the view) or click in a zone on the view which does not contain any elements.

It is possible to select a vertical representation of the variable (check box).
The selected variable can be moved using the mouse or the arrow keys.
It can be deleted by pressing the **Delete** button.

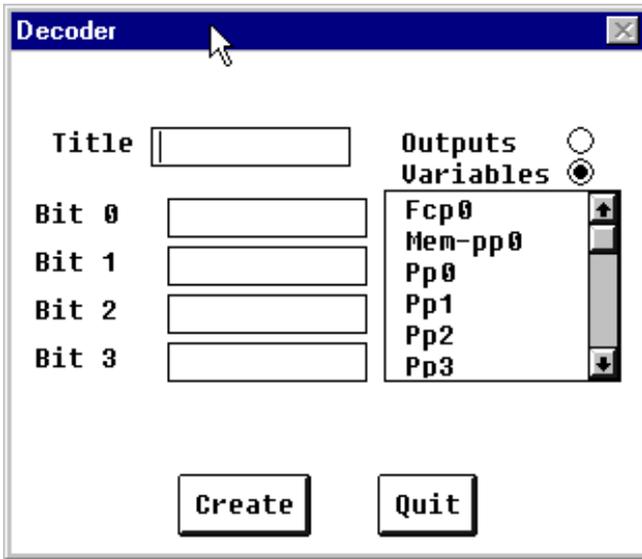
5.2-3 Text

Free text can be placed in the view by positioning it using the mouse.



5.2-4 Decoders

In the same way as with operator panels, hexadecimal displays can be used in the view. The value displayed is given by 4 bits which can be outputs or variables.



5.2-5 Lines

Solid lines can be added to enhance the views.

5.3 Editing Aids for Views

5.3-1 Copy/Paste

A copy/paste mechanism is available :

1. Select the part of the view to be copied, using the left mouse button to draw a box around the zone to be copied.
2. Click on the **Copy** button.
3. Click on the **Paste** button then click on the required position in the view.

5.3-2 Moving Elements

It is possible to move elements in the view by clicking on them and moving the mouse without releasing the button, or by using the arrow keys (movement pixel by pixel).

5.3-3 Printing Views

This command is used to print a view on the default printer. The document created by SIMTSX corresponds to a simple copy of the image of the selected view.

Note

Before printing a view, the user must ensure that this view is displayed in full on the screen. If not, only the part displayed will be printed.

D

Section	Page
1 Presentation of Validation Possibilities	1/1
1.1 Presentation	1/1
2 Using Mechanical Grafcet Charts	2/1
2.1 Syntax	2/2
2.2 Interface for Editing Grafcet Charts	2/4
2.3 Printing Grafcet Charts	2/10
3 Validation by Simulation in Offline Mode	3/1
3.1 Animation in Offline Mode	3/1
4 Simulation Environment	4/1
4.1 Background Window for the Simulation Environment	4/2
4.2 Actions on Operator Panels and External Variables	4/5
4.2-1 External Variables	4/5
4.2-2 Operator Panels	4/7
4.2-3 Concept of "Context"	4/9
4.2-4 Concept of a "Scenario"	4/9
4.3 Display Tools	4/10
4.3-1 Dynamic Pages	4/10
4.3-2 Views	4/12
4.4 Forcing Description Variables	4/13

Section	Page
4.5 Examination or Analysis of Situations	4/14
4.5-1 Examination of Variables Whose State is Determined by a Boolean Expression	4/15
4.5-2 Examining Numeric Variables	4/15
4.5-3 Examining Movements	4/16
4.5-4 Examining Sensors	4/17
<hr/>	
4.6 Operation Traps	4/18
4.6-1 Characteristics of a Trap	4/18
4.6-2 Evaluating Traps	4/19
4.6-3 Action of Traps	4/19
<hr/>	
4.7 Faults	4/20
4.7-1 Sticking and Forcing	4/20
4.7-2 Sensor Bounce	4/21
4.7-3 Notes	4/21
<hr/>	
4.8 Scenarios	4/22
4.8-1 Sequences of Actions on Operator Panels and External Variables	4/22
4.8-2 Editing a Scenario	4/25
4.8-3 Concept of Wait Periods	4/26
4.8-4 Verifying Reactions	4/28
<hr/>	
4.9 Initialization	4/30
<hr/>	
4.10 Saving and Restoring the Application State	4/31
<hr/>	
4.11 Description	4/32

Section	Page
5 Editing Trend Diagrams	5/1
5.1 Configuration	5/1
5.1-1 Editing a Module	5/2
5.2 Presentation Elements - General Functions	5/3
5.2-1 General Presentation	5/3
5.2-2 General Principle	5/5
5.2-3 Page Header	5/7
5.2-4 Moving Around in the Trend Diagram	5/7
5.2-5 Precision	5/8
5.2-6 Indications of Time	5/8
5.3 Axis Diagram	5/10
5.3-1 Types of Element Shown	5/10
5.3-2 Changing the Reference Point	5/10
5.3-3 Selecting a Module	5/11
5.3-4 Selecting an Axis	5/11
5.3-5 Exiting the Axis Diagram	5/11
5.4 Module Movement Diagram	5/12
5.4-1 Types of Element Shown	5/12
5.4-2 Selecting the Elements Shown	5/12
5.4-3 Exiting the Movement Diagram	5/12
5.5 Axis I/O Diagram	5/13
5.5-1 Types of Element Shown	5/13
5.5-2 Selecting the Elements Shown	5/14
5.5-3 Exiting the I/O Diagram	5/14

Section	Page
5.6 Representation in a Functional Analysis Table	5/15
5.6-1 Access	5/15
5.6-2 Presentation	5/15
5.6-3 Functions	5/15
5.6-4 Filtering	5/16
5.6-5 Exiting the FA Representation	5/16
<hr/>	
5.7 Comparing Cycles	5/17
5.7-1 Principle	5/17
5.7-2 Creating a Reference Cycle	5/18
5.7-3 Activating the Comparison	5/20
5.7-4 Displaying the Durations for the Evolutions of the Reference Cycle	5/20
5.7-5 Displaying Time Deviations in Structured Table Mode	5/20
5.7-6 Moving the Reference Record	5/21
5.7-7 Displaying Reference Cycle Synchronization Information	5/21
5.7-8 Finding a Deviation	5/22
<hr/>	
5.8 Printing Diagrams	5/24
<hr/>	
5.9 Summary of the Keys Used	5/25
5.9-1 Description of the Keyboard	5/25
5.9-2 Axis Diagram	5/25
5.9-3 Movement Diagram	5/26
5.9-4 I/O Diagram	5/27
5.9-5 Functional Analysis Diagram	5/28

1.1 Presentation

Offline simulation is used to check that the behavior of the simulated application corresponds to reality.

Offline simulation is generally executed using a Grafcet chart, which sequences the movements to reproduce the operating cycle of the simulated application.

The "Mechanical Grafcet chart" is used to execute offline simulation without having to configure the I/O. These Grafcet charts are called "MGRAF<i>".

I/O are implicitly created to enable communication between the Grafcet chart and the application model.

Inputs are created for any variable appearing in a transition condition and which do not correspond to an internal variable in the Grafcet chart, such as the end of a time delay or a step state.

In the same way, "fictitious" outputs are created for any step action for which no variable of the same name has been defined in the application model.

If an I/O configuration and a relay interface have been described, they are taken into account, even if this concerns simulation with a Grafcet chart. However, if the Grafcet chart actions have the same name as the relays, the relays are "skipped" during simulation.

Note :

It is possible to execute an offline simulation in two ways :

- The actuators are controlled "manually" by acting on the outputs, or directly on the relays or solenoid valves if the relay interface has not been described : these entities will appear as "external variables".
- With a Mechanical Grafcet chart.

E

A mechanical Grafcet chart is a Grafcet chart whose interface with the application is not defined explicitly by an I/O configuration.

Communication is performed by acting on the undefined variables of the application model via Grafcet chart actions : the actions for the steps whose name is not that of a variable defined in the model are "externalized" in the form of "implicit outputs" which may have an effect on the application.

This is also true for the relays in the relay interface which are not part of the power supply: these relays will not be taken into account if actions with the same name already exist in the mechanical Grafcet chart.

Conversely, the variables used in the transition conditions are directly the sensors and other variables in the application, as well as the operator panel variables. Additional external variables may be introduced if required.

Such a Grafcet chart can be created if the I/O have been configured : therefore inputs must not be used in the transition conditions.

To access the interface for creating mechanical Grafcet charts, click on the CHART button in the SIMTSX configuration interface.

2.1 Syntax

Grafcet charts are made up of steps and transitions linked by arcs. Actions are associated with steps and conditions are associated with transitions.

Step actions

The basic actions are actuator activations. They can also involve internal variables, or affect the time delays and counters.

Actions on Boolean variables

The actions can be set during activation of the step with which they are associated, or memorized and deactivated. In this case, the syntax is as follows :

<action> = 1 for memorization

<action> = 0 for deactivation

Managing numeric variables

On Grafcet charts, it is possible to assign a value to a numeric variable, provided that it is declared external, in a step action.

Example : speed = 4

The value of a numeric variable can be tested in a transition or an action condition.

Activating time delays

An action can also be the start of a time delay. The syntax is as follows :

<Lti> if the time delay is activated with its default duration (duration = 1)

or

<Lti> <duration>, where <duration> is a positive numeric value.

Operation of the time delay differs according to whether its activation is conditioned or not. If there is no condition, the time delay is started when the step with which it is associated is activated and continues to run even if this step is deactivated.

However, if a condition is defined, the time delay is started when the step changes to the true state of the "logic and" formed by the activity of the step and this condition : this time delay is thus initialized on activation of the step if the condition becomes true at this moment, or when the condition is true when the step is active. If the condition is invalidated or the step deactivated, the time delay is reset to 0.

Operations on the counters

Several actions are possible on counters. These are assignment, incrementation and decrementation, and the arithmetic operations of addition, subtraction and multiplication.

The assignment syntax is as follows :

<counter> = <value>

where <value> can be a number or another counter.

These actions can be conditioned by Boolean expressions, where the syntax is the same as that for the transition conditions.

The syntax for incrementation/decrementation is as follows :

<counter> + and <counter> -

If no condition is associated with this operation, the operation is executed on activation of the step. If a condition is defined, the incrementation or decrementation is executed on the change to the true state of the "logic and" formed by the activity of the step and this condition : either on activation of the step if the condition becomes true, or each time the condition changes to "1" when the step is active.

For arithmetic operations, the syntax is :

<counter> <operator> <value>

where <operator> can be "+", "-", or "*" and <value> is defined as for assignment.

These operations are performed when the step is active - and not on activation as is the case for incrementation/decrementation - and when the condition which may be associated becomes true.

If the action "cnt1 + 1" is defined in a step and the activity of this step is not "transient" in a single execution cycle, there will be an overflow on counter "cnt1".

Transition conditions

These transition conditions are defined by "Boolean" expressions, in which predicates may be applied to the counter values.

Tests on the counter values are written as follows :

<counter> <operator> <value> where :

<counter> is the counter concerned,

<operator> is a comparison operator which can be : <, >, <=, >=, = or <>.

"Xi" terms take into account the activity of a step, in this case, step "n° i", in a transition condition.

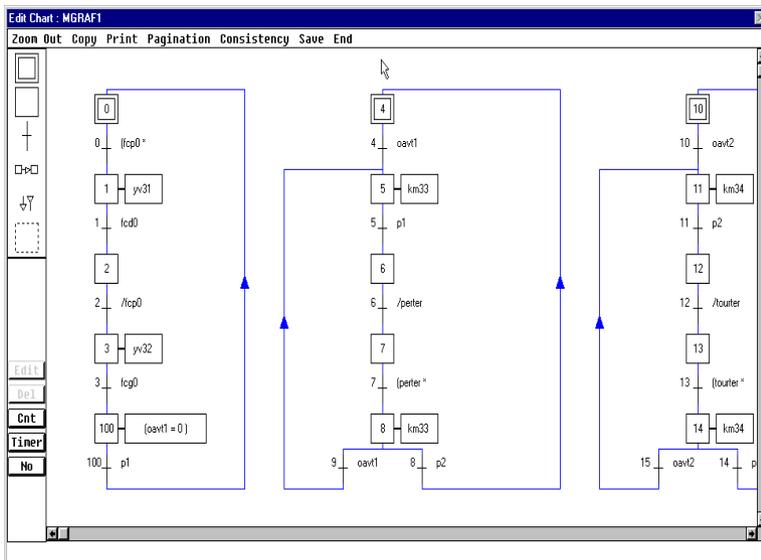
Edges on inputs or step activities can be tested by prefixing the corresponding variables with a "+" or "-" depending on whether it is a rising or falling edge.

2.2 Interface for Editing Grafcet Charts

The edit window is composed of a main zone, dedicated to the drawing of the Grafcet chart. This graphic page is actually 25 times larger than the part which is displayed on screen. It can be scrolled in order to move the zone displayed.

On the left-hand side of the edit window there are 5 icons symbolizing, from top to bottom, an initial step, a "normal" step, a transition, the link between two elements and a zone delimited by dotted lines.

There are also 5 buttons in the lower left-hand part of the window. The first 2 are used to edit and delete the components of a Grafcet chart, that is, the steps and transitions and the links between these elements. The next 2 buttons are reserved for creating counters and time delays.



Creating steps and transitions

To create a step or transition, first select the corresponding icon on the left-hand side of the window using the mouse : this icon changes to reverse video when in create mode.

This mode can be exited by deselecting the icon using the right-hand mouse button.

The element is created simply by clicking the left-hand mouse button in the graphic zone at the position in which you wish a step or transition to appear.

These elements are positioned at regular intervals on the graphic page based on a grid.

The element which has just been created is named in the left-hand side of the window, above the column of buttons.

It can be moved in the graphic page using the arrow keys on the keyboard, or directly using the mouse.

If a step or transition has been created by mistake, it can be deleted, using the **Del** button, after having selected the element which is not required. A confirmation is requested before the element is deleted definitively.

By default, the index of a step or transition recently created is the integer $n+1$, where n is the largest integer numbering an existing step or transition. The **No.** button is used to select the indexes from which the steps and transitions created are numbered.

Links between steps and transitions

The link between the elements can be established in two ways, depending on the graphic representation selected.

The link can be represented by an arc : in this case, the icon representing two elements linked by an arrow must be selected first in the left-hand side of the window.

The link also can be represented by a connector : this is useful when the trace of an arc in the graphic page would reduce clarity. In this case, select the icon representing two arrows, located below that selected previously.

The link is then established by first selecting the upstream element using the mouse : this is then identified by a small black square.

The downstream element is then selected. This is designated by the marker as being the upstream element for the next link. If this is not required, deselect the element using the mouse.

A link between two elements can be deleted if it has already been selected with the left-hand mouse button : the arc concerned is then designated in the left-hand side of the window and the **Del** button is used to delete it.

Selection field

The graphic interface offers the possibility of manipulating sets of steps and transitions. To do this, first click on the icon representing a square in dotted lines on the left-hand side of the screen, which then changes to reverse video.

Steps and transitions are designated by framing them within a rectangle.

The top-left hand corner of this rectangle is positioned by clicking on the graphic page without releasing the mouse button.

If you drag the mouse while holding the button down, a rectangle will appear. The lower right corner of this rectangle follows the movements of the mouse. When the button is released, the rectangle is fixed on the screen : a set of steps and transitions is then designated.

The whole block can be moved at once using the mouse. It can also be deleted using the **Del** button, after confirmation.

A set of steps and transitions can also be duplicated. Having made the selection as explained earlier, click on the **Copy** button at the top of the window.

Clicking the mouse in the graphic zone positions the top left-hand corner of the block to be copied, which is then copied after confirmation.

Zoom out

We have seen that the Grafcet chart is described in a graphic page of which 1/25th is displayed.

To be able to have a global view of the Grafcet chart, **azoom out** function is offered, which can be activated using the button at the top of the window.

In **zoom out**, the zone displayed in edit mode is represented by a rectangle. This rectangle can be moved using the mouse and confirmation is used to return to edit mode.

The function for managing blocks, described earlier, can also be accessed in zoom out.

Editing steps and transitions

The graphic page is used to describe the structure of the Grafcet chart. For this to be useful, we need to define its "interpretation", that is to associate actions with the steps and conditions with the transitions.

This is achieved using the windows provided for this purpose, which are activated using the **Edit** button, having first selected a step or transition in the graphic zone. The **Previous** and **Next** buttons in these windows are used to review all the elements in the Grafcet chart belonging to the selected type.

Describing the steps

The upper part of the description window includes a selection box indicating whether it is an initial step or not : this characteristic can be modified by clicking on the box with the mouse.

The actions associated with the step appear in a scroll-down list on the left-hand side of the window. If one of these actions is selected, it is displayed in the **Action** editor, and the condition which may be associated with it appears in the corresponding editor.

The action and its condition can then be modified, the substitution being made using the **Replace** button. The selected action can also be deleted using the corresponding button.

To create a new action, it must be defined in the **Action** and **Condition** editors and then added using the **Add** button.

There are Help functions available for defining actions. The **Action Help** button is used to select an internal action, a time delay or a counter.

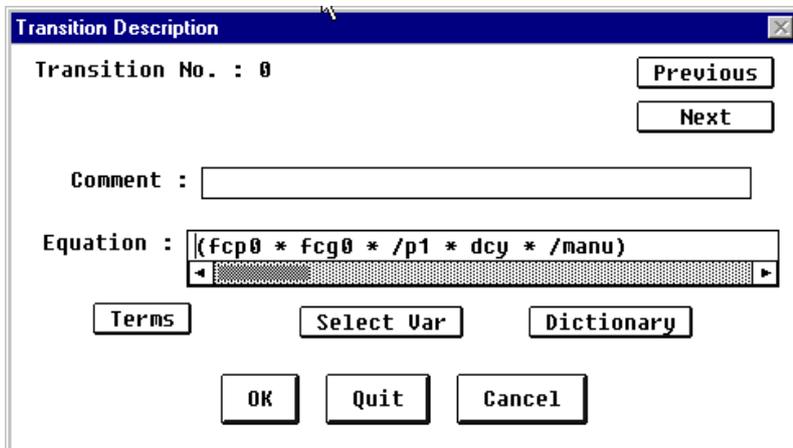
The **Condition Help** button is used to select terms which may be used when expressing the condition of an action, ie. internal actions, counters and end of time delay variables.

The screenshot shows a dialog box titled "Step Action Description". It has a blue title bar with a close button. The main area contains the following elements:

- Step No :** 1
- Initial Step**
- Comment :** [Empty text box]
- Action List:** A scrollable list box containing "yv31 (*) (oavt1 = 1)".
- Action :** [Empty text box]
- Condition :** [Empty text box]
- Action Help** and **Condition Help** buttons.
- Terms**, **Replace**, **Add**, and **Delete** buttons.
- OK**, **Quit**, and **Cancel** buttons.

Describing the transitions

This description consists of defining the condition associated with the transition. The syntax is the same as for that of the step action conditions, and a similar Help function is offered.



Creating the counters and time delays

In the Grafset chart steps, time delays can be archived or operations can be performed on the counters.

In the transitions, end of time delay variables can be taken into account and counter values can be tested.

These entities must be previously defined to be available for use in the actions and conditions : this is done using the **Cnt** and **Timer** buttons.

Counters

The **Cnt** button is used to open a window where the counters defined for the Grafset chart being edited are displayed.

To create a new counter, enter an integer index in the editor in the right-hand side of the window and press the **Add** button.

A counter name is always in the format "Cnt<i>" where <i> is an integer. If integer <i> is such that counter "Cnt<i>" already exists, a message signals the clash and the first unassigned integer is offered in the editor.

Time delays

The window is used to create new time delays and to modify the duration of existing time delays.

To modify the duration of a time delay, select the term " $t_{<i>$ " corresponding to the activation of this time delay. The duration of this delay is then displayed in an editor where it can be modified : this is taken into account using the **Modify** button.

To create a time delay, use the **New Time Delay** button : the smallest integer not corresponding to an existing time delay then appears in the **Time Delay** editor. The default duration is 1, although it can, of course, be modified.

The time delay to be created can be entered directly in the **Time Delay** editor.

The **Add** button is used to take account of the new time delay, provided that the same time delay does not already exist.

A time delay can be added to the actions of a step by the step editor interface. Simply enter its name, that is " t " followed by its number, or select the list of time delays using the **Action Help** button.

The end of the time delay in a transition equation is defined by the variable " t " followed by the number of the time delay.

Note :

It is not possible to delete counters and time delays from the interface. Deletion is actually implicit, only the time delays and counters used in the Grafcet chart are saved when the user exits the edit screen.

2.3 Printing Grafcet Charts

SIMTSX can be used to print any selection block created using the mouse. The **Print** button in the top part of the window triggers, after confirmation, printing of the contents of this block. However, if the size of the rectangle representing the block is not compatible with the printer, an error message signals that this command has been aborted.

To simplify the printing of Grafcet charts, SIMTSX provides the user with an automatic page setup function. This is activated using the **Pagination** button at the top of the window. The Grafcet chart is then displayed in zoom out by showing each page created by a rectangle with a "shadow". The size and locations of these rectangles are calculated according to :

- the location and proximity of "sub-charts"
- the parameters defined for the default printer : size and source of the paper

Printing, triggered using the **Print** button, involves several options :

- Print all pages : all pages created are printed.
- Print selected pages : a page is selected by clicking in its rectangle ; the page is then grayed out.
- Print a selection zone : the user may require this function when a chart is too long or too wide to be printed on a single sheet of paper. In this case, the chart is printed manually in successive selections of zones to be printed.

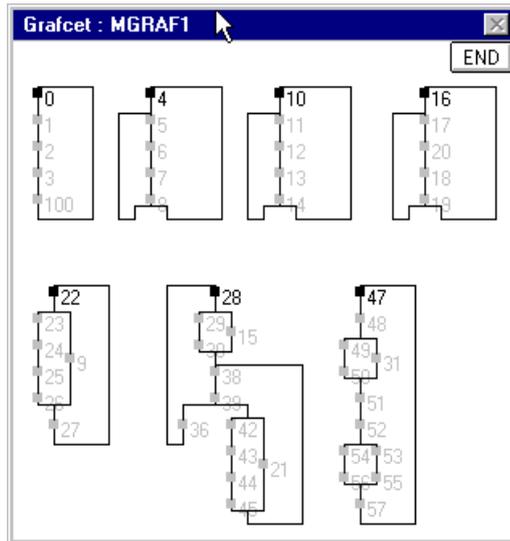
The document printed by SIMTSX is made up of :

- One page containing the graphic representation of the GRAFCET CHART in which the step and transition references are found
- One or more pages detailing each step and each transition for the selected part of the GRAFCET CHART

3.1 Animation in Offline Mode

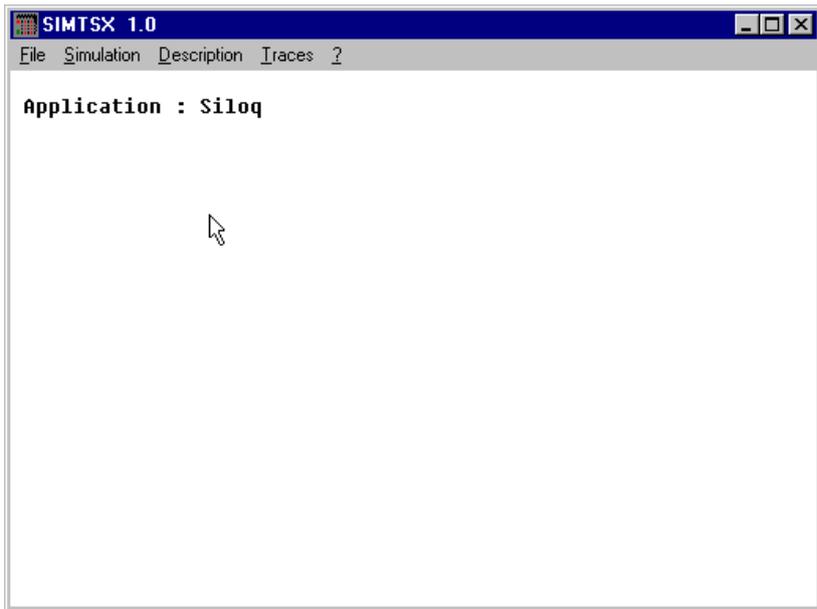
If a Grafcet chart is being executed, a view of the same name can be displayed. It contains the Grafcet chart as it has been described, in **Zoom out** mode (reduction to 1/5th) with the steps, transitions and links. The active steps are animated as the Grafcet chart evolves.

Once simulation has stopped, it is possible to open the examination window for a step or a transition directly by clicking on this step or this transition.



E

The **File** and **Simulation** menus in the main window of SIMTSX are used to define the simulation environment.



The **File** menu is used to load the application to be simulated and the local control chart :

Application - Selects and loads the application to be simulated.

Charts - Accessible, once a machine has been chosen, for selecting the offline simulation Grafcet chart ("Cancel" to deselect).

The **Simulation** menu is used to start simulation in offline or online mode :

Offline - Starts simulation in offline mode.

If no Grafcet chart is selected, SIMTSX suggests selecting a Grafcet chart ("Cancel" to deselect).

Online - Starts simulation in online mode, sets parameters and selects communication mode.

If a Grafcet chart is selected, this Grafcet chart will be loaded and then executed at the same time as I/O exchanges. If an actuator is controlled by both a Grafcet chart action and a PLC output at the same time, it is the Grafcet chart action which is propagated by SIMTSX.

4.1 Background Window for the Simulation Environment

The main window for the simulation environment comprises three zones. The title bar contains the various commands which can be used in simulation. The zone situated below this bar is divided into three parts, dedicated to the operating modes, displaying the time and choice of period, and displaying the simulation state. The bottom of the window is reserved for displaying the background view of the simulated application.

The simulator operating modes are Step by Step, Periodic and Continuous. They are selected by checking the corresponding boxes.

E

- In **Step by Step** mode, simulation is interrupted each time output state changes.
- **Periodic** mode enables simulation to be stopped according to a machine time step. The value of this step is entered via a dialog box which is opened by clicking on the **Period** button. This window appears automatically if Periodic mode is selected and the Period is not defined.
- **Continuous** mode : simulation continues without interruption as long as there are active movements.

If you are in periodic mode and you then select continuous mode, the period which was entered is lost. On the other hand, if periodic mode is deselected, the period is retained.

The **Simulation Time** is displayed in the central part of the zone situated under the title bar. By default, this advances according to the events generated by the evolution of the application model. For relatively slow processes, this "advancement of time" can be much faster than in reality. The "real" time, according to which simulation is executed, depends on the number of calculations to be performed and not on the movement time defined during configuration of the model.

It is possible to slow simulation by defining a period with a low value using the Period button. Another way is to select **Delayed Mode** by checking the box below the time. This enables, as far as possible, the evolution of the simulation time and the PC clock to be synchronized. In other words, the simulation time does not advance any faster than "real" time. This is only possible if the calculations to be carried out - and the exchanges with the PLC - are not too numerous. A "+" sign appears to the left of the "Delayed Mode" command if this is the case.

The <<>> Num button is used to introduce an update period for input registers and words. By default, input registers and words are written each time a value is changed by SIMTSX. This may lead to numerous exchanges between the PLC and the simulator, slowing down simulation but without affecting the debugging of the PLC program.

This period can then be used to time the writing of input registers according to a time step expressed as "application time" :

- If it is set at 0 (by default), numerical exchanges are performed each time %IW and word values are changed.
- Otherwise, updates are performed :
 - at the same time as discrete inputs
 - if there is no change in the state of discrete inputs, as soon as the simulation time progresses by more than the numerical exchange period (since the last exchange), or if the simulation time is not moving (movements are blocked, etc)

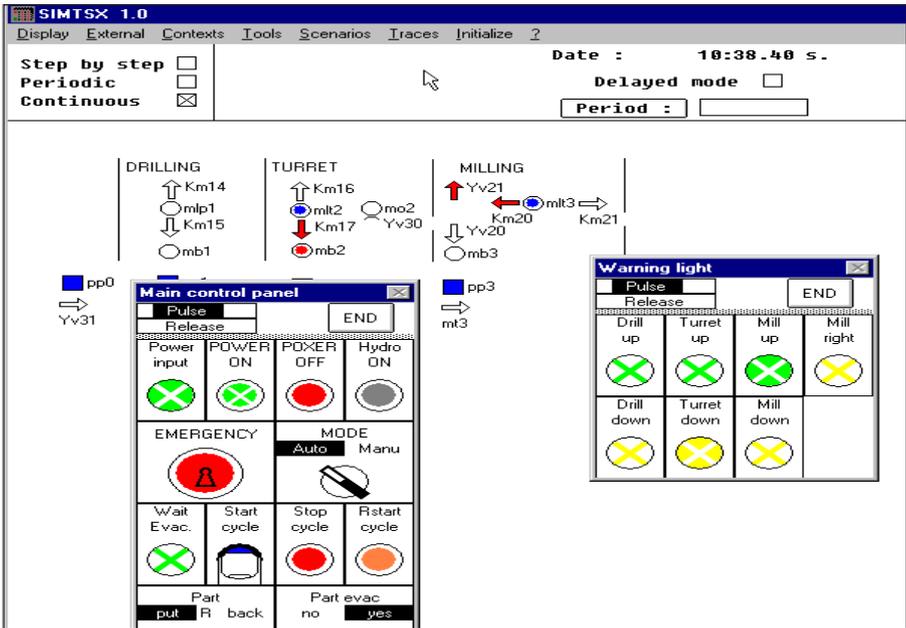
The right side of the window displays the simulation state. When no movement is produced, the text **Blocking** appears on a black background. During simulation, the **Interrupt** button enables evolution of the model, and therefore the time, to be interrupted. When simulation is stopped - for example on the appearance of an output when Step by Step mode is selected - the **Restart** button enables simulation to be restarted.

A specific state of SIMTSX is **Interrupt on Model Problem**. This occurs when an inconsistency in the modeling process prevents simulation continuing. Possible causes are specified in the Error Messages section in the Appendix (see section 2 of part I).

The table below refers, for each of the simulation environment commands, to the relevant section of the documentation.

Simulation environment commands

Operator panels	See E 4.2-2.
External variables	See E 4.2-1.
Contexts	See E 4.2-3.
Views	See E 4.3-2.
Tools	
Evolutions	Opens a window displaying I/O exchanges.
Event prediction	Opens a window displaying SIMTSX event prediction.
Examination	See E 4.5.
Pages	See E 4.3-1.
Traps	See E 4.6.
Forcing	See E 4.4.
Description	See E 4.11.
Faults	See E 4.7.
Scenarios	See E 4.8.
Trace	See E 5.
Help	Opens the simulation online help.
Initialization	Reinitializes simulation offering a context (see E 4.9).
Quit	Quits simulation on request if the machine state is to be saved. (see E 4.10).
I/O	In online mode, displays the communication server. See F1.5 and F1.6.



4.2 Actions on Operator Panels and External Variables

4.2-1 External Variables

"External variables" is the name given to terms which affect the model logic equations and whose origin is not specified (undefined variables of the description part). It generally refers to machine variables representing energy supply conditions, numeric variables, sensors not activated by axes, such as safety sensors, or the presence of parts at the entry point of a machine.

By creating implicit inputs, internal Grafcet chart variables which are not managed also enter into this category : these can then be forced.

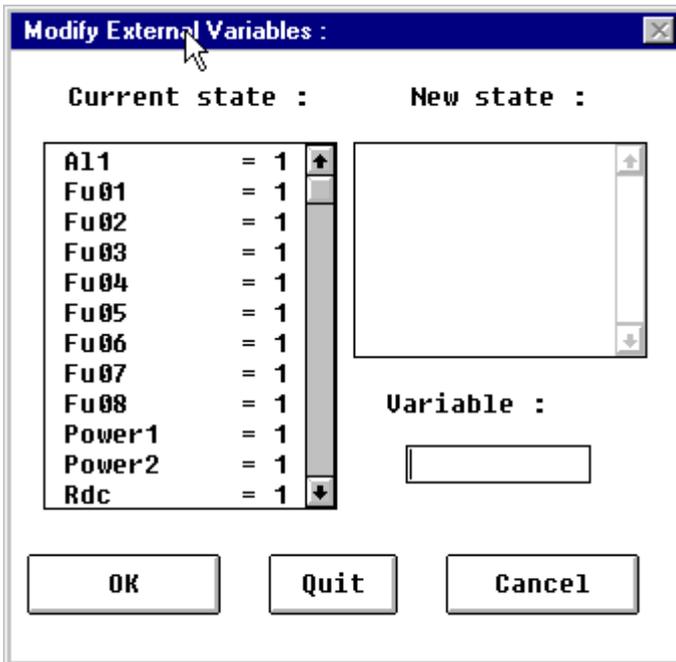
The outputs which are not controlled by the Grafcet chart in internal simulation but appear in the model, change to "external variables".

The **External** button at the top left of the simulation screen produces a window which enables the state of these variables to be modified.

The external variables are presented in the left part of this window together with their logic state.

If one of the variables is selected from the list on the left of the screen, it will appear on the right side of the screen.

The variable can be removed from the right side of the screen by clicking on it in the same way. Another way of selecting a variable to modify its state is to type it in the editor field in the window ; if the name entered is not that of an external variable, the first external variable with an alphabetically higher initial letter than the typed variable appears at the top of the list on the left side of the window. The **OK** button enables the selected changes of state to be sent to the simulator.



4.2-2 Operator Panels

In the simulator operating environment, operator panels appear as windows. Operations on elements comprising these operator panels is carried out using a mouse.

Manipulating operator panels

The **Panels** button at the top left hand side of the simulation window produces a menu for choosing the operator panel to be displayed.

The **All** item activates, in a single command, all the operator panels configured. The selected operator panels appear in the position in which they were located in the last simulation. By default, they appear in the lower right hand part of the screen.

The **End** button at the top right of an operator panel removes it from the screen.

Actions on operator panel elements

Operator action involves pushbuttons and switches for running the automated system.

Pushbuttons

The usual use of a pushbutton is to generate a pulse to the control system of the automated system. This can be done in SIMTSX by clicking on the pushbutton. The operator panel button remains pushed in as long as the mouse button is held down. The pushbutton returns to its original state.

The latching pushbutton is a particular type of button which remains depressed when it is pushed in : this is the case for emergency stop buttons which sometimes require a key to release them. This type of button is found in SIMTSX : one click of the mouse latches it, and a special command releases it.

The button can be released in two ways. The first is by clicking the **Release** command at the top of the operator panel : this changes to reverse video and by clicking the button concerned it can be released. The second is by holding down the <Ctrl> key on the keyboard and clicking the button.

It is possible to hold down a pushbutton. This can be useful for simultaneously activating several pushbuttons or for selecting a simulator function using the mouse while keeping a button depressed.

As for releasing a pushbutton, there are two ways of holding the pushbutton down :

- The first way is by clicking on the **pulse** command at the top of the operator panel : this is then replaced by the text **hold**. If the pushbutton is clicked, it changes to the hold state which is signalled by a finger icon on the button.
- The second way is by clicking on the button while pressing the <Shift> key on the keyboard.

A pushbutton which is held down can be released, as for latching buttons, by using the operator panel **release** command or by the combined action of holding the <Ctrl> key and clicking the mouse.

Switches

In SIMTSX switches are available with 2, 3 or n positions. Switches are positioned using the left and right mouse buttons.

2 or 3 position switches may be equipped with a spring return for certain positions, for example, the left position of a 2 position switch. In this case, the position is indicated by an arrow and the switch acts as a pushbutton in this position : one click of the mouse generates a pulse after which the button returns to its original position, in other words the left position for a 2 position switch or the central position for a 3 position switch.

In the same way as for pushbuttons, a switch can be held in a spring return position : the arrow indicating the spring return no longer appears.

Thumbwheels

The selected position is incremented by clicking on the "+" box, and decremented by clicking on the "-" box.

Analyzing the states of indicator lamps and displays

Operator panel components can be "interrogated" on their state. It is possible to find out why an indicator is on (or lit) or according to which variables the numeric value is displayed in a hexadecimal decoder. For this, the mouse cursor must be positioned on the element in question and the left and right buttons must be clicked simultaneously. For an indicator lamp, there will be an "examination" of the variable controlling it. In the case of a hexadecimal display, the state of 4 data bits and the optional validation variable will be displayed.

Note

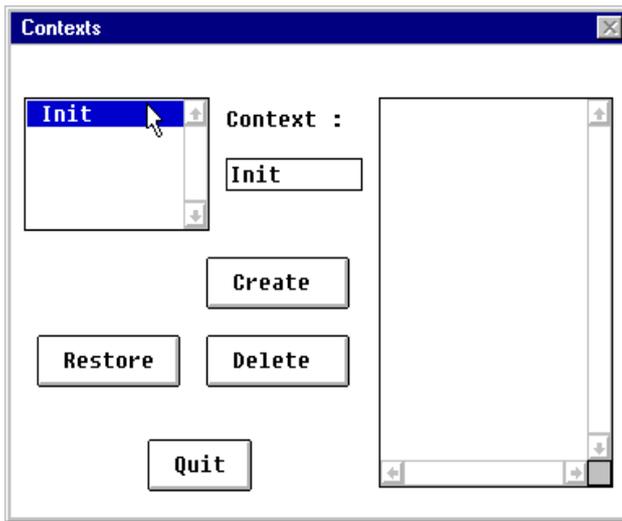
To "interrogate" an indicator pushbutton, the 2 mouse buttons must be clicked and the <Ctrl> key on the keyboard must be pressed in order to avoid activation of the **button** function of this element.

4.2-3 Concept of "Context"

We have just seen how to manipulate operator panels and perform operations on external variables.

The **Context** function, which can be accessed via the button at the top of the simulation window, enables a current state of operator panels and external variables to be saved : the recorded context is restored when simulation is initialized.

During simulation, this function can be used to check the state of the operator panel and external variables : by selecting a context from the window, enabling them to be created, the difference between the current situation and the context is shown in the right hand part of this window. Variables with a state which does not conform to the context are signaled and by clicking on one of these variables, the element, and the operator panel to which it is attached, are displayed. The **Restore** button enables the state of these variables to be made consistent with the selected context.



4.2-4 Concept of a "Scenario"

The concept of context is completed by the possibility of recording and restoring a series of actions on Boolean and numeric external variables and operator panel elements in order to automate command sequences (start sequence, restart cycles, etc).

This is a particular case of the use of scenarios. The concept of a scenario is explained in section 4.8 of this part.

4.3 Display Tools

We have just seen in the previous section how the simulator can be controlled via operator panels and actions on external variables. It is also important to be able to observe evolutions of the simulated model under good conditions, other than via the simple exchange of I/O or through the prediction of expected events.

Functions have been designed in order to do this : these are the "dynamic pages" and "views" functions.

The **examination** function also enables simulation to be observed. Due to its special nature, this function will be outlined in a separate section (see section 4.5 of this part).

E

4.3-1 Dynamic Pages

This tool enables the state of the previously selected variables to be permanently displayed.

This function can be accessed by selecting the **Pages** item from the **Tools** menu located in the lower part of the simulation screen.

The first window is then displayed, enabling a new page to be created, and previously configured pages to be edited, displayed and deleted.

The **Create** button opens a window where the name of the page to be configured is entered. After confirmation, the page can be edited.

Editing a page

The edit window comprises a zone on the right where the variables which constitute the edited page are displayed.

This zone is empty if a new page is to be created. The center of the window enables the selection of the type of variable to be shown in the page.

The following choices are available :

- I/O configuration
- relays and general power supply to the relay interface
- description variables of the "moving" part
- operation traps

Selecting the type of variable makes the list appear in the left part of the window.

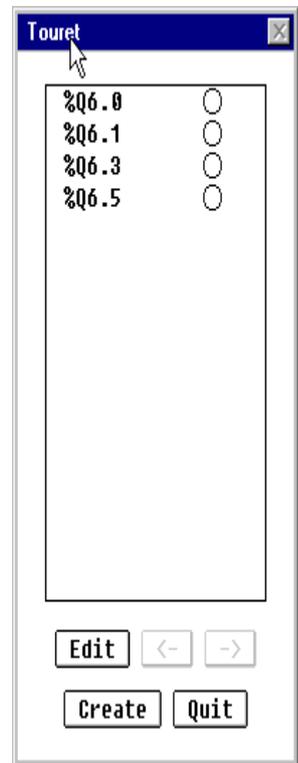
The page can be easily configured by selecting variables from this list using the mouse.

A variable can be removed from the page in the same way by clicking on it in the right part of the window. A variable can also be added to a page using the editor located at the bottom middle part of the window.

If the entered variable is correct, it is transferred to the right part of the window during validation.

As well as the choice of variables previously described, sensors, bistables, movements or external variables can also be incorporated in the page.

If the entered variable does not correspond to one of the variables of the type selected, the list of variables presented is scrolled alphabetically.



Display of dynamic pages

Clicking the **Display** button in the main window of the **Pages** function displays the variables which make up the selected page in the right part of the background simulator window. Variables are arranged in a column and their state is indicated by a "Radio button" icon.

The display window has **—>** and **<—** buttons in order to access the following or previous page.

The **Edit** button allows the list of variables which make up the page to be modified.

It is also possible to create a new page using the appropriate button.

Note

The reason for the logic state of a variable displayed on the page can be analyzed using the Examination function. This function can be accessed when the simulator is stopped or blocked by clicking on the variable concerned.

4.3-2 Views

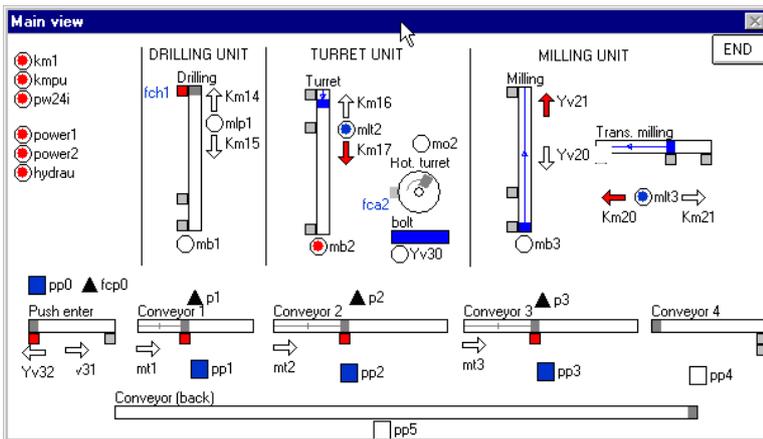
In the simulation environment, previously described views appear in the form of windows (views), except for the view titled **Background** which appears in the main simulation window (see section 4.1 of this part). The Grafcet chart view windows are accessed in the same way as other views. Their use is described in section 3 of this part.

Using views

If the views have been described, the **Views** button accesses a menu for selecting the view(s) to be displayed.

The **All** item displays all views with a single command. The **End** button enables the view displayed to be deleted.

When the user quits the simulation, the positions of the views on the screen are maintained.



Animating views

Animating axes

An axis is represented by a contour (vertical or horizontal rectangle or circle) in which the current position evolves by means of a cam or a bargraph.

The states of axis sensors are "illuminated" when they change to 1.

Animating variables

The variables of a view are "illuminated" when they change to 1 and "off" when they change to 0.

Animating decoders

A calculation is made according to the value of the decoder bits. A figure indicates the current reconstructed value.

Examining view variables

When the simulator is stopped or in the event of blocking, it is possible to examine the view variables, by double-clicking on the graphic element concerned.

4.4 Forcing Description Variables

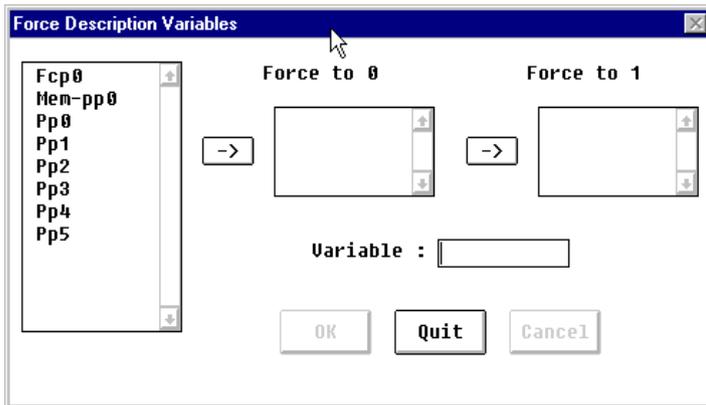
These variables are equivalent to relays in the internal operation of SIMTSX. They are described in the "Axes" of the configuration interface.

They cannot be subjected to faults in the simulation environment.

Forcing enables direct action on the logic state of these variables which is normally defined by a Boolean expression.

This enables for example, in the case where these variables represent the presence of parts, the insertion of a part into any location in the simulated application without this having been explicitly planned. This also enables a modeling error to be temporarily corrected (incorrect variable equation).

If not used in a controlled manner, this "forcing" may cause inconsistencies in the modeling.



4.5 Examination or Analysis of Situations

"Examination" is a function which can be accessed at various levels. It can be accessed when simulation is "stopped", in other words on blocking, or at a stop.

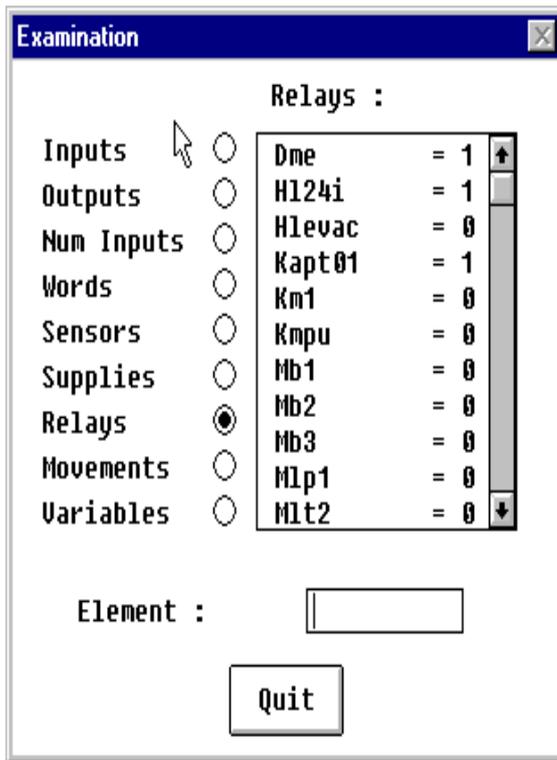
The main window of this function is activated using the **Examination** item of the **Tools** menu.

This window comprises a number of buttons and an editor. Clicking the mouse on one of these buttons displays the logic states of the type of variables selected.

An examination of one of the variables present can be made by clicking on it with the mouse.

Another way of examining a variable is to enter it in the editor of the **Examination** window.

Validation starts an examination of the variable if that variable is known to the simulator. If not, the list of variables present is scrolled alphabetically according to the position of the variable entered.



4.5-1 Examination of Variables Whose State is Determined by a Boolean Expression

This is the case for the input configuration, relays or variables of the moving parts.

If the variable is at logic state "0", the terms which are missing for checking the variable equation are presented.

If the variable is at "1", the explanation of this state, based on the analysis of the equation, is given. The **Set to 0** button produces the conditions used to reset the variable to "0".

The list of variables missing or explaining the true state is presented in a scroll-down list in the right part of the window. By selecting one of these variables, the examination is carried out in more depth.

E

4.5-2 Examining Numeric Variables

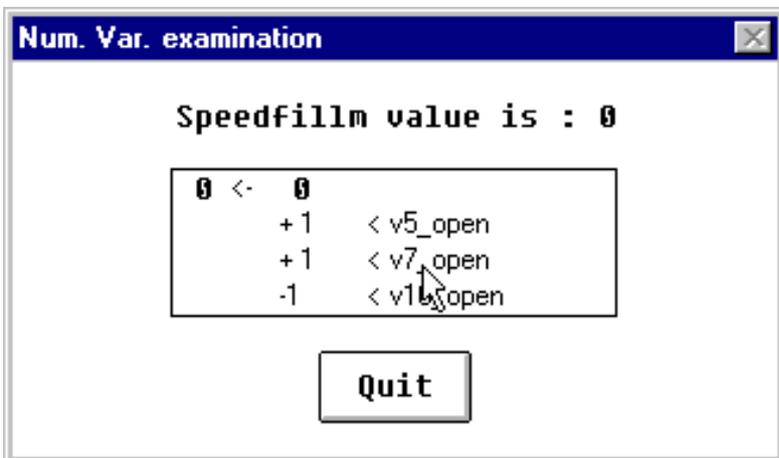
There are two possible ways to access examination of numeric variables :

- from the **Examination** item of the main menu
- by double-clicking in a view on the box associated with the required numeric variable (it should have been declared during description of the views)

The current value of the numeric variable is displayed.

The table associated with the variable is also displayed. Assignments whose associated condition has been checked are displayed in bold with the intermediate result.

Examination can be performed by clicking on a variable used in an assignment or in the condition which may be associated with an assignment.

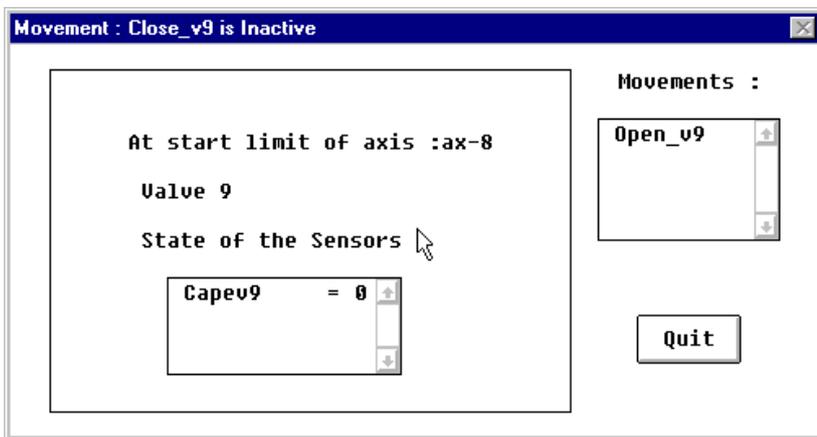


4.5-3 Examining Movements

When a movement is examined, the following is possible :

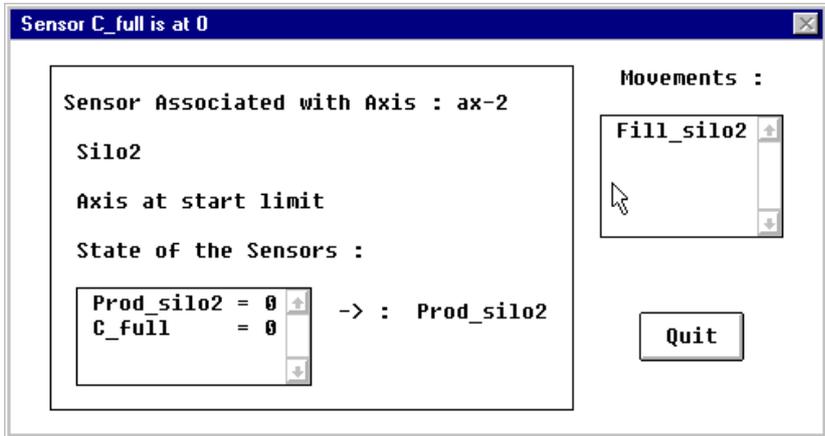
- If the axis with which the movement is associated is linear and the movement is at the end limit, this is indicated and the movement equation is not analyzed.
- If the movement is inactive but another movement is active for the associated axis, this movement is indicated and an examination of this is started when the window is quit.
- If the movement examined is inactive without being at the end limit, the analysis of the equation provides the missing conditions.

The examination indicates "impossible movement" : this occurs when the sensors of the associated axis are involved in the missing conditions.



4.5-4 Examining Sensors

Examination indicates the state of a sensor. In the right part of the window movement(s) are displayed which enable the sensor to be quit or attained depending on whether it is active or inactive.



4.6 Operation Traps

Using the traps, the simulator can be stopped if a particular situation should arise. Traps also enable the occurrence of events to be counted, the time lapse between two successive occurrences or the duration of a particular situation to be measured.

E

4.6-1 Characteristics of a Trap

A trap is defined by an event and/or a condition. The variables to which these terms apply are I/O, relays and description variables. If the condition is not defined, it is always true. The trap is checked if, on the occurrence of the event, the condition is validated. If the event does not exist, the condition is systematically tested. Text of 40 characters can be added to the trap mnemonic.

The numeric variables and the sampled axes can be used in the condition of these traps by comparing their values.

Example : threshold trap condition : (position > 150)

Edit Trap

Mnemonic :

Label :

State to Trap

Event :

Condition :

Stop Simulation : Threshold : with RST :

The up arrow (and down arrow respectively) is used to select the change of the variable to be trapped to 1 (to 0 respectively).

4.6-2 Evaluating Traps

Inputs, outputs and, if required, relays are involved in the definition of a trap.

SIMTSX evaluates the traps after reception of outputs from the PLC, and propagation in the relays.

4.6-3 Action of Traps

A trap is activated by a message composed of a mnemonic and the trap text. The elapsed time (cumulative total of the machine evolution times) since the last trap occurrence is indicated. If the trap does not contain a releasing element, its deactivation is signaled, as well as the duration of this activation (time during which the condition has been verified).

The threshold associated with a trap enables the start of an action to be delayed until after a certain number of occurrences.

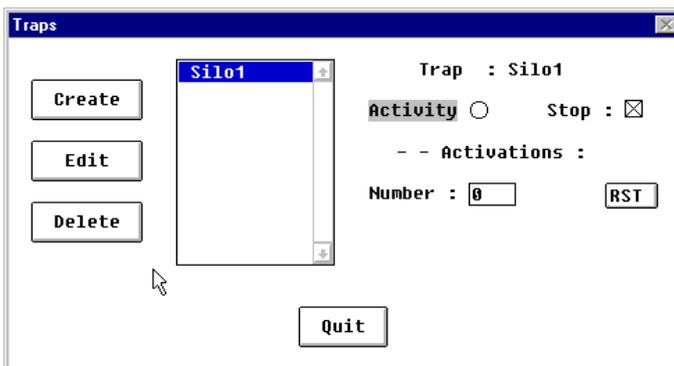
If the **Reset (RST)** option has been selected, the action will start every n occurrence, n being the threshold value.

The simulation stop caused by the trap can be deactivated : its activation will be "silent" but a cumulative total of its occurrences will nevertheless be kept : the trap can therefore act as a counter.

Note

A trap can be incorporated in a page of variables ; its state is therefore displayed dynamically.

When the simulator is stopped, an "examination" can be started by clicking on the name of the trap. If the trap possesses a condition, the examination corresponds to the analysis of this condition.



4.7 Faults

There are three types of fault :

- Sticking at 0 or at 1 of the sensors, relays and bistables : when the element involved changes to the corresponding state, it remains this way until the fault is corrected (or forced to the opposite state).
- Forcing involving the same elements : the change to the selected state is immediate.
- Bounces on the sensors, which may occur on activation or deactivation.

These functions can be accessed by the **Faults** item in the **Tools** menu.

E

4.7-1 Sticking and Forcing

In the left part of the window which enables this type of fault to be set, there appears a list of the entities concerned.

After selecting one of these variables, the chosen fault is set using one of the buttons.

The variable on which the fault is to be placed can be selected using the editor which is below the list in the left part of the window.

This fault is cleared by clicking on a variable displayed in the fault zone.

Note

The **Cancel** button does not delete faults which have been set, but indicates their current state.

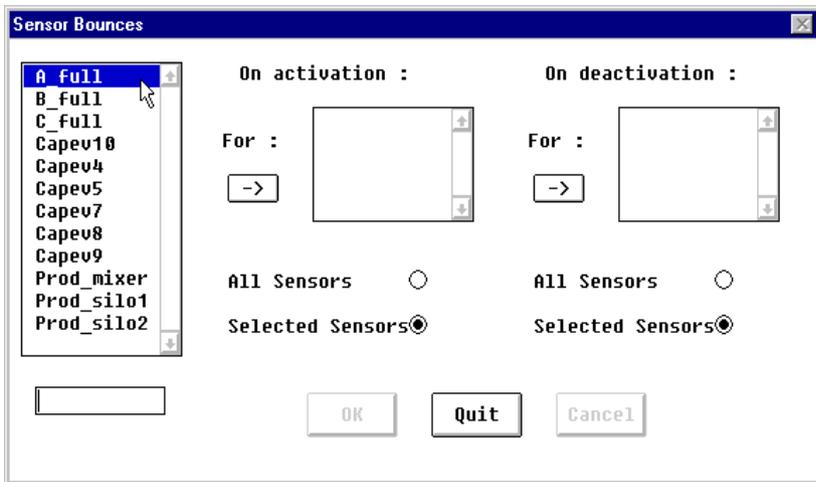
Sticking affecting sensors is taken into account in the input configuration equations and in the relays in the relay interface where these elements are involved. However, it is not perceived in the axis movements nor the "description variables" of the mechanical application description.

4.7-2 Sensor Bounce

When a sensor affected by a "bounce" fault changes state, this fault sends to the PLC 3 successive changes of state of the inputs where the sensor occurs. The state of this sensor changes twice after changing to the correct state.

At least one PLC scan elapses before these input changes appear on the I/O bus : the PLC therefore has the time to perceive them and advance its program. The outputs are not sent to the simulator until after the three successive elements have been received.

The "sensor bounce" can be set either by selecting the sensors on which the fault must be set, or by setting it on all the sensors, except for a few. Sensors are selected via a list or by direct entry using an editor.



4.7-3 Notes

- It is possible to simultaneously delete all faults which have been set. To do this, use the **Delete All** item in the **Faults** sub-menu.
- Upon initialization, the simulator requests if all faults which have been set are to be deleted.

4.8 Scenarios

A scenario is used to automatically restore sequences of actions on the operator panels and external variables. This facilitates the automated system starting procedures - power-up, start-up - or changes of operating mode during operation.

These sequences are created by saving and their operation can be controlled by "wait periods".

A scenario can be used to define breakdowns, which can be useful from the point of view of operator training.

These scenarios, set up using an edit tool, enable faults to be automatically set during simulation according to certain events. To meet this requirement, the number of trap activations during a wait period can be tested. A sensor can therefore be broken after a certain number of parts have passed, for example.

Finally, the concept of "reaction" is used to check the conformity of an evolution of an automated system subassembly with a reference model.

This reference is a scenario composed of reactions. It is established by recording the changes in state of variables associated with the scenario. During operation, these changes in state are considered as reactions which must be reproduced. The scenario must be executed under the same conditions : the wait periods enable it to be synchronized with the evolution of the automated system.

4.8-1 Sequences of Actions on Operator Panels and External Variables

Record

Firstly a scenario must be created by giving it a name and, if required, a comment.

Select the scenario and click on the **Record** button of the scenario management window.

After confirmation, recording is activated. The **Scenario** button of the Scenario window is then grayed out and its text changes to **Record**.

Simultaneously a subsequent window appears. **Quit** is used to close this window. The **Record** button in the background window enables it to be recalled. The **Save** button is used to end and save the recording.

As the actions are performed on the operator panels and external variables, the changes in state for the corresponding variables are displayed in this window.

The syntaxes are as follows :

- <var> <- 1
- <var> <- 0

for the change to 1 or 0 of a variable.

In the case of pushbuttons, the syntax is as follows, depending on whether the contact is normally open or normally closed :

- <var> <> 1 **or** - <var> <> 0

In the case of switches with n positions and thumbwheels, several changes of state of variables may occur simultaneously : these are "synchronous actions" whose syntax is as follows :

- <var1> <- 0
- <var2> <- 1

If actions are performed at different simulation times, an indication of the time elapsed since the previous action - or since the beginning of recording if it is the first action - is added to the recorded scenario line. For example, the syntax in the case of a change to 1 of a variable is :

- <var> <- 1 d: <duration>

In the case of synchronous actions, the indication of duration follows the first change of state displayed.

A reaction involving a numeric variable can be inserted into a scenario. The syntax is as follows :

<numeric variable> <predicate> <value>

with an indication of the duration if necessary, if there is a delay in the appearance of the reaction.

Note

The maximum recording capacity for a scenario is 150 lines. Above this, recording is interrupted and the user is requested to save.

Restoring the sequence of actions

In order to do this, click on the **Execute** button of the Scenarios management window. After confirmation, execution is launched. The **Scenarios** button is grayed out and its text changes to **Execution**.

During execution of a sequence, the operator panels are locked and modification of the state of external variables and setting of faults is inhibited.

The recorded actions are restored, respecting possible duration differences, unless blocking occurs : in this case the action is executed without the associated wait period being taken into account. The time reference used is "machine time" and not "realtime", which renders inoperative those sequences where PLC time delays are used.

If the **Execution** button is clicked a window appears in which the contents of the current scenario are displayed. The next action to be carried out is then indicated in reverse video. The **Stop** button of this window enables the execution of the scenario to be stopped before it reaches its end.

Other recorded actions

Besides the changes in state of the external variables and operator panels, the setting and clearing of faults on sensors, relays and bistables are also recorded together with the forcing of Boolean modeling variables.

For example, the sticking at 1 of a sensor has the following syntax :

- <sensor> : stuck at 1

Enhancing a scenario

The **Complete** button is used to add actions to an already existing scenario. The existing scenario is executed and the new actions are then recorded.

4.8-2 Editing a Scenario

A scenario consists of a series of lines. During execution these lines are processed sequentially.

As well as actions, a scenario can include wait periods and reactions.

The edit function can be accessed via the **Edit** button in the Scenarios management window.

The edit tool edits an existing scenario - obtained by recording - and also enables a new scenario to be created.

Deleting a line removes undesirable actions.

A line can be modified.

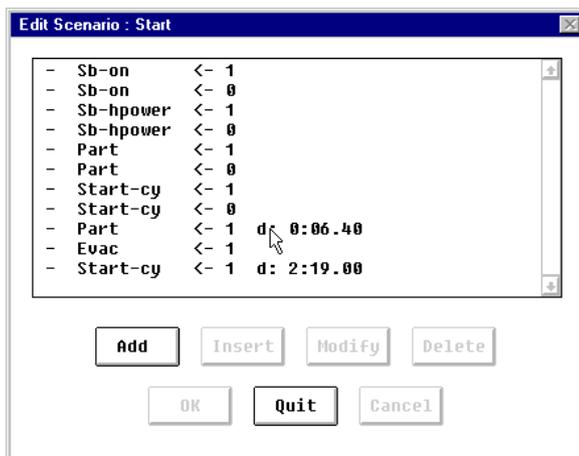
Finally, it is possible to add a new line at the end of the scenario or insert a line before a selected line.

Modifying a line

Select the line to be edited and click on the **Modify** button. An editor, located in the right part of the window, contains the relevant text for the line. The modification can be made directly in this editor : the syntax is checked on validation. The modification can also be confirmed by clicking on the **OK** button.

The **Revert** button in the edit window restores the various line elements starting at the end of the line.

Each time the **Revert** button is used, the substitute choices are displayed in the left part of the edit window. Simply select an element from this list in order to reconstruct a line, for which the syntax will, of course, be correct.



Creating a line

Two buttons are used for adding lines at the end of a scenario or inserting lines. The creation window is the same as for modification.

Initially, the editor containing the line text is empty and the selection list contains the various types of line which can be added or inserted into a scenario, depending on the type of line preceding and following the line to be created.

A new line can be created, in the same way as it is modified, by directly using the editor and keeping to the syntax, or by successive selections from the list, the content of which is modified as the line is created.

Note

After confirming the creation of a line, another line can be added afterwards without quitting the edit window.

4.8-3 Concept of Wait Periods

Sequences obtained by recording do not contain synchronization conditions for controlling their execution.

"Wait periods" are used to postpone the execution of a scenario until an event occurs or a condition has been checked.

Inserting wait periods by editing

To edit the scenario obtained by recording, select the action - or the first of a group of synchronous actions - whose execution is to be controlled by a wait period, click on the **Insert** button and select the **Wait** item.

The types of variable which may occur in a wait line are then displayed.

For Boolean variables possible wait periods are the change to 1 or to 0 of a variable (event) - and verification of logic state 1 or 0 of the variable (condition).

Corresponding syntaxes are as follows :

 <- <var> -> 1 or <- <var> -> 0
and
 <- <var> :: 1 or <- <var> :: 0

Traps :

A distinction is made between traps which possess an event and those which do not. In the first case, the change to 1 of a trap, whose activity is transient, is expected. However, if the trap has no event, its activity - represented by its condition - is processed in the same way as other Boolean variables.

A wait period can be formed by testing the counter associated with a trap by comparing it to an integer value. A specific action enabling the reset to 0 of the trap counter is available : it must be inserted in the scenario using the edit tool.

Recording wait periods

Another way of including a wait period in a scenario is to "prepare" the scenario before recording it.

Preparation of a scenario consists of assigning it variables - these can only be Boolean variables - before recording.

The preparation window, which can be accessed by the **Prepare** button, contains a selector switch in the top part of the window containing the **Wait** and **Reaction** items. The default setting is **Wait**.

The central part of the window is used to select the required type of variable : variables concerned appear on the left. When a variable is selected, it is placed in the list of **Wait periods** on the right of the window. It can be removed by clicking on it with the mouse. The required variable can be entered directly via the editor. It must then be confirmed so that the selected variables can be associated with the scenario.

During recording of the scenario, changes in state of the associated variables are automatically inserted in the actions performed in the form of wait periods.

Executing a scenario which contains wait periods

During execution of a sequence of actions, operator panels are locked and setting of faults is inhibited.

If the scenario contains a wait line, the user has time to modify the interfaces as long as the wait period has not expired. This makes it possible to create a scenario which is not executed until a wait period expires, and then triggers a series of actions.

Enhancing scenarios with wait periods

The principle of enhancement consists of rerecording a scenario which has already been executed.

If the line currently being executed is a wait instruction, it is reinserted in the scenario and control is passed over to the operator panels and to faults. Actions performed are not recorded until the wait period has expired. This enables a recording to be conditioned by including a wait period in a scenario.

4.8-4 Verifying Reactions

The use of scenarios has, up to now, only been discussed in the form of sequences of actions whose execution can be controlled via wait periods.

This function can also be used to verify the behavior of automated system subassemblies using the "reaction" concept.

These reactions are the changes in state, obtained by recording, of variables associated with the scenario in the "preparation" phase.

In order to execute this scenario correctly later, the changes in state of variables, or "reactions", have to be reproduced in the same way as during recording.

Recording reactions

Firstly, we must define variables for which the changes of state are to be considered as reactions to be monitored.

This is done using the preparation window. The **Reaction** item in the upper part of the window must be selected.

Note

A special variable entitled "Blocking" can be monitored. It is true when no movement occurs.

When recording is launched, the following window displays the scenario. It comprises, besides the changes in state of variables resulting from actions on the operator panels and external variables, changes in state affecting variables associated with the scenario in the form of reactions.

Reactions give an indication of duration. During recording, the duration which is displayed in the window corresponds to the time elapsed since the last action or the beginning of recording. Once the scenario is saved, only the last line in a series of consecutive reactions retains this duration indication.

The syntax of a "reaction" line for the change to 1 of a variable is :

```
=> <var> -> 1 d:<duration>
```

As is the case for actions, reactions can be synchronous. This is the case if variables, whose changes in state are immediate consequences of one another, are associated with scenarios. On the appearance of an output, the latching of the relay and activation of the movement it controls are events which occur simultaneously with the change in state of the output.

The syntax is :

```
=> <var1> -> 1 d:<duration>  
    <var2> -> 1
```

Notes

- A wait period enables the beginning of a recording to be conditioned on completion of a scenario. This is also true if the scenario contains reactions which are associated during preparation. Monitoring the evolution of corresponding variables is delayed until the wait period has expired.
- Wait periods can be recorded, if they are associated with a scenario via the preparation function. In this case, the "wait" variables and "reaction" variables are mutually exclusive.

Conformity of reactions

The execution of a scenario containing reactions may or may not conform to criteria. Consider a scenario beginning with a wait period enabling the execution to be synchronized and containing only reactions. The wait period can be a trap characterizing the beginning of a cycle of the automated system subassembly, the reactions and movements of axes forming this module. This scenario could be constituted by extending the wait period defined during editing.

The execution of a scenario consists, once the "start of cycle" wait period has expired, of verifying that the reactions - here the changes in state of movements of module axes - reappear in the same order and before a limit time corresponding to the duration associated with the last reaction recorded.

Sequencing actions and reactions

If the scenario contains actions, the principle of execution is as follows :

- Actions are executed according to the wait period which may be assigned to them. While an "action" line is being processed - its execution wait period not yet elapsed - a check is made that no evolution of the variables associated with the scenario takes place.
- When a reaction is being processed, SIMTSX checks that it occurs well before the assigned wait period and that no other scenario variable evolution takes place.

Execution report

The execution of a scenario can be displayed by clicking on the **Execution** button in the simulation window. Lines currently being processed appear in reverse video in the window.

In the case of scenarios with reactions, an execution report window is systematically displayed, it can be removed once the scenario has finished.

E

If the scenario execution is not correct, in other words if the reactions do not appear within the anticipated wait periods or if untimely evolutions occur, the "anomalies" are indicated in this window.

In this case simulation is interrupted and the scenario terminated. The window enables the user to see at what level in the scenario the deviation occurred.

Enhancing scenarios with reactions

In this case enhancement consists of rerecording the scenario - actions and evolutions of variables associated as wait periods or reactions - by firstly executing the actions of the initial scenario.

If the scenario contains wait periods, only those which are not after a reaction are retained for execution. In fact, since recording is postponed during wait processing, these reactions can no longer figure in the new scenario.

The essential use of enhancement in the case of reaction monitoring is to be able to correct a scenario : it is sufficient therefore to modify these variables and to "replay" the scenario using its enhancement function.

4.9 Initialization

The **Initialize** command enables the application to be returned to how it was when the simulation was launched.

Everything takes place as if a new simulation is being started : the context selection window is displayed. If no context is selected, the operator panel variables and external variables remain in their original state.

If a machine state - or "mechanical context" - should have been saved during a previous simulation, it is possible to restore it.

In online mode, the PLC program should probably be initialized in order that simulation can be correctly restarted.

4.10 Saving and Restoring the Application State

When a simulation is launched offline or online, the axes are in the initial position. All variables described by Boolean expressions are at logic state "0".

This may be disadvantageous however, when simulating a transfer system for example. Saving the application state enables simulation to be resumed if the application is found in the state obtained during a previous simulation.

The principle consists of being able to save, when quitting the simulation, the position of application axes and the logic state of all the variables described in the "moving" part of the application. Offline simulation can be performed using Grafcet charts, whose state is saved : in other words active steps, positioned internal actions and the value of counters.

The logic state of external variables or variables activated by operator panels is also retained : saving an application state thus includes the concept of context.

This save may be requested when quitting simulation. To do this, check the **Save Machine State** box which appears in the confirmation window. A save made previously is lost.

The application state can be restored by checking the **Restore Saved State** box located in the context selection window.



4.11 Description

The "Description" function is used to display the description of the modeled application during simulation and to make certain modifications.

This function can be accessed via the corresponding item from the **Tools** menu.

It is possible to modify the logic equations for the inputs, relays and variables as well as the characteristics of the movements.

These modifications can all be made during offline simulation as this is a methodology phase aimed at debugging the model.

However, during online simulation, the modifications which can be made are limited to the "electrical" part of the model, that is, the input and relay logic equations.

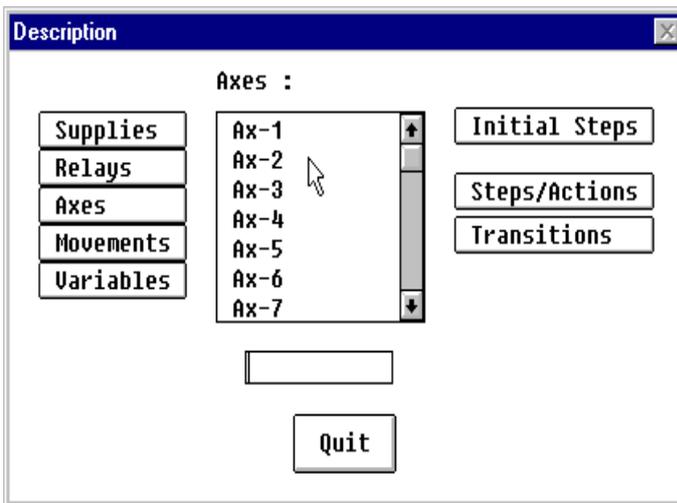
The sensors associated with the axes cannot be modified but they can be displayed in the form of a trend diagram or "evolutions".

During offline simulation, it is also possible to modify the description of the control system. This means it is possible to correct the actions associated with the steps or transition conditions.

The initial steps can also be modified, which makes it possible to test Grafcet charts independently.

Once simulation is complete, if modifications have been made, the simulator asks if these changes should be saved.

The old description files are then renamed with the extension .BAK.

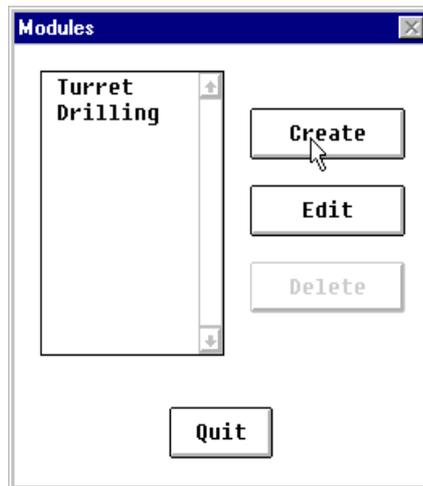


5.1 Configuration

Before executing a trace on an installation, the first operation is to configure the modules in order to structure the representation of the application. To do this, select the **Configure** item from the **Trace** menu in the main window in the **SIMTSX** simulation environment.

An automated installation is often made up of mechanical subassemblies which may be relatively independent of the animation program. The "modules" function is used to give the simulator this type of structure. A module is a set of axes. It is said to be active as soon as a movement is made on one of the axes which make up the module. It is said to be inactive if there is no movement.

The module configuration has some similarities with that of the pages of variables (see section 1.7, part D). The **Create** button is used to construct a module ; after having entered the name of the module, the user moves to the edit screen.



5.1-1 Editing a Module

The window for editing modules comprises, on the left-hand side, all the axes of the simulated application. On the right-hand side of the screen are the axes which comprise the module being edited.

If an axis is selected from one of these two lists, its label appears in the top center of the window and the buttons become active.

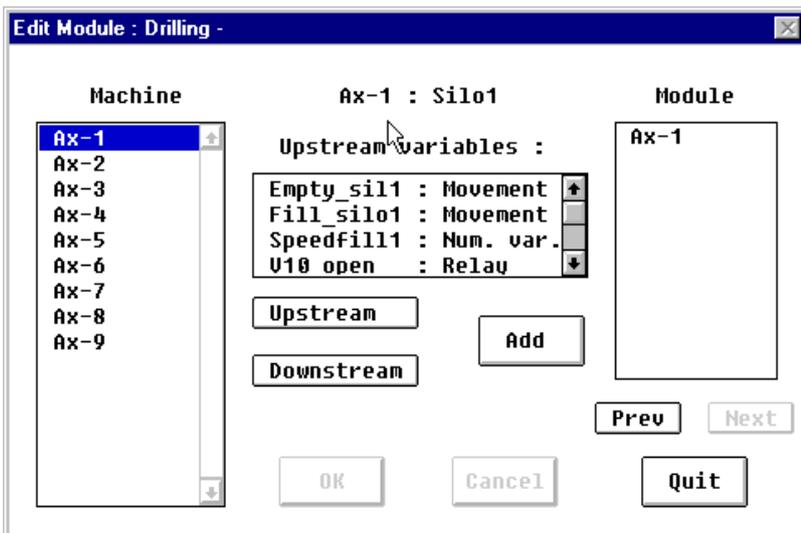
The **Downstream** button is used to display in the center of the window all the variables which are located "downstream" of the axis, that is those on which any evolutions to the axis are likely to have any effect.

This involves the axis sensors, the combinations in which these sensors are used, and any description variables or relays and inputs concerned with the axis.

Similarly, the **Upstream** button is used to display the variables located "upstream" of the axis, that is those used in the evolutions to the axis. This involves movements, relays, bistables and outputs controlling the axis.

These variables are obtained from the existing chart of the description variables, excluding the relays which are part of the general power supply.

The **Add** button is also active if an axis is selected from the list on the left-hand side (all the application axes) : this button is used to add the axis to the module. If an axis module is selected (right-hand side of the window), the application axis will be inserted before this axis in the module, otherwise it is simply added at the end of the module. However, if it is an axis belonging to the module which is selected, the button is used to delete it from the module.



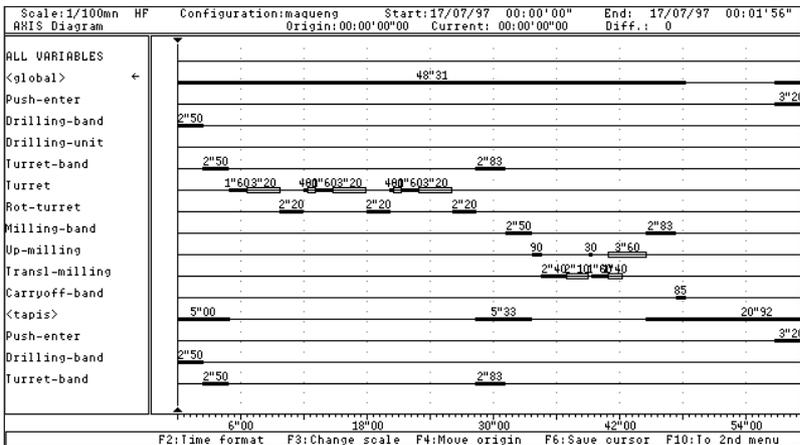
5.2 Presentation Elements - General Functions

5.2-1 General Presentation

This section explains the basic principles required for representing trend diagrams. It describes the following :

- the elements in the representation window
- how to order the representation
- how to print all or part of a trend diagram
- how to create and use the reference cycles

The first diagram presented to the user when he views a record is called the **Axis Diagram**.



This diagram offers an overall view of the operation giving the changes in state of the modules (active, inactive), and, for each of them, the activity of the associated axes. The first line, **All variables**, is used to access a diagram of the evolutions of all the I/O and application movements.

If no module has been created before the record, the diagram simply shows the activity of the axes.

Moving within this trend diagram the user can select a specific instant in the record (horizontal cursor keys) and a module or an axis (vertical cursor keys).

If a module is selected, pressing the **<Enter>** key displays the **diagram of the module movements**.

If an axis is selected, pressing the **<Enter>** key displays the **diagram of the axis I/O** and a summary in the first line of the activity of the axis itself.

Function keys can also be used to switch to a functional representation, known as an **FA representation**, from the **movement diagrams** and the **I/O diagrams** (see below).

```

I/O Grps      HF      Module: <tourelle>      Start: 17/07/97  00:00'00"      17/07/97  00:01'56"
FA Representation With filter      Origin: 00:00'00"00      Current: 00:01'10"75      1'10"75
20"20      mvt+  dtour
20"60      /mvt+ dtour
20"60      mvt-  mtour
21"40      /mvt- mtour
21"40      mvt+  dtour
23"00      /mvt+ dtour
23"00      mvt-  mtour
26"20      /mvt- mtour
26"20      mvt+  rtour
28"40      /mvt+ rtour
28"40      mvt+  st2
31"23      /mvt+ st2
***** END OF CYCLE *****
1'10"75      mvt+  st2
>> ***** START OF CYCLE *****
1'13"25      /mvt+ st2
1'13"25      mvt+  dtour
1'14"85      /mvt+ dtour
1'14"85      mvt-  mtour
F2:Outputs      F3:no filter      F4:Filter      F6:Save cursor      F10:To 2nd menu
  
```

In addition to changing the time base, various functions are used to :

- calculate the time difference between two events
- organize the sensors/actuators used according to numerical, chronological or structural criteria
- modify the reference point of the diagrams shown
- print diagrams

5.2-2 General Principle

The general principle of the man-machine interface and the functions which can be accessed at each level are summarized in the table below :

Type of diagram	Fx function keys	Shift + Fx function keys	Alt + Fx function keys
Axis diagram	F1: Information F2: Time/Mechanical Format F3: Zoom F4: Change ref. point F6: Memorize cursor F8: Print screen F9: Printout Tab: Menu window	None	None
Movement diagram I/O diagram	F2: I/O or Outputs F3: Change scale F4: Num/Chronological order F6: Memorize cursor F7: I/O label F8: Print screen F9: Printout Tab: Menu window	Shift+F1: FA representation Shift+F2: Change time format Shift+F3: Input color Shift+F4: Change ref. point Shift+F5: Zoom 1 min Shift+F8: Cont/evolution mvmt	Alt+F1: Save reference Alt+F2: Comparison active Alt+F3: Ref. cycle duration Alt+F4: Activate ref. mvmt
FA Representation	F2: Sensor/Act. or Actuator F3: With or without filtering F4: Filter/add selected I/O F6: Memorize cursor F8: Print screen F9: Printout Tab: Menu window	Shift+F1: FA representation Shift+F2: Change time format Shift+F3: Input color Shift+F4: Change ref. point Shift+F5: Zoom 1 min Shift+F8: Cont/evolution mvmt	Alt+F1: Save reference Alt+F2: Comparison active Alt+F3: Ref. cycle duration Alt+F4: Activate ref. mvmt Alt+F5: Activate resynchro Alt+F6: Find deviation Alt+F7: Change ref. ref. point Alt+F8: Reference cycle info

Pressing the **<Tab>** key displays a menu which contains all the functions available for the current representation.

Each function can thus be accessed either via a function key (**<F2>** for example) a combination of keystrokes (**<Alt+F2>** for example), or the menu described below :

This menu is made up of 2 parts :

- The first part contains the title of the function keys.
- The second part contains information relating to the pages in the menu. In our example :
 - ← indicates that pressing the LEFT arrow key will display the previous page in the menu.
 - → indicates that pressing the RIGHT arrow key will display the next page in the menu.
 - The menu windows are defined in a dropdown list as shown in the illustration below (once the last window has been displayed, the user is returned to the first). The number of the window in a menu differs depending on the type of representation.



Once this menu is displayed on screen, it is possible to :

- Select a function :

by selecting a function key (eg : by pressing **<F3>** to change the scale)

by placing the cursor (represented by an arrow to the right of the menu label) before the chosen function using the up or down arrow keys (\uparrow \downarrow) then confirming the operation with **<Enter>**(↵)

The menu will close and the selected action will be executed. The next time the **<Tab>** key is pressed, the menu will reappear, with the cursor next to the last function selected.

- **View another page in the menu by :**

pressing the left and right arrow keys (\leftarrow \rightarrow)

- **View a particular page in the menu :**

<Shift> : to display the **<Shift+Fx>** type functions

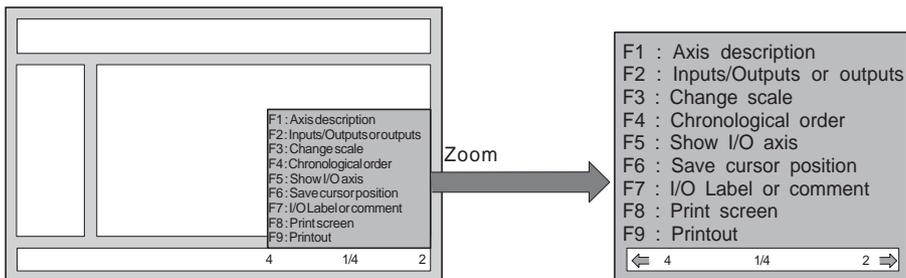
<Alt> : to display the **<Alt+Fx>** type functions

<END> : to display the arrow, **<Ctrl>**, **<Esc>**, etc, keys

- **Clear the menu window by :**

pressing **<Esc>** or **<Tab>** (no action is performed)

selecting a function as explained above



5.2-3 Page Header

The top of the screen page contains the following elements :

- indication of the current time scale
- the name of the application in the **axis diagram**, the label of the module in the **module movement diagram**, the name of the axis in the **axis I/O diagram**
- the record start and end dates and times
- the type of diagram shown
- indication of the current duration format :
 - TF : Time Format (hh : mn' ss» xx)
 - MF : Mechanical Format (hh: mm' yyy)
- indications of the reference point and the current position of the cursor

By default, the reference point corresponds to the record start time. The deviation represents the difference between the current position of the cursor and a previously memorized position (by default, this position corresponds to the reference point).

5.2-4 Moving Around in the Trend Diagram

- **The cursor**

The cursor is represented by a vertical bar which can be moved around on the page using the left and right arrow keys in fifths of a time unit in the **axis diagram**. In the **movement** and **I/O diagrams**, pressing a left or right arrow key moves the cursor between evolutions (this includes all the evolutions to all the I/O and movements).

With each movement, information relating to the deviation and the current position in the page header are updated.

Keys :

<Shift+F8> : used to switch from a continuous mode (in fifths of a time unit) to an evolution mode (move from evolution to evolution).

<Ctrl+→> or <Ctrl+@> : quicker movement (10 units).

<Ctrl+Home> or <Ctrl+End> : moves the cursor to the start or end of the trend diagram.

- **Page break**

A page represents 100 time units in a record (6 s, 1 min, 10 min, 100 min depending on the scale). When the cursor is moved to 5 time units from the end of a page, a page break is generated which shifts the window shown on screen by 50 time units.

5.2-5 Precision

The precision of the diagram is directly linked to the modeling of the application. The times displayed in the trend diagrams and the FA representations are calculated based on the distances and speeds declared in **SIMTSX**.

The time scales available are :

- in the axis diagram :
 - 1 min or 60 s
 - 1/10 min or 6 s
 - 1 / 100 min or 600 ms
- in the movement diagram and the I/O diagram :
 - 1 min or 60 s
 - 1/10 min or 6 s
 - 1/100 min or 600 ms
 - 1/1000 min or 60 ms

The graph is positioned on screen with a precision of 1/5 of a time unit or 12 ms maximum.

Keys :

<F3> changes the time scale.

<Shift+F5> changes to the 1 min time scale in the movement and I/O diagrams.

5.2-6 Indications of Time

• The duration

The principle for constructing a diagram consists of tracing the horizontal segments representing the activation periods for each input, output or movement, or periods during which an active or inactive state of a module is maintained.

For each segment, the duration is indicated in the current format. By default, the time format is used and only the significant figures are shown.

Two types of format are possible :

- Time Format (TF) hh : mn' ss» xx, (example: 2'7"38)
- Mechanical Format (MF) hh: mm' yyy, (example: 2'138)

The indication > followed by a duration above a segment (for example : > 2"38) signifies that the corresponding element was already in the state indicated before the reference point or stays in this state after the end of the record.

Indication X above a segment (example : X) indicates that the activation is less than or equal to :

- 10 ms in time format
- 1/100th of a minute in mechanical format

The precision of the duration is always the same regardless of which time scale is used.

Keys :

<F2> : changes the date format in the modules diagram.

<Shift+F2> : changes the date format in the movement and I/O diagrams.

• Start and end of activation

When the cursor is placed at the start or end of the segment, the indication of the difference in the header corresponds to the difference between the instant when the event concerned occurred and the reference point.

The position indication corresponds to exact time at which the event occurred.

The reference point in the axis diagram represents the record start time by default. It can be modified via the "change reference point" function :

Keys :

<F4> changes the reference point of a record in the axis diagram.

<Shift+F4> changes the reference point of a record in the movement and I/O diagrams.

• Time difference between two events

The<F6> key is used to memorize the current cursor position. Pressing this key places a triangular marker on the time scale at the current cursor position. If the cursor is moved, this automatically calculates the difference between the memorized and the current position.

For example :

Ref. point : 15:37'53"32 Current : 15:47'26"11 Difference : 3"10/15:47'23"1

Key :

<F6> memorizes the position of the cursor.

• Vertical movement

The window shown on screen can be moved vertically by one line using the up and down arrow keys.

Keys :

<PgDn> and <PgUp> are used to move the window from module to module in the axis diagram or by 10 lines in the other diagrams.

<Home> and <End> are used to position the cursor on the first or last line.

5.3 Axis Diagram

5.3-1 Types of Element Shown

A screen page contains 16 lines representing the activity of the modules and axes over one time period. The activity of an axis is represented by a thick line (forwards) or a triple line (backwards) with a time representing the duration.

- **The "all variables" module**

The first module named "all variables" is fictitious. It contains all the declared I/O in **SIMTSX**. It is used to display all the I/O in the same diagram.

E

5.3-2 Changing the Reference Point

If a record contains more than 3000 changes in state of a module, the diagram represents the first 3000 from the reference point. This constitutes a sector.

Changing a reference point consists of moving the reference point of the diagram to the current cursor position, which moves the sector displayed in the overall record.

Note

Changing the reference point when the cursor is placed on the reference point repositions the reference point at the start of the record. The I/O diagram is limited to the first 3000 changes in state of the module I/O selected from the instant corresponding to the current cursor position. To display the remainder of the record, simply move to the end of the sector, then change the reference point.

Keys :

<F4> changes the reference point in the axis diagram.

5.3-3 Selecting a Module

Selecting a module displays the movements corresponding to the selected module, shown axis by axis.

The position of the cursor in the axis diagram will set the cursor reference point in the module movement diagram.

Keys :

- <↑>, <↓> move up or down.
- <Home> and <End> move to the first or last module.
- <Enter> selects the module.

5.3-4 Selecting an Axis

Selecting an axis displays the I/O and movements of the axis. The axis movements are placed at the end of the list. The first line summarizes the activity of the axis movements. The position of the cursor in the axis diagram will set the cursor reference point in the axis I/O diagram.

Keys :

- <↑>, <↓> move up or down.
- <Home> and <End> move to the first or last module.
- <Enter> selects the axis.

5.3-5 Exiting the Axis Diagram

Pressing the <Esc> key exits the axis diagram. This action can also be used to exit the representation software.

5.4 Module Movement Diagram

Selecting a module and an instant in a record in the axis diagram displays the module movement diagram.

The reference point of this diagram corresponds to the current cursor position in the axis diagram at the moment of confirmation. Any cursor movement will be reflected in the axis diagram.

E

5.4-1 Types of Element Shown

- **Axis status line**

The top of the trend diagram shows the module status line as it appears in the axis diagram. The user can then make a direct correspondence between the activity of the module and the activity of the movements.

5.4-2 Selecting the Elements Shown

The order in which the elements are shown on a trend diagram can be selected by the user. The types of element shown are always given in the page header.

The sort in chronological order can be performed by comparing the time at which the first edges of each element appear starting from the memorized position (<F6> key). Elements which have never been active or which are already active at the corresponding instant are shown at the end of the list.

Key :

<F4> is used to switch from numerical order to chronological order and vice versa.

5.4-3 Exiting the Movement Diagram

Pressing the <Esc > key exits the movement diagram. This action can also be used to return to the axis diagram.

5.5 Axis I/O Diagram

Selecting an axis and an instant in a record in the axis diagram displays the I/O diagram for the installation.

The reference point of this diagram corresponds to the current cursor position in the axis diagram at the moment of confirmation. Any cursor movement will be reflected in the axis diagram.

5.5-1 Types of Element Shown

- **Selecting inputs or I/O**

By default, the elements shown on screen are the I/O then the movements and the axes to be monitored for the module. The I/O are sorted in numerical order of their PLC denomination.

Key :

<F2> is used to switch between the diagram of the outputs and that of the I/O and vice versa.

- **Differentiating the colors of the input and output evolutions**

By default, the I/O evolutions are shown in yellow. However, to simplify reading of the trend diagram, it is possible to modify the colors of the input evolutions. Three colors are available : yellow, blue, white.

Key :

<Shift+F3> is used to change the colors of the input evolutions by switching from yellow to blue, blue to white then white to yellow respectively.

- **Axis status line**

The top of the trend diagram shows the axis status line, as it appears in the axis diagram. The user can then make a direct correspondence between the activity of the axis movements and the evolutions of the I/O.

5.5-2 Selecting the Elements Shown

The order in which the elements are shown on a trend diagram can be selected by the user. The types of element shown are always given in the page header.

The sort in chronological order can be performed by comparing the time at which the first edges of each element appear starting from the memorized position (<F6> key). Elements which have never been active or which are already active at the corresponding instant are shown at the end of the list.

Key :

<F4> is used to switch from numerical order to chronological order and vice versa.

5.5-3 Exiting the I/O Diagram

Pressing the <Esc> key exits the I/O diagram. This action can also be used to return to the axis diagram.

5.6 Representation in a Functional Analysis Table

5.6-1 Access

The structured table type representation is accessed by pressing <Shift+F1> in the I/O diagram or the movement diagram.

5.6-2 Presentation

This representation displays the same information as the trend diagram in the form of structured tables. It is used to follow the evolutions of the I/O chronologically and to label synchronous evolutions instantaneously.

5.6-3 Functions

The current cursor position corresponds to the time at which the first evolution is displayed on screen (marked by an arrow) and can be memorized by pressing <F6>. All the changes made to the FA representation remain valid in the trend diagram.

Note that most of the keys which can be used to represent the movements or I/O can also be used in the FA representation.

Keys :

<↑> or <↓> move the cursor evolution by evolution.

<PgUp>, <PgDn> move through the list of evolutions page by page.

5.6-4 Filtering

If the evolutions of one or more I/O "disrupt" the analysis or are not helpful in understanding the sequence of evolutions, the user can delete them temporarily from the representation. This operation is known as filtering.

To execute a filtering operation, simply switch to "FA representation with filtering" mode (indication in the header) by pressing **<F3>**.

In "FA representation" mode, if filtering is not active, any I/O which have already been filtered are marked with an *. In this case, pressing the **<F4>** key redisplayes the evolutions of the selected I/O in the representation with filtering.

Filtering only operates with the FA representation. However, it is possible to use this filtering function in the I/O diagram.

Keys :

<F3> activates or deactivates the representation with filtering.

<F4> is used to filter the representation of the I/O selected by the arrow, or to restore I/O which have already been filtered (marked by *).

5.6-5 Exiting the FA Representation

Pressing the **<Esc>** key exits the FA representation. This action can also be used to return to "standard" mode in the movement or I/O diagrams.

5.7 Comparing Cycles

5.7-1 Principle

This function is used to show two trend diagrams on screen saved at different times. The simultaneous representation of these two records is used to display the deviations in the operation of an installation at a given moment, in relation to the reference operation.

A reference trend diagram is memorized in the trend diagram by selecting all or part of this trend diagram and saving it.

The comparison can then be activated at any time. The reference trend diagram is placed next to the trend diagram already on screen. It is possible to move one trend diagram in relation to another to match up the changes in state of the I/O.

During the comparison of the trend diagrams, all functions are still available : changing the time base, chronological sort, selecting inputs or I/O, etc.

This comparison of cycles can be used in "I/O diagram", "movement diagram" or in "FA representation" mode.

5.7-2 Creating a Reference Cycle

The reference cycle is defined by the memorized position (accessible via <F6>), and the current cursor position.

If the two are mixed together, the whole record is saved (1500 evolutions maximum). A reference cycle can either be created in the I/O trend diagram, or in the structured table representation.

The following table summarizes the steps for creating a reference cycle :

E

Action	Keys
Place the cursor in the position at which you wish to start the reference cycle. For example, at the "START CYCLE" instant	<←>, <→>, <Ctrl+←>, <Ctrl+→> in I/O representation mode <↑>, <↓>, <PgUp> or <PgDn> in FA mode
Memorize this position. Place the cursor in the position at which you wish to end the reference record. For example, at the "END CYCLE" instant	<F6> (memo cursor) <←>, <→>, <Ctrl+←>, <Ctrl+→> in I/O representation mode <↑>, <↓>, <PgUp> or <PgDn> in FA mode
Save the reference record for the 2 positions indicated.	<Alt+F1>
Confirm the reference cycle record.	<O> to confirm <N> to cancel creation
The reference cycle will then be resynchronized automatically to the same start cycle conditions in the record.	Automatic

Once the operation has been confirmed, the reference cycle is automatically saved. It can be recreated at any time.

A reference cycle record includes :

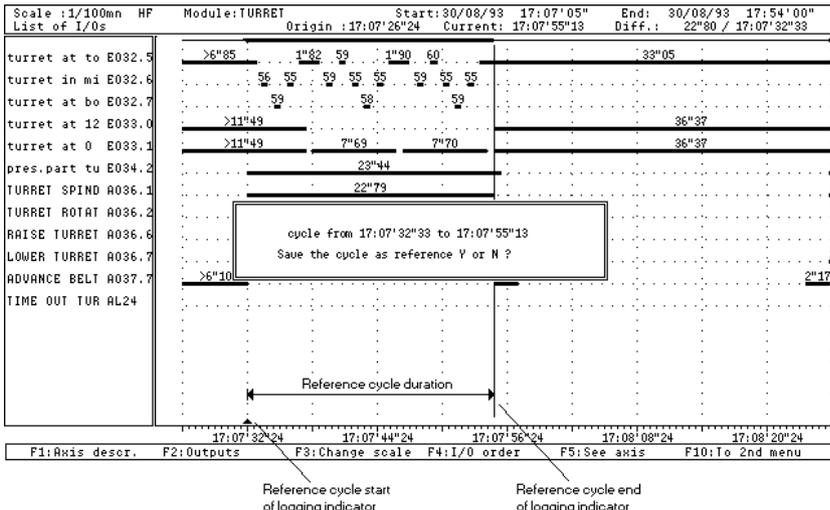
- the evolutions of all the I/O (within the limit of 1500 evolutions)
- data relating to the instant at which the record starts
- the first change in state of the I/O (from all the axis I/O)
- the logic state of the I/O shown on screen when this changes state

This information can be displayed by pressing <Alt+F7>.

It is possible to create as many reference cycles as there are elements in the axis diagram. A single reference cycle can be created for each element.

Keys affecting the reference cycles :

- <Alt+F2> deactivates or activates the reference cycle display (if it exists)
- <Alt+F3> displays the durations for the reference cycle
- <Alt+F4> moves the reference cycle over time
- <Alt+F5> performs an automatic search for deviations
- <Alt+F6> changes the reference point of the reference cycle over time
- <Alt+F7> displays the synchronization information for the reference cycle
- <+> (plus) resynchronizes the reference cycle



5.7-3 Activating the Comparison

The reference trend diagram is displayed in blue with a different marker to distinguish it in trend diagram mode, or on the right-hand side in structured table mode.

In comparison mode, it is always possible to modify the time base (key <F3>), display the outputs only or the I/O (key <F2>), etc.

Key :

<Alt+F2> superposes the reference and current cycles.

5.7-4 Displaying the Durations for the Evolutions of the Reference Cycle

By default, the durations for the evolutions displayed are those of the current cycle. However, it is possible to display the durations for the evolutions of the reference cycle in trend diagram mode.

Keys :

<Shift+F2> displays the durations for the evolutions of the reference cycle in time or mechanical format.

<Alt+F3> alternates between indications of the duration for the current record with those of the reference record.

5.7-5 Displaying Time Deviations in Structured Table Mode

The time indicated for each evolution of the current record (that on the left-hand side) corresponds to the deviation calculated between the current time and the reference time. Consider the following example :

- **input %I2.5** : active after 4'10"45 on the current record and active after 4'10"35 on the reference record will display a positive time deviation of 10 hundredths of a second (meaning that the record at the time of this evolution, has a delay of 10 hundredths of a second on the reference cycle).
- **input %I2.3** : active after 4'13"05 on the current record and active after 4'13"10 on the reference record will display a negative time deviation of (-5) hundredths of a second (meaning that the record at the time of this evolution, is 5 hundredths of a second ahead of the reference cycle).

Key :

<Alt+F3> displays the time deviations.

5.7-6 Moving the Reference Record

To simplify comparison, it is possible to move the reference record in the diagram. This can be done in three different ways :

- **Each time the cursor is moved**, simply press **<Alt+F4>** and use the **<<->** or **<->** keys in trend diagram mode, **<↑>** or **<↓>** in structured table mode. The reference record then moves in the diagram with the cursor.
- **Between cycles**, press the **<+>** (plus) or **-** (minus) keys to find the next or previous cycle from the cursor position. If the start of the reference record corresponds to the start of a cycle, the reference record is then moved from cycle to cycle in the trend diagram.
- **By shifting the reference record to the current cursor position** by pressing **<Alt+F6>**.

Moving the reference cycle from cycle to cycle using the **<+>** or **-** keys consists of finding in the current record the same conditions as the instant of the start of the reference record. This is the change in state corresponding to the first evolution with the same I/O logic states displayed during the save.

Any evolution of the current record which does not correspond to an evolution of the reference cycle is considered a deviation.

The functional deviations are displayed in red and marked with the symbol "#".

5.7-7 Displaying Reference Cycle Synchronization Information

The following information is displayed :

- the date and time of the record
- data relating to the instant of the start of the record
- the first change in state of the I/O (from all the module I/O)
- the logic state of the I/O shown on screen at the instant of this change in state

Key :

<Shift+F7> displays the reference cycle information.

5.7-8 Finding a Deviation

A search for a deviation can be executed automatically in order to compare the **full record** with the reference cycle previously created. The search starts from the current cursor position and finishes at the end of the record. Each time a deviation is found, the search stops and displays the type of deviation found.

Keys :

- <Shift+F5> launches the automatic search for deviations.
- <Esc> ends the current automatic search for deviations.

- Principle of finding a deviation

Each time a cycle is resynchronized, the list of evolutions of the current cycle is compared to the list of evolutions for the reference cycle.

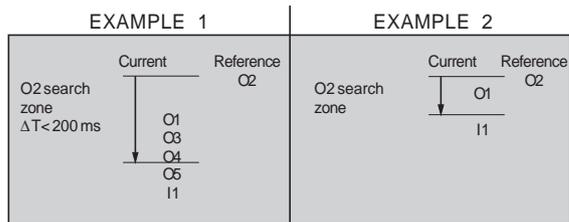
In FA representation, where filtering is also taken into account, the deviation displayed may differ from that in the trend diagram.

- 1st step

Starting from the first change in state of the cycle, the current cycle and the reference cycle are compared evolution by evolution. The first different change in state in the reference cycle is searched for in the current cycle among the following evolutions.

On a change in state of an output

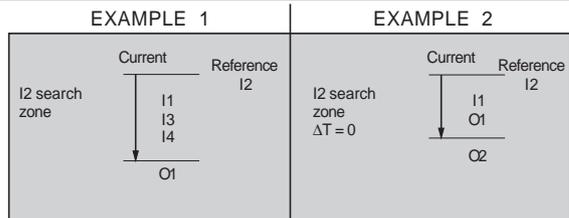
The search stops at the first change in state of an input OR at the first evolution of an output separated by more than 200 ms.



Comparison of cycle evolution by evolution for an output

On an input

The search stops at the first change in state of an output or at the first different time.



Comparison of cycle evolution by evolution for an input

• **2nd step**

If the search for an evolution finishes, it is considered that there is no deviation. The user is then interested in the corresponding change in state in the current cycle which can be searched for in the same way in the reference cycle. Similarly, if an evolution is found, there is no deviation and it goes on to the next different evolution.

For a group of deviations, the search zone starts at the first different evolution.

• **Displaying deviations**

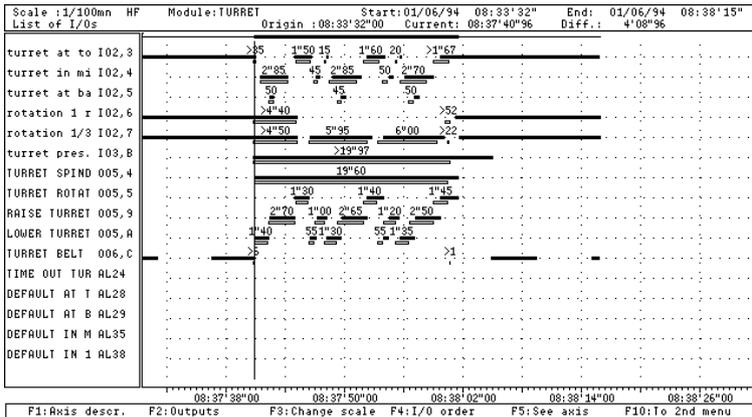
A deviation found in this way is signaled by a message at the bottom of the screen.

"Deviation : %I2.4 - CENTER TOOL POST ==> edge ↑ not expected"

or

"Deviation : %I2.4 - CENTER TOOL POST ==> edge ↓ missing"

This concerns either an unexpected edge in the current cycle, or an edge in the reference cycle which is missing in the current cycle. The deviation is marked in the trend diagram by "DEV" in red or "dev" in blue. In FA representation mode, the line corresponding to the deviation is shown in purple.



5.8 Printing Diagrams

A trend diagram can be printed at any time, whether it is the axis diagram, the movement diagram, the I/O diagram or the FA representation.

2 types of printer can be used : EPSON or compatible printers, and HP LaserJet printers.

The start of the print zone corresponds to the reference point of the record, and the end of the print zone corresponds to the current cursor position.

Printing is from the first element displayed in the list. 32 elements can be printed on an A4 sheet.

To print **part** or **all** of a trend diagram :

Action	Keys
Place the cursor in the position at which printing should start	<←>, <→>, <Ctrl+←>, <Ctrl+→> in axis diagram, movement diagram or I/O diagram mode <↑>, <↓>, <PgUp> or <PgDn> in FA mode
Change the reference point to mark the start of printing	<F4> in the axis diagram <Shift+F4> in movement diagram, the I/O diagram or in FA representation
To print part of a diagram : place the cursor in the position at which printing should finish	<←>, <→>, <Ctrl+←>, <Ctrl+→> in movement, I/O, or axis diagram mode
To print all of a diagram : do not move the cursor (leave it at the reference point of the record)	<↑>, <↓>, <PgUp> or <PgDn> in FA mode
Start printing	<F9>
Select the type of printer	<1> : EPSON printer <2> : HP LaserJet printer <Esc> : cancels printing

Printing can be stopped at any point by pressing <Esc>.

The choice of printer is only required for the first print.

Keys :

<F8> prints the screen.

<F9> prints all or part of the trend diagram.

5.9 Summary of the Keys Used

5.9-1 Description of the Keyboard

The illustration below shows the various keys used when representing trend diagrams. These keys are outlined in gray and are labeled (the layout and labels of the keys may differ depending on the type of keyboard used).



5.9-2 Axis Diagram

Fx type function keys

Functions	Explanation
F1 : Software information	Gives the software version
F2 : Change HMS or HM format	Time (HH:MM'SS" XX) or mechanical (HH:MM'XXX) format
F3 : Change scale	Scale : 1 min (60 s), 1/10 min (6s), 1/100 min (600 ms)
F4 : Change ref. point	Changes the reference point of the diagram
F6 : Memorize cursor	Memorizes the cursor position
F8 : Print screen	Prints the part displayed on screen
F9 : Printout	Prints the diagram (between the memorized and current cursor positions)

Keystrokes

Keystrokes	Explanation
Home, End	: First/last module Accesses the first or last module of the diagram
Ctrl+Home	: Start of record Accesses the first module of the diagram
Ctrl+End	: End of record Accesses the last module of the diagram
PgUp, PgDn	: Scroll up/down Moves the window 10 lines up or down
← →	: Prev./next instant Moves 1/5 of a time unit to the previous or next instant
Ctrl ← →	: Rapid scroll Moves quickly (10 time units) to the previous or next instant
↑ ↓	: Prev./next module Previous or next module
Return	: Select module or axis Displays the corresponding I/O diagram for the selected module.
Tab	: Display menu window Displays the menus in a window
Esc	: Exit Exits the module diagram

5.9-3 Movement Diagram

Fx type function keys

Functions	Explanation
F3 : Change scale	Scale : 1 min, 1/10 min (6s), 1/100 min (600 ms), 1/1000 min (60 ms)
F4 : Chronolog./numerical order	Displays the movements in numerical or chronological order
F6 : Memo. cursor position	Memorizes the current cursor position
F7 : I/O label	All or part of label
F8 : Print screen	Prints the part displayed on screen
F9 : Printout	Prints the diagram (between the memorized and current cursor positions)

Shift+Fx type function keys

Functions	Explanation
SF1 : FA representation	Accesses FUNCTIONAL ANALYSIS mode
SF2 : Change HMS or HM format	Time (HH:MM'SS" XX) or mechanical (HH:MM'XXX) format
SF3 : Change Input color	Changes the color of the inputs (yellow, white or blue)
SF4 : Change ref. point	Moves the ref. point of the diagram to the current position
SF5 : ZOOM 1 min	Zoom by 1 minute
SF8 : Cont./evolution mvmt	Mode for moving the cursor vertically continuously or by evolution

Alt+Fx type function keys

Functions	Explanation
AF1 : Save reference	Accesses FUNCTIONAL ANALYSIS mode
AF2 : Comparison active	Time (HH:MM'SS»XX) or mechanical (HH:MM'XXX) format
AF3 : Reference cycle duration	Changes the color of the inputs (yellow, white or blue)
AF4 : Change ref. point	Moves the ref. point of the diagram to the current position
AF5 : Activate resynchro.	Zoom by 1 minute
AF6 : Find deviation	Mode for moving the cursor vertically continuously or by evolution
AF7 : Change reference ref. point	Modifies the reference point of the reference cycle
AF8 : Info reference cycle	Displays information on the reference cycle

Keystrokes

Keystrokes	Explanation
Home, End : First/last I/O	Accesses the first or last movement of the trend diagram
Ctrl+Home : Start of record	Accesses the first movement of the trend diagram
Ctrl+End : End of record	Accesses the last movement of the trend diagram
PgUp,PgDn : Scroll up/down	Moves the window 10 lines up or down
-, + : Prev./next cycle	Moves to the previous or next cycle (in comparison only)
← → : Prev./next instant	Moves 1/5 of a time unit to the previous or next instant
Ctrl ← → : Rapid scroll	Moves quickly (10 time units) to the previous or next instant
↑ ↓ : Prev/next I/O	Previous or next movement
Shift, Alt : Other menus	Displays the menus for the Shift+Fx or Alt+Fx functions
Tab : Display menu window	Displays the menus in a window
Esc : Exit	Exits the movement diagram

5.9-4 I/O Diagram

Fx type function keys

Functions	Explanation
F2 : I/O or Outputs	Displays the I/O or only the outputs
F3 : Change scale	Scale : 1 min, 1/10 min (6s), 1/100 min (600 ms), 1/1000 min (60 ms)
F4 : Chronolog./numerical order	Displays the I/O in numerical or chronological order
F6 : Memo. cursor position	Memorizes the current cursor position
F7 : I/O label	All or part of the label + I/O address
F8 : Print screen	Prints the part displayed on screen
F9 : Printout	Prints the diagram (between the memorized and the current cursor positions)

Shift+Fx type function keys

Functions	Explanation
SF1 : FA representation	Accesses FUNCTIONAL ANALYSIS mode
SF2 : Change HMS or HM format	Time (HH:MM'SS" XX) or mechanical (HH:MM'XXX) format
SF3 : Change Input color	Changes the color of the Inputs (yellow, white or blue)
SF4 : Change ref. point	Moves the ref. point of the diagram to the current position
SF5 : ZOOM 1 min	Zoom by 1 minute
SF8 : Cont./evolution mvmt	Mode for moving the cursor vertically continuously or by evolution

Alt+Fx type function keys

Functions	Explanation
AF1 : Save reference	Accesses FUNCTIONAL ANALYSIS mode
AF2 : Comparison active	Time (HH:MM'SS»XX) or mechanical (HH:MM'XXX) format
AF3 : Reference cycle duration	Changes the color of the Inputs (yellow, white or blue)
AF4 : Change ref. point	Moves the ref. point of the diagram to the current position
AF5 : Activate resynchro.	Zoom by 1 minute
AF6 : Find deviation	Mode for moving the cursor vertically continuously or by evolution
AF7 : Change reference ref. point	Modifies the reference point of the reference cycle
AF8 : Info reference cycle	Displays information on the reference cycle

Keystrokes

Keystrokes	Explanation
Home, End : First/last I/O.	Accesses the first or last I/O of the trend diagram
Ctrl+Home : Start of record	Accesses the first I/O of the trend diagram
Ctrl+End : End of record	Accesses the last I/O of the trend diagram
PgUp,PgDn : Scroll up/down	Moves the window 10 lines up or down
-, + : Prev./next cycle	Moves to the previous or next cycle (in comparison only)
← → : Prev./next instant	Moves 1/5 of a time unit to the previous or next instant
Ctrl ← → : Rapid scroll	Moves quickly (10 time units) to the prev. or next instant
↑ ↓ : Prev./next I/O	Previous or next I/O
Shift, Alt : Other menus	Displays the menus for the Shift+Fx or Alt+Fx functions
Tab : Display menu window	Displays the menus in a window
Esc : Exit	Exits the I/O diagram

5.9-5 Functional Analysis Diagram

Fx type function keys

Functions	Explanation
F2 : I/O or Outputs	Displays the I/O or only the outputs
F3 : with or without filtering	Activates or deactivates the filter
F4 : Filter/add selected I/O	Filter or add selected I/O
F6 : Memo. cursor position	Memorizes the current cursor position
F7 : I/O label or comment	All or part of the label + I/O denomination
F8 : Print screen	Prints the part displayed on screen
F9 : Printout	Prints the diagram (between the memorized and the current cursor positions)

Shift+Fx type function keys

Functions	Explanation
SF2 : Change HMS or HM format	Time (HH:MM'SS" XX) or mechanical (HH:MM'XXX) format
SF4 : Change ref. point	Moves the ref. point of the diagram to the current position

Alt+Fx type function keys

Functions	Explanation
AF1 : Save reference	Accesses FUNCTIONAL ANALYSIS mode
AF2 : Comparison active	Time (HH:MM'SS" XX) or mechanical (HH:MM'XXX) format
AF3 : Reference cycle duration	Changes the color of the Inputs (yellow, white or blue)
AF4 : Change ref. point	Moves the ref. point of the diagram to the current position
AF5 : Activate resynchro.	Zoom by 1 minute
AF6 : Find deviation	Mode for moving the cursor vertically continuously or by evolution
AF7 : Change reference ref. point	Modifies the reference point of the reference cycle
AF8 : Reference cycle info	Displays information on the reference cycle

Keystrokes

Keystrokes	Explanation
Home, End : First/last I/O.	Accesses the first or last evolution
PgUp,PgDn : Scroll up/down	Moves the window 10 lines up or down
-, + : Prev./next cycle	Moves to the previous or next cycle (in comparison only)
↑ ↓ : Prev./next evol.	Previous or next evolution
Shift, Alt : Other menus	Displays the menus for the Shift+Fx or Alt+Fx functions
Tab : Display menu window	Displays the menus in a window
Esc : Exit	Exits FA representation

Section	Page
1 Validating the PLC Program	1/1
1.1 Principle of PLC/SIMTSX Exchanges	1/1
1.1-1 Introduction for Micro/Premium	1/1
1.1-2 Basic Principle for Micro/Premium	1/1
1.1-3 Introduction for Quantum	1/2
1.1-4 Basic Principle for Quantum	1/2
1.2 Setup	1/3
1.2-1 Hardware Setup	1/3
1.2-2 Simulating I/O Modules on Micro/Premium	1/3
1.3 Simulation Modules	1/4
1.3-1 Operating the Simulation Module on Micro/Premium	1/4
1.3-2 Limits in the Use of Simulation Modules on Micro/Premium	1/5
1.3-3 Operating the Simulation EFB on Quantum	1/6
1.4 Communication Parameters	1/7
1.5 Displaying Discrete I/O	1/7
1.6 Available Commands	1/10

Section

Page

F

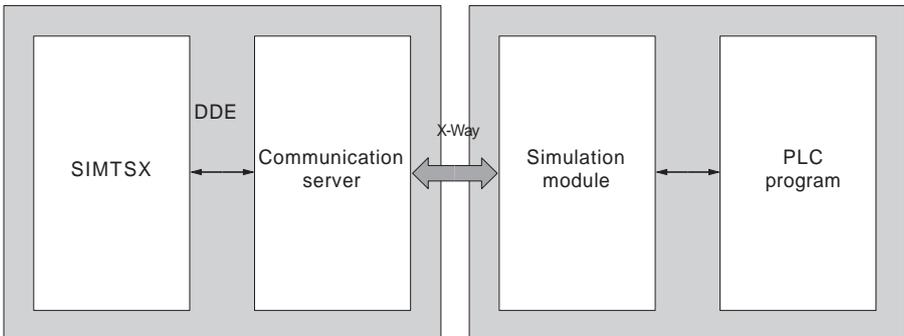
1.1 Principle of PLC/SIMTSX Exchanges

1.1-1 Introduction for Micro/Premium

In online mode, SIMTSX exchanges I/O with the PLC to simulate the behavior of industrial devices. These exchanges are performed using two software modules : the communication server installed on the simulation PC with SIMTSX, and the simulation module installed on the PLC with the PL7 program. SIMTSX communicates with the communication server which exchanges I/O with the simulation module. The simulation module is responsible for providing the PLC with the I/O evolutions. The diagram below illustrates the I/O exchanges between SIMTSX and the PLC.

PC + Windows 95 or NT4.0

Micro/Premium



1.1-2 Basic Principle for Micro/Premium

When SIMTSX advances the simulation model, the evolutions of the inputs are communicated to the server. The server then communicates these to the simulation module which then copies them to the PLC. The evolutions of the outputs resulting from the reaction of the PLC program are then returned to SIMTSX by the simulation module and the server.

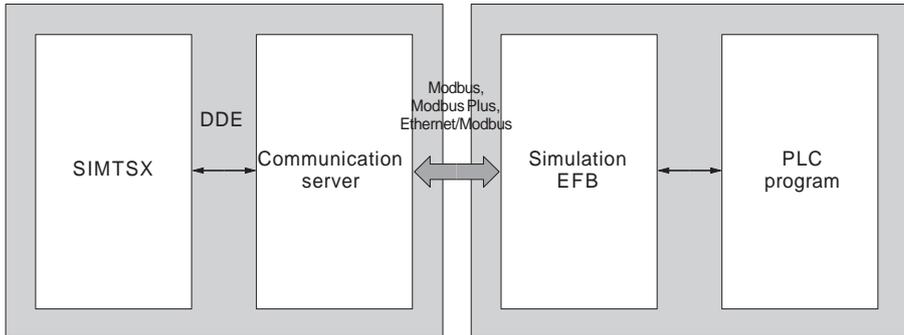
To provide the PLC with the changes in state of the inputs, the module has commands for writing directly to the PLC memory. The evolutions of the outputs are obtained via a direct read command from the PLC output image memory. By comparing this to the former contents memorized previously, the module deduces the outputs whose state has changed. Thus the module keeps its own image memory of the I/O.

1.1-3 Introduction for Quantum

In online mode, SIMTSX exchanges I/O with the PLC to simulate the behavior of devices. These exchanges are performed using two software modules : the communication server installed on the simulation PC with SIMTSX and the simulation EFB integrated in the CONCEPT program. SIMTSX communicates with the communication server which exchanges I/O with the simulation EFB. The simulation EFB is responsible for providing the PLC with the I/O exchanges.

PC + Windows 95 or NT4.0

Quantum



1.1-4 Basic Principle for Quantum

When SIMTSX advances the simulation model, the evolutions of the inputs are communicated to the server. The server then communicates these to the EFB (via the 4x registers) which then copies them to the PLC. The evolutions of the outputs resulting from the reaction of the PLC program are then returned to SIMTSX by the server.

To provide the PLC with the changes in state of the inputs, the EFB has commands for writing directly to the PLC memory. The evolutions of the outputs are obtained via a direct read command from the PLC output image memory by the communication server. By comparing this to the former contents memorized previously, the communication server deduces the outputs whose state has changed. Thus the communication server keeps its own image memory of the I/O.

1.2 Setup

1.2-1 Hardware Setup

Micro/Premium connection

Simulation on Micro or Premium requires PLC/SIMTSX communication which relies on X-Way services. All X-Way services are compatible (terminal port, UNI-TELWAY, Fipway, Ethway), as long as the corresponding physical supports are used (CPU port, PLC module, PC card, etc).

Quantum connection

Simulation on Quantum requires PLC/SIMTSX communication be set up using Modbus, Modbus plus or Ethernet Modbus. All these services are available as long as the corresponding physical supports are used (CPU port, PLC module, PC card, etc).

Simulation in online mode can also be performed on Quantum entirely on PC by connecting to the CONCEPT IEC/SIMULATOR.

F

1.2-2 Simulating I/O Modules on Micro/Premium

During SIMTSX simulation, the user can select various in-rack and remote I/O modules, and then define which modules are to be simulated by SIMTSX. The various configuration possibilities and their checks are detailed below.

- I/O modules simulated by SIMTSX : this type of configuration is the most commonly used and enables SIMTSX to simulate the I/O of these modules. SIMTSX checks the presence of these modules in the PLC configuration and blocks the startup of the simulation if the modules do not exist or differ from those declared in SIMTSX. The modules may or may not be physically present.
- I/O modules not simulated by SIMTSX and declared as such in SIMTSX : this is especially used with output modules as it may be necessary, during installation on site, to wire certain modules. This is used to retrieve in SIMTSX the state of their outputs in order to advance the simulation model. SIMTSX checks the presence of these modules in the PLC configuration and blocks the startup of the simulation if the modules do not exist. It displays a non-blocking message for the simulation if the unsimulated modules are present in the PLC configuration but are not physically present (for modules on Bus X only).
- I/O modules not declared in SIMTSX but present in the PLC configuration : this is used to leave any physically wired I/O modules which are not used in the evolution of the simulation model. SIMTSX displays a non-blocking message for the simulation if the modules are not physically present (for modules on Bus X only).

Caution

There are two possible cases for the simulation of I/O :

- All the I/O modules declared in PL7 are simulated by SIMTSX : in this case, the driver which updates the I/O is inhibited. This leads to improved performance time for tasks used.
- In all other cases, the driver which updates the I/O is not inhibited. This means that the simulated outputs are updated if the corresponding module is physically wired.

Note on "mixed : online + local chart" simulation

It is possible to perform a simulation connected to the PLC while keeping a SIMTSX chart for all or part of the SIMTSX application. For example, when the program is not completely finished, or in a multi-PLC application, one part of the application is controlled by the PLC program, and another part by a chart. In this case, the application or the part of the application concerned with SIMTSX chart transitions and actions changes according to this chart, while the other part changes according to the PLC I/O. If there is a conflict of common PLC/SIMTSX chart actions, the SIMTSX chart has priority for the evolutions generated by the outputs (for a PLC output wired on a relay, corresponding to a chart action).

F

1.3 Simulation Modules

1.3-1 Operating the Simulation Module on Micro/Premium

Before starting a simulation, the simulation module must be loaded in the PLC program. To do this, simply add a fictitious simulation module to the configuration on the PL7 side and the simulator side.

Important

If there is at least one remote input or output to be simulated, the simulation module must be positioned after the last Fipio connection point.

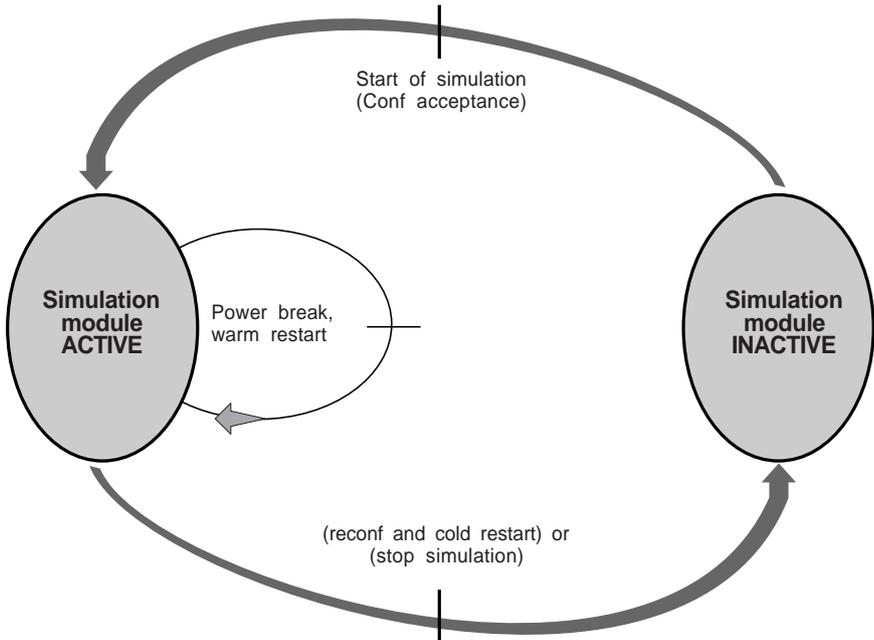
If not, it must be positioned on a slot in the rack.

For all cases, the simulation module must be located on the same slot on the PL7 configuration side and the simulator configuration side.

The Simulation module is available in the "**Add a Module**" dialog box in the **Simulation** family.

When the simulation module is loaded in the PLC program, it is inactive until simulation starts. It is thus possible to leave it in the PLC program if you wish to execute operations other than simulation.

When simulation starts, the simulation module becomes active in order to manage the I/O exchanges between the PLC and SIMTSX. At the end of simulation and until the next simulation, it remains inactive. However, in certain cases, the simulation module can remain active (in the event of a power break during simulation, for example). In this case, the PLC program must be completely reloaded to retrieve its initial conditions and thus deactivate the simulation module. The various operating modes of the simulation module are explained below.



1.3-2 Limits in the Use of Simulation Modules on Micro/Premium

The simulation module added to the PL7 configuration uses the PLC memory. The user must therefore ensure that there is enough space to integrate the module. It should be noted that the Bus X simulation module is half the size of the Fipio simulation module. As a result, the simulation module on Bus X should be added if the application to be simulated only has I/O on Bus X.

1.3-3 Operating the Simulation EFB on Quantum

Before starting a simulation, a certain number of simulation EFBs must be added in the PLC program. To do this, follow the steps below :

- Create a SIMAC_PARAM variable in CONCEPT.
- Fill in this variable with the 4x register corresponding to the first adjacent input field to be simulated (information can be found in the **EFB Link** tab of the simulator I/O configurator).
- Repeat these two steps for all the adjacent input fields to be simulated (all the SIMTSX EFBs in the EFB Link tab of the I/O configurator).
- Create a new SIMTSX section in FBD language.
- Put the SIMTSX section first in the execution sequence for the sections.
- Open the SIMTSX section.
- Insert a SIMTSX EFB available from the EFB library.
- Fill in the top field with the first SIMAC_PARAM variable created.
- Fill in the lower field with the direct value 1 or 3 depending on whether the corresponding adjacent input field processes discrete or analog inputs (information can be found in the EFB Link tab of the I/O configurator).
- Repeat the procedure for inserting EFBs for all the adjacent input fields to be simulated.

Each EFB is then responsible for copying the contents of the 4x working registers to the corresponding discrete or analog input fields at the start of the cycle.

1.4 Communication Parameters

Two parameters are required to establish communication between SIMTSX and the PLC :

- the address of the PLC which communicates with the simulator
- the type of driver used for communication

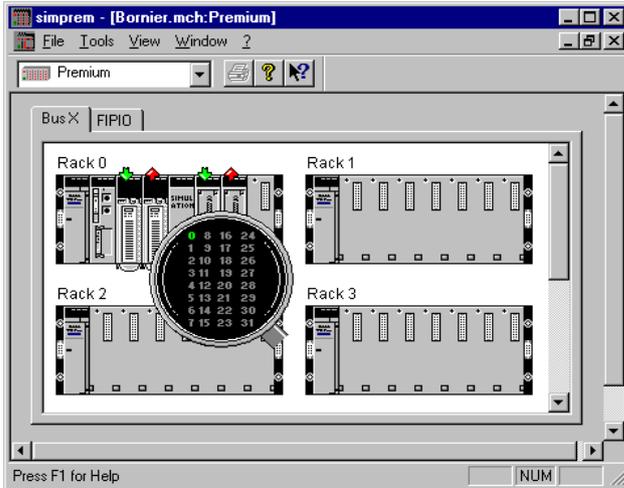
These two parameters should be entered in SIMTSX at the start of simulation.

During simulation, one or more views are assigned representing the PLC(s) running a simulation. These views offer various functions.

1.5 Displaying Discrete I/O

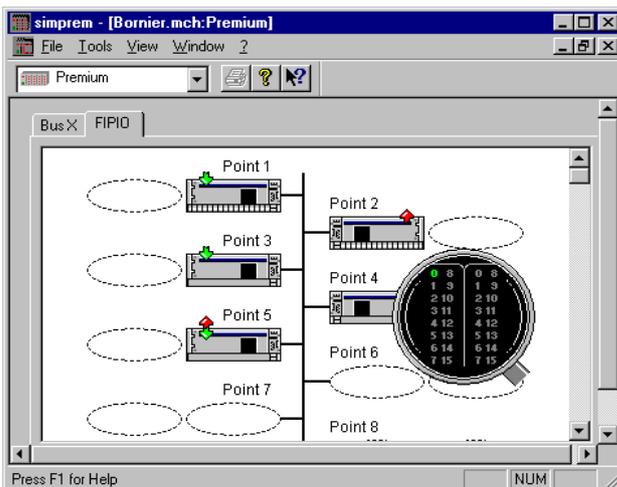
When the mouse is positioned on the display for a power supply unit/CPU on Micro or on a discrete I/O module on Premium or Quantum, after a few seconds a magnifying glass appears which can be used to display the state of the discrete channels.

Bus X magnifying glass

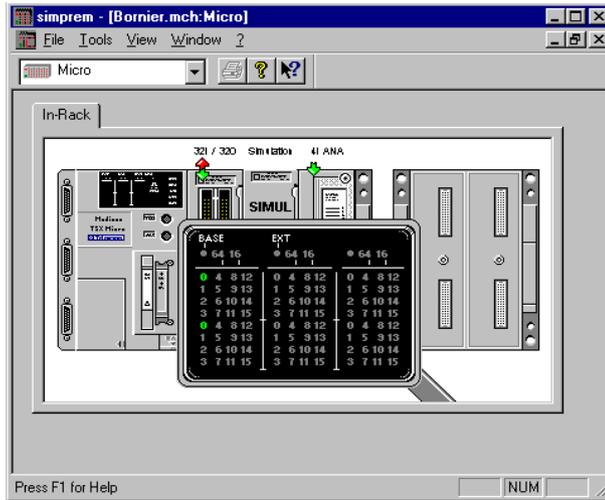


To display inputs or outputs 32 to 63 in 64-channel modules, simply activate the "+32" command in the module shortcut menu.

Fipio magnifying glass



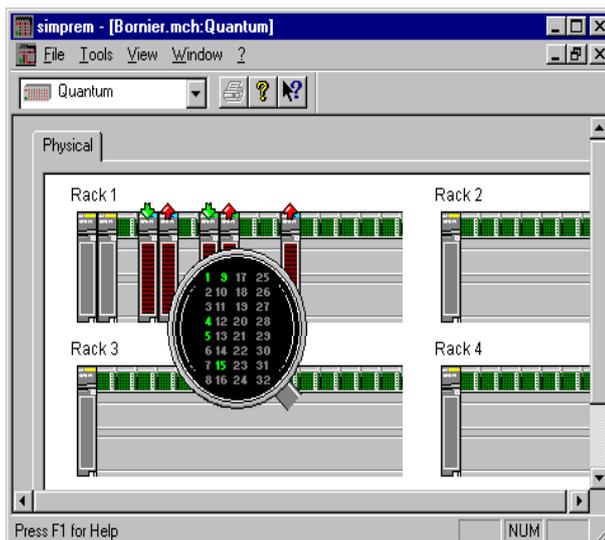
Micro magnifying glass



To display extension channels, simply deactivate the "Base" command in the power supply unit/CPU shortcut menu.

To display channels 16 to 31 in 32-channel modules, simply activate the "+16" command in the power supply unit/CPU shortcut menu.

Quantum magnifying glass



1.6 Available Commands

Tools menu

The **Tools** menu comprises the following commands :

- **Historic** : Displays the historic of the I/O.
- **Curves** : Accesses a window for displaying the analog I/O in the form of curves.

Historic command

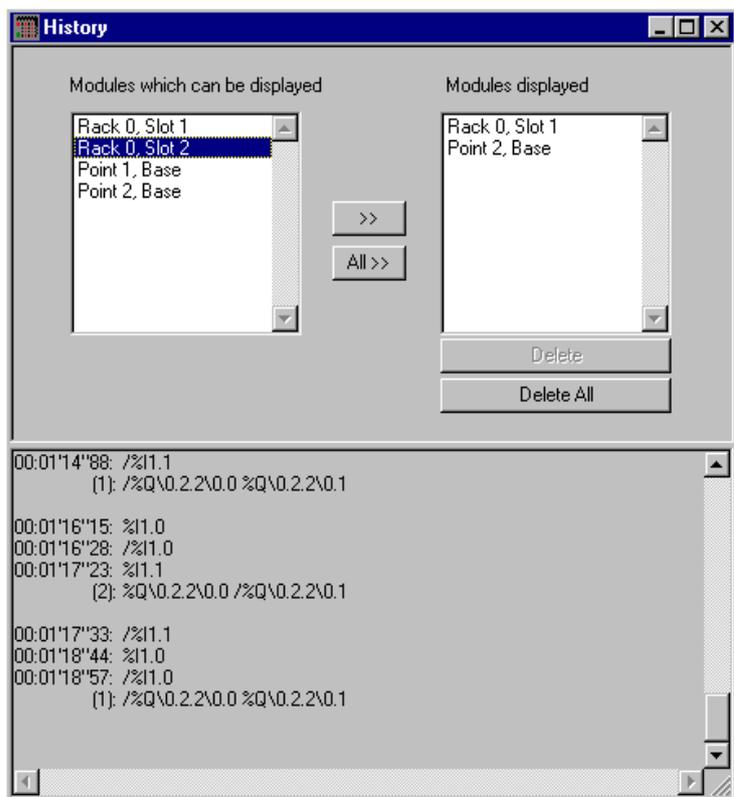
The Historic command is used to display a monitoring window which scrolls through the successive evolutions of the discrete I/O during simulation. These changes are timed in two different ways. For the inputs sent by SIMTSX, the time from the start of simulation is displayed before each series. For the reception of output evolutions, for Micro and Premium the number of PLC scans required to make these outputs appear from the last input transmission is displayed. For Micro and Premium, the **(a)** sign in front of some outputs means that these outputs appear in the PLC after an output feedback but before the input evolutions are sent by the simulator.

The I/O are displayed via the **Edit** command in the **Utilities** menu ; they belong to the modules configured in the upper part. They are shown by their address or by their mnemonic using the **Mnemonics** command in the **Utilities** menu.

The upper part of the window is used to configure the historic. The list on the left shows the modules which can be displayed and the list on the right shows the modules displayed. The list on the right shows the I/O modules displayed in the **Historic** window of the server when the **Edit** command is active. By default, there are no I/O modules displayed.

There are various ways of adding or removing modules, as follows :

- Use the left mouse button to select an element then the >> button or Delete.
- Double-click on an element to add or remove it.
- SHIFT + left mouse button to select several adjacent modules then >> button or Delete.
- CTRL + left mouse button to select several non-adjacent modules then >> button or Delete.



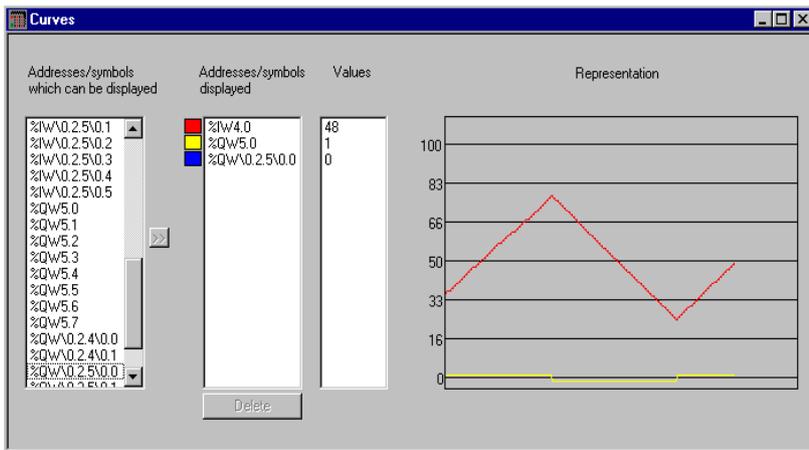
Curves command

This command is used to open a window which displays analog I/O in the form of curves. The list on the left represents the I/O which can be displayed and the list on the right the I/O displayed. Next to each analog channel, a color pad can be used to identify the corresponding channel on the curve. The color can be modified by double-clicking on the pad. The scale can be modified using the **Scale** command in the **Utilities** menu, and the channels can be displayed in the form of symbols using the **Symbols** command in the **Utilities** menu.

It is possible to change from decimal display to hexadecimal display using the **Hex** command in the **Utilities** menu.

The time scale for tracing curves is set by the simulator. During one simulation cycle (transmission of inputs, reception of outputs), two evolutions appear on the curve.

F



The Utilities menu

The **Utilities** menu is displayed when the Historic window is active or when the Curves window is active.

The **Utilities** menu contains the commands which relate to the active tool.

The following commands are available with the **History** tool :

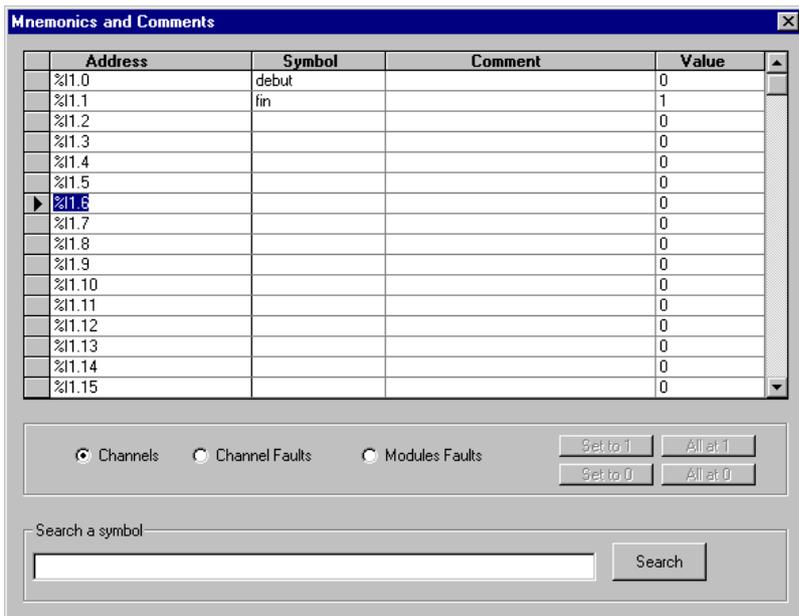
- **Edit** : Displays exchanges of the selected I/O between SIMTSX and the PLC.
- **Symbols** : Used to select the I/O display mode.

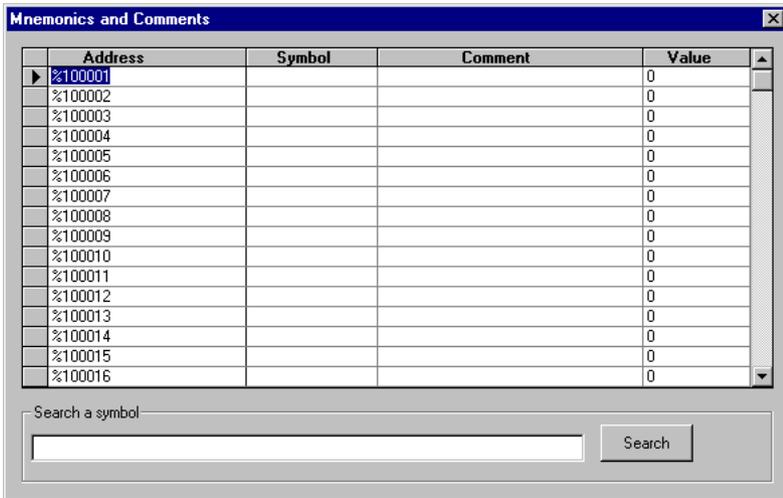
The following commands are available with the **Curves** tool :

- **Scale** : Used to select the scale on the ordinate axis.
- **Symbols** : Used to select the I/O display mode.
- **Hex.** : Used to display values in decimal or hexadecimal.

Displaying I/O values

Double-clicking on an I/O module opens a grid representing the list of I/O with their symbols, comments and values. It can be used to determine the state of a discrete or analog variable at any given time. It is possible to search for occurrences of a character string using the editor and the **Find** button.



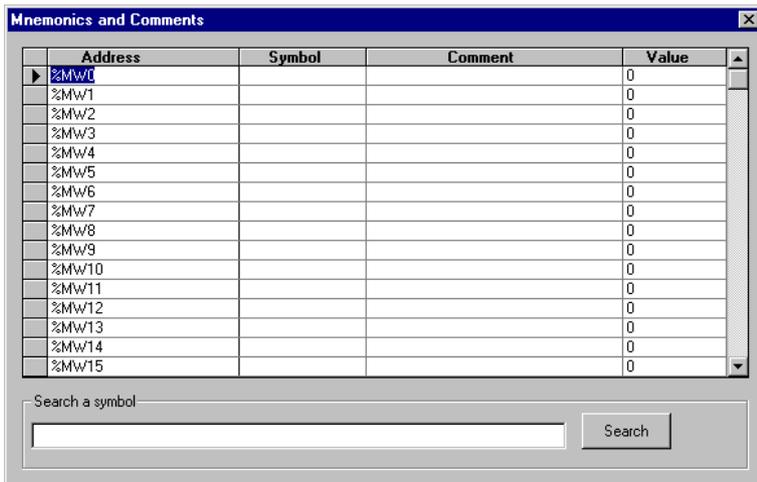


Address	Symbol	Comment	Value
%I00001			0
%I00002			0
%I00003			0
%I00004			0
%I00005			0
%I00006			0
%I00007			0
%I00008			0
%I00009			0
%I00010			0
%I00011			0
%I00012			0
%I00013			0
%I00014			0
%I00015			0
%I00016			0

Search a symbol

Displaying the values of simulated internal words

Double-clicking on the PLC processor opens a grid representing the list of simulated internal words with their values. It can be used to determine the value of an internal word at any given time. It is possible to search for occurrences of a character string using the editor and the **Find** button.



Address	Symbol	Comment	Value
%Mw0			0
%Mw1			0
%Mw2			0
%Mw3			0
%Mw4			0
%Mw5			0
%Mw6			0
%Mw7			0
%Mw8			0
%Mw9			0
%Mw10			0
%Mw11			0
%Mw12			0
%Mw13			0
%Mw14			0
%Mw15			0

Search a symbol

Channel and module faults on Micro and Premium

During a simulation on Micro and Premium PLCs, channel and module faults are managed by SIMTSX. When the simulation starts in online mode, these faults are updated by SIMTSX for simulated modules (set to no fault). During simulation, they can be set to 1 or 0 using the communication server.

Channel or module faults can be accessed using the correspondence grid for I/O, symbols, comments and values. This means that the state of these variables can be set in the relevant PLC and the reactions of the PLC program to these types of faults can be tested. Click on the radio button to display the list of channel faults or module faults. Select the line corresponding to the channel or module which should be faulty. Press the "Set to 1" or "Set to 0" button for the desired action. For module faults, it is possible to set them all to one or to zero using the corresponding buttons.

Address	Symbol	Comment	Value
%I1.0 ERR			0
%I1.1 ERR			0
%I1.2 ERR			0
%I1.3 ERR			0
%I1.4 ERR			0
%I1.5 ERR			0
%I1.6 ERR			0
%I1.7 ERR			0
%I1.8 ERR			0
%I1.9 ERR			0
%I1.10 ERR			0
%I1.11 ERR			0
%I1.12 ERR			0
%I1.13 ERR			0
%I1.14 ERR			0
%I1.15 ERR			0

Channels
 Channel Faults
 Modules Faults

Search a symbol:

Address	Symbol	Comment	Value
%I1.MOD ERR			0
%I2.MOD ERR			0
%I4.MOD ERR			0
%I5.MOD ERR			0

Channels
 Channel Faults
 Modules Faults

Search a symbol:

Caution

To ensure that channel and module faults are updated properly, do not touch the I/O cards which are physically present during simulation in online mode. Touching the cards would cause channel faults and module faults on the simulated cards in the course of a PLC scan (the time when SIMTSX updates them).

F

Section	Page
1 Tool for Transferring from an Application to a Model	1/1
1.1 Introduction	1/1
1.2 Defining Models	1/1
1.2-1 Editing a Generic Model	1/2
1.2-2 Parameter Setting	1/4
1.2-3 Example of a Model	1/5
1.3 Defining a Model using an Application	1/6
1.3-1 Transferring from an Application to a Model	1/6
1.3-2 Reducing the Model	1/7
1.3-3 Setting the Parameters for a Model	1/8
2 Transferring Files	2/1
2.1 Introduction	2/1
2.2 Saving to Floppy Disk	2/2
2.3 Loading from a Floppy Disk	2/3
3 Documentation for SIMTSX Models and Applications	3/1
3.1 Introduction	3/1
3.2 The File Menu	3/6
3.2-1 New Command	3/6
3.2-2 Open Command	3/7
3.2-3 Save Command	3/8
3.2-4 Save As Command	3/8
3.2-5 Print Command	3/9

Section	Page
3.2-6 Print Preview Command	3/10
3.2-7 Page Setup Command	3/11
3.2-8 Last File Opened	3/12
3.2-9 Exit	3/12
<hr/>	
3.3 The Dossier Menu	3/13
3.3-1 Open Application Command	3/13
3.3-2 Open Model Command	3/14
<hr/>	
3.4 The Format Menu	3/15
3.4-1 General Command	3/15
3.4-2 Table of Contents Command	3/18
3.4-3 I/O Configuration Command	3/19
3.4-4 Relays Command	3/20
3.4-5 Axis Command	3/21
3.4-6 Panel Command	3/22
3.4-7 Paragraphs Command	3/23
<hr/>	
3.5 View Menu	3/25
3.5-1 Toolbar Command	3/25
3.5-2 Information Bar Command	3/26
<hr/>	
3.6 The Help Menu	3/27
3.6-1 Contents Command	3/27
3.6-2 Search for Help on... Command	3/27
3.6-3 About SIMTSX Command	3/27
<hr/>	
3.7 Printing Problems and Advice	3/28

1 Tool for Transferring from an Application to a Model

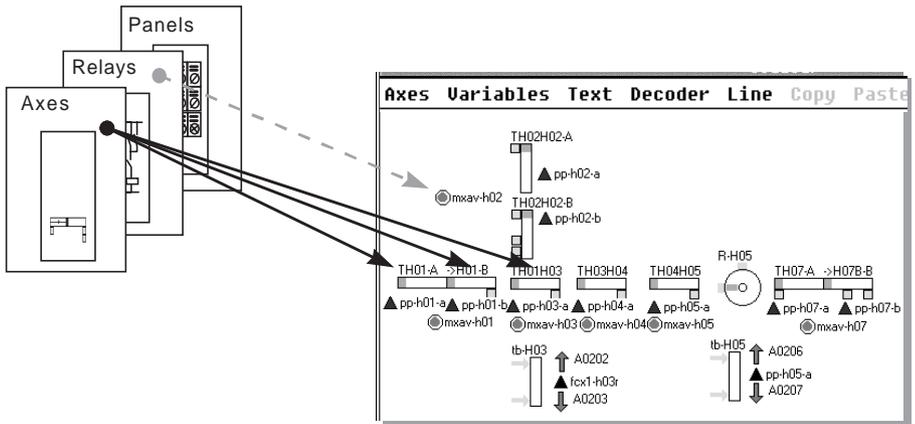
1.1 Introduction

The similarity of applications, and the desire to retain the expertise gained in the description of elements makes it useful to have a library of generic models.

The use of this function involves two aspects :

- first, **defining a model**, similar to describing an application (described in this section)
- second, **transferring from an application to a model** which is saved in a library (see part C, section 1)

1.2 Defining Models



A model is made up of the same components as a machine : axes, relays, operator panels and views, with the exception of I/O. However, a model can contain the definition of input equations, and the designation of these inputs is totally free. A model does not provide the explicit definition of outputs, so these variables are "undefined variables". When a model is incorporated in a machine, these variables can be identified with defined outputs.

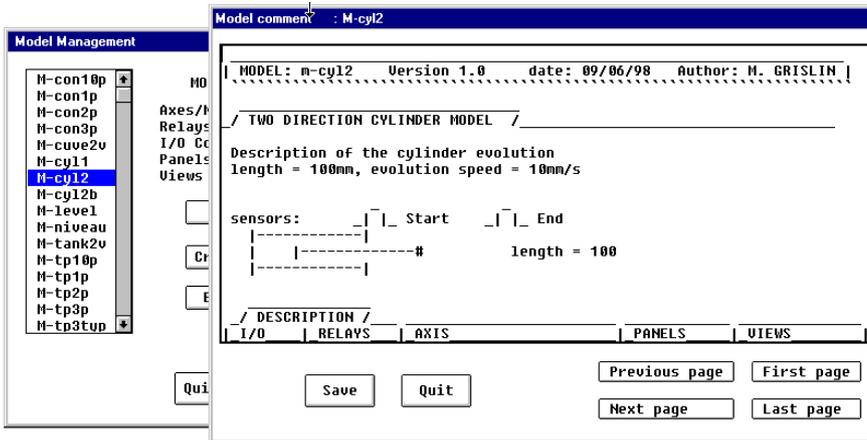
If the nature of the model requires some of the undefined variables to be identified with outputs, it is possible to identify them during incorporation.

The generic model is first described in SIMTSX to enable confirmation during offline simulation. Then, once it is ready, it can be simply transformed into a model which can be stored in the library with any settings and comments (Tools menu > Transfer from Application to Model).

Once the transfer is complete, the model must be loaded from the library (File menu > Model), so that it can be edited.

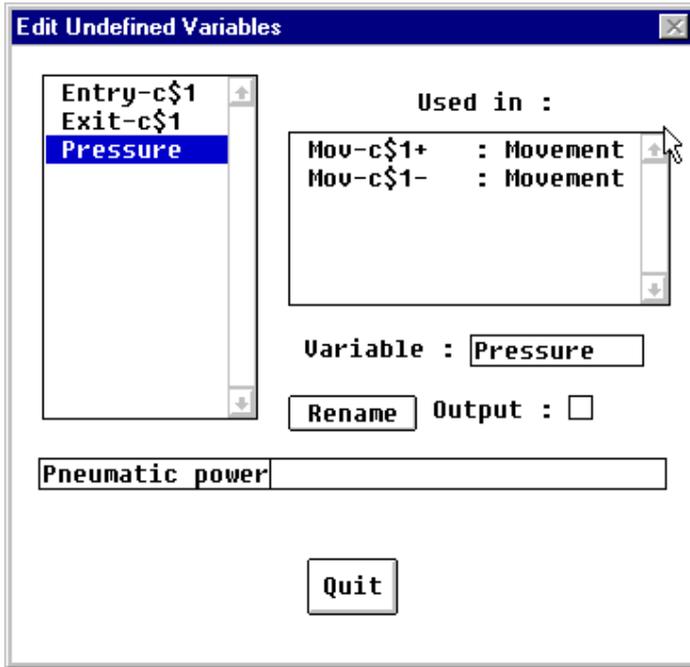
1.2-1 Editing a Generic Model

A generic model is edited using the same interfaces as for the description of an application. The only difference is that it is possible to add comments to description variables to ensure that the model is easy to understand during subsequent use.



A general comment can also be associated with the model, as text which can be accessed using the **Comment** button.

In the same way as the other entities which form the description of a model, comments can be added to the "undefined variables" using the window which accesses these variables.



This window can also be used to introduce a requirement that some of these variables must be linked with outputs. This is done by checking the **Output** box when an undefined variable is selected.

1.2-2 Parameter Setting

Parameters can be set for a model, i.e. the name of the model variables can contain one or more parameters. A parameter is designated by the character "\$", which may be followed by a number between 0 and 9 (up to 11 different parameters can be used in 1 model). Apart from the name of the variables, parameters can be used in the labels of axes, the input mnemonics and comments, the titles and labels for operator panel elements and in the associated comments.

Some undefined variables can be identified with outputs during the incorporation of the model in an application. The name of the undefined variable in the model and its comment become the mnemonic and comment of the application output (unless these have already been defined).

This assignment takes place after the parameters have been substituted. This means that parameters can also be set for the comment associated with an undefined variable in a model.

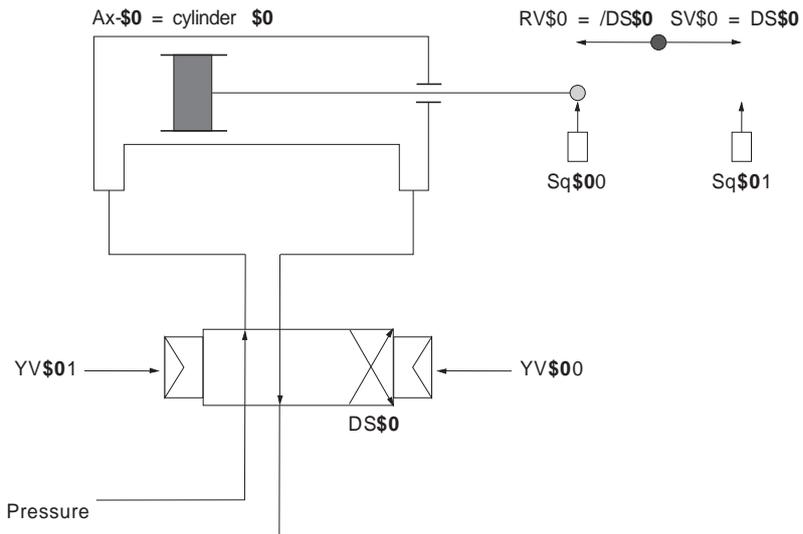
G

The **Parameters** button can be used to access the parameters used in the model and check in which description entities they are used. Comments can also be associated with the parameters.

1.2-3 Example of a Model

Consider the model of a cylinder controlled by a bistable directional control valve.

- The axis describing the course of this cylinder is axis "Ax-1", and its name is "Cylinder \$0".
- The sensors at the start and end limit are Sq\$00 and Sq\$01 respectively. The bistable directional control valve has the name D\$0.
- The movements of the outlet and inlet are SV\$0 and RV\$0 respectively, and their equations are D\$0 and /D\$0. The latch term for bistable D\$0 is YV\$01, and its release term is YV\$00, conditioned by the pressure variable.
- The solenoid valves have the following equations :
 YV\$01 = Outlet-\$0 . Power-supply
 YV\$00 = Inlet-\$0 . Power-supply



The parameter (here \$0) indicates that a link exists between the names of the various variables. This link will be propagated in the application in which the model will be incorporated.

Thus parameters are only set for the significant model variables. In the above example, these are the sensors, the bistable and the solenoid valves. Parameters are not set for the name of the axis itself.

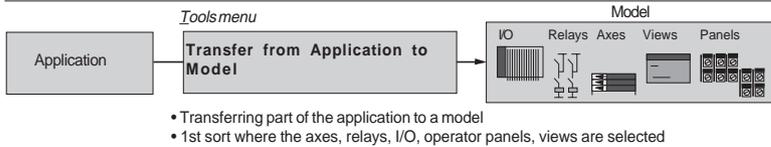
The cylinder model has the undefined variables Outlet-\$0, Inlet-\$0 and Power-supply. The latter represents a condition of the electricity supply for the solenoid valves which control the directional control valve for the cylinder. The two other variables can be identified with outputs in the application in which the model will be incorporated.



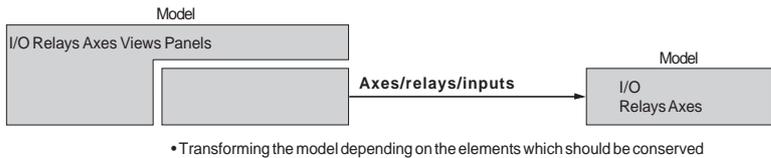
1.3 Defining a Model using an Application

It is often only after a particular application has been defined that it appears useful to store part of this application in the form of a generic model. It is possible to extract part of an application to make it into a reusable model. This procedure involves several steps.

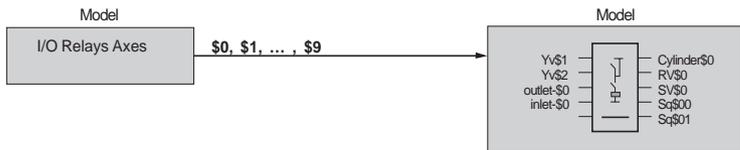
1. TRANSFER



2. REDUCTION



3. PARAMETER SETTING



1.3-1 Transferring from an Application to a Model

This consists of copying the various subassemblies which make up an application as a model : axes, relays, I/O and operator panels. This function can be accessed using the **Tools** menu in the description interface.

1.3-2 Reducing the Model

Unless the application has been described to enable a model, for example using offline simulation, only some of the elements which appear in the description form the model. A large part of the description must therefore be eliminated in order to extract the model. To make this operation easier, "reduction" functions are available in the model description interface, using the axes, relays and I/O menus.

Reducing axes

The **Reduce** button in the main edit window for model axes can be used to select the axes which should be retained in the model. The relevant axes are selected in the left-hand part of the reduction window, and then appear in the right-hand part of this window. The reduction of axes leads to the elimination of axes which are not selected and of variables specifically associated with these axes.

Thus the variables which appear in the up/downstream associations of the selected axes are retained, as are the variables which are not associated with any axis in the description.

By default, this reduction is applied to relays and inputs. This option can be deselected, which means that the reduction is only applied to the variables described in the axes/movements part.

Reducing relays

For the general power supply, the reduction simply consists of completely deleting it.

For other relays, the reduction can be used to retain only those relays which are associated with the model axes, or linked with the operator panels.

The relays used in axis control or assigned by their evolutions are retained: these are the ones which appear in "upstream" and "downstream" associations. The relays in which variables activated by pushbuttons and switches are used, or which are used for the control of indicator lamps and display units, are also retained.

Reducing inputs

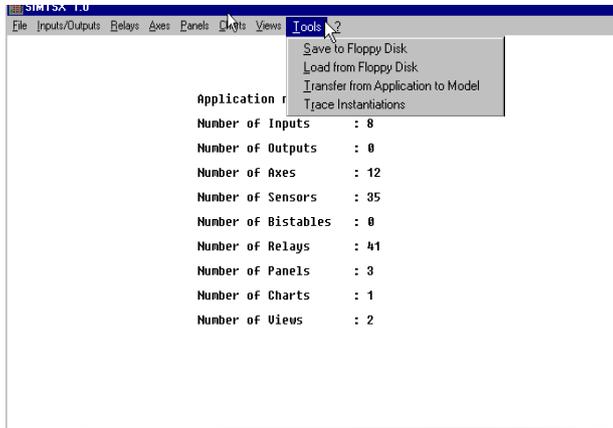
This consists of eliminating the inputs which are not connected to description elements: the inputs in equations which are only used in "General power supply" relays or "undefined" variables which are not used are deleted.

1.3-3 Setting the Parameters for a Model

So that the subassembly can be saved as a reusable model, variables must also be eliminated. Comments can be added to the variables which are retained and any necessary parameters can be set using the **Rename** function.

2.1 Introduction

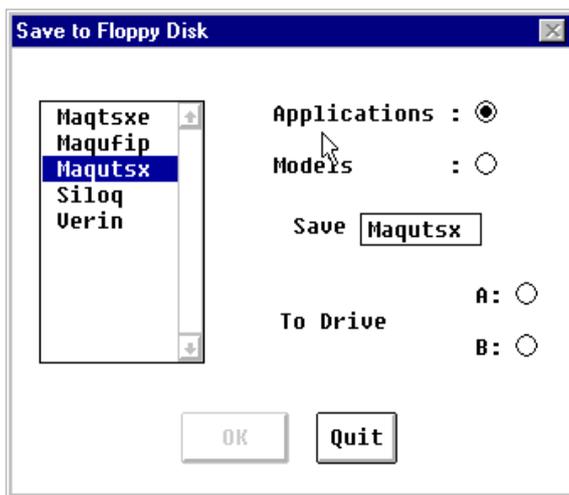
The functions described below are used to transfer applications. To access them, select the **Tools** button in the main **description interface** window.



2.2 Saving to Floppy Disk

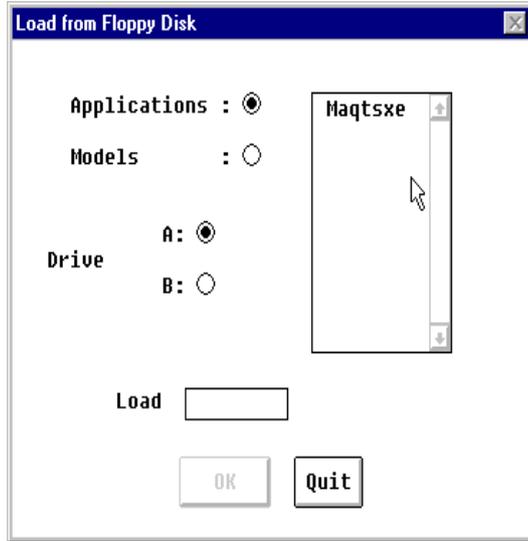
Select the application to be saved from the scroll-down list in the top left-hand part of the window. Its name is displayed in the editor on the right. Select the disk drive to which you wish to copy the application by clicking one of the radio buttons **A:** or **B:**. The **OK** button triggers the command.

A Windows error message appears if there is not enough space on the disk or there is no disk in the selected drive. In this case, solve the problem and retry.



2.3 Loading from a Floppy Disk

Select the disk drive from which you wish to copy the application by clicking one of the radio buttons **A:** or **B:**. Select the application to be loaded from the scroll-down list in the top right-hand part of the window. This scroll-down list contains all the applications stored on the floppy disk. The **OK** button triggers the command.



3.1 Introduction

SIMTSX models and applications can be printed using the **Print** command in the **File** menu of the description interface.

Application or model dossiers which are to be printed can be customized in terms of layout and contents. These customized settings can be stored for later use.

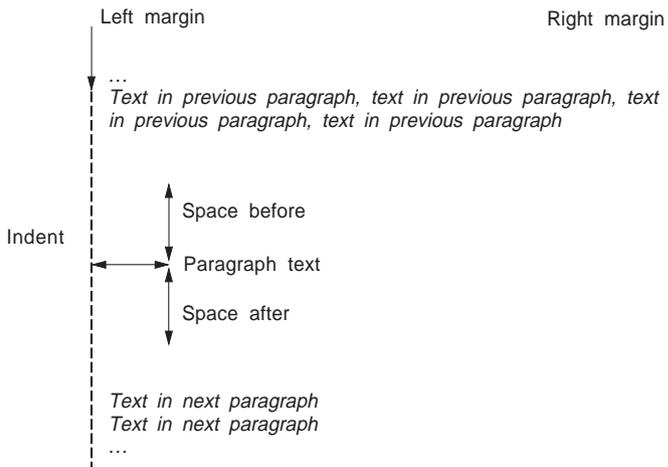
The commands for customizing formats are divided as follows :

- The File menu contains commands for managing customized formats.
- The Format menu contains commands for editing a format.
- The General, Relays, Axis and Panel commands are used to define the contents of the dossier to be printed.
- The Dossier menu is used to open an application dossier.

The printed dossier is structured into paragraphs with which a **style** is associated.

For each style, the following characteristics can be set :

- page break : inserts a page break before the paragraph
- indent : configures the free space to the left of the paragraph
- before : configures the free space above the paragraph
- after : configures the free space below the paragraph
- font : selects the character font to be used for the paragraph



The styles used are as follows :

- Main title
- Identification
- Heading 1
- Heading 2
- Heading 3
- Element
- Trend diagram
- Normal text
- Header
- Footer
- Panel & View
- TOC 1
- TOC 2
- TOC 3

G

Main title

This style defines how to print the title of the print dossier (the first line of the dossier).

Identification

This style defines how to print the text in the fields on the first page. The fields printed are those which have been filled in in the main application window. Empty fields are ignored.

Heading 1, Heading 2, Heading 3

These styles define how to print the various titles contained in the dossier. Heading 1 relates to the various sections of the print dossier (I/O configurations, power supplies, relays, axes, variables, bistables, operator panel variables, operator panels and views). Headings 2 and 3 correspond to the subsections within these sections. The correspondence between text and styles depends on the print options selected by the user.

Element

Use this style to define how to print the names of the description elements (variable, relay, etc).

Trend diagram

This style defines how to print the trend diagrams for the sensors on the axes.

Normal text

This style defines how to print normal text in the print dossier.

Header

This style defines how to print the text in the header on each page of the dossier. The only useful parameter in this style is the character font.

Footer

This style defines how to print the text in the footer on each page of the dossier. The only useful parameter in this style is the character font.

Panel & View

This style defines the space required around the drawings of the operator panels and views. Selection of the character font has no effect on this style.

TOC 1

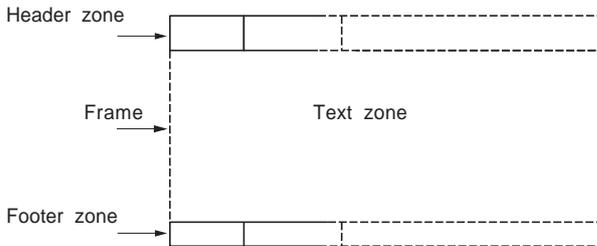
This style defines how to print level 1 headings in the table of contents. These correspond to the Heading 1 style in the print dossier.

TOC 2

This style defines how to print level 2 headings in the table of contents. These correspond to the Heading 2 style in the print dossier.

TOC 3

This style defines how to print level 3 headings in the table of contents. These correspond to the Heading 3 style in the print dossier.

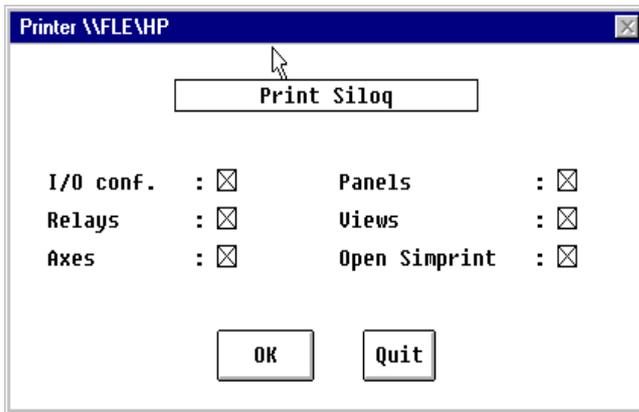


To start printing a dossier, use the **Print** item in the **File** menu of the description interface. If the **Printer <Default Printer>** window does not appear immediately, click on the **OK** button in this window.

The **Printer <Default Printer>** window is used to open the following :

- I/O Conf** Prints the I/O configurations.
- Relays** Prints the relays and the power supplies.
- Axes** Prints the axes, sensors, movements and all other description elements associated with the axes.

- Panels** Prints the operator panels.
- Views** Prints the views.
- Open** If this box is not checked, the dossier will be printed using the default print format on the default printer.
- Simprint**



The contents of the fields in the window are used to identify the dossier. They are printed on the first page, if they have been filled in, and saved in the print formats. Only the Application field is filled in automatically when a **SIMTSX** application or model is opened. The fields used are as follows :

- **Company** : Name of the company which produced the original dossier.
- **Department** : Department within the company responsible for the dossier.
- **Date** : Machine dossier creation date.
- **Author** : Name of the author of the description.
- **Application** : Read-only field. The name of the machine appears automatically.
- **Sector** : Name of the sector of the machine or industrial process.
- **Client code** : Client code number.
- **Classification code** : Archive number, information storage medium.
- **Version** : Number of the current version of the dossier.

At the bottom of the main application window is a comment zone in which the user can add additional information relating to the machine description. This zone is printed after the fields.

The screenshot shows a Windows-style application window titled "Simprint - DEFAUT.FMT". The menu bar includes "File", "Dossier", "Format", "View", and "Help". Below the menu is a toolbar with icons for file operations and help. The main area contains several labeled input fields:

- Machine** : siloq
- Company** : [empty text box]
- Department** : [empty text box]
- Date** : [empty text box]
- Author** : [empty text box]
- Sector** : [empty text box]
- Client Code** : [empty text box]
- Classification Code** : [empty text box]
- Version** : 6.3

Below these fields is a large text area containing the following text: "Format d'impression par défaut. Vous pouvez modifier ce format pour l'adapter à vos propres besoins." At the bottom of the window, there is a status bar with the text "For help, press F1" and a "NUM" button.

3.2 The File Menu

The File menu contains all the commands for managing customized formats. It includes the following commands :

New	Creates a new print format.
Open...	Opens an existing print format file.
Save	Saves any modifications made to the active print format.
Save As...	Saves the active print format under a new name.
Print...	Prints the print dossier.
Print Preview	Displays the dossier on-screen as it will be printed.
Page Setup...	Selects a printer and its connection mode.
Last file opened	Opens one of the four most recently used print formats.
Exit	Exits.

G

3.2-1 New Command

This command creates a new print format using the default options in the application. The default character font is "Courier New". If this font is not installed, SIMTSX uses the system font on your computer.

The **Open** command is used to open an existing print format.

Shortcuts : Toolbar :  Keys : CTRL+N

3.2-2 Open Command

This command opens an existing print format from the **Open** dialog box. The name of the selected print format appears in the title bar.

You can also create a new print format using the **New** command.

Shortcuts : Toolbar : 
 Keys : CTRL+O

- **Open dialog box**

In the dialog box, the following options are used to select the print format to be opened :

File Name

Contains the name of the file to be opened. This list only displays files with an **extension** which matches the selection (see below).

List Files of Type

Selects the type of files you wish to open. The print formats generally have the extension ***.fmt**.

Directories

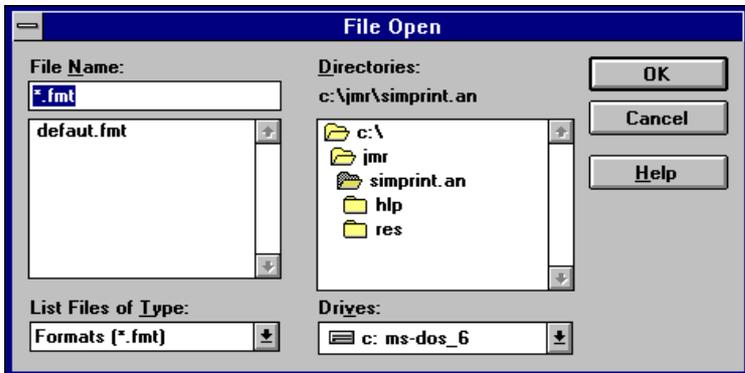
Selects the directory in which the print format to be opened is stored.

Drives

Selects the drive in which the print format to be opened is stored.

Network (1)

Connects to another network node.



(1) This option only appears if the computer is connected to a local area network.

3.2-3 Save Command

This command saves the active print format with its name and current location. When a print format is saved for the first time, the **Save As** dialog box opens which is used to name the print format. This command will be grayed out if no modification has been made to the active print format.

The **Save As** command changes the name and the location of a print format.

Shortcuts : Toolbar : 
 Keys : CTRL+S

3.2-4 Save As Command

This command saves the active print format under a new name and/or to another location. It opens the **Save As** dialog box.

The **Save** command is used to save any modifications made to the active print format.

- **Save As dialog box**

The **Save As** dialog box contains the following options :

File Name

Contains the new name of the archive file for the active print format. A file name includes a maximum of eight characters for the name and three characters for the extension. SIMTSX automatically adds the extension specified in the list of formats to the file name.

Drives

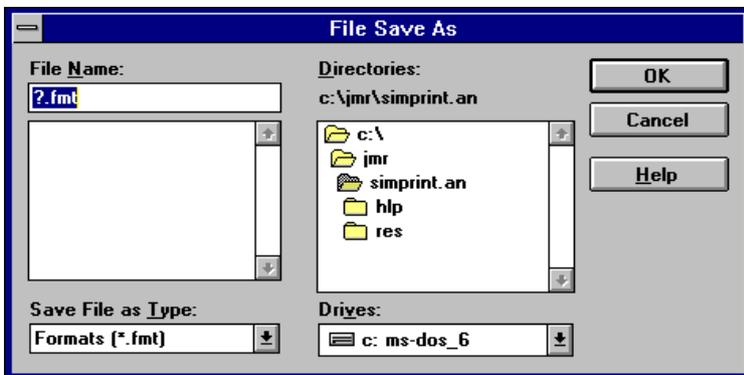
Selects the drive on which the print format is to be archived.

Directories

Selects the directory in which the print format is to be archived.

Network (1)

Connects to another network node.



(1) This option only appears if the computer is connected to a local area network.

3.2-5 Print Command

This command prints a **SIMTSX** dossier from the **Print** dialog box. This dialog box is used to select the pages to be printed, the number of copies and the printer, and to access the other print options.

This command is grayed out when there is no **SIMTSX** application dossier or model available.

Shortcuts : **Toolbar :** 
Keys : **CTRL+P**

- **Print dialog box**

The dialog box contains the following options :

Printer

Shows which printer is selected and the connection mode used. The **Setup** option modifies these parameters.

Setup

Opens the **Print Setup** dialog box which can be accessed directly using the **Page Setup** command (see section 3.2-7 of this part). This option is used to select a printer and its connection mode.

Print Range

Determines the pages to be printed :

- All** Prints the whole dossier.
- Selection** Option not compatible.
- Pages** Prints the pages between the number entered in the **From** field and that entered in the **To** field.

Copies

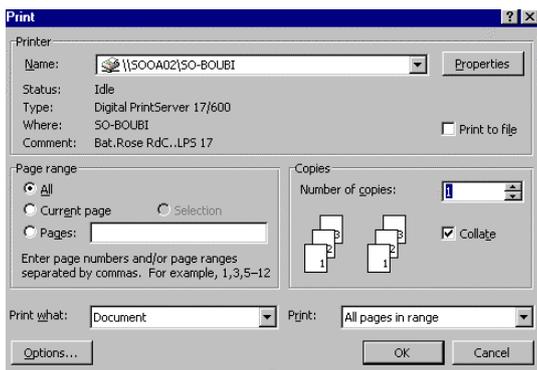
Shows the number of copies to be printed.

Collate Copies

Prints the copies in order instead of printing several copies of the same page at once.

Print Quality

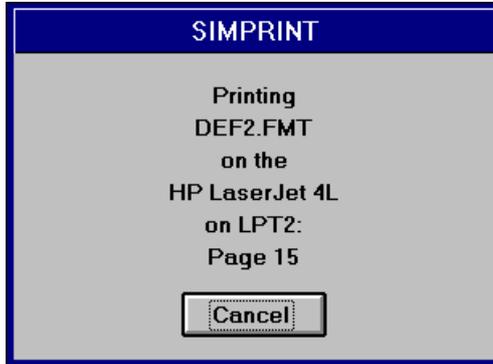
Selects the print quality. Generally, a poorer print quality requires less time.



- **Printing dialog box**

This dialog box is displayed when SIMTSX sends the dossier to the printer. The page number indicates how printing is progressing.

The **Cancel** button is used to abort the print function before it finishes.



G

3.2-6 Print Preview Command

This command displays the dossier on-screen as it will be printed. The main window is replaced by a window in which one or two pages are displayed in the selected print format. The Print Preview toolbar is used to display one or two pages at once, to move up and down within the dossier displayed, to zoom in on a page and to start printing.

This command is grayed out when there is no **SIMTSX** machine dossier or model available.

Shortcuts : Toolbar : 
 Keys : ALT+V

- **Print preview toolbar**

The print preview toolbar offers the following options :

Print

Opens the Print dialog box to start printing the dossier.

Next Page

Displays the next page in the dossier.

Previous Page

Displays the previous page in the dossier.

One Page/Two Pages

Displays one or two pages at once.

Zoom +

Changes the display to give a more detailed view of the page displayed.

Zoom -

Changes the display to give a less detailed view of the page displayed.

Close

Closes the print preview and returns to the main window.

3.2-7 Page Setup Command

This command selects a printer and its connection mode. It opens a **Print Setup** dialog box.

- **Print Setup dialog box**

The following options are used to select a printer and a connection mode :

Printer

Selects the printer to be used. This is either the default printer, or another printer in the list. The Windows Control Panel is used to install new printers.

Orientation

Selects portrait or landscape.

Size

Selects the size of paper on which the dossier will be printed.

Source

Some printers have several paper supply sources. This option selects the required source.

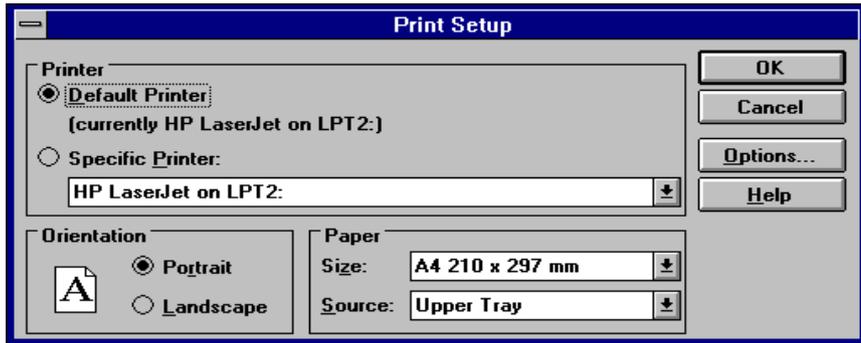
Options

Displays a dialog box used to modify other options. This dialog box is specific to the printer used.



Network(1)

Use this option to connect to another network node.



(1) This option only appears if the computer is connected to a local area network.

G

3.2-8 Last File Opened

The numbers and names of the files listed at the bottom of the file menu are used to open any of the last four print formats most recently used.

3.2-9 Exit

This command is used to exit the document functions. A box is displayed suggesting that any modifications made to the active print format should be saved.

Shortcuts : Mouse : Double-click on the Application Control button
 Keys : ALT+F4

3.3 The Dossier Menu

The Dossier menu contains the commands for opening the **SIMTSX** application dossiers and models. These commands are :

- Open Application Opens a **SIMTSX** application dossier.
- Open Model Opens a **SIMTSX** model.

3.3-1 Open Application Command

This command opens an application dossier from those listed in the **Open Application Dossier** dialog box.

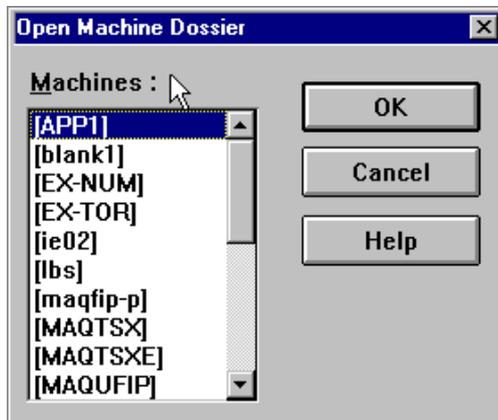
If an application dossier is already open, a marker is positioned before the **Open Application** command in the menu.

Note :

The name of the **SIMTSX** application loaded using this command is displayed in the application field in the main window. If the application has not already been opened from **SIMTSX**, the images representing the operator panels and views of the application may be out of date or missing.
This command is not available if a model is currently being edited.

- **Open Application Dossier dialog box**

The list in the dialog box contains all the dossiers available on your computer. To open a dossier, select it from the list and click on the **OK** button or double-click directly on the element. The **Cancel** button is used to abort this command.



3.3-2 Open Model Command

This command opens a **SIMTSX** model from those listed in the **Open Model** dialog box. If a **SIMTSX** model is already open, a marker is positioned before the **Open Model** command in the menu.

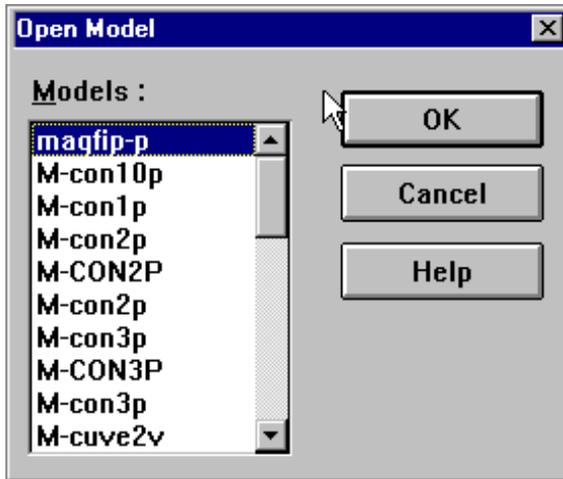
Note :

The name of the model loaded using this command is displayed in the application field in the main window. If the application has not already been opened from **SIMTSX**, the images representing the operator panels and views of the application may be out of date or missing.

This command is not available if a model is currently being edited.

- **Open Model dialog box**

The list in the dialog box contains all the **SIMTSX** models located in the same directory as the one which has been loaded. To open a model, select it from the list and click on the **OK** button or double-click directly on the element. The **Cancel** button is used to abort this command.



3.4 The Format Menu

The Format menu contains the options for printing the print dossiers in the application.

- General Sets the margins, the header and the footer in the print format.
- Table of Contents Configures the printing of the table of contents.
- I/O configuration Configures the printing of the inputs and outputs.
- Relays Configures the printing of the relays and power supplies.
- Axis Configures the printing of the axes.
- Panel Configures the printing of the elements in the operator panel.
- Paragraphs Selects the style for each type of paragraph.

3.4-1 General Command

This command accesses all the configuration parameters. It opens a **General Configuration** dialog box which is used to set the various options suggested.

- **General Configuration dialog box**

The options in the dialog box are used to configure the print option used. You can display immediately the result obtained using the **Print Preview** command in the **File** menu (see section 3.2-6 of this part).

Margins

Each checkbox in this group contains the value of one of the margins in centimeters. The header will be printed above the top margin. The footer will be printed below the bottom margin. The left and right margins correspond to the free space required at each side of the pages in the dossier.

Elements

Each checkbox in this group corresponds to a part in the print dossier. If the box is checked, the corresponding part in the print dossier is printed.

To change the state of a box, simply click on it with the mouse.

The options which can be accessed are as follows :

Instances	Prints the instances of the model.
I/O Config.	Prints the I/O configurations.
Relays/Supplies	Prints the relays and power supplies.
Axes	Prints the axes, sensors, movements and all other description elements associated with the axes.
Op. Panels	Prints the operator panels.
Table of Contents	Prints the table of contents.
Border	Prints a border around each page in the dossier. The border is split into three zones : the header, a central zone where the dossier is printed, and the footer.
Header	Prints the header at the top of each page.
Footer	Prints the footer at the bottom of each page.
View	Prints the views.

Header and Footer

The header and footer of the print dossiers are split into three zones (right, center and left). For each of these zones, you can type some text which will be printed with the dossier.

These text zones can also contain the following commands :

- &P** This sequence of characters is replaced during printing by the page number.
- &M** This sequence of characters is replaced during printing by the name of the application or model.
- &D** This sequence of characters is replaced by the current date.
- &&** This sequence of characters is replaced by the character &.

Note :

The **Table of Contents, I/O Configuration, Relays, Axis** and **Panel** commands have an effect on the checkboxes in the General Configuration screen. For example, if you select a Panel configuration where no element in the operator panel should be printed, the **Op. Panels** checkbox in the **General Configuration dialog box will be deactivated automatically**. Conversely, if you deactivate the checkbox in the **General Configuration** dialog box, the **Panel** item in the menu will no longer be accessible (grayed out).

General Configuration

Margins :

Top : 1.5

Bottom : 1.5

Left : 0.5

Right : 0.5

Elements :

Instances Table of Contents

I/O Config. Border

Relays/Supplies Header

Axes Footer

Op. Panels Views

Header :

Right : Machine &M

Center :

Left : &D

Footer :

Right : Pag. &P

Center :

Left :

OK

Cancel

Help

G

3.4-2 Table of Contents Command

This command directly accesses the parameters relating to printing the table of contents. It opens the "Configure Table of Contents" dialog box.

- **Configure Table of Contents dialog box**

This dialog box contains the following options :

Level of Detail

This dropdown list is used to select the level of detail required for the table of contents.

Level 1 : Table of contents containing the list of Heading 1 styles.

Level 2 : Table of contents containing the list of Heading 1 and Heading 2 styles.

Level 3 : Table of contents containing the list of Heading 1, Heading 2 and Heading 3 styles.

Position

Selects the position of the table of contents in relation to the rest of the dossier.

Start of dossier : Positions the table of contents at the start of the dossier.

End of dossier : Positions the table of contents at the end of the dossier.

Page Numbers on the Right

If this box is checked, the page numbers will be aligned to the right in the table of contents.

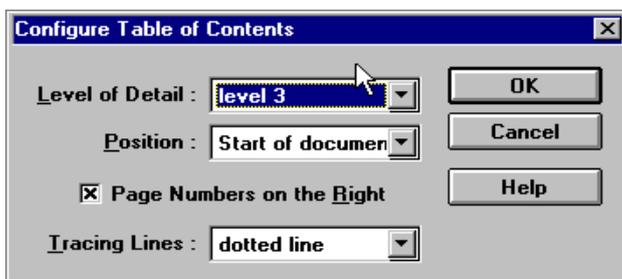
Tracing Lines

Select the type of tracing lines which will be used between the headings and page numbers. This list is only active if the **Page Numbers on the Right** checkbox is selected.

None : No tracing lines in the table of contents.

Underscore : Draws an unbroken line between the heading and the page number.

Dotted Line : Draws a dotted line between the heading and the page number.



3.4-3 I/O Configuration Command

This command directly accesses the parameters relating to printing the I/O configurations. It opens the **I/O Configuration** dialog box.

- **I/O Configuration dialog box**

This dialog box contains the following options :

I/O Cards

When this box is checked, the list of I/O cards is added to the print dossier.

Inputs

Select one of the three radio buttons in the group to define the required level of printing for the inputs.

- No** No inputs are printed.
- List** Only the names of the inputs along with their mnemonic and comment are printed.
- Complete** Includes the input logic equation in the print dossier.

Outputs

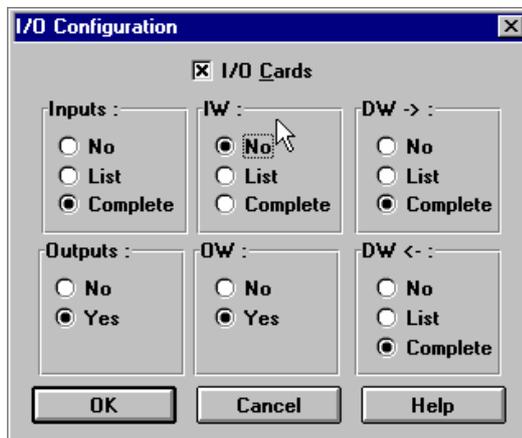
Select one of the two radio buttons in the group to define the required level of printing for the outputs.

- No** No outputs in the print dossier.
- Yes** Adds the name of each output accompanied by its mnemonic and comment to the print dossier.

Note :

If all these options are deactivated (no printing of I/O cards, inputs or outputs), the **I/O Config.** checkbox in the **General Configuration** dialog box is automatically deactivated.

This function can only be accessed if the **I/O Config.** box is checked in the **General Configuration** dialog box (see section 3.4-1 of this part).



3.4-4 Relays Command

This command directly accesses the parameters relating to printing the relays. It opens the **Relay Configuration** dialog box.

- **Relay Configuration dialog box**

This dialog box configures the printing of relays.

Supplies

Select one of the three radio buttons in the group to define the required level of printing for the power supplies.

Relays

Select one of the three radio buttons in the group to define the required level of printing for the relays.

Radio button options

- | | |
|-----------------|---|
| No | No printing for this type of element. |
| List | Only prints the list of names of elements of this type. |
| Complete | Prints all the elements in the print dossier. |

Note :

The model comments are printed next to the name of the elements (list mode and complete mode).

If all the options in the dialog box are deactivated (no printing of power supplies or relays), the **Relays** checkbox in the **General Configuration** dialog box is automatically deactivated.

This function can only be accessed if the **Relays/Supplies** box is checked in the **General Configuration** dialog box (see section 3.4-1 of this part).



3.4-5 Axis Command

This command directly accesses the parameters relating to printing the axes. It opens the **Axis Configuration** dialog box.

- **Axis Configuration dialog box**

This dialog box configures the printing for the axes and the other mechanical description elements in **SIMTSX**. There are several levels of printing for each of the following types of element :

Sensor

Selects the level of printing for the sensors. The sensors are printed after the axis to which they are attached.

Movement

Selects the level of printing for the movements. The movements are printed after the axis to which they are attached.

Variable

Selects the level of printing for the variables.

Bistable

Selects the level of printing for the bistables.

Axis

Selects the level of printing for the axes.

Radio button options

No No printing for this type of element.

List Adds the list of names of elements of this type to the printed dossier.

Complete Adds the full description of elements of this type to the printed dossier.

Trend diagram

When this box is checked, the trend diagram for the sensors of each axis is added to the print dossier.

Note :

The model comments are printed next to the names of the elements.

This function can only be accessed if the **Axes** box is checked in the **General Configuration** dialog box (see section 3.4-1 of this part).



3.4-6 Panel Command

This command directly accesses the parameters relating to printing the operator panels. It opens the **Panel Configuration** dialog box.

- **Panel Configuration dialog box**

This dialog box is used to configure the printing of the operator panels.

Display of Panels

Selecting this checkbox adds the drawings of the operator panels to the print dossier.

Variables

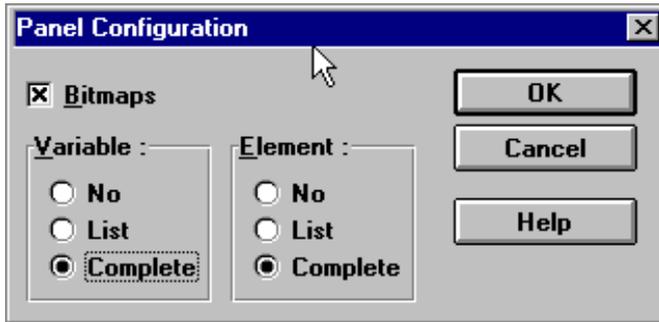
This group of radio buttons is used to select the level of printing for the operator panel variables.

Elements

This group of radio buttons is used to select the level of printing for the operator panel elements.

Radio button options

- No** The mechanical description elements are not printed.
- List** Adds the list of names of elements of this type to the printed dossier.
- Complete** Adds the full description of elements of this type to the printed dossier.



3.4-7 Paragraphs Command

This command directly accesses the parameters relating to printing the paragraphs. It opens the **Paragraph Configuration** dialog box.

- **Paragraph Configuration dialog box**

This dialog box changes the character font and the other options for each type of paragraph.

Dropdown list (Title, etc)

The elements in the dropdown list are the various paragraph styles used. Each style has corresponding lines of text in the print dossier. The styles defined are as follows :

Main Title This style defines how to print the title of the print dossier (the first line of the dossier).

Identification This style defines how to print the field names on the first page. The fields printed are those which have been filled in in the main application window. Empty fields are ignored.

Heading 1

Heading 2

Heading 3

These styles define how to print the various titles contained in the dossier. Heading 1 relates to the various sections of the print dossier (I/O configurations, power supplies, relays, axes, variables, bistables, operator panel variables, operator panels and views). Headings 2 and 3 correspond to the subsections within these sections. The correspondence between text and styles depends on the print options selected by the user.

Element Use this style to define how to print the names of the description elements (variable, relay, etc).

Trend Diagram This style defines how to print the trend diagrams for the sensors on the axes.

Normal text This style defines how to print normal text in the print dossier.

Header This style defines how to print the text in the header on each page of the dossier. The only useful parameter in this style is the character font.

Footer This style defines how to print the text in the footer on each page of the dossier. The only useful parameter in this style is the character font.

Panel & View This style defines the space required around the drawings of the operator panels and views. Selection of the character font has no effect on this style.

Page Break

Inserts a page break before each paragraph using this style if this box is checked. This option has no effect on the Header and Footer styles.

Spacing

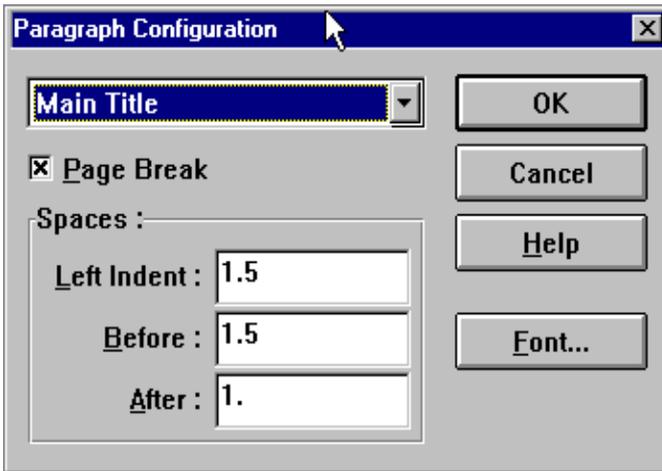
Left Indent Type the size of the indent required in relation to the left margin for all paragraphs using this style. The size must be given in centimeters.

Before Type the size of the space required above the first line of text in each paragraph using this style. The size must be given in centimeters.

After Type the size of the space required below the last line of text in each paragraph using this style. The size must be given in centimeters.

Font

Opens the Windows dialog box for selecting a character font.



3.5 View Menu

The View menu contains the following commands :

- Toolbar Shows or hides the toolbar.
- Information bar Shows or hides the information bar.

3.5-1 Toolbar Command

This command shows or hides the toolbar which contains all the buttons for accessing the most commonly used commands. A marker appears next to the element in the menu when the toolbar is displayed.

The toolbar is displayed across the whole width of the application, just below the menu bar, and provides quick access to the main commands.

-  Opens a new print format.
-  Opens an existing print format.
-  Saves the print format under its current name. If the print format is not already named, the **Save As** dialog box is opened.
-  Displays a print preview of the dossier using the active print format.
-  Prints the dossier using the active print format.
-  Opens the **About SIMTSX** dialog box.
-  Changes the mouse pointer tool to the contextual Help pointer tool.

3.5-2 Information Bar Command

This command shows and hides the information bar. A marker appears next to the command in the menu if the information bar is displayed.

The left-hand part of the information bar explains, when you move through the menus using the arrow keys, the actions associated with the menu elements. In the same way, this zone displays messages relating to the toolbar buttons when you press these buttons without releasing them. If, after having read the message associated with a toolbar button, you do not wish to execute the command, release the mouse button outside the button zone.

The right-hand part informs the user of the status of the following keys :

- CAP** Caps Lock is active.
- NUM** Num Lock is active.
- SCRL** Scroll Lock is active.

3.6 The Help Menu

The Help menu contains commands for accessing the online Help function :

- Contents Opens the online Help at the page containing the index.
- Search for Help on... Opens the general Windows Help on using the Help function.
- About SIMTSX Opens the **About SIMTSX** dialog box.

3.6-1 Contents Command

This command opens the initial Help screen. From this screen, it is possible to access all the online Help for the application.

Wherever you are in the online Help, you can click on the **Index** button to return to this initial screen.

3.6-2 Search for Help on... Command

This command opens the general Windows online Help function.

3.6-3 About SIMTSX Command

This command opens the **About SIMTSX** dialog box containing the version number as well as the copyright ® of the SIMTSX application.

3.7 Printing Problems and Advice

1. If printing is slow. The print quality depends on several parameters. Depending on the character fonts used and the type of printer, the time required can vary within a range from one to ten. This is also true for the size of the file sent to the Print Manager. For example, PostScript printers send basic graphics to draw a line, while other printers need to send bitmaps. It is not possible to define simple rules. The user must configure the print formats according to his hardware and requirements.
2. If the last character (the furthest character to the right) in the boxes on the right-hand side of the header and/or footer is not printed. This is due to the problem in Windows of calculating the size of a character string according to the font used. To resolve this problem, simply add a space after the last character in the string in question.

Section	Page
1 SIMTSX Model Library	1/1
1.1 Cylinder Models	1/2
1.1-1 Single-acting Cylinder Model : m-cyl1	1/2
1.1-2 Double-acting Cylinder Model : m-cyl2	1/3
1.1-3 Double-acting Bistable Cylinder Model : m-cyl2b	1/4
1.1-4 Example : Mechanical Interlocking	1/5
1.2 Models for Conveying Parts	1/6
1.2-1 Model for Conveying 1 Part in 1 Direction : m-con1p	1/7
1.2-2 Model for Conveying 2 Parts in 1 Direction : m-con2p	1/8
1.2-3 Model for Conveying 3 Parts in 1 Direction : m-con3p	1/9
1.2-4 Model for Conveying 10 Parts in 1 Direction : m-con10p	1/10
1.2-5 Principle of Describing Conveying	1/11
1.2-6 Example : Simple Forward Operation	1/13
1.2-7 Example : Forward and Reverse Operation	1/17
1.2-8 Example : Taking the Length of the Part into Account	1/19
1.3 Tank Level Model	1/21
1.3-1 Model for a 2 Valve Tank : m-tank2v	1/21
1.3-2 Model for a Level : m-level	1/22
1.3-3 Example : Single Tank	1/23
1.4 Model for Proportional Action Valves	1/24
1.4-1 Model for Single Proportional Action Valve : m-valve1	1/24
1.4-2 Model for Variable Opening Proportional Action Valve : m-valve2	1/25
1.4-3 Example : Single Tank With 1 Downstream Proportional Action Valve	1/26
1.5 Blank Data Sheet for Describing a Model	1/27

Section	Page
2 Machine With Three Machining Stations	2/1
2.1 SIMTSX Description	2/1
2.1.1 Introduction	2/1
2.1.2 Modeling	2/2
2.2 Electrical and Mechanical Dossier	2/7

1 SIMTSX Model Library

SIMTSX has a library of generic models which represent description elements which are commonly used in automated installations :

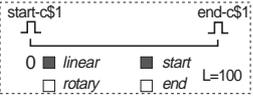
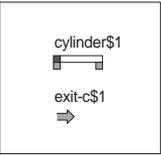
- Several cylinder models : single-acting, double-acting, bistable (evolution of the position of the cam)
- Several models for conveyors, belts, roller tables (evolution of the position of parts)
- Tank model (evolution of the level depending on the regulated flow through 1 valve upstream and 1 valve downstream)
- Generic level model (evolution of the level depending on the sum of the incoming flow and outgoing flow)
- Several proportional valve models (evolution of the opening of the valve)

These models can be used as they are, or copied and adapted to a particular context before being incorporated in an application. The library can also be expanded with new standard models which are application-specific or are derived from applications developed using SIMTSX (see part G, Tool for Transferring from an Application to a Model).

At the end of this section, a blank data sheet is provided for describing models.

1.1 Cylinder Models

1.1-1 Single-acting Cylinder Model : m-cyl1

Model:m-cyl1		Parameters: \$1 = cylinder number		
I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O	No relays	<p>ax-c\$1 : cylinder\$1</p>  <p> <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input type="checkbox"/> rotary <input type="checkbox"/> end L=100 </p> <p>Sensor start-c\$1 Position : start Logic : pulse Activation : 1 end-c\$1 Position : end Logic : pulse Activation : 1</p> <p>Movement mov-c\$1+= pressure*exit-c\$1 direction : positive, speed : 10 mov-c\$1-= /exit-c\$1 +/pressure direction : negative, speed : 10</p>	No operator panel	<p>Screenbackground</p> 

The mechanical and pneumatic description of a single-acting cylinder is given by a single linear axis. The cylinder output is enabled if the solenoid valve for output **exit-c\$1** is active. The axis feedback is enabled either when the solenoid valve for output **exit-c\$1** is inactive or when the pressure is insufficient.

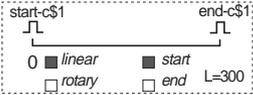
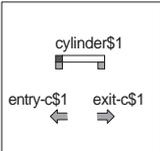
Instantiation :

Parameter which should be filled in during instantiation of the model in a SIMTSX application : \$1 = axis and cylinder number

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Evolution speed can be modified in the axis>movement menus, in the "Speed" field.
- Start and end limit sensors can be connected to the PLC inputs.
- The "exit" variable corresponds to the cylinder output control relay.

1.1-2 Double-acting Cylinder Model : m-cyl2

Model:m-cyl2		Parameters: \$1=cylinder number		
I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O		ax-c\$1:cylinder\$1  Sensor start-c\$1 Position : start Logic : pulse Activation : 1 end-c\$1 Position : end Logic : pulse Activation : 1 Movement mov-c\$1+= pressure*exit-c\$1 direction : positive, speed : 10 mov-c\$1-= pressure*entry-c\$1 direction : negative, speed : 10	No operator panel	

The mechanical and pneumatic description of a double-acting cylinder is given by a single linear axis. The cylinder output or input is enabled if there is pressure and one of the solenoid valves for output **exit-c\$1** or input **entry-c\$1** is active. If both solenoid valves are active at the same time, with pressure, there is no evolution and a message signals that there is a movement modeling problem.

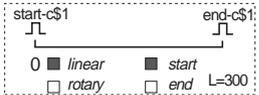
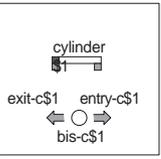
Instantiation:

Parameter which should be filled in during instantiation of the model in a SIMTSX application : \$1 = axis and cylinder number

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Evolution speed can be modified in the axis>movement menus, in the "Speed" field.
- Start and end limit sensors can be connected to the PLC inputs.
- The "exit" and "entry" variables correspond to the cylinder control relays.

1.1-3 Double-acting Bistable Cylinder Model : *m-cyl2b*

<i>Model</i> : m-cyl2b		<i>Parameters</i> : \$1 = cylinder number		
I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O	No relays	<p>ax-c\$1 : cylinder \$1</p>  <p>Sensor start-c\$1 Position : start Logic : pulse Activation : 1 end-c\$1 Position : end Logic : pulse Activation : 1</p> <p>Bistable Bis-c\$1 Latch : exit-c\$1 Release : entry-c\$1</p> <p>Movement mov-c\$1+= pressure*Bis-c\$1 direction : positive, speed : 10 mov-c\$1-= pressure*/Bis-c\$1 direction : negative, speed : 10</p>	No operator panel	<p>Screenbackground</p> 

The mechanical and pneumatic description of a bistable cylinder is given by a single linear axis with a bistable **Bis-c\$1**. The cylinder output or input is enabled if there is pressure and a pulse on one of the solenoid valves for output **exit-c\$1** or input **entry-c\$1**. If both solenoid valves are active at the same time, with pressure, there is no evolution and a message signals that there is a bistable modeling problem.

Instantiation :

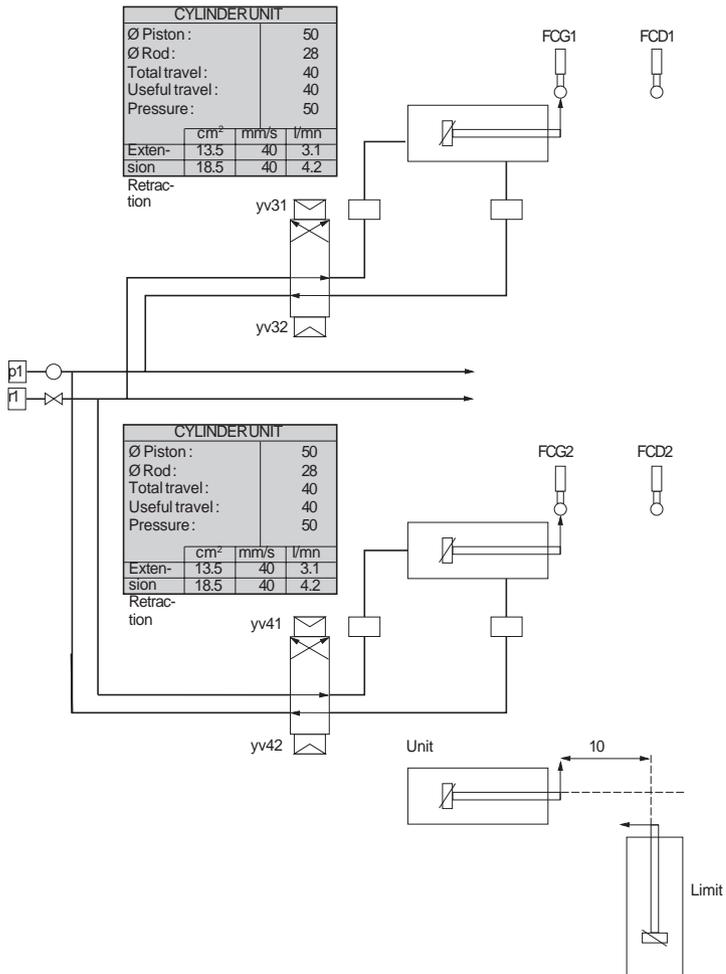
Parameter which should be filled in during instantiation of the model in a SIMTSX application : \$1 = axis and cylinder number

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Evolution speed can be modified in the axis>movement menus, in the "Speed" field.
- Start and end limit sensors can be connected to the PLC inputs.
- The "exit" and "entry" variables correspond to the cylinder control relays.

1.1-4 Example : Mechanical Interlocking

Mechanical and pneumatic description of the actual installation

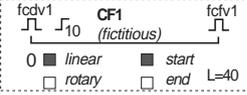
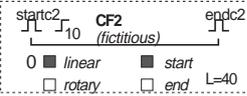
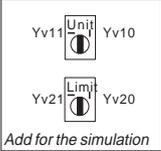
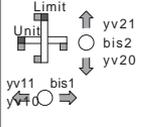


This example is described using two single axes, where one is the mechanical lock for the other. This interaction is taken into account by adding two fictitious sensors (described as sensors) representing the free travel of the cylinders (axis 1 and axis 2) and by integrating them into the equations for movement $mov-c1(2)+$ and $mov-c1(2)+$. These sensors are internal to the description, and do not exist from the point of view of the PLC (this example does not consider problems experienced during the backward movement of the cylinders).

For this example, the model "m-cyl2b" is used twice : instances 1 and 2. The axes must now be edited to add 2 theoretical sensors FC1 and FC2, and the movement equations must be modified to take the clashes into account.

SIMTSX description of the application

Application:	UNIT-LIMIT
--------------	-------------------

I/O config	RELAYS	AXES	PANELS	VIEWS
NoI/O	No relays	<p>ax-c1: Unit</p>  <p>Bistable: Bis1 E: Yv31, D: Yv32</p> <p>Movement Mov-c1+= $p1 * bis1 * (/cf1 + cf1 * /cf2)$ positive, speed = 4 Mov-c1-=$p1 * bis1$ negative, speed = 1</p> <p>ax-c2: Limit</p>  <p>Bistable: Bis2 E: Yv41, D: Yv42</p> <p>Movement Mov-c2+= $p1 * bis2 * (/cf2 + cf2 * /cf1)$ positive, speed = 4 Mov-c2-=$p1 * bis2$ negative, speed = 1</p>	 <p style="text-align: center;">Add for the simulation</p>	<p>Screenbackground</p> 

Notes

The commands yv31, yv32, yv41 and yv42 activating the movements mov-c1+, mov-c1-, mov-c2+ and mov-c2- are SIMTSX relays. "pneu" is a general SIMTSX power supply term. For simulation, it is possible to add 2 switches to control the movements by hand. It is also possible to describe a description trap to signal the clash instant.

1.2 Models for Conveying Parts

These generic conveying models are designed to be used together. The parameter settings take into account the arrival of parts upstream and the departure of parts downstream of the belt. To instantiate these models consecutively, simply fill in the upstream-downstream parameters with the numbers of the corresponding belts.

1.2-1 Model for Conveying 1 Part in 1 Direction : m-con1p

Model: m-con1p		Parameters: \$0 = set down part or previous belt \$1 = belt and motor number \$9 = remove part or next belt		
I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O	Relay mforw\$1 = 380v * q\$1 * km\$1	<p>ax-c\$1-0: Conveyorbelt \$1</p> <p>Sensor dx\$1 = start, pulse Activation : 1 fx\$1 = end, pulse</p> <p>Movement mov-c\$1 = mforw\$1 * (pp\$1 * /fx\$1 + fx\$1 * /pp\$1) direction : positive, speed : 1</p> <p>Variable pp\$1 : presence of a part in section 1 = (dx\$1 * fx\$0 + pp\$1) * / (fx\$1 * dx\$9)</p>	No operator panel	<p>Conveyorbelt \$1</p> <p>pp\$1</p> <p>mforw\$1</p>

Description of the evolution of a single part on a belt or a roller table. The part occupies the entire belt.

Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application :

- \$0 = set down part or previous belt
- \$1 = belt and motor number
- \$9 = remove part or next belt

After instantiation :

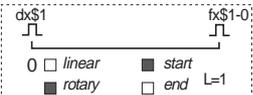
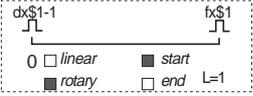
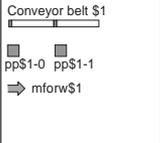
- Length can be modified in the axis menu, in the "Length" field.
- Speed can be modified in the axis>movement menus, in the "Speed" field.
- The "mforw" variable corresponds motor movement control relay.



1.2-2 Model for Conveying 2 Parts in 1 Direction : m-con2p

Model :m-con2p

Parameters: \$0 = set down part or previous belt
 \$1 = belt and motor number
 \$9 = remove part or next belt

I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O	Relay mforw\$1 380v * q\$1 * km\$1	<p>ax-c\$1-0:(0)</p>  <p>0 <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input checked="" type="checkbox"/> rotary <input type="checkbox"/> end L=1</p> <p>Movement mov-c\$1-0 = mforw\$1 * (pp\$1-0 * /fx\$1-0+ fx\$1-0 * /pp\$1-0) direction : positive, speed : 1</p> <p>Variable pp\$1-0 : presence of a part in section 0 = (dx\$1 * fx\$0 + pp\$1-0) * / (fx\$1-0 * dx\$1-1) ax-c\$1-1:(1)</p>  <p>0 <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input checked="" type="checkbox"/> rotary <input type="checkbox"/> end L=1</p> <p>Movement mov-c\$1-1 = mforw\$1 * (pp\$1-1 * /fx\$1+ fx\$1 * /pp\$1-1) direction : positive, speed : 1</p> <p>Variable pp\$1-1 : presence of a part in section 1 = (dx\$1-1 * fx\$1-0 + pp\$1-1) * / (fx\$1- * dx\$9)</p>	No operator panel	 <p>Conveyor belt \$1 pp\$1-0 pp\$1-1 => mforw\$1</p>

Description of the evolution of two parts on a belt or a roller table. The belt has a capacity of 2 parts maximum, occupying the entire belt.

Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application :

- \$0 = set down part or previous belt
- \$1 = belt and motor number
- \$9 = remove part or next belt

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Speed can be modified in the axis>movement menus, in the "Speed" field.
- The "mforw" variable corresponds motor movement control relay.

1.2-3 Model for Conveying 3 Parts in 1 Direction : m-con3p

Model:m-con3p		Parameters: \$0 = set down part or previous belt \$1 = belt and motor number \$9 = remove part or next belt		
I/O config.	RELAYS	AXES	PANELS	VIEWS
No/I/O	Relay mforw\$1= 380v * q\$1 * km\$1	<p>ax-c\$1-0:(0)</p> <p>ax-c\$1-1:(1)</p> <p>ax-c\$1-2:(2)</p> <p>Movements mov-c\$1-0 = mforw\$1 * (pp\$1-0 * fx\$1-0+ fx\$1-0 * pp\$1-0) / pp\$1-0 mov-c\$1-1 = mforw\$1 * (pp\$1-1 * fx\$1-1+ fx\$1-1 * pp\$1-1) / pp\$1-1 mov-c\$1-2 = mforw\$1 * (pp\$1-2 * fx\$1-2+ fx\$1-2 * pp\$1-2) / pp\$1-2 direction : positive, speed : 1</p> <p>Variables pp\$1-0 : presence of a part in section 0 = (dx\$1 * fx\$0 + pp\$1-0) * / (fx\$1-0 * dx\$1-0) pp\$1-1 : presence of a part in section 1 = (dx\$1-1 * fx\$1-0 + pp\$1-1) * / (fx\$1-1 * dx\$1-2) pp\$1-2 : presence of a part in section 2 = (dx\$1-2 * fx\$1-1 + pp\$1-2) * / (fx\$1 * dx\$9)</p>	No operator panel	

Description of the evolution of three parts on a belt or a roller table. The belt has a capacity of 3 parts maximum, occupying the entire belt.

Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application :

- \$0 = set down part or previous belt
- \$1 = belt and motor number
- \$9 = remove part or next belt

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Speed can be modified in the axis>movement menus, in the "Speed" field.
- The "mforw" variable corresponds motor movement control relay.



1.2-4 Model for Conveying 10 Parts in 1 Direction : m-con10p

I/O config.		RELAYS	AXES	PANELS	VIEWS
No I/O		Relay mforw\$1 = 380v * q\$1 * km\$1	<p>ax-c\$1-0:(0)</p> <p>0 <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input checked="" type="checkbox"/> rotary <input type="checkbox"/> end L=1</p> <p>ax-c\$1-i: (i) <i>where i = 1 to 8</i></p> <p>0 <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input checked="" type="checkbox"/> rotary <input type="checkbox"/> end L=1</p> <p>ax-c\$1-9:(9)</p> <p>0 <input type="checkbox"/> linear <input checked="" type="checkbox"/> start <input checked="" type="checkbox"/> rotary <input type="checkbox"/> end L=1</p> <p>Movements mov-c\$1-0 = mforw\$1 * (pp\$1-0 * /fx\$1-0+ fx\$1-0 * /pp\$1-0) ... mov-c\$1-i = mforw\$1 * (pp\$1-i * /fx\$1-i+ fx\$1-i * /pp\$1-i) where i = 1 to 8 ... mov-c\$1-9 = mforw\$1 * (pp\$1-9 * /fx\$1-9+ fx\$1-9 * /pp\$1-9) direction : positive, speed : 1</p> <p>Variables pp\$1-0 : presence of a part in section 0 = (dx\$1 * fx\$0 + pp\$1-0) * / (fx\$1-0 * dx\$1-1) pp\$1-i : presence of a part in section i = (dx\$1-i * fx\$1- [i-1]+ pp\$1-i) * / (fx\$1-i * dx\$1- [i-1]) where i = 1 to 8 pp\$1-9 : presence of a part in section 9 = (dx\$1-9 * fx\$1-8+ pp\$1-9) * / (fx\$1 * dx\$9)</p>	No operator panel	<p>pp\$1-0 pp\$1-i pp\$1-9</p> <p>mforw\$1</p> <p>Belt \$1</p> <p>pp\$1-0 pp\$1-i pp\$1-9</p> <p>mforw\$1</p>

Description of the evolution of ten parts on a belt or a roller table. The belt has a capacity of 10 parts maximum, occupying the entire belt.

Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application :

\$0 = set down part or previous belt

\$1 = belt and motor number

\$9 = remove part or next belt

After instantiation :

- Length can be modified in the axis menu, in the "Length" field.
- Speed can be modified in the axis>movement menus, in the "Speed" field
- The "mforw" variable corresponds motor movement control relay.
- There are two possible model views : displaying the evolution of parts only or displaying the modeled axes.

1.2-5 Principle of Describing Conveying

In materials handling processes, the PLC receives information from the machine on the presence and movement of parts. An application is represented, in SIMTSX, by axes and Boolean variables. The parts are not represented explicitly. The SIMTSX description simulates the presence of parts by calculating their actions on the PLC inputs according to the PLC outputs and the time elapsed.

The following section presents the principle of modeling a conveyor system. Examples of transferring in one, then in two operating directions are given. The detection of conveyed parts, taking account of their length, is also explained.

The suggested models are based on this principle.

Principle of modeling

The modeling of a conveyor system consists of dividing the system into zones where only one part may be found : this is the concept of "section".

The presence of a part in a zone is represented by a Boolean variable "PPi". The movement takes a certain amount of time, and an axis associated with each "section" is used to represent this time : the position on the axis is the "front edge" of the moving part. The movement of this axis represents the movement of the part in the zone described by the axis. The equation for this movement uses the variable "PPi" and a term - for example "KMVTi" representing the rotation of the motor driving the conveyor.

The movement of the part in the zone described by the axis is considered to be a process made up of three states. The initial state corresponds to the availability of the conveyor for a part arriving upstream. The second state is the movement - and also stopping at a given position - of the part on the section. The final state is the end of the transfer in this zone, where the part is made available to the next section. This process is cyclical, the axis is thus rotary. A fictitious sensor "DXi" of width zero, placed at the start of the axis, represents the availability of the section. Similarly, a sensor "FXi", also with a zero activation width, is placed at the end of the rotary axis to represent making the part available for the rest of the transfer.

The evolution of the transfer process is described via a movement equation. The part is moved when the motor is running and :

- a part is present and the end of the section has not been reached
- the end of the section has been reached and the part is taken into the next section or removed

The following equation is obtained for the movement :

$$MVT_i = KMVT_i * (PPI * /FX_i + FX_i * /PPI)$$

Once it reaches "FX_i" and the conveyed part - "PPI" - is reset to 0, the rotary axis returns to the initial position - on "DX_i" - and is then available for a new part.

A part enters the section if the section is at its initial state and the previous section makes a part available. Taking account of the representation of states of the section via sensors "DX_i" and "FX_i", the latch term of the part present variable "PPI" is :

$$DX_i * PPI_{i-1} * FX_{i-1} * KMVT_{i-1}$$

The presence of "KMVT_{i-1}" keeps the part in the previous section if the motor for that section is not running - this should definitely not happen.

Once the transfer is finished on a section - the corresponding axis is then at "FX_i" - the part can move on to the next section if this is available - the axis describing it is then "DX_{i+1}". However, it is important to check that this section does not already contain a part. In fact, if the motor of the next section is not controlled, the axis will have been able to stay in "DX_{i+1}" even though a part is present on this section. In these conditions, the release term of variable "PPI" has the following expression :

$$RPP_i = FX_i * DX_{i+1} * /PPI_{i+1} * KMVT_i$$

The equation of the part present variable "PPI" has the following final expression :

$$PPI = (DX_i * PPI_{i-1} * FX_{i-1} * KMVT_{i-1} + PPI) * /RPP_i$$

1.2-6 Example : Simple Forward Operation

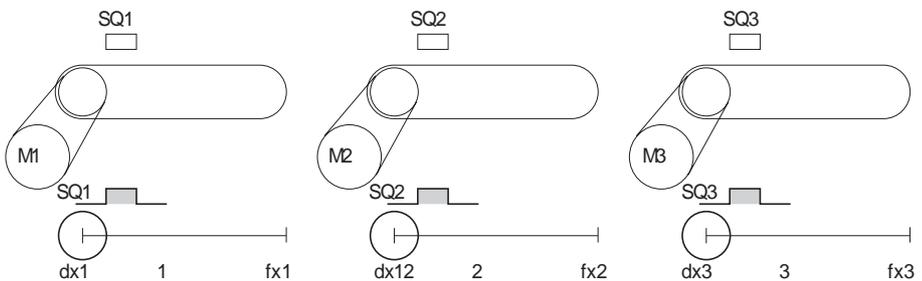
The following example shows the movement of parts on three conveyor belts in forward operation. The size of the parts is considered negligible, there is only one part per belt and the motors for the belts are in continuous operation. The belts are represented by rotary axes. The presence of a part is represented by a Boolean variable.

The movements take place when the following two conditions are true :

- there is a part on the axis
- the conveyor belt motor is in forward operation

For a part to move from one conveyor belt to another, the installation must also meet the following two criteria :

- the downstream axis must be at the reference point (waiting for parts)
- there is no part on the downstream axis (avoid clashes)



Description of the first conveyor belt

The first conveyor belt is at the beginning of the conveyor system, a variable "BPP1" is used to introduce a part in the system.

The part appears on axis 1 when the pushbutton is pressed and the axis is at the start limit. This state maintained until RPP1 (Boolean variable for resetting the part present) is true. RPP1 is true when axis 1 is at the end limit, axis 2 at the start limit, there is no part on the next conveyor belt and the motor of conveyor belt 1 is active.

Movement MVT1 is active as long as the motor is active, PP1 is true, or to move automatically from the end limit position to the start limit position.

Equations for the first conveyor belt

- $PP1 = (+BPP1 * DX1 + PP1) * /RPP1$
- $RPP1 = FX1 * DX2 * /PP2 * KMVT1$
- $MVT1 = (PP1 * /FX1 + FX1 * /PP1) * KMVT1$

where :

- PP1 : part present on conveyor belt 1
- PP2 : part present on conveyor belt 2
- DX1 : start limit sensor of axis 1
- DX2 : start limit sensor of axis 2
- FX1 : end limit sensor of axis 1
- +BPP1 : rising edge of fictitious pushbutton used to place parts on the first conveyor belt
- RPP1 : resets part present
- KMVT1 : controls the motor on conveyor belt 1

Description of the second conveyor belt

The part is considered to be present on axis 2 when axis 2 is at the start limit, axis 1 is at the end limit, there is a part on axis 1 and the motor of axis 1 is active. The part present on axis 2 is held in the same way as the part present on axis 1.

The other equations are similar to those of axis 1.

Equations for the second conveyor belt

- $PP2 = (DX2 * FX1 * PP1 * KMVT1 + PP2) * /RPP2$
- $RPP2 = FX2 * DX3 * /PP3 * KMVT2$
- $MVT2 = (PP2 * /FX2 + FX2 * /PP2) * KMVT2$

where :

- PP1 : part present on conveyor belt 1
- PP2 : part present on conveyor belt 2
- DX2 : start limit sensor of axis 2
- DX3 : start limit sensor of axis 3
- FX1 : end limit sensor of axis 1
- FX2 : end limit sensor of axis 2
- RPP2 : resets part present
- KMVT1 : controls the motor on conveyor belt 1
- KMVT2 : controls the motor on conveyor belt 2

Description of the third conveyor belt

We assume that the parts disappear automatically at the end of conveyor belt 3 (removed to part of the installation which has not been simulated, for example). For the rest, the equations are similar to those described previously.

Equations for the third conveyor belt

- $PP3 = (DX3 * FX2 * PP2 * KMVT2 + PP3) * /RPP3$
- $RPP3 = FX3 * KMVT3$
- $MVT3 = (PP3 * /FX3 + FX3 * /PP3) * KMVT3$

where :

- PP2 : part present on conveyor belt 2
- PP3 : part present on conveyor belt 3
- DX3 : start limit sensor of axis 3
- FX2 : end limit sensor of axis 2
- FX3 : end limit sensor of axis 3
- RPP3 : resets part present
- KMVT2 : controls the motor on conveyor belt 2
- KMVT3 : controls the motor on conveyor belt 3

1.2-7 Example : Forward and Reverse Operation

This example uses the description given above but includes the reverse operation of the conveyor belt. The equations for forward movements remain the same.

SIMTSX description

In the part present equations, in comparison with the previous example, an additional term is added. When a part is moving from one conveyor belt to another, it is important to check that the conveyor belt which is to receive the part is turning in the same direction as the conveyor belt which is bringing the part to it. If not, the part must not change section.

The SIMTSX description elements are as follows :

- PP1 : Boolean variable for part present on conveyor belt 1
- PP2 : Boolean variable for part present on conveyor belt 2
- PP3 : Boolean variable for part present on conveyor belt 3
- DX1 : start limit sensor of axis 1
- DX2 : start limit sensor of axis 2
- DX3 : start limit sensor of axis 3
- FX1 : end limit sensor of axis 1
- FX2 : end limit sensor of axis 2
- FX3 : end limit sensor of axis 3
- +BPP1 : rising edge of fictitious pushbutton used to place parts on the first conveyor belt
- RPP1 : Boolean variable to reset part present on conveyor belt 1
- RPP2 : Boolean variable to reset part present on conveyor belt 2
- RPP3 : Boolean variable to reset part present on conveyor belt 3
- MTAV1 : controls motor for moving conveyor belt 1 forward
- MTAV2 : controls motor for moving conveyor belt 2 forward
- MTAV3 : controls motor for moving conveyor belt 3 forward
- MTAR1 : controls motor for moving conveyor belt 1 backward
- MTAR2 : controls motor for moving conveyor belt 2 backward
- MTAR3 : controls motor for moving conveyor belt 3 backward
- MVAV1 : forward movement of axis 1
- MVAV2 : forward movement of axis 2
- MVAV3 : forward movement of axis 3
- MVAR1 : backward movement of axis 1
- MVAR2 : backward movement of axis 2
- MVAR3 : backward movement of axis 3

Equations for the first conveyor belt

- $PP1 = (+BPP1 * DX1 + FX1 * DX2 * PP2 * MTAR2 * /MTAV1 + PP1) * /RPP1$
- $RPP1 = FX1 * DX2 * /PP2 * MTAV1 * /MTAR2 + DX1 * MTAR1$
- $MVAV1 = (PP1 * /FX1 + FX1 * /PP1) * MTAV1$
- $MVAR1 = (PP1 * /DX1 + DX1 * /PP1) * MTAR1$

Equations for the second conveyor belt

- $PP2 = (DX2 * FX1 * PP1 * MTAV1 * /MTAR2 + FX2 * DX3 * PP3 * MTAR3 * /MTAV2 + PP2) * /RPP2$
- $RPP2 = FX2 * DX3 * /PP3 * MTAV2 * /MTAR3 + DX2 * FX1 * /PP1 * /MTAV1 * MTAR2$
- $MVAV2 = (PP2 * /FX2 + FX2 * /PP2) * MTAV2$
- $MVAR2 = (PP2 * /DX2 + DX2 * /PP2) * MTAR2$

Equations for the third conveyor belt

- $PP3 = (DX3 * FX2 * PP2 * MTAV2 * /MTAR3 + PP3) * /RPP3$
- $RPP3 = FX3 * MTAV3 + DX3 * FX2 * /PP2 * /MTAV2 * MTAR3$
- $MVAV3 = (PP3 * /FX3 + FX3 * /PP3) * MTAV3$
- $MVAR3 = (PP3 * /DX3 + DX3 * /PP3) * MTAR3$

1.2-8 Example : Taking the Length of the Part into Account

The example described here has three elements. It concerns roller tables 6000 mm in length on which sleds of length 5000 mm move.

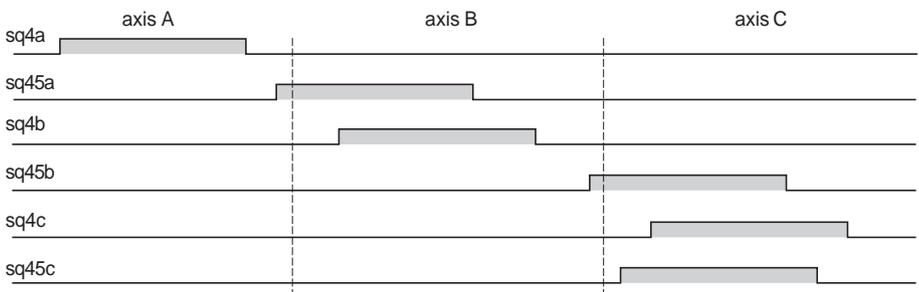


The presence of parts is represented by Boolean variables, the position on an axis describing a section corresponding to the front edge of the part. In this modeling operation, there is no mention of the length of the part, the only constraint imposed is that there should only be one part per section.

However, these parts signal their movement to the PLC via detectors. At this level at least, the length of these parts should be taken into account : in fact, the detectors are active the whole time the parts are moving.

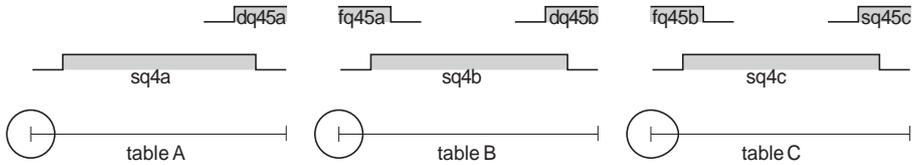
This cannot be done by simply placing real sensors on the axes. In particular, when one part starts to enter the next section, it continues to be detected in the section it is leaving, because of its length.

This means that the detectors must be modeled by "relay" logic equations used in fictitious sensors placed on the axes, conditioned by part present variables to reproduce, as a function of the movement of the parts, the trend diagrams for activating the detectors shown below.



SIMTSX description

The installation described using SIMTSX is similar to that previously explained in this section. Because of the size of the parts, the real sensors SQ45A and SQ45B are activated on one axis after the other. These sensors are represented by relays activated by the part present variables and fictitious sensors on the axes.

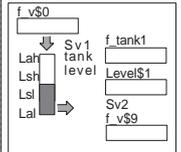


Equations for the relays representing the sensors :

- $SQ45A = DQ45A * PPA + FQ45A * PPB$
- $SQ45B = DQ45B * PPB + FQ45C * PPC$

1.3 Tank Level Model

1.3-1 Model for a 2 Valve Tank : m-tank2v

I/O config.	RELAYS	AXES	PANELS	VIEWS
No I/O	<p>Relay Sv\$0 (undefined) Sv\$9 (undefined)</p>	<p>Parameters: \$0=upstream valve number \$1=tank number \$9=downstream valve number</p> <p>ax-t\$1: Tank level</p> <div style="border: 1px dashed black; padding: 5px;"> <p>Lal\$1 Lsl\$1 Lsh\$1 Lah\$1</p> <p><input type="checkbox"/> linear <input type="checkbox"/> start <input type="checkbox"/> rotary <input type="checkbox"/> end L=100</p> <p><input checked="" type="checkbox"/> sampled, step=1</p> </div> <p>Sensors Lal\$1 = 10, latch Lsl\$1 = 20, latch Lsh\$1 = 80, latch Lah\$1 = 90, latch t\$1_empty = start, pulse</p> <p>Movement f&e_t\$1 = () positive, speed = f_tank1</p> <p>Numeric variable I_t\$1 : tank level ax_t\$1 f_tank1 : level evolution speed 0 + f_v\$0 - f_v\$9 if /t\$1_empty f_v\$0 : tank filling rate 0 10 if /sv\$0 f_v\$9 : tank emptying rate 0 5 if /sv\$9 * /t\$1_empty</p>	No operator panel	

Description of the evolution of level in a tank, capacity of the tank = 100 liters, filling rate = twice the emptying rate.

Instantiation:

Parameter which should be filled in during instantiation of the model in a SIMTSX application :

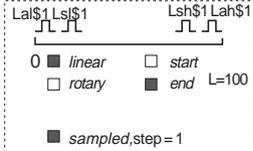
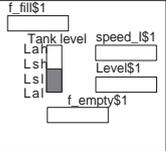
\$0 = upstream valve number

\$1 = tank number

\$9 = downstream valve number

- The variable "I_t\$1" corresponds to the value of the level in the tank.
- The capacity of the tank can be modified by editing the tank axis ("length"). Please note when selecting the axis sampling that steps of 1 should be avoided for axes with length 1000.
- Upstream and downstream valve flows can be modified (constants, external variables, PLC settings), simply edit the numeric variables "f_v\$0" for filling and "f_v\$9" for emptying, and replace "10" and "5" with the corresponding flow rates.
- Valves can be added in the assignment table for numeric variables "f_v\$0" and "f_v\$9" (for example, for several inflows).

1.3-2 Model for a Level : m-level

Model:m-level		Parameters: \$1 = tank number, level n filling/emptying rate					
I/O config	RELAYS	AXES	PANELS	VIEWS			
No I/O		<p>ax-\$1: Tank level\$1</p>  <p>Sensors LaL\$1 = 10, latch LsL\$1 = 20, latch Lsh\$1 = 80, latch Lah\$1 = 90, latch \$1_empty = start, pulse</p> <p>Movement mov-\$1 = () positive, speed = speed_\$1</p> <p>Numeric variable level\$1 : tank level ax-\$1 f_fill\$1 : sum of the filling flows f_empty\$1 : sum of the emptying flows speed_\$1 : tank level evolution speed</p> <table border="1" data-bbox="407 837 562 901"> <tr><td>0</td></tr> <tr><td>+ f_fill\$1</td></tr> <tr><td>- f_empty\$1 if /\$1_empty</td></tr> </table>	0	+ f_fill\$1	- f_empty\$1 if /\$1_empty	No operator panel	
0							
+ f_fill\$1							
- f_empty\$1 if /\$1_empty							

Description of the evolution of level in a tank, capacity of the tank = 100 liters, filling rate = sum of upstream flow, emptying rate = sum of downstream flow.

Instantiation :

Parameter which should be filled in during instantiation of the model in a SIMT SX application :
 \$1 = tank, level, filling/emptying rate

The variable "level\$1" corresponds to the value of the level in the tank.

The capacity of the tank can be modified by editing the tank axis ("length"). Please note when selecting the axis sampling that steps of 1 should be avoided for axes with length 1000.

Filling and emptying rates which are defined as external variables by default can be modified during simulation or adapted to the target application by editing the assignment tables for flows "f_fill\$1" and "f_empty\$1".

1.3-3 Example : Single Tank

Modeling the level in a tank comprising 2 parallel upstream valves (sv1, flow = 5 and sv2, flow = 10), 1 downstream valve (sv3, flow = 3).

To create this application, simply instantiate the "m-level" model once (parameter to be instantiated : \$1 = "1").

Modifications to be made in the application once the model has been instantiated :
Edit the numeric variable F-fill1 and complete it as below :

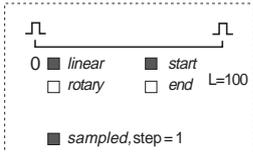
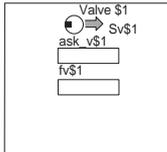
0		assign 0 without condition to reset the value
+ 5	< sv1	operation (F-fill1 + 5) if valve control sv1
+ 10	< sv2	operation (F-fill1 + 10) if valve control sv2

Edit the numeric variable F-empty1 and complete it as below :

0		assign 0 without condition to reset the value
+ 3	< sv3	operation (F-empty1 + 3) if control valve sv3

1.4 Model for Proportional Action Valves

1.4-1 Model for Single Proportional Action Valve : m-valve1

Model : m-valve1		Parameters : \$1 = valve number, control, setpoint					
I/O config.	RELAYS	AXES	PANELS	VIEWS			
No I/O	Relay v\$1 (undefined)	<p>ax-v\$1 : valve \$1</p>  <p>Movement o&c-v\$1 = v\$1 positive, speed = open_v\$1</p> <p>Numeric variable fv\$1: valve current flow ax-v\$1 open_v\$1: valve evolution speed</p> <table border="1" data-bbox="407 742 568 798"> <tr><td>0</td></tr> <tr><td>+ 5 if ax-v\$1 < ask_v\$1</td></tr> <tr><td>- 5 if ax-v\$1 > ask_v\$1</td></tr> </table> <p>ask_v\$1: valve opening setpoint</p>	0	+ 5 if ax-v\$1 < ask_v\$1	- 5 if ax-v\$1 > ask_v\$1	No operator panel	
0							
+ 5 if ax-v\$1 < ask_v\$1							
- 5 if ax-v\$1 > ask_v\$1							

Description of the evolution of the opening position of a valve : opening = 0-100%, opening speed = 5 mm/s (if the units for the developed application are millimeters and seconds).

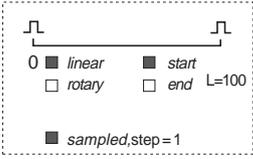
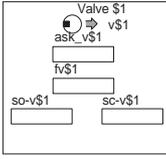
Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application :

\$1 = valve number, control, setpoint.

- The variable "fv\$1" corresponds to the valve flow, which depends on its opening position, to be connected to a numeric PLC input.
- The opening setpoint "ask_v\$1" is an external numeric variable which should be connected to a numeric PLC output.
- The opening and closing time for the valve can be modified in the assignment table of "open_v\$1" (in this example = 5mm/s).

1.4-2 Model for Variable Opening Proportional Action Valve : *m-valve2*

Model : <i>m-valve2</i>		Parameters: \$1 = valve number, control, setpoint		
I/O config	RELAYS	AXES	PANELS	VIEWS
No I/O	Relay v\$1 (undefined)	<p>ax-v\$1 : valve \$1</p>  <p>Movement o&c-v\$1 = v\$1 positive, speed = open_v\$1</p> <p>Numeric variable fv\$1: valve current flow ax-v\$1 open_v\$1: valve evolution speed</p> <div style="border: 1px solid black; padding: 2px;"> <p>0 + so-v\$1 if ax-v\$1 < ask_v\$1 - sc-v\$1 if ax-v\$1 > ask_v\$1</p> </div> <p>askv\$1: valve opening setpoint so-v\$1 : valve opening speed sc-v\$1 : valve closing speed</p>	No operator panel	

Description of the evolution of the open position of a valve : opening = 0-100%, opening and closing time can be modified by an external variable.

Instantiation:

Parameters which should be filled in during instantiation of the model in a SIMTSX application : \$1 = valve number, control, setpoint.

- The variable "fv\$1" corresponds to the valve flow, which depends on its opening position, to be connected to a numeric PLC input..
- The opening setpoint "ask_v\$1" is an external numeric variable which should be connected to a numeric PLC output.
- The opening and closing time for the valve can be modified during simulation using the external variables "so-v\$1" and "sc-v\$1". These numeric variables can also be set or calculated using an assignment table.



1.4-3 Example : Single Tank With 1 Downstream Proportional Action Valve

Modeling the level in a tank comprising 2 parallel upstream valves (sv1, flow = 5 and sv2, flow = 10), 2 downstream valves in series (sv3, flow = 3 and sv4, flow = 0-100%).

To create this application, simply instantiate the "m-level" model once (parameter to be instantiated : \$1 = "1"), and the "m-valve1" proportional action valve model once (parameter to be instantiated : \$1 = "4").

Modifications to be made in the application once the models have been instantiated :
Edit the numeric variable f_fill1 and complete it as below :

0		assign 0 without condition to reset the value
+ 5	< sv1	operation (f_fill1 + 5) if valve control sv1
+ 10	< sv2	operation (f_fill1 + 10) if valve control sv2

Edit the numeric variable f_empty1 and complete it as below :

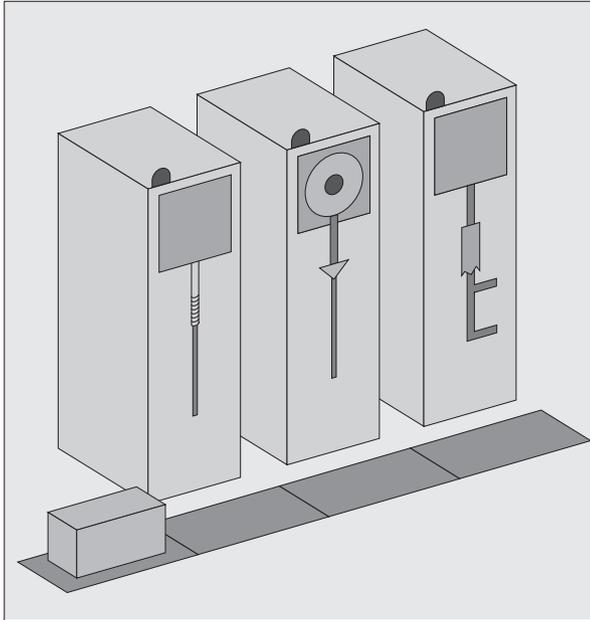
0		assign 0 without condition to reset the value
+ 3	< sv3	operation (f_empty1 + 3) if control valve sv3
+ Qv4	< sv4	operation (f_empty1 * flow value of proportional action valve 4)
/ 100		operation (f_empty1 / 100).

H

2.1 SIMTSX Description

2.1.1 Introduction

The SIMTSX software environment is supplied with an application example illustrating the various concepts outlined in this manual : ModelTSX.



This model comprises three machining stations placed in a line. Before each station is a conveyor belt on which the part to be machined is placed. These conveyor belts are used to transfer parts from one station to the next, from entry at the beginning of the line to removal at the end.

The parts placed at the entry point are fed onto the first conveyor belt, which in this case is for the drilling machine, by means of a feeder.

2.1.2 Modeling

This section explains the modeling process performed using SIMTSX.

• The I/O configuration

The suggested application does not include an I/O configuration, but operates in local simulation with a simple chart. It is therefore possible to connect it to Micro, Premium and Quantum PLCs by linking the relays, sensors and operator panel variables to the corresponding I/O.

• The relays

The relays serve as power intermediaries between the outputs and the actuator control and the electrical wiring for the indicator lamps.

• Axis description

The description of the model includes 12 axes. Axes 1 to 7 correspond to the actuators for the 3 stations. Thus, axis 1 represents the rise/fall of the drilling machine.

This axis has 3 sensors : FCH1 corresponds to the top position of the drilling machine, FCM1 to the middle position and FCB1 to the bottom position. “Dperc” is the downward movement of the drilling machine, controlled by relay Km15, and relay Km14 controls the upward movement “Mperc”, both via the intermediary of motor MLP1.

Similarly, axis 2 represents the rise/fall of the 3-tool turret tool post.

The rotation of the turret tool post is described by axis 3. This rotary axis activates 2 sensors via different cams. The sensor “FCA1” is active once per turn, while “FCA2” is active 3 times during one full rotation of the turret tool post.

The description of the milling machining station is made up of 2 axes, axis 4 corresponding to the rise/fall, axis 5 to the translation.

Axis 6 represents the feeder used to supply the parts. As there are only 2 sensors placed at the start and end, this is a “standard” axis, and its evolution is controlled by solenoid valves “YV31” and “YV32”.

Axis 7 is, like axis 6, a “standard” axis. It models a mechanical lock used to block the turret tool post during machining. This mechanical locking is achieved by prohibiting the “RTRN” rotation movement of the turret tool post if sensor “FCVE” is active.

The next axes, t1-0 to t5-0, correspond to the modeling of the conveyor belts which move the parts between the stations. They are taken from models in the SIMTSX library : 5 instantiations of the “mcon1p” model.

In these conditions, the equation of the presence of a part on the drilling machine conveyor belt is :

$$pp1 = (dx1 \cdot Mem-pp0 + pp1) \cdot / (fx1 \cdot dx2)$$

The feeder is represented by a “standard” axis. The two sensors at either end of this element are “Fcg0” and “Fcd0”, corresponding to the left and right positions of the feeder respectively.

The variable corresponding to the presence of a part on the feeder is “pp0”. A part may be placed at the entry point if the feeder is in the left position.

The latch term of pp0 will be : fcg0 . part, where “part” is an external variable representing the presence of a part at the entry point.

The part will be available to the drilling machine conveyor belt if the feeder has moved to the right position. This part will then be taken over by the conveyor belt when the axis representing that belt is at “dx7”, thus the equation of the variable Mem-pp0 is :

$$Mem-pp0 = fcd0 \cdot pp0 + Mem-pp0 \cdot / dx1$$

The variable pp0, representing the presence of a part at the entry point, will also be released, provided that the availability of the drilling machine conveyor belt has been memorized, or that the drilling machine conveyor belt is available when the feeder arrives at the right position.

The equation of this variable is thus :

$$pp0 = (fcg0 \cdot part + pp0) \cdot / [(fcd0 + Mem-p0) \cdot dx1]$$

When the parts arrive at the end of the milling machine conveyor belt, which is the last station on the line, they enter an exit zone. The variable corresponding to the presence of a part at the exit point is PP10, for which the equation is :

$$pp4 = (dx4 \cdot fx3 + pp4) \cdot / (fx4 \cdot exit)$$

where “exit” is an external variable used to remove the parts present at the exit point. To inform the control system of the presence of a part on one of the conveyor belts, sensors are placed in front of each of the stations.

Activation of these sensors then appears in axes t1-0, t2-0 and t3-0 which represent the evolution of a part on these conveyor belts. The exit zone sensor is placed at the end of axis t4-0. The sensor corresponding to the presence of a part at the entry point, Fcp0, is described by an equation :

$$Fcp0 = (fcg0 * pp0 + fcp0) \cdot / pp1$$

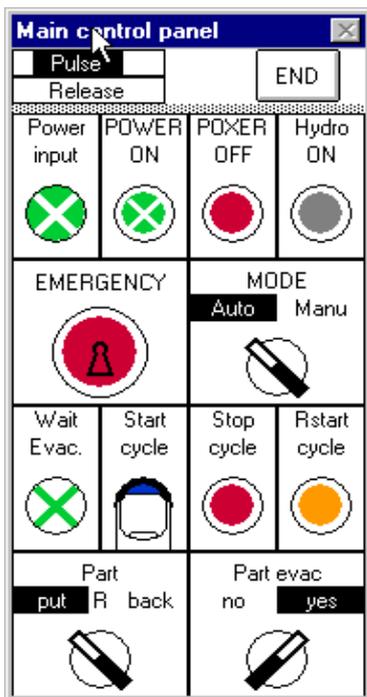
pp0 is the variable representing the presence of a part at the entry point. It is then used logically in the expression of fcp0.

Notes

This description is completed by axis t5-0 which is used to loop the parts moving from the exit to the entry point.

The loop is made when the external variable “Return” is at logic state “1”.

- The operator panel

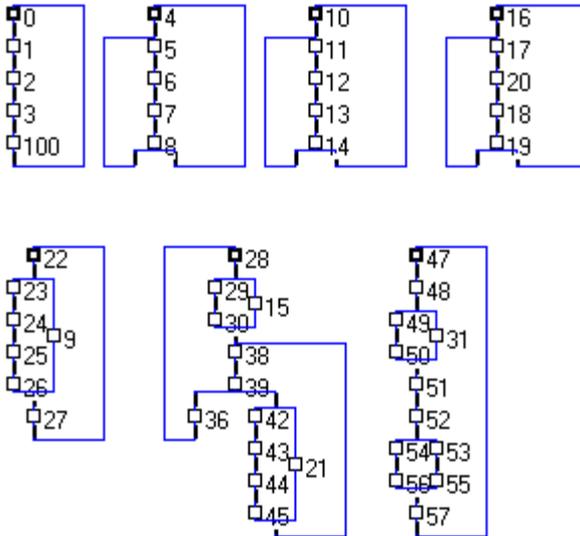


The operator panel is used to control this model : as well as the buttons for switching the voltage/hydraulic power on and off, it has Auto/Man switches, start cycle (required for each machining cycle), stop cycle and reset cycle buttons.

There is also a fictitious “Parts” switch which is used to introduce parts in the machining line and to loop them to the entry point. The “Exitpart” switch is used to remove parts from the line once they have been machined.

• The Grafcet charts

The file MGRAF1 contains Grafcet charts used to animate the model in local mode.



The first 4 Grafcet charts manage the supply of parts and the 3 conveyor belts. The 3 other Grafcet charts each control one of the stations.

Notes

These Grafcet charts known as “Mechanical” only manage simple operating modes.

During local mode simulation with MGRAF1, simply switch on the installation voltage and power, introduce a part at the entry point via the “Parts” switch on the operator panel, then enable the cycle using the “start cycle” button to activate the model.

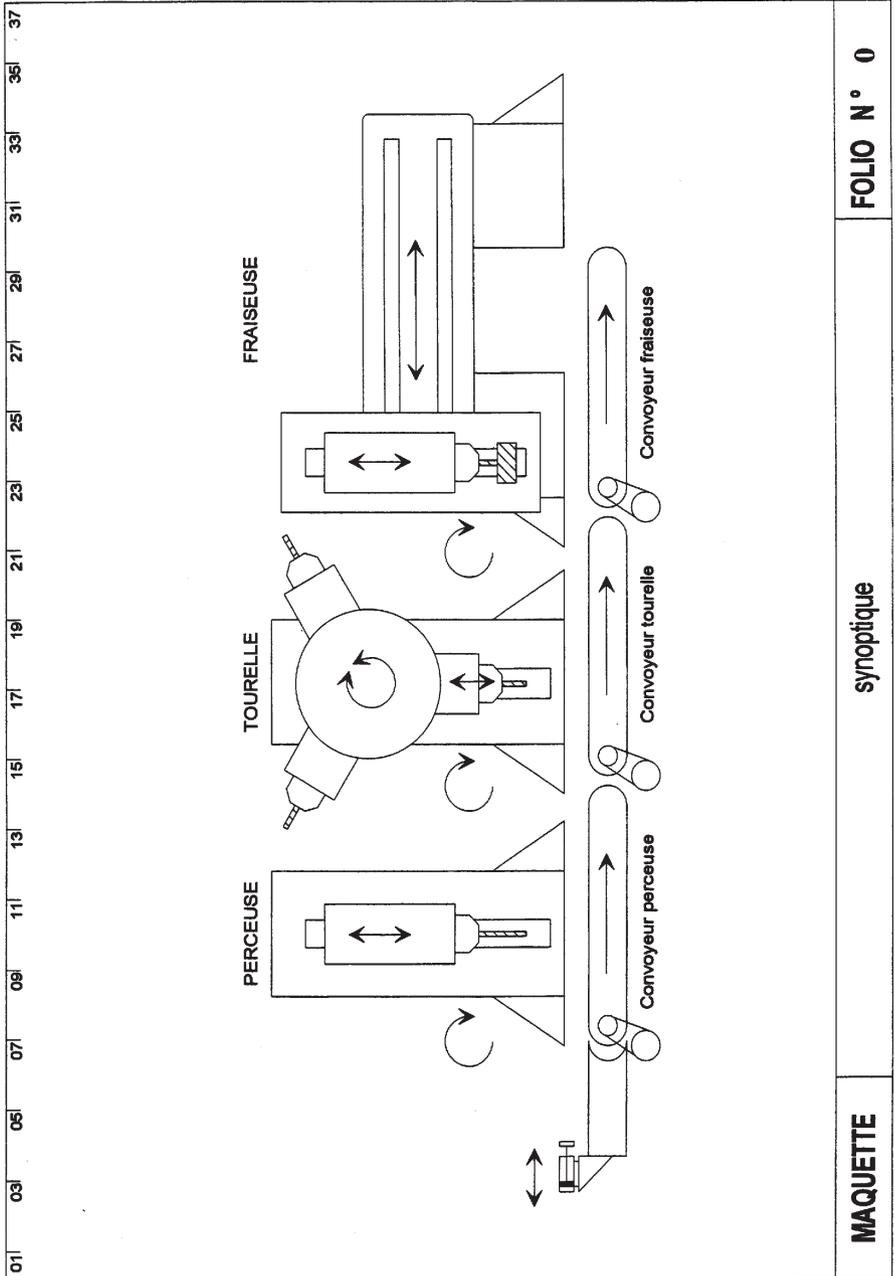
Note that the “init” context can be used to initialize all the variables on installation startup and that 3 operating scenarios are available :

Production : places parts on the installation production line continuously, then removes them once they are machined.

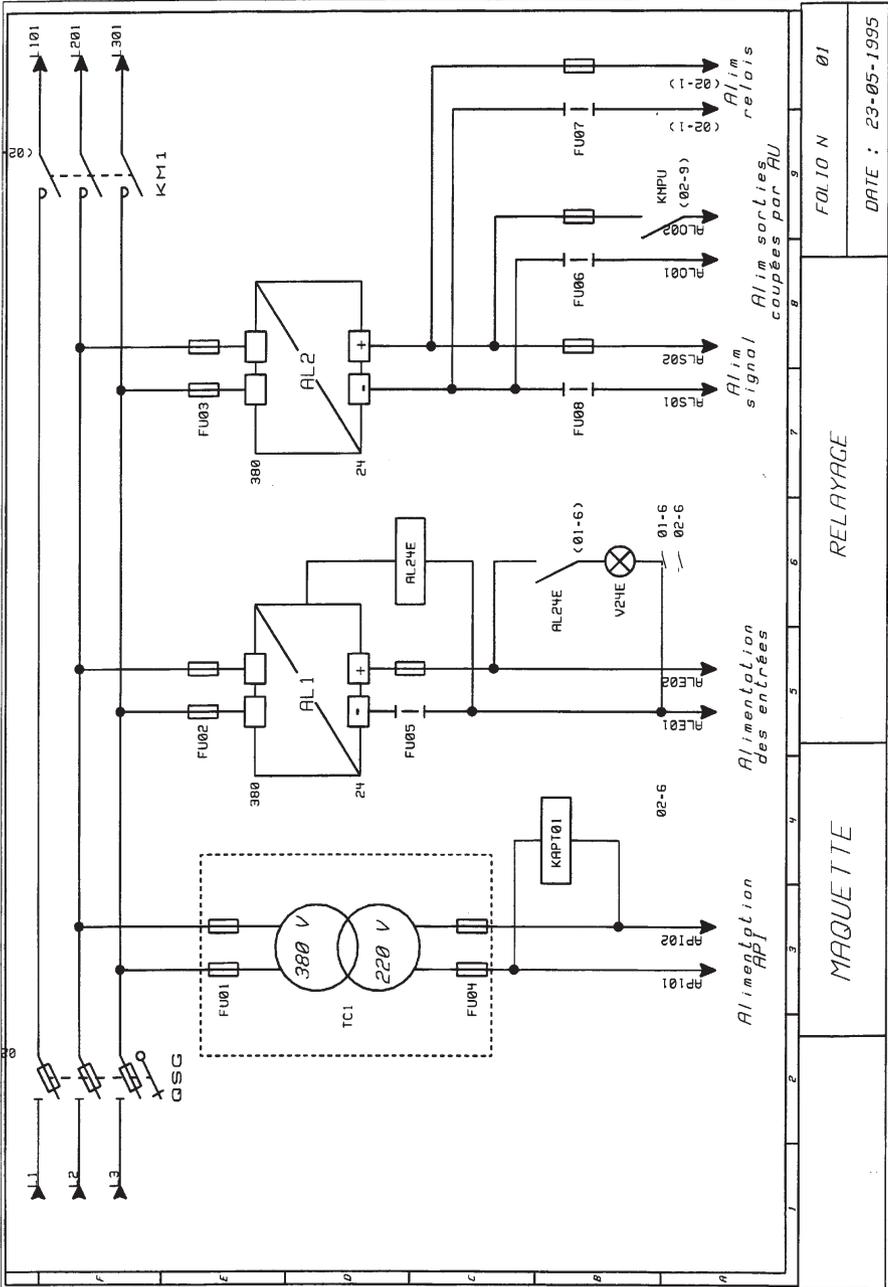
Return-p : machines a part then returns it to the entry point.

1part : machines a part, then waits for action from the operator to remove or return it.

2.2 Electrical and Mechanical Dossier



H

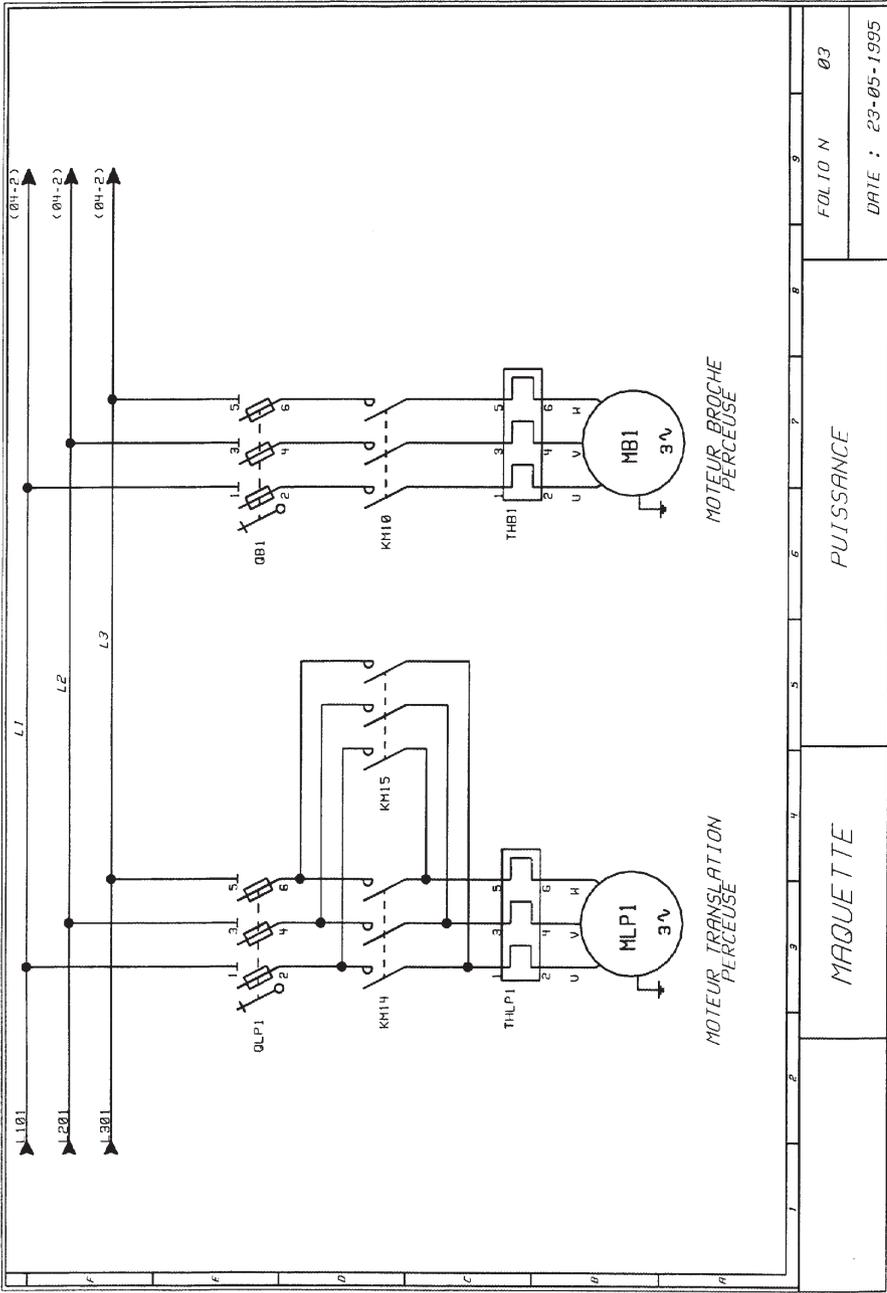


FOLIO N 01

DATE : 23-05-1995

RELAYAGE

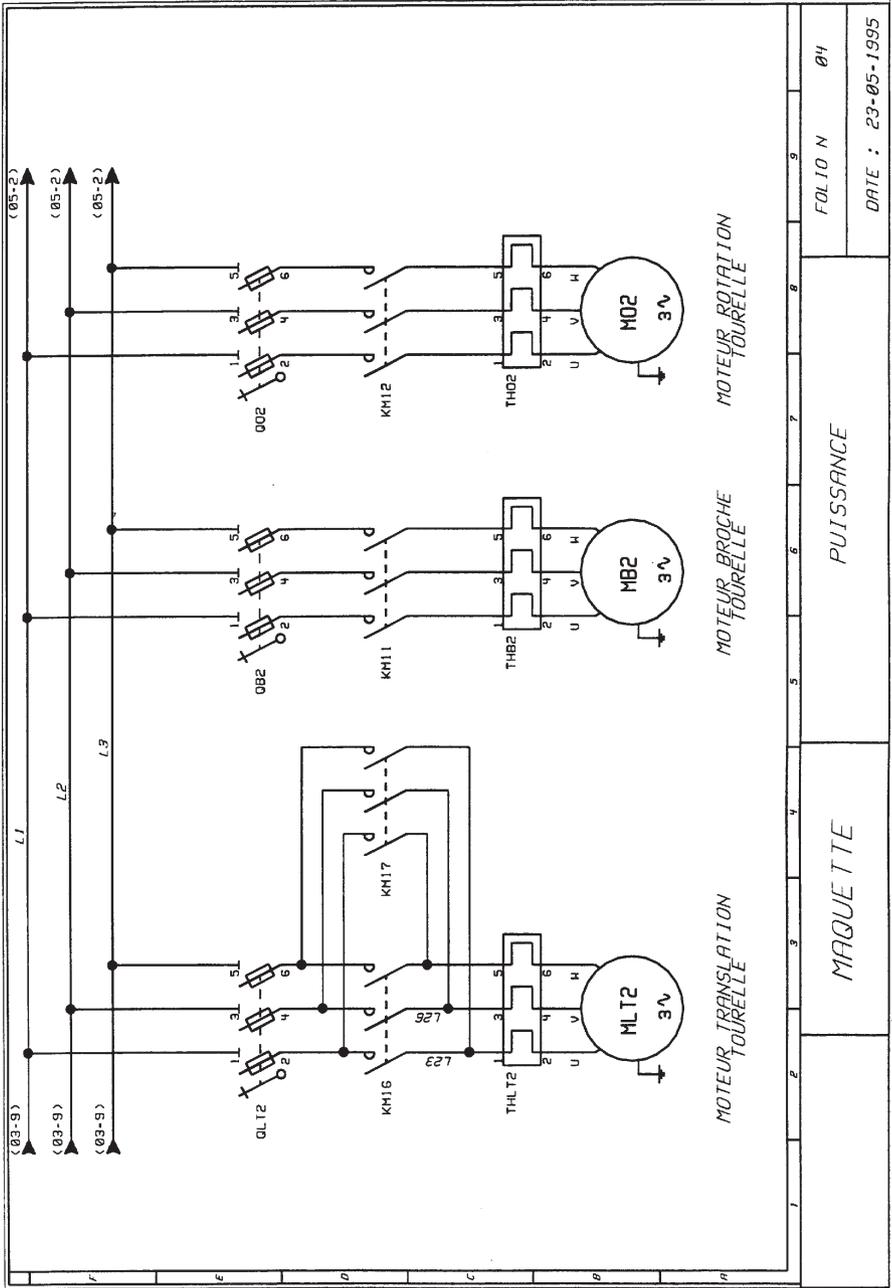
MAQUETTE



FOLIO N 03
DATE : 23-05-1995

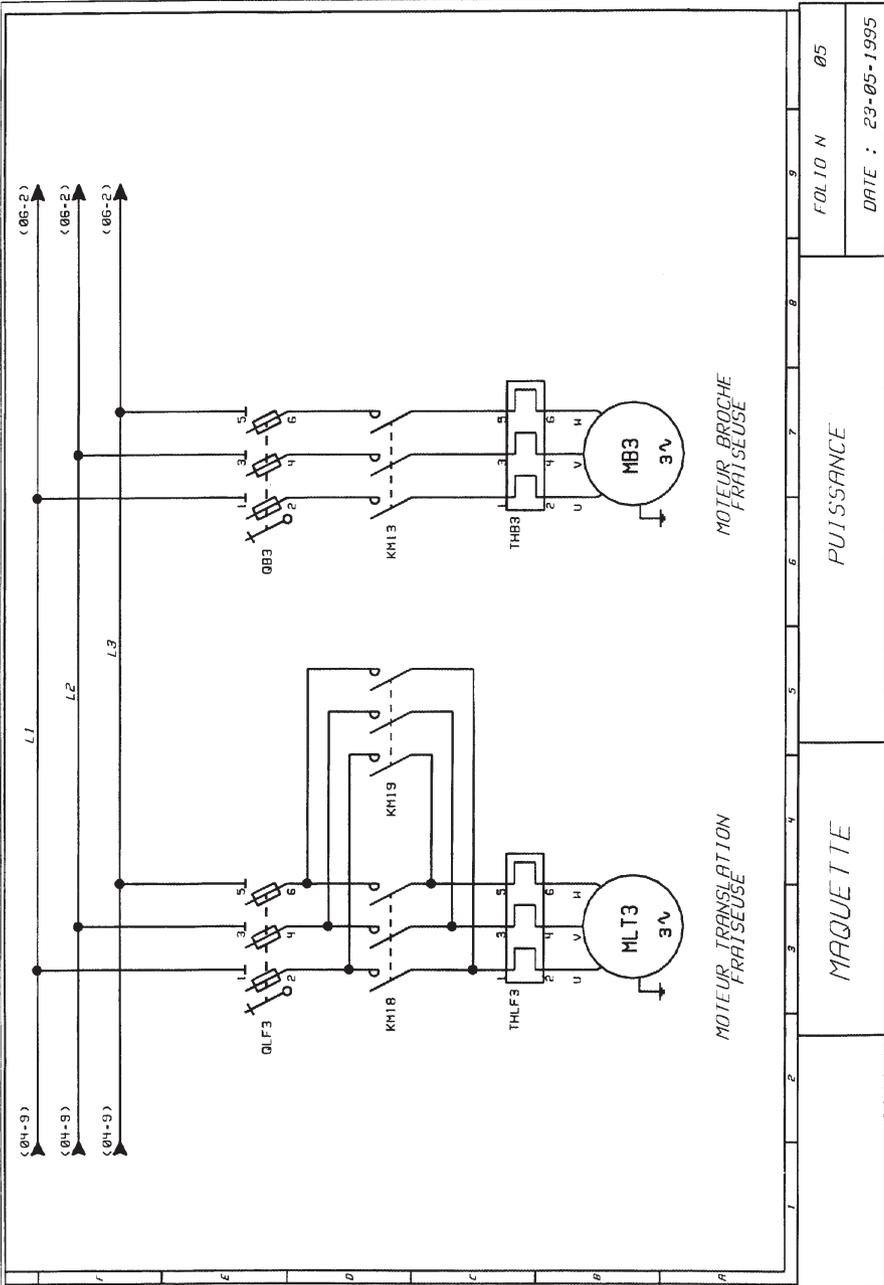
PUISSANCE

MAQUETTE



										FOLIO N 04									
										DATE : 23-05-1995									
										PUISSANCE									
										MAQUETTE									

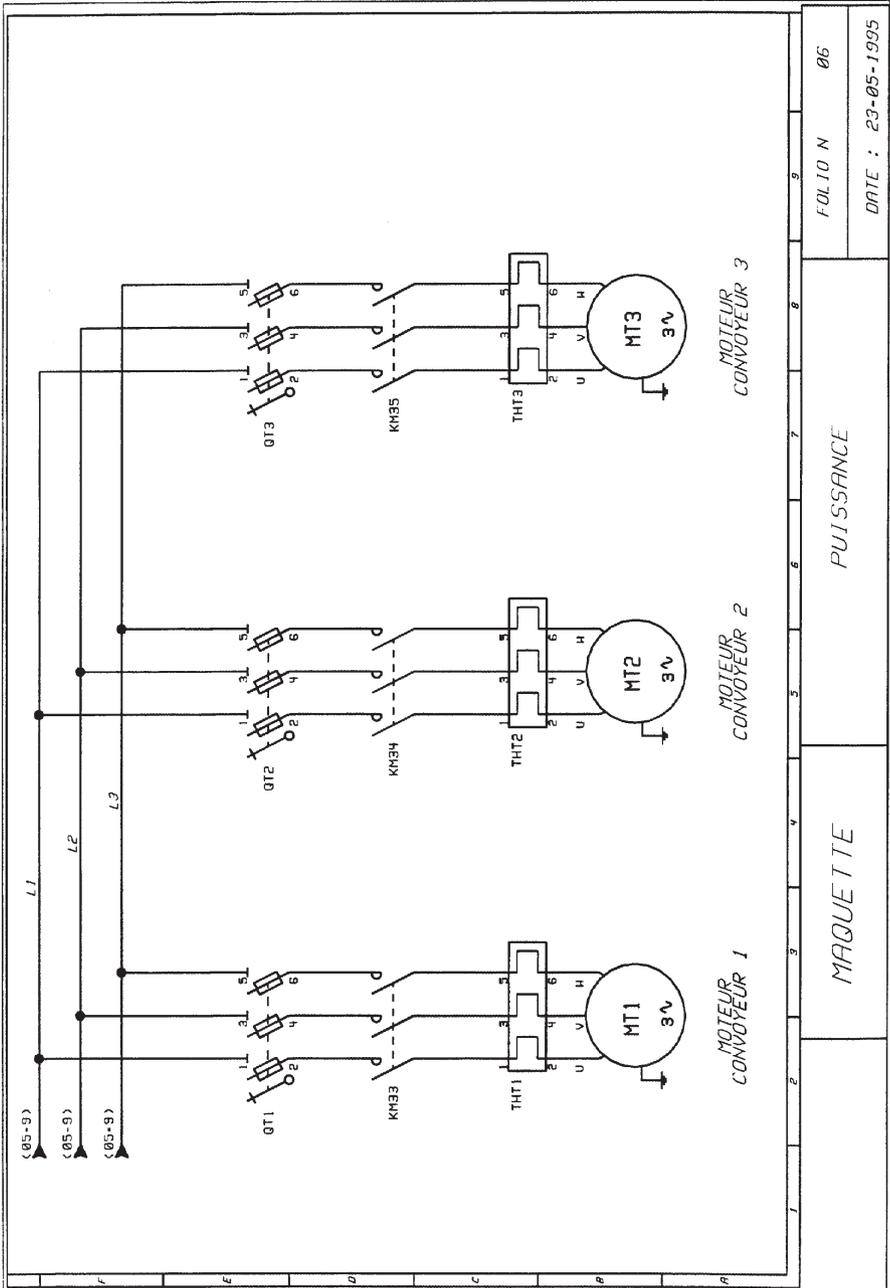




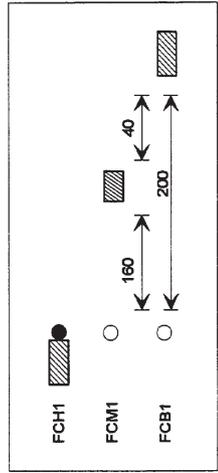
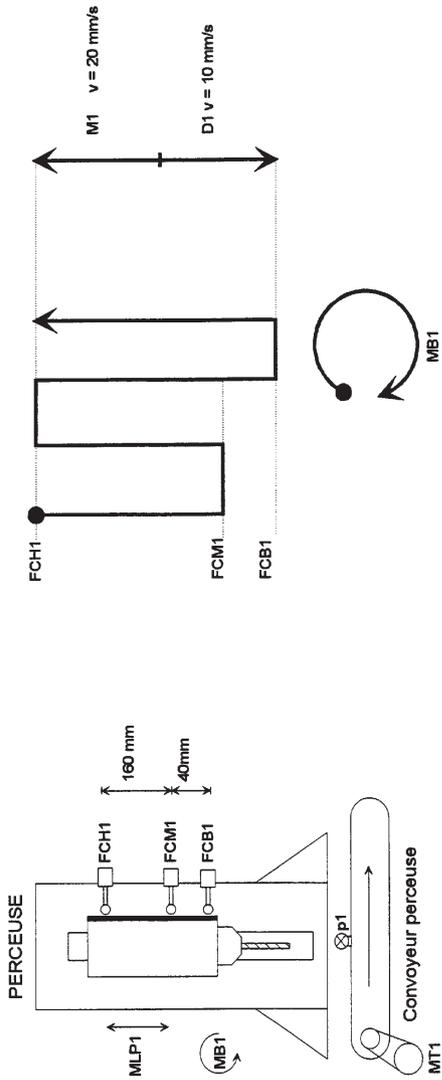
MOTEUR BROCHE
FRAISEUSE

MOTEUR TRANSLATION
FRAISEUSE

1	2	3	4	5	6	7	8	9	
MAQUETTE				PUISSANCE				FOLIO N	Ø5
								DATE : 23-05-1995	



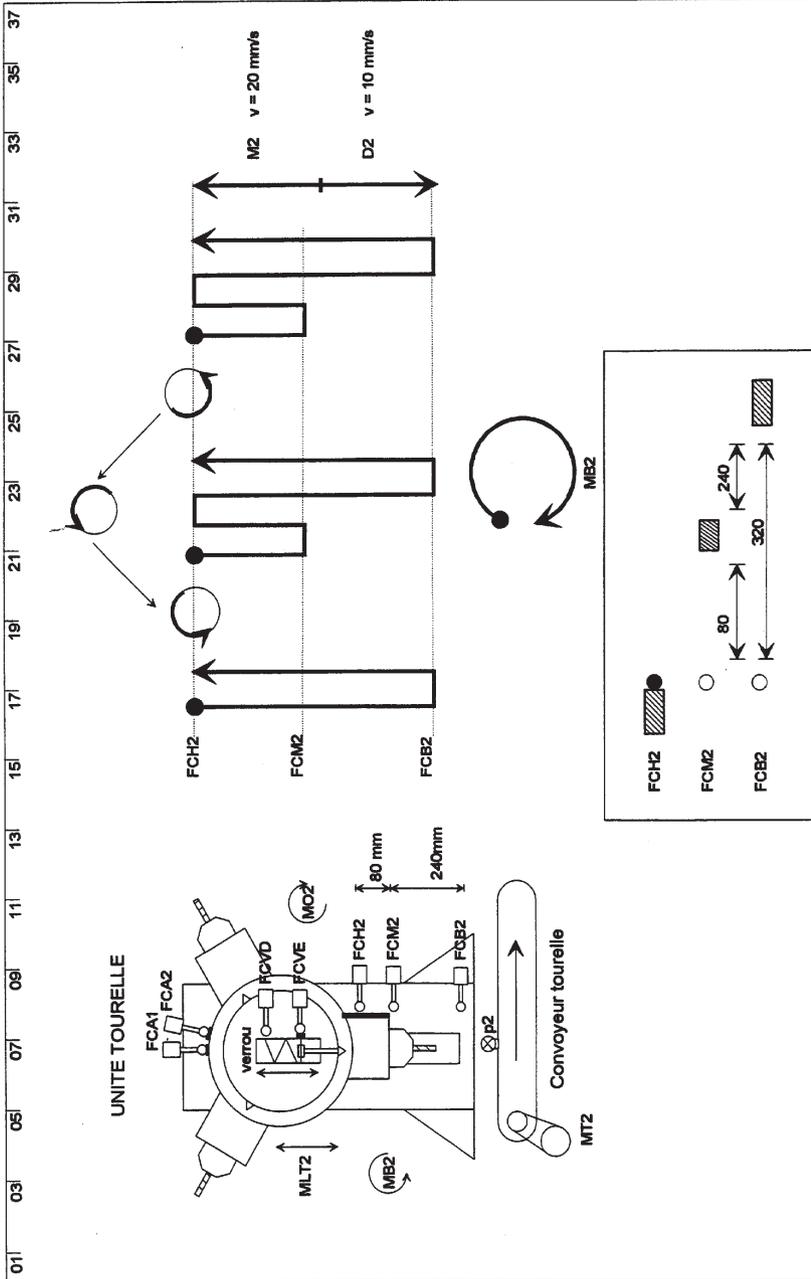
01 03 05 07 09 11 13 15 17 19 21 23 25 27 29 31 33 35 37



MAQUETTE

séquence de perçage

FOLIO N° 13

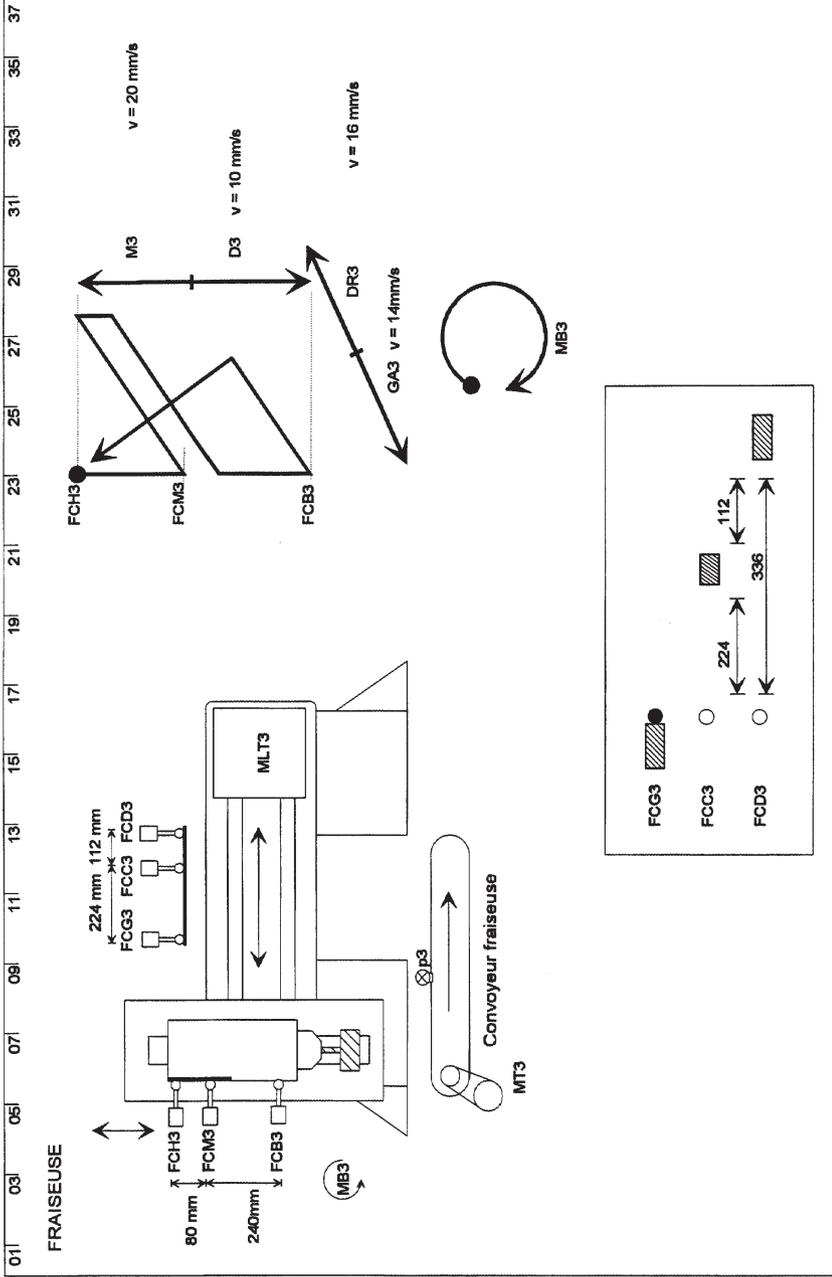


FOLIO N° 14

séquence de unité tourelle

MAQUETTE

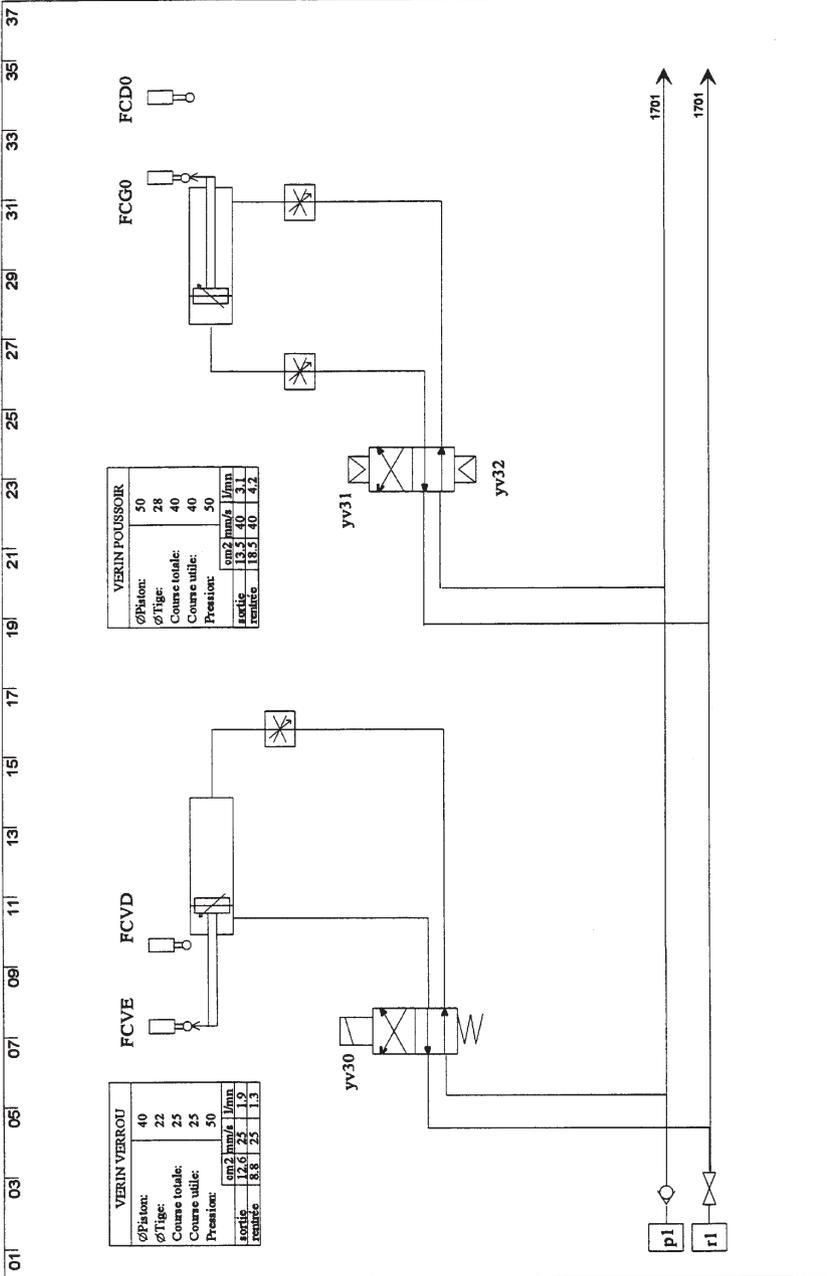




FOLIO N° 15

séquence de fraiseage

MAQUETTE



FOLIO N° 16

hydraulique

MAQUETTE



Section	Page
1 Performance	1/1
1.1 Introduction	1/1
1.2 Effects of Simulation on the Size of the PL7 Application	1/1
1.3 Effect of Simulation on the Scan Time of a Premium PLC	1/1
1.4 Simulation Cycle	1/3
1.5 Simulation of an Application With an Input Configured in Run/Stop	1/4
2 Error Messages	2/1
2.1 Messages Linked With the Environment	2/1
2.2 Messages Linked With Modeling	2/2
2.3 Messages Linked With PLC Exchanges	2/3
2.4 Messages Linked With Printing	2/4
3 "Axis" principle	3/1
3.1 Using SIMTSX	3/1
3.2 Principle for Describing Devices in SIMTSX	3/1
3.2-1 "Axis" Principle	3/1
3.2-2 Describing Wired Systems	3/3

Section	Page
4 Recommendations for Use	4/1
4.1 Simulation by Events	4/1
4.1-1 General Principle	4/1
4.2 Recommendations for Use	4/2
5 Multi-PLC Simulation	5/1
5.1 Multi-PLC Simulation	5/1
6 D.D.E. Links	6/1
6.1 General Information	6/1
6.2 List of D.D.E. Exchanges	6/2
6.3 Information on Simulation	6/4
6.3-1 Simulation Status : Status	6/4
6.3-2 Simulation Mode : Mode	6/4
6.3-3 Simulation Period : mode:period	6/5
6.3-4 Delayed Mode : Realtime	6/5
6.3-5 Output Pulse : Pulse	6/5
6.3-6 Simulation Date : date	6/6

Section	Page
6.4 Information on Description Elements	6/7
6.4-1 Axis	6/7
6.4-2 Flipflop	6/8
6.4-3 External variables	6/9
6.4-4 Input	6/10
6.4-5 Movement	6/11
6.4-6 Numerical	6/12
6.4-7 Output	6/13
6.4-8 Power Supply	6/14
6.4-9 Relay	6/15
6.4-10 Switch	6/16
6.4-11 Trap	6/17
6.4-12 Variable	6/18
6.5 Example of D.D.E. Communication With Excel	6/19
6.6 Advice and Problems Relating to D.D.E.	6/20

Section

Page

1.1 Introduction

To perform a simulation in online mode on Quantum/IEC Simulator, EFBs has to be included into the first section of the Concept project.

The size of the project will be increased by $820+40*(x-1)$ bytes where x means the number of EFBs (1EFB requested per continuous simulated I/Os).

To perform a simulation in online mode on Micro or Premium PLCs, a simulation module must be added to the PL7 configuration of the application.

This logic module can be used to perform exchanges between SIMTSX and the PLCs.

Simulation has the following effects :

- It increases the size of the application.
- It affects the PLC scan time.

1.2 Effects of Simulation on the Size of the PL7 Application

Micro	Delta code	Delta data		
	10.5 KB	1.25 KB		
	Bus X I/O simulation		Bus X and Fipio I/O simulation	
Premium	Delta code	Delta data	Delta code	Delta data
	15 KB	11.6 KB	23.1 KB	20.4 KB

1.3 Effect of Simulation on the Scan Time of a Premium PLC

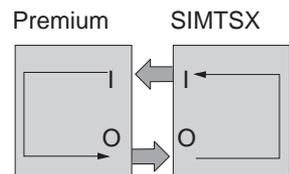
The effect of simulation on the PLC scan time is limited if all of the I/O modules are simulated.

If there is a combination of simulated and non-simulated modules, the effect will be more severe.

The graphs on the following pages give the figures relating to these effects on the PLC scan time.

The context of the measurements is as follows :

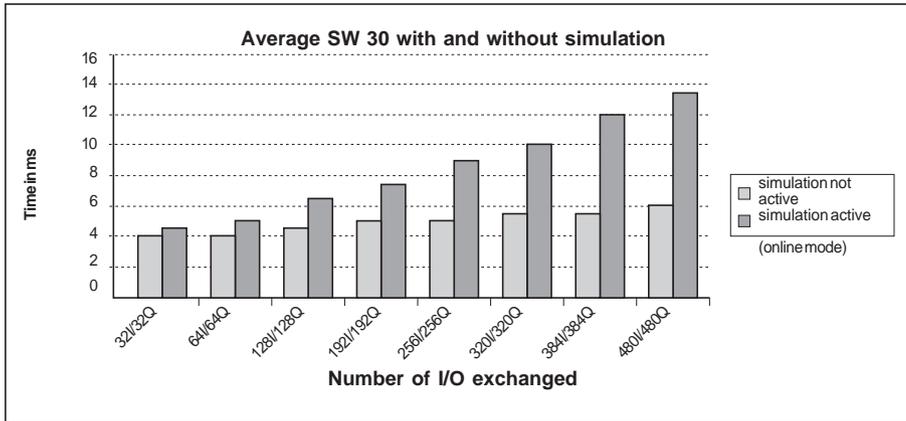
- Systematic feedback of I/O in increasing numbers on all of the configured I/O



- All of the I/O are configured in the MAST task

Measurement principle : comparison of scan time %SW30 with and without the simulation IOB. Remember that the scan time measured using SW30 represents the real execution time for the PLC scan and not the time configured during the creation of the application.

The results obtained are as follows :



In the graph above, it should be noted that the average scan time without simulation has been measured while the configured I/O were not really present in the PLC racks. This means that the circulation time on the bus for updating I/O modules has not been taken into account. In reality, the scan time will be greater than that measured in this test.

- To ascertain the real context of the simulation by event, 5% of the inputs and 10% of the outputs configured are fed back systematically.

The measurement principle remains the same. It appears that the effect on the scan time is the same as in the previous case. However, the simulation cycle time is much faster considering the number of I/O exchanged in each cycle.

Recommendation :

During simulation in online mode, the simulation module slightly disturbs the PLC scan time. In some cases, this could lead to a watchdog overflow and thus a PLC block. In this case, the scan time(s) (MAST, FAST) must be increased to prevent this overflow. To diagnose a watchdog overflow, first reset the PLC and then simply check in PL7 that system word %sw124 equals 128 (80H). The scan time for the configured tasks must be increased to prevent this trip.

1.4 Simulation Cycle

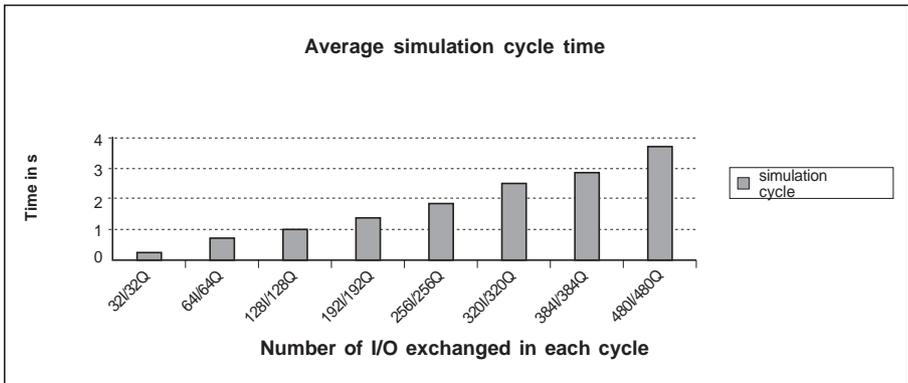
The simulation cycle time includes :

- the data exchange time between SIMTSX and the PLC
- the data processing time in the PLC
- the data processing time in SIMTSX (e.g. evolution of the installation)

The graph below provides an indication of the simulation cycle time, depending on the number of I/O exchanged in each cycle (UNI-TELWAY connection).

Note :

The simulation cycle time depends mainly on the number of data requests between SIMTSX and the PLC.



The number of requests can be evaluated using the following rule :

(Number of discrete I/O evolving in a simulation cycle x 2

+

Number of numeric I/O evolving in a simulation cycle x 4)

/ 116

= Number of requests

In most applications, 2 requests are exchanged per simulation cycle. The cycle time is about 300 ms.

1.5 Simulation of an Application With an Input Configured in Run/Stop

In this case, simulation in online mode is only possible if the input can be forced to 1 or to 0 in PL7. If not, simulation in online mode is not possible unless this input is deconfigured. In fact, during simulation in online mode, SIMTSX requests a PLC STOP at the start of the connection and a PLC RUN when the connection has been made. SIMTSX has a window for managing these PLC states. If an input is configured as Run/Stop, then SIMTSX is unable to change the PLC state. If the state cannot be changed by PL7, simulation is impossible if the PLC is not in Run.

2.1 Messages Linked With the Environment

These messages concern the availability of resources required for the correct operation of SIMTSX.

- **"Windows resource problem - close windows"**
 - Windows windows are required to display the operator panels and views, particularly because of the number of elements represented in the views. When a request is made to display operator panels or views, a test of the available resources is made. If resources are low - this can be checked in the "About" window in Program Manager - this message is displayed. You should then close one or more of the views or operator panels displayed on screen.
- **"Copy (or Save or Load) failed"**
 - These messages may appear when using tools for copying an application or model, transferring an application to a model and saving or loading an application or model to - or from - a floppy disk. The most likely cause of this is a lack of space available on the hard disk or the floppy disk during a save operation. The disk format may also be incorrect.

2.2 Messages Linked With Modeling

These messages may appear during online or offline simulation.

Their origin stems from an anomaly in the modeling process, preventing the continuation of simulation, which is then interrupted.

- **"Unknown state of variables"**

- This message appears when SIMTSX is not in a position to determine the state of the description variables or relays. A simple example is the following equation :

$$a = /a$$

This phenomena appears, for example, if a Reset/Set switch is described in SIMTSX using two logic equations and they activate the Set and Reset inputs at the same time!

This anomaly can be resolved by correcting the expressions of the variables in question or by reestablishing a stable state using one of the terms in the equations.

- **"Simultaneous activities for "axis : .."**

- This message signals an anomaly due to the non exclusive nature of movement equations associated with a single axis. In this case, you need to modify the movement equations concerned so that they differ.

- **"Infinite evolution of :"**

- This message appears when a loop occurs in a rotary axis changing at an infinite speed. The problem is that the equation is never invalidated during the movement. This equation needs to be corrected so that the movement to which an infinite speed has been given can be deactivated by the change in state of one of the sensors in the axis.

- **"Chart instability"**

- This message signals that a "stationary situation" appears in a Grafcet chart. This means that a closed cycle of transitions for which the conditions are simultaneously true has occurred. Some transitions in the loop therefore need to be locked. Another possibility is that there is a counter overflow : this concerns an anomaly in an arithmetic operation - which should be replaced by an incrementation - made in a step.

2.3 Messages Linked With PLC Exchanges

The various error messages linked with exchanges between SIMTSX and the PLC and their causes are explained below.

- **"Connection Failed"**

- SIMTSX has not succeeded in launching the communication server.

- **"Connection Error"**

On Micro/Premium, this message is accompanied by other additional messages from the communication server. The list of additional messages is explained below :

- **"The simulation module in PL7 does not allow the simulation of the FIP I/Os".**
- **"Check the drivers used are loaded and the PLC(s) is (are) connected".**
- **"Check the SIMULATION module is in the PL7 program. Make sure the SIMULATION module is the same slot in PL7 and SIMTSX".**
- **"The SIMULATION module is missing from the SIMTSX configuration".**
- **"Non Simulated Modules Missing : Rack i, Pos. j"** : this means that the I/O modules which have not been declared in the SIMTSX configuration (non simulated modules) but which have been declared in the PL7 configuration are not present in the PLC rack.
- **"SIMTSX Configuration <> PLC Configuration : Rack i, Pos. j"** : this means that for the slots marked, the SIMTSX configuration does not correspond to the configuration described in the PLC program.

2.4 Messages Linked With Printing

The various error messages linked with printing and their causes are explained below.

- **"Cannot open the application dossier".**
 - SIMTSX has not succeeded in opening the application dossier requested. This message appears when the software cannot read the information contained in the application description files. If this message appears, contact your technical support center.
- **"Cannot open the model".**
 - SIMTSX has not succeeded in opening the model requested. This message appears when the software cannot read the information contained in the model description files. If this message appears, contact your technical support center.
- **"Incompatibility between the print format, the application dossier and the printer characteristics. SIMTSX cannot print your application dossier ! PLEASE CHANGE YOUR PRINT FORMAT..."**
 - This message appears when SIMTSX cannot print the document with the print format requested. This happens when one of the sections in the document is too large to be printed on a single page. To correct this problem, select a smaller character font so that all sections can fit on one page.
- **"Not enough memory ! Close unused applications to free the memory."**
 - When too many WINDOWS applications are open at the same time, the computer does not have enough free memory to execute SIMTSX correctly. Close as many applications as possible to free some memory when this message appears.
- **"Another SIMPRINT instance has been launched ! Close this first before using the print tool"**
 - This message appears when the print tool is opened when the print application is already active. To correct this problem, close the active print application, then use the new print tool.
- **"Could not start print job"**
 - This message appears when a WINDOWS print error occurs. To correct this problem, refer to the WINDOWS user manual.
- **"The *.dib files are too big for SIMPRINT !" and "Cannot open *.dib file!"**
 - These messages appear when SIMTSX encounters problems printing operator panels and views. Contact your technical support center to correct this problem.

3.1 Using SIMTSX

The methodology for using **SIMTSX** can be summarized in three steps :

- Describe the devices in **SIMTSX** to form the simulation model.
- Validate the model obtained by the first level of simulation in "offline mode", i.e. not connected to the PLC.
- Once the simulation model has been validated, connect **SIMTSX** to the PLC and let the simulation scenarios proceed in "online mode", according to the desired purpose.

3.2 Principle for Describing Devices in SIMTSX

The general modeling principle in **SIMTSX** is to describe the devices as objectively as possible, by following the mechanical, pneumatic and electromechanical design as closely as possible.

It is never a question of programming the expected behavior using a particular computing or control system language, which would constitute a subjective description taking precedence over normal operation.

A new and specific description method has therefore been introduced, based on the "Axis" concept (see part D, section 1).

An axis represents a basic evolution possible in an automated system. An automated system can therefore be described by a group of axes.

3.2-1 "Axis" Principle

When an automated system is operating, the PLC controls and monitors the evolutions, whether they are :

- mechanical movements : forward/backward movement of a cylinder, opening of a valve, movement of a load on an automated transfer system, etc.
- evolutions of physical measurements : temperature, pressure, flow, level of a product in a tank, etc.

With this in mind, the device designer has provided the actuators and preactuators required for control, and the detection methods (sensors, cells, etc.) required for monitoring.

The principle of **SIMTSX** is thus to create an "Axis" object for each of these devices or basic measurements which is capable of reproducing the associated evolutions.

An "axis" is characterized by :

- its length
- sensors
- one or more movements :
 - direction
 - speed
 - activation condition

More details of the characteristics of an "axis" :

- The **length** corresponds to the amplitude of the possible evolutions : for example, a cylinder is described by an axis with a length which corresponds to the rod travel.
- The **associated sensors** (for example, start and end limit sensors) are placed at a given position on the axis, and are described by their logical behavior :
 - Normally open, or positive logic sensors : their logic state changes to 1 when the mechanical element which is moving is positioned on the sensor.
 - Normally closed, or negative logic sensors : their logic state changes to 0 when the mechanical element which is moving is positioned on the sensor.
 - "Latch" sensors change to state "1" when the physical measurement which they take into account is greater than the value given by their position on the axis.
 - The operation of "release" sensors is the opposite of that of latch sensors.

Note

It is however sometimes necessary to follow the evolutions of a measurement "continuously", and not just using the evolution of sensors connected to "discrete" inputs on the PLC (for example, monitoring the position of a moving part using an encoder, monitoring the weight of a product on an analog input on the PLC, etc).

In this case, the axis created in **SIMTSX** can be declared as a "sampled" axis with which a "**step**" value must be associated : it is a question of the precision with which **SIMTSX** makes the measurement evolve (for example, level of a product in a tank, length of the axis : 200 cm, steps of 10 cm). To reduce **SIMTSX** processing, the number of steps on a sampled axis must be kept to a minimum.

-
- The **movements** associated with the axis describe the evolutions of the physical measurement in terms of :
 - **direction** of evolution
 - **speed** : this is a constant value, or a numeric variable which – simply described by a calculation table – can be used to reproduce specific evolution speeds on the axis (acceleration ramp, speed reference from the PLC, etc.)
 - **activation conditions** : this is a Boolean equation which groups together the conditions required for movement. This is often used to indicate the presence of energy and the control preactuator (for example, pilot control solenoid valve of the directional control valve associated with a cylinder).

Several movements can be described on one axis, but only one of them can be activated at any given time during simulation : if the equations associated with two different movements are not exclusive, or if the PLC controls two conflicting outputs, a message is sent to the user.

3.2-2 Describing Wired Systems

The axis principle can be used to describe any "preactuator – actuator – sensor" subassembly in the installation and, by extension, any measurement which is able to evolve.

As the interface between the PLC and **SIMTSX** is at I/O level, links must be established between :

- the control outputs and the preactuators
- the sensors and the PLC inputs

These links are described in **SIMTSX** by means of Boolean equations which make use of intermediate electrical components (contactors, supplies, isolating switches, etc.) and – by extension – all of the wired systems in the control cabinet (notably for the management of safety controls, as well as local control operator panels).

This description is based directly on the electrical diagrams for the installation. The description increases the test possibilities on the **SIMTSX** platform because it is then possible to act and to position faults on all of the wired devices.

It should be noted that :

- Boolean equations can also be used to describe the behavior of variables, to reproduce for example the evolution logic of "part present" on automated transfer systems.
- Boolean equations on relays and description variables are evaluated "in parallel" in **SIMTSX**, ie. a variable evolution affects the equations in which it is used synchronously : changes in state are propagated by "layers" of successive equations.
- The switching times for relays are considered negligible in relation to axis evolutions : the equations representing relays are thus evaluated at a set simulation time, and until a stable state is obtained for these elements.
- It is possible to use "edges" of variables in these equations (designated by the prefix "+" or "-" in front of the name of the variable).

4.1 Simulation by Events

In contrast to the continuous processes of the chemical industry for example, logic and sequential aspects dominate the operation of systems controlled by PLCs.

In this respect, the evolution of a control system can be described as a series of "discrete" events.

Specifically, events are changes in the state of the I/O which ensure communication between the control system and the "application" of the PLC system.

4.1-1 General Principle

Once connected to the PLC, **SIMTSX** is capable of animating the model and reproducing its evolution dynamics using the I/O declaration, description of wired systems (relays, safety, etc) and axes.

Therefore :

- Knowing the state of the command outputs at each instant, it calculates movement activity associated with the axes.
- It is therefore capable of projecting over time and sequencing all future events which will occur in the installation.
- This event prediction, which can also be questioned by the operator at each instant (by pressing a pushbutton, setting a fault, etc) enables **SIMTSX** to act on the PLC inputs while ensuring that the chronological order of real events is maintained.
- Sending a date-stamped event on a PLC input advances the simulation date (example : if only one cylinder is exiting and the end limit sensor appears after 30 seconds (depending on the model), sending an input to the PLC "ages" the simulated machine by 30 seconds, irrespective of the "true" time at which **SIMTSX** sends the data).
- As **SIMTSX** is generally faster than the real installation, the "time-delay" mode is used to time the transmission of events whilst maintaining simulation times (example : **SIMTSX** waits 30 seconds before sending "end limit of the cylinder" data, although it already has this data a few milliseconds after having detected the cylinder output order).

4.2 Recommendations for Use

Exchange of I/O between SIMTSX and the PLC

The principle of Simulation by Events applied in **SIMTSX** consists of predicting the evolution of inputs according to the state of the outputs. Outputs received from the PLC are stabilized in order to :

- Retain events caused by the evolutions of these outputs on the installation.
- Avoid propagating the transient evolutions of an output in the model of the installation. This would not correspond to reality due to the inertia of the actuators.
- SIMTSX-PLC communication in offline mode simulation

SIMTSX offers the possibility of checking the model of the machine to be simulated by controlling the machine with the aid of a mechanical Grafcet chart : this is known as "Simulation in offline mode".

This simulation is performed in line with the Grafcet standard. In particular, the Grafcet chart is executed in a manner which seeks stability on the change of state of an input. Only the final, and therefore stable, state of the outputs is perceived by the model of the simulated machine.

- I/O exchange with a PLC

In order for simulation to work correctly, the prediction of input evolutions must depend on a stable output state.

Communication then takes place in the following manner : each time the simulator sends inputs to the function block, the latter supplies them to the PLC and receives output evolutions, resulting from the PLC reaction, for a time period equal to several scans.

SIMTSX sends inputs to the PLC for each simulation cycle. The PLC then applies them to the program on several PLC scans. The PLC memorizes the resulting output evolutions for each program processing.

The time corresponding to the processing of several PLC scans with the same input values is used to present a stable output state. The outputs received are sent to the simulator which can then continue its processing. If the outputs continue to evolve, they are stored by the PLC. The PLC (new simulation cycle) will then supply them to the simulator when this sends the changes in state of the next inputs.

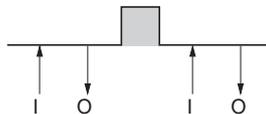
- Acceptance of PLC time delays

Communication between the simulator and the PLC is based on the hypothesis that when the PLC perceives a change in the input state, the outputs that it will cause to evolve reach a stable state after several program execution scans.

However the use of time delays in the PLC program may cause this hypothesis to fail.

Time delays can be used in different ways :

- Some are used to carry out "open loop" actions, which do not generate any feedback from the application. This is the case for auxiliary functions such as the flashing of LEDs. These time delays cause no communication problems, although the period of flashing must not be too short to ensure that the active/inactive state of the output involved can be correctly perceived.
- Time delays can also be used to generate output pulses of a sufficient duration to overcome the inertia of a preactuator such as a directional control valve. The transmission of these pulsed commands can also be delayed in relation to the reception of inputs by the PLC. The problem is therefore to prevent the outputs from being filtered by the communication process between the simulator and the PLC.



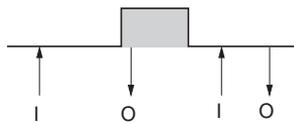
Delayed pulse not perceived by the simulator

In effect, if the exchange of I/O takes place in a way that the rising and falling edges of a pulsed output are sent by the PLC in the same group of outputs, the temporary change to the active state of the output concerned will not be taken into account and will therefore not have the required effect on the simulated application.

To solve this problem, the pulse must not be produced between two successive I/O exchanges, which can sometimes happen if the change to 1 of the output is delayed in relation to the reception of inputs.

The duration of the pulse must therefore be greater than the simulation cycle time (interval of time separating two I/O exchanges).

Since this time depends on the processing carried out by the simulator and can therefore vary greatly, it is wiser to eliminate any delay between the reception of inputs and transmission of outputs by modifying the time delays in the PLC program.



Adaptation for correct operation

- Finally, time delays can also be used to monitor installation evolutions. They are used as "watchdogs" for movements or part of a cycle, enabling, for example, the detection of a sensor sticking at 0 or an actuator blocking.

Although generally simulation is "speeded up" in relation to the actual machine, for certain rapid movements, the simulator can sometimes be slower than reality. It is better therefore to initially increase the value of these "watchdog" time delays. The sequential part of the animation program can thus be validated independently of the time-based aspects of the monitoring. The correct operation of monitoring time delays can then be checked by adjusting their value according to the actual response time of the simulator.

Analog simulation and process control

The principle of sampled axes can be used to simulate evolutions of certain measurements "continuously", such as the mechanical position of a moving part, level in a tank, weight of a product, physical measurements of temperature, pressure, etc.

The principle of simulation is still that of simulation by events : any change of step on an axis is classed as a change of sensor state type event.

It is thus clear that, for performance reasons, sampled axes must be used with caution. It is particularly advisable to **limit the number of steps on an axis as much as possible**, in each case taking into consideration the **real need** of the systems engineer responsible for debugging the application.

Moreover, sampled axes and numeric variables can be used to reproduce the behavior of certain physical measurements, after preliminary analysis. In the case of a "2nd order" measurement, it is possible, for example, to create an axis simulating the variable for which the speed of evolution will itself be given by the position of a second axis corresponding to the "derivative" of the variable! On the second axis, the corresponding speed is therefore the "second derivative".

For modeling via a transfer function (the case for linear systems), transforms (especially the "Z transform") are used to obtain simple equations (to transcribe to the assignment tables) calculating the successive samples of a variable. A rotary axis can therefore generate "0 markers" corresponding to the sampling period.

As **SIMTSX** is still based on the principle of simulation by events, and not on the principle of realtime simulation, these modeling methods should be used with great care, simply because it is impossible to guarantee a sampling frequency for the calculated variables in all cases. Consequently, it is not possible, in principle, to use **SIMTSX** to test the behavior of control loops on a test bed.

Experience also shows that the gains made are not in relation to the efforts necessary for modeling since :

- On the one hand, it is often difficult to obtain enough information for detailed modeling.
- On the other hand, loop adjustments must always be made on site with the actual process.

As for the selection of a step on a sampled axis, the **SIMTSX** user must analyze the real need for tuning on a test bed in order to define the required, sufficient, level of modeling for the physical measurements.

SIMTSX limitations

SIMTSX is used for debugging the application for all types of Micro, Premium and Quantum PLC architectures.

The specific limits of SIMTSX are :

- Number of Internal Words (%MW) which can be simulated : 128 in write mode and 128 in read mode
- Number of analog channels of Fipio devices which can be simulated : ≤ 1920
- Restriction on the debugging of an application containing PL7 event-triggered processing

For Micro :

- Number of Internal Words (%MW) which can be simulated : 32 in write mode and 32 in read mode

For Micro (software version 3.0 only) :

- If more than 58 outputs evolve simultaneously in a simulation cycle, certain output evolutions may not be taken into account by the simulator.

Simulation with the I/O declared in a PL7 task but used in another task

If an input/output is configured in a PL7 task but used in a different task in the program, the value of this input/output may not be compatible with the simulation. It is therefore advisable to use the I/O in the tasks in which they are configured.

5.1 Multi-PLC Simulation

Multi-PLC simulation consists of developing a single **SIMTSX** application controlled by a control architecture distributed over several PLCs. It should be used when :

- the programs on each PLC are extremely interdependent (COM word exchanges, redundancy management, production flow control strategy, etc)
- "level 2" functions, which interact with PLCs and are also to be validated, exist in the architecture (supervision, production management),
- there is a high degree of interaction between the application zones controlled by various PLCs (production flows, handling systems, etc)

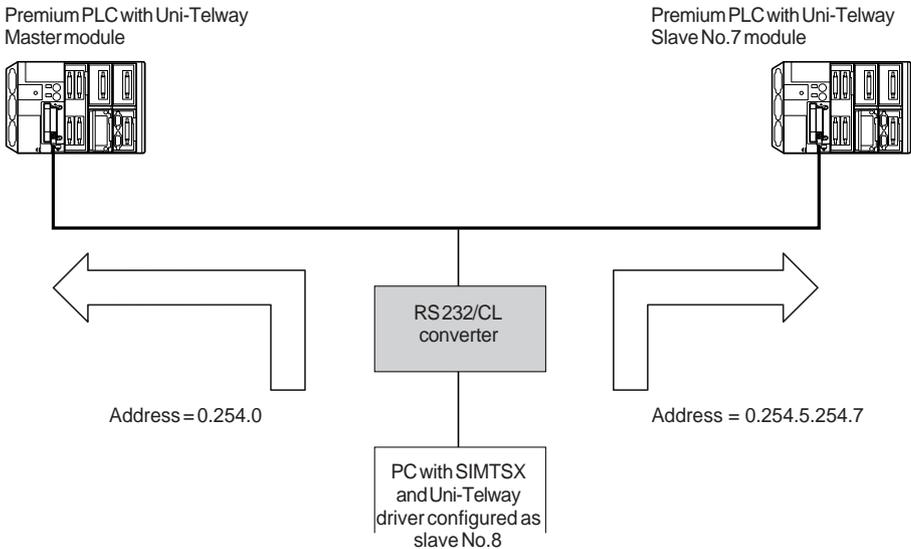
On a methodological level, it is highly advisable to :

- begin by performing single-PLC simulation, on one PLC at a time
- then use multi-PLC simulation for other specific tests, linked to the architecture, which will have already been defined

This structured approach must be followed so that the user is not confronted by a multitude of highly diverse non-conformances on starting multi-PLC simulation and also to ensure that the multi-PLC tests are sufficiently exhaustive (ie. offer specific scenarios depending on the tests being carried out).

Multi-PLC simulation on Uni-Telway bus

To perform a multi-PLC simulation via a Uni-Telway fieldbus, see the diagram below :



The Uni-Telway driver on the SIMTSX simulation station should be configured as a slave (default configuration) with the required slave number (Base = Slave no., Number = number of slaves to which the PC must respond from the Base). On the SIMTSX parameter page, select the Uni-Telway driver for the two PLCs and enter the address 0.254.0 to simulate on the master PLC and the address 0.254.5.254.i to access slave PLC No.i (No.7 in example above).

Note

To simulate a SIMTSX application, using the terminal port on the master PLC, the addressing of the slave PLC in SIMTSX is as follows :
0.254.5.<RackModule>.<ChannelAd0>

- The <RackModule> field describes the master PLC. It must be filled in as follows :
(Master rack number*16) + Master module number.
- The <ChannelAd0> field describes the slave PLC. It must be filled in as follows :
(Master channel number*100) + Slave Ad0 number.

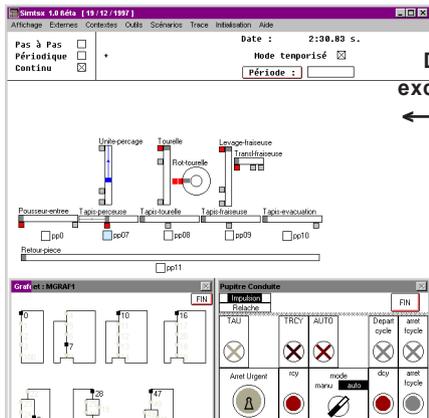
6.1 General Information

Dynamic data exchange (D.D.E.) is an inter-process communication mode which is used to exchange data between programs. This involves transferring information, or even informing a program of the data changes in another program.

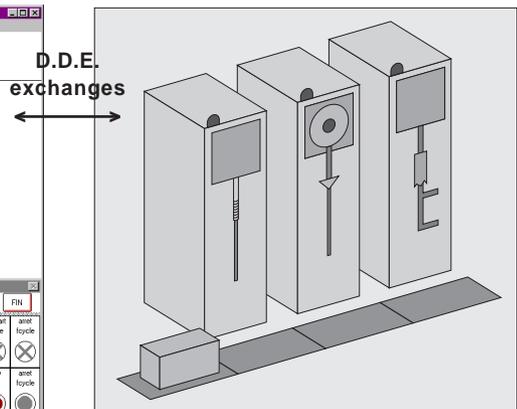
SIMTSX is a "D.D.E. server". This means that it is possible for any "D.D.E. client" application to know the state of the SIMTSX variables, to follow their evolutions, and to change their values. The SIMTSX D.D.E. server is available for client WINDOWS programs **only when the main simulation window is activated** (simulation in offline mode or online simulation with the PLC).

The name of the subject for which a link can be opened is the name of the application on which the simulation is performed (eg : Maqtsx). The syntax of the subject name is the same as for the SIMTSX application names ; the first letter in the application name must be in upper case with the rest in lower case.

SIMTSX simulation environment



Example of monitoring images



There are three types of request on this data which are understood by the SIMTSX D.D.E. server :

- read the value of the data item (READ)
- receive information on each modification to the data item (INFORMATION)
- change the value of the data item (WRITE)

Depending on the data, only one or some of these requests are available. For each data item, the possible requests are detailed in the remainder of this section.

Note :

To respect the D.D.E. standard as closely as possible, SIMTSX differentiates between upper and lower case characters. Similarly, accented characters are distinguished from non accented characters.

Only the CF_TEXT format is currently recognized by the SIMTSX server.

6.2 List of D.D.E. Exchanges

The table below shows the list of data which can be exchanged with the SIMTSX D.D.E. server.

The name of the SIMTSX D.D.E. server is : **SIMAC**.

Standard syntax of the D.D.E. request : **SIMAC|Application!request**.

The "Request " column shows the syntax corresponding to various exchange data. "**elt**" must be replaced by the name used in the SIMTSX application for the corresponding variable.

Example : SIMAC|Maqutsx!axis:ax-1:position -> followed by the evolution of the axis 1

Request	Read	Write	Info.	Comment
AXIS				
axis	yes			See section 6.4-1 list of axes
axis:num	yes			number of axes
axis:elt:state	yes		yes	state of an axis
axis:elt:length	yes			length of an axis
axis:elt:position	yes		yes	position of an axis
axis MOVEMENTS				
movements	yes			See section 6.4-5 list of movements
movements:num	yes			number of movements
movement:elt:state	yes		yes	state of a movement
FLIPFLOP				
flipflop	yes			See section 6.4-2 list of bistables
flipflop:num	yes			number of bistables
flipflop:elt:state	yes		yes	state of a bistable
flipflop:elt:default	yes	yes	yes	faults set on a bistable
EXTERNAL variables				
external	yes			See section 6.4-3 list of external variables
external:num	yes			number of external variables
external:elt:state	yes	yes	yes	state of an external variable
INPUTS				
inputs	yes			See section 6.4-4 list of inputs
input:elt:state	yes	yes	yes	state of an input
inputs:num	yes			number of inputs
NUMERICAL				
numerical	yes			See section 6.4-6 list of numeric variables
numerical:num	yes			number of numeric variables
numerical:elt:value	yes	yes	yes	value of a numeric variable
OUTPUTS				
outputs	yes			See section 6.4-7 list of outputs
outputs:num	yes			number of outputs
outputs:elt:state	yes		yes	staeet of an output
IMPULSE				
impulse	yes		yes	See section 6.3-5 option to stop on an output pulse

Request	Read	Write	Info.	Comment
RELAY				
relay	yes			See section 6.4-9 list of relays
relay:num	yes			number of relays
relay:elt:state	yes		yes	state of a relay
relay:elt:default	yes	yes	yes	faults set on a relay
SWITCHES				
switches	yes			See section 6.4-10 list of sensors
switches:num	yes			number of sensors
switch:elt:state	yes		yes	state of a sensor
switch:elt:default	yes	yes	yes	fault set on a sensor
switch:elt:rebounds		yes		bounce set on a sensor
simulation MODE				
mode	yes	yes		See section 6.3-2 the current simulation mode
mode:period	yes	yes		the period
simulation STATUS				
status	yes	yes	yes	See section 6.3-1 state of the simulation
realtime	yes	yes		realtime option
simulation TRAPS				
traps	yes			See section 6.4-11 list of simulation traps
traps:num	yes			number of simulation traps
trap:elt:state	yes		yes	state of a simulation trap
trap:elt:max	yes			max. time difference between 2 activations
trap:elt:min	yes			min. time difference between 2 activations
trap:elt:average	yes			average time difference between 2 activations
trap:elt:number	yes			number of activations of a trap
SUPPLIES				
supplies	yes			See section 6.4-8 list of supplies
supplies:num	yes			number of supplies
supply:elt:state	yes		yes	state of a supply elt
supply:elt:default	yes	yes	yes	faults set on a supply
DATE				
date	yes			See section 6.3-6 evolution of the simulation date
date:unit	yes			date unit
VARIABLES				
variables	yes			See section 6.4-12 list of variables
variables:num	yes			number of variables
variable:elt:state	yes		yes	state of a variable
variable:elt:default	yes	yes	yes	faults set on a variable

6.3 Information on Simulation

6.3-1 Simulation Status : Status

Requests : READ, INFORMATION, WRITE

Operating status of the SIMTSX D.D.E. server. The value returned belongs to all the following character strings :

- "current" if simulation is in progress
- "interrupted" if simulation is interrupted
- "blocked" if simulation is blocked

To modify the simulation status, the value to be sent to the SIMTSX server is one of the following character strings :

- "interrupt" : interrupts simulation
- "restart" : restarts simulation

Note

It is not possible to modify the simulation status when the current status is "blocked".

6.3-2 Simulation Mode : Mode

Requests : READ, WRITE

Simulation operating mode. The value returned by the SIMTSX D.D.E. server is one of the following character strings :

- "step by step" if step by step simulation mode is the current simulation mode
- "periodic" if the current simulation mode is periodic mode
- "continuous" if the current simulation mode is continuous mode

To modify the data item mode, the value to be sent to the SIMTSX D.D.E. server is one of the three character strings listed above.

Note

To select periodic simulation mode, a period must already have been defined.

6.3-3 Simulation Period : mode:period

Requests : READ, WRITE

Period associated with periodic operating mode. The value returned by the SIMTSX server is a character string containing the value of the period (eg : "0.001"). Similarly, to change the value of the simulation period, the client application sends a character string containing a number to the SIMTSX server (the number formats which can be understood by SIMTSX are those accepted by the simulation interface).

The value returned is the character string "UNKNOWN" if no period is set in the SIMTSX server.

6.3-4 Delayed Mode : Realtime

Requests : READ, WRITE

Indicates whether the simulation is in delayed mode. The value returned (to be sent to the SIMTSX server) is one of the following character strings :

- "1" if the delayed mode option is active, "0" otherwise

6.3-5 Output Pulse : Pulse

Requests : READ, WRITE

Returns the state of the "Stop if Output Pulse" check box. The value returned (to be sent to the SIMTSX server) is one of the following character strings :

- "1" if the option is active, "0" otherwise

6.3-6 Simulation Date : date

Requests : READ, INFORMATION

Returns the current time value in the form of a character string. Depending on the type of representation selected, the value returned is one of the following :

- "HH:MM:SS.XX" (HH : hours, MM : minutes, SS : seconds, XX : remainder)
- "MM:SS.XX" (MM : minutes, SS : seconds, XX : remainder) if the time sent < 1 hour

Note

If the SIMTSX model is calculated in hundredths of a minute, the time sent by the D.D.E. link is automatically converted to the hours, minutes, seconds format.

«**date:unit**» returns the unit used to calculate the time in the form of a character string (READ request). The value returned belongs to the following character strings :

- "1/100 min", or "sec."

6.4 Information on Description Elements

6.4-1 Axis

Axis Request : READ

Returns the list of application axes in the form of a character string. Each axis name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Axis:num Request : READ

Returns the number of application axes in the form of a character string containing a number.

Axis:elt:state Requests : READ, INFORMATION

State of the axis **elt**, where **elt** is the name of a SIMTSX axis written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the axis is active (a movement is being executed)
- "0" if the axis is inactive (no current axis movement)

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX axis.

Axis:elt:position Requests : READ, INFORMATION

Current position of the axis **elt**, where **elt** is the name of a SIMTSX axis written in lower case. The value returned is in the form of a character string containing a number between 0 and 1. The value returned is the percentage of outputs for the axis **elt**. The character string "UNKNOWN" is returned if **elt** is not a SIMTSX axis.

Axis:elt:length Request : READ

Length of the axis **elt**, where **elt** is the name of a SIMTSX axis written in lower case. The value is returned in the form of a character string containing a number. The character string "UNKNOWN" is returned if **elt** is not a SIMTSX axis.

6.4-2 Flipflop

Flipflop

Request : READ

Returns the list of application bistables in the form of a character string. Each bistable name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Flipflop:num

Request : READ

Returns the number of application bistables in the form of a character string containing a number.

Flipflop:elt:state

Requests : READ, INFORMATION

State of a bistable **elt**, where **elt** is the name of a SIMTSX bistable written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the bistable is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX bistable.

Flipflop:elt:default

Requests : READ, INFORMATION, WRITE

Faults set on the bistable **elt**, where **elt** is the name of a SIMTSX bistable written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "stick at 0"
- "stick at 1"
- "force to 0"
- "force to 1"
- "" (empty character string) if no fault has been set on the bistable *elt*

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX bistable.

To set a fault, the client application sends one of the character strings listed above. To remove all the faults set on the bistable **elt**, the client application sends the character string "none" to the SIMTSX server.

Note

The write request has no effect if the SIMTSX interface window for setting faults on bistables is open.

6.4-3 External variables

External Request : READ

Returns the list of external application variables in the form of a character string. Each external variable name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

External:num Request : READ

Returns the number of external variables in the form of a character string containing a number.

External:elt:state Requests : READ, INFORMATION, WRITE

State of the external variable **elt**, where **elt** is the name of a SIMTSX external variable written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the external variable is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX external variable.

To set the value of an external variable, the client application sends one of the two character strings defined above.

Note

The write request has no effect if the SIMTSX interface window for modifying external variables is open.

6.4-4 Input

Inputs

Request : READ

Returns the list of application inputs in the form of a character string. Each input name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Inputs:num

Request : READ

Returns the number of application inputs in the form of a character string containing a number.

Input :elt:state

Request : READ, INFORMATION

State of the input **elt**, where **elt** is the name of a SIMTSX input written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the input is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX input.

6.4-5 Movement

Movements **Request : READ**

Returns the list of application movements in the form of a character string. Each movement name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Movements:num **Request : READ**

Returns the number of application movements in the form of a character string containing a number.

Movement:elt:state **Requests : READ, INFORMATION**

State of the movement **elt**, where **elt** is the name of a SIMTSX movement written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the movement is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX movement.

6.4-6 Numerical

Numerical **Request : READ**

Returns the list of application numeric variables in the form of a character string. Each variable name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the **SIMTSX** application does not contain any elements of this type.

Numerical:num **Request : READ**

Returns the number of application numeric variables in the form of a character string containing a numeric number:**elt.value**.

Numerical:elt:value **Requests : READ, WRITE, INFORMATION**

State of **elt**, where **elt** is the name of a **SIMTSX** numeric variable written in lower case. The value is returned by **SIMTSX** in the form of a character string. The character string "UNKNOWN" is returned if **elt** is not a numeric variable.

Note

The request in write mode only operates for external numeric variables.

6.4-7 Output

Outputs **Request : READ**

Returns the list of application outputs in the form of a character string. Each output name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Outputs:num **Request : READ**

Returns the number of application outputs in the form of a character string containing a number.

Output:elt:state **Requests : READ, INFORMATION**

Returns the value of the output **elt**, where **elt** is the name of a SIMTSX output written in lower case. The value belongs to the following character strings :

- "1" if the output is at the high state, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX output.

6.4-8 Power Supply

Supplies

Request : READ

Returns the list of the application power supplies in the form of a character string. Each supply name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Supplies:num

Request : READ

Returns the number of the application power supplies in the form of a character string containing a number.

Supply:elt:state

Requests : READ, INFORMATION

State of the supply **elt**, where **elt** is the name of a SIMTSX power supply written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the supply is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX power supply.

Supply:elt:default

Requests : READ, INFORMATION, WRITE

Faults set on the supply **elt**, where **elt** is the name of a SIMTSX power supply written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "stick at 0"
- "stick at 1"
- "force to 0"
- "force to 1"
- "" (empty character string) if no fault is set on the supply **elt**

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX power supply.

To set a fault, the client application sends one of the character strings listed above. To remove all the faults set on the supply **elt**, the client application sends the character string "none" to the SIMTSX server to the SIMTSX server.

Note

The write request has no effect if the SIMTSX interface window for setting faults on supplies is open.

6.4-9 Relay

Relay Request : READ

Returns the list of application relays in the form of a character string. Each relay name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Relay:num Request : READ

Returns the number of application relays in the form of a character string containing a number.

Relay:elt:state Requests : READ, INFORMATION

State of the relay **elt**, where **elt** is the name of a SIMTSX relay written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the relay is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX relay.

Relay:elt:default Requests : READ, INFORMATION, WRITE

Faults set on the relay **elt**, where **elt** is the name of a SIMTSX relay written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "stick at 0"
- "stick at 1"
- "force to 0"
- "force to 1"
- "" (empty character string) if no fault has been set on the relay **elt**

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX relay.

To set a fault, the client application sends one of the character strings listed above. To remove all the faults set on the relay **elt**, the client application sends the character string "none" to the SIMTSX server.

Note

The write request has no effect if the SIMTSX interface window for setting faults on relays is open.

6.4-10 Switch

Switches

Request : READ

Returns the list of application sensors in the form of a character string. Each sensor name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Switches:num

Request : READ

Returns the number of application sensors in the form of a character string containing a number.

Switch:elt:state

Requests : READ, INFORMATION

State of a sensor **elt**, where **elt** is the name of a SIMTSX sensor written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the sensor is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX sensor.

Switch:elt:default

Requests : READ, INFORMATION, WRITE

Faults set on the sensor **elt**, where **elt** is the name of a SIMTSX sensor written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "stick at 0"
- "stick at 1"
- "force to 0"
- "force to 1"
- "bounce on activation"
- "bounce on deactivation"
- "" (empty character string) if no fault has been set on the sensor **elt**

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX sensor.

The result of the request can be several concatenated character strings if, for example, forcing and contact bounce are set at the same time.

To modify the faults set on the sensor **elt**, the client application sends one of the following character strings :

- "stick at 0"
- "stick at 1"
- "force to 0"
- "force to 1"
- "none"

Note

The write request has no effect if the SIMTSX interface window for setting faults on sensors is open.

To set contact bounce from the client application, use the "bounce" request.

Switch: *elt:rebounds* **Request :** WRITE

Sets the bounce on **elt**, where **elt** is the name of a SIMTSX sensor written in lower case. The values sent to the SIMTSX server belong to the following character strings :

- "bounce on activation"
- "bounce on deactivation"
- "none" to remove all the contact bounces set on the sensor **elt**

Note

The write request has no effect if the SIMTSX interface window for setting bounce on sensors is open

6.4-11 Trap

Traps **Request :** READ

Returns the list of application traps in the form of a character string. Each trap name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Traps:num **Request :** READ

Returns the number of application simulation traps in the form of a character string containing a number.

Trap:elt:state **Requests :** READ, INFORMATION

State of **elt**, where **elt** is the name of a SIMTSX simulation trap written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "1" if the trap is active, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX simulation trap.

Trap:elt:number **Request :** READ

Number of recorded activations of **elt**, where **elt** is the name of a SIMTSX simulation trap written in lower case. The value obtained is a character string containing a number. The character string "UNKNOWN" is returned if **elt** is not a SIMTSX simulation trap.

Trap:elt:average **Request :** READ

Average time difference recorded between two activations of **elt**, where **elt** is the name of a SIMTSX simulation trap written in lower case. The value obtained is a character string containing a number.

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX simulation trap.

Trap:elt:min **Request :** READ

Minimum time difference recorded between two activations of **elt**, where **elt** is the name of a SIMTSX simulation trap written in lower case. The value obtained is a character string containing a number.

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX simulation trap.

Trap:elt:max**Request : READ**

Maximum time difference recorded between two activations of **elt**, where **elt** is the name of a SIMTSX simulation trap written in lower case. The value obtained is a character string containing a number.

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX simulation trap.

6.4-12 Variable**Variables****Request : READ**

Returns the list of application variables in the form of a character string. Each variable name is delimited by the separators "CR" and "LF" (\r\n). An empty character string is returned if the SIMTSX server does not contain any elements of this type.

Variables:num**Request : READ**

Returns the number of application variables in the form of a character string containing a number.

Variable:elt:state**Requests : READ, INFORMATION**

Returns the value of the variable **elt**, where **elt** is the name of a SIMTSX output written in lower case. The value belongs to the following elements :

- "1" if the output is at the high state, "0" otherwise

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX variable.

Variable:elt:default**Requests : READ, INFORMATION, WRITE**

Faults set on the variable **elt**, where **elt** is the name of a SIMTSX variable written in lower case. The values obtained from the SIMTSX server belong to the following character strings :

- "force to 0"
- "force to 1"
- "" (empty character string) if no fault has been set on the variable **elt**

The character string "UNKNOWN" is returned if **elt** is not a SIMTSX variable.

To set a fault, the client application sends one of the character strings listed above. To remove all the faults set on the variable **elt**, the client application sends the character string "none" to the SIMTSX server.

Note

The write request has no effect if the SIMTSX interface window for setting faults on variables is open.

6.5 Example of D.D.E. Communication With Excel

The example below shows a D.D.E. exchange table with the SIMTSX "maqutsx" application. To activate D.D.E. exchanges, simply enter this table in Excel and develop the SIMTSX "maqutsx" application in offline or online simulation. The values in the second column will evolve depending on simulation.

For exchanges with another application, such as a supervisor for example, simply enter the D.D.E. links in the database or the corresponding exchange table.

Comment	D.D.E. link	Syntax to be entered in the fields in the D.D.E. link column
state of an axis	0	=SIMAC!'Maqutsx'!axis:ax-1:state'
length of an axis	20	=SIMAC!'Maqutsx'!axis:ax-1:length'
current position of an axis	0.125	=SIMAC!'Maqutsx'!axis:ax-1:position'
list of axes	ax-1	=SIMAC!'Maqutsx'!axis
number of axes	11	=SIMAC!'Maqutsx'!axis:num'
fault on sensor	0	=SIMAC!'Maqutsx'!switch:a1:default'
state of a sensor	1	=SIMAC!'Maqutsx'!switch:a1:state'
list of sensors	a1	=SIMAC!'Maqutsx'!switches
number of sensors	31	=SIMAC!'Maqutsx'!switches:num'
date	1.2540625	=SIMAC!'Maqutsx'!date
date unit	sec.	=SIMAC!'Maqutsx'!date:unit'
state of an input	0	=SIMAC!'Maqutsx'!input:%i2.0:state'
list of inputs	%i2.0	=SIMAC!'Maqutsx'!inputs
number of inputs	71	=SIMAC!'Maqutsx'!inputs:num'
state of an external variable	0	=SIMAC!'Maqutsx'!external:a4:state'
list of external variables	a4	=SIMAC!'Maqutsx'!external
number of external variables	31	=SIMAC!'Maqutsx'!external:num'
current simulation mode	continuous	=SIMAC!'Maqutsx'!mode
value of the period	UNKNOWN	=SIMAC!'Maqutsx'!mode:period'
state of a movement	0	=SIMAC!'Maqutsx'!movement:arfrai:state'
list of movements	arfrai	=SIMAC!'Maqutsx'!movements
number of movements	16	=SIMAC!'Maqutsx'!movements:num'
list of relays	km22	=SIMAC!'Maqutsx'!relay
fault on relay	0	=SIMAC!'Maqutsx'!relay:km22:default'
state of a relay	0	=SIMAC!'Maqutsx'!relay:km22:state'
number of relays	25	=SIMAC!'Maqutsx'!relay:num'
state of an output	0	=SIMAC!'Maqutsx'!output:%q5.0:state'
list of outputs	%q5.0	=SIMAC!'Maqutsx'!outputs
number of outputs	48	=SIMAC!'Maqutsx'!outputs:num'
fault on variable	0	=SIMAC!'Maqutsx'!variable:mem-pp0:default'
state of a variable	0	=SIMAC!'Maqutsx'!variable:mem-pp0:state'
variables	mem-pp0	=SIMAC!'Maqutsx'!variables
number of variables	8	=SIMAC!'Maqutsx'!variables:num'

6.6 Advice and Problems Relating to D.D.E.

1. The "mode" request does not operate when continuous and periodic modes are selected at the same time. Only one of the current modes is returned by the D.D.E. link.
2. There are two D.D.E. link modes, synchronous and asynchronous links. With a synchronous link, the client and server processes are executed at the same rate, one waiting for the other. All evolutions in the server appear in the client. As a result, the SIMTSX server execution is slowed down by the connection. This mode is used by Microsoft EXCEL D.D.E. client, for example. However, with an asynchronous link, client and server evolve at their own rates. In this configuration, it is possible for the client to "miss" some evolutions of the server which are too quick. This connection mode is that of NetD.D.E. for Windows for WorkGroups.