StateWORKS

Specifying RTDB (execution environment) - Tutorial

Introduction

- The tutorial teaches you how to specify the RTDB which is the base of any StateWORKS application.
- We assume that you know about the specification of a virtual finite state machine (see [5]) and about the specification of a system of virtual finite state machines (see [6]).

- The result of the RTDB specification is a set of files that may be read by a StateWORKS run-time system.
- To test the specified system you may use the SWLab and Monitors available in StateWORKS Studio under the menu **Tools**.

What is this process for?

- A very powerful feature of the StateWORKS concept is the way state machines (VFSM objects), once they are designed, can be used, in several instances, in a project and can also be taken over to new projects very easily.

- All state machines (VFSM) are designed to function in a "virtual environment", and you now need to learn how to configure them to run them in the real environment, using the RTDB. For this, the project window is used.

- The VFSM you have designed - and in fact each instance of any such design – must be placed in your project and linked to the real input-output signals, commands, etc. which will be used.

Introduction

- The tutorial uses the project ***Pumps*** from the book [1] to illustrate the design steps. In the book you find detailed requirements and analysis of the control task.

- For the purpose of this tutorial some partial specifications are provided which may be loaded to accelerate the training (observe corresponding notes).

Terminology

- **Always (table)**
  A table used for specification of combinational systems or Input actions valid for all states
- **Entry action**
  An Output name describing an action performed by entering a state
- **Exit action (written also as eXit action)**
  An Output name describing an action performed by exiting a state
- **Id name**
  A name of an object
- **I/O Object Dictionary**
  A list of all defined objects
- **I/O Object Id**
  see: Id name
- **Init (flag)**
  A flag: if marked instructs the execution system (RTDB) to initialize the virtual input to that value
- **Init (state)**
  A default state which cannot be deleted but can be renamed
- **Input**
  see: Input Name
- **Input (tab)**
  see: Input Name Dictionary
- **Input action**
  An Output name describing action performed if an Input action condition is due

Terminology

- **Input action condition**
  A condition defined using Input names linked by AND and OR operators
- **Input action expression**
  Input action condition and Input action
- **Input action priority**
  The sequence of Input action expressions in the ST table; used for documentation purpose
- **Input Name**
  A name of a control condition (defined on an Input Value)
- **Input Name Dictionary**
  A list of all defined Input Names
- **Input Value**
  Object input value
- **MyCmd**
  A default Input Name of a type CMD which cannot be deleted but can be renamed
- **Next State priority**
  The sequence of state transitions in the ST table; determines the execution sequence
- **Operators: AND (&), OR (|)**
  Boolean operators
- **Output**
  see: Output Name
- **Output (tab)**
  see: Output Name Dictionary
- **Output Name**
  A name describing an action (defined on an Output Value)

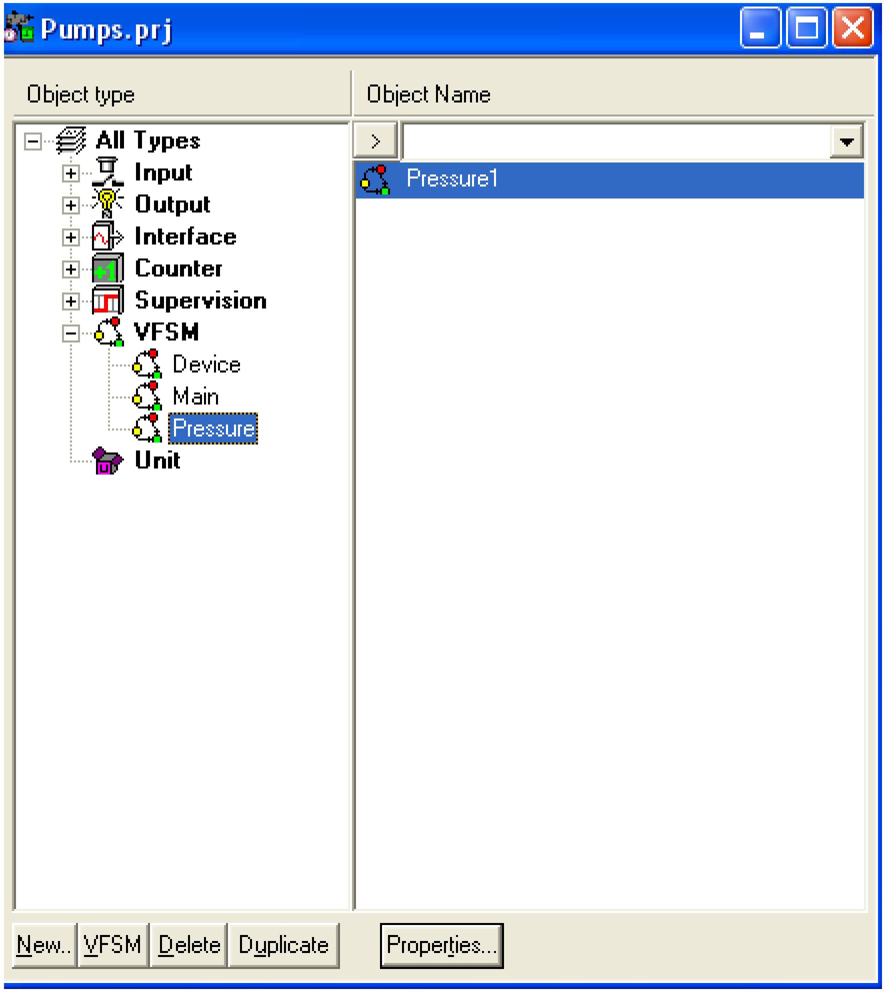
Terminology

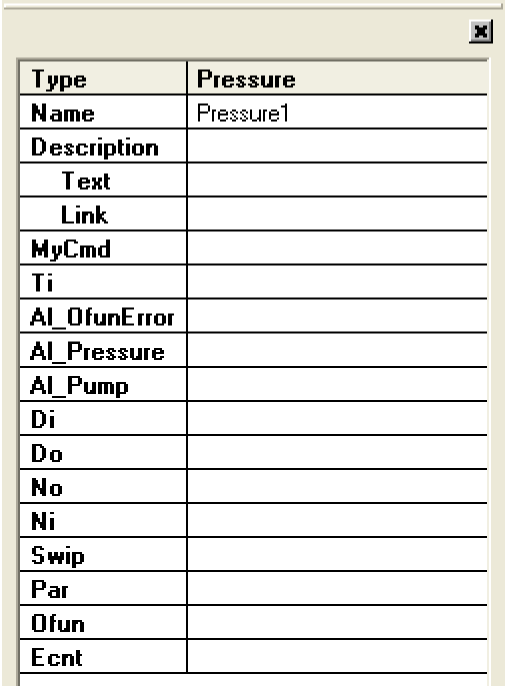
- **Output Name Dictionary**
  A list of all defined Output Names
- **Output Value**
  An Object output value
- **Prefix**
  A VFSM specific prefix used in h-files generated for each VFSM
- **ST diagram**
  A state transition diagram used for graphic presentation of a state machine behavior
- **ST table**
  A state transition table used for detailed specification of a state.
- **State**
  see: State Name (drawn as a circle on the ST diagram)
- **State Name**
  A state name
- **State Name Dictionary**
  A list of all defined State Names
- **Transition**
  A transition between two states (drawn as an arrow on the ST diagram)
- **Transition condition**
  A condition defined using Input names linked by AND and OR operators
- **Transition expression**
  Next state and Transition condition

Creating a VFSM object

- Note: The text below applies to figures on the next slide.
- Most of this work is carried out in the project window.
- Expand the tree VFSM in the Project pane **Object type**.
- Select an object type, for instance ***Pressure***.
- Clicking on the button **New..** create a VFSM object in the pane **Object Name**. The object gets the default name ***Pressure1***.
- You may edit the object name in the **Property** window which opens automatically by creating or selecting the object.
- You will define all **Properties** of the VFSM object later when all required objects are created. At that moment you may define the name and prepare the **Description** (**Text** and **Link**).

Creating a VFSM object

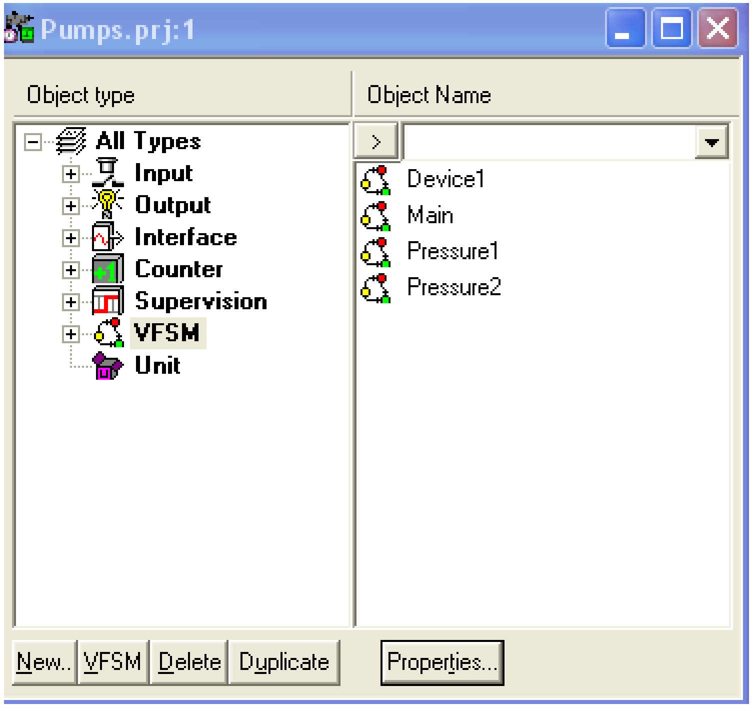


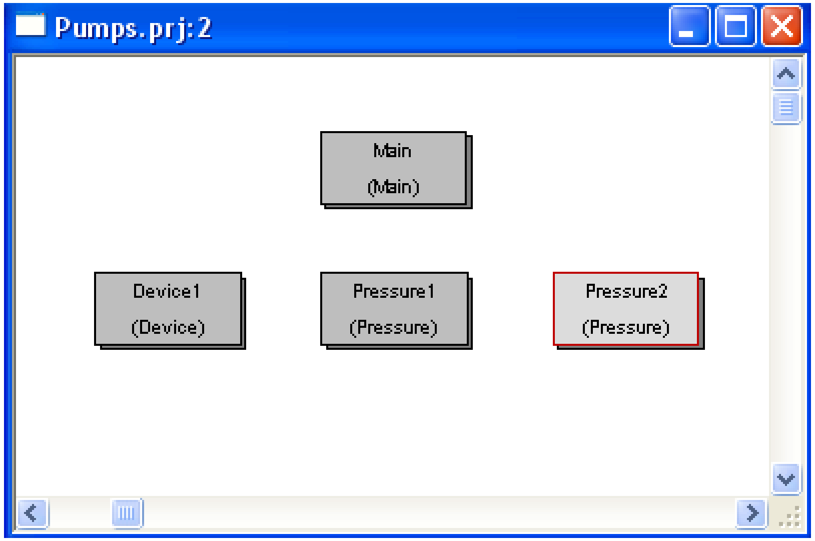


Type	Pressure
Name	Pressure1
Description	
Text	
Link	
MyCmd	
Ti	
AI_OfunError	
AI_Pressure	
AI_Pump	
Di	
Do	
No	
Ni	
Swip	
Par	
Ofun	
Ecnt	

Creating a VFSM object

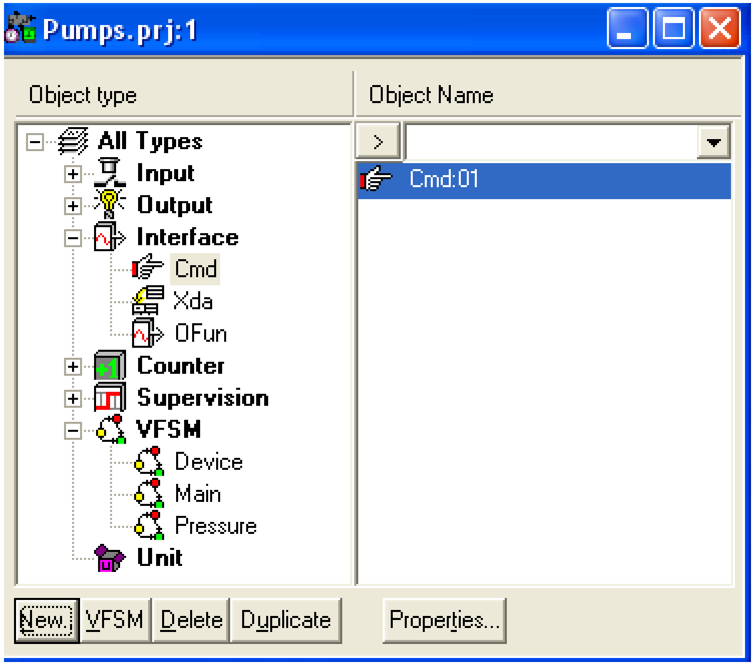
- Similarly, create other required state machines: ***Main***, ***Device1***, ***Pressure1*** and ***Pressure2***.
- You may control the progress by opening the **ST diagram** (shape the diagram according to your preference).

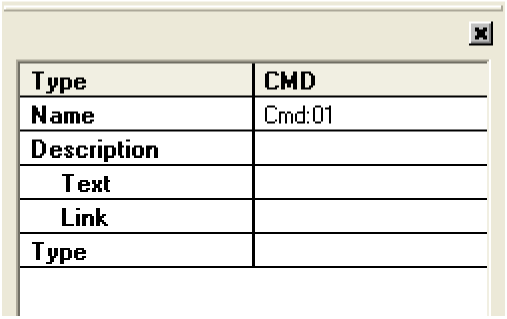




Creating a CMD object

- Selecting any state machine, you see in the **Property** window a list of all its objects.
- Start creating objects, for instance ***MyCmd*** for ***Pressure1***.
- Expand the tree **Interface** in the pane **Object type**.
- Select the type **Cmd** and create the object ***Cmd:01*** by clicking on the button **New…**

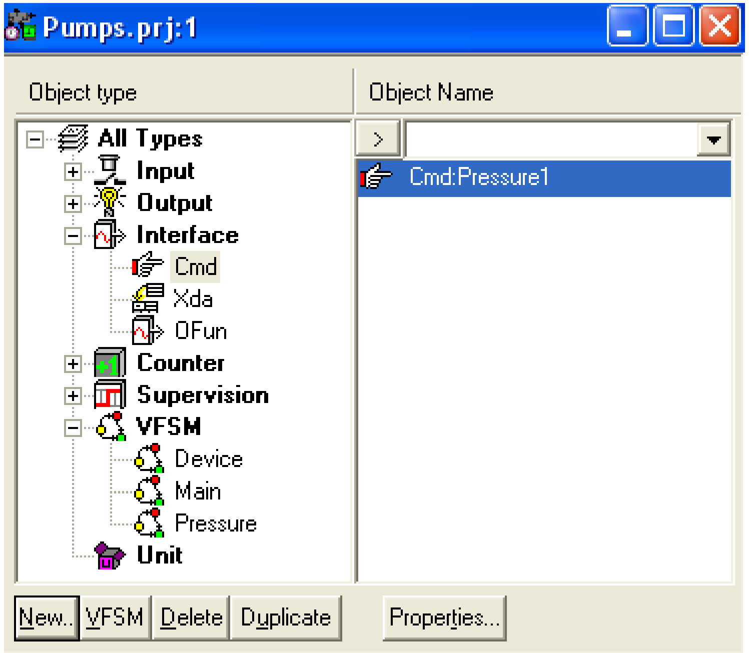


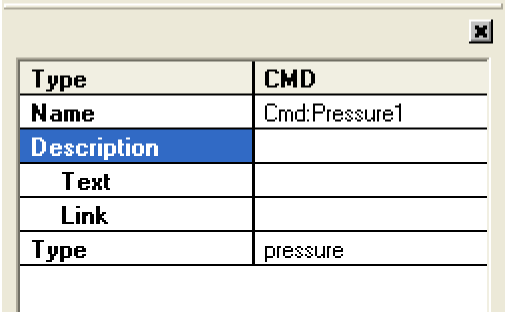


Type	CMD
Name	Cmd:01
Description	
Text	
Link	
Type	

Creating a CMD object

- Define properties of the newly created object in the **Property** window:
  - Give the object a more expressive name, for instance ***Cmd:Pressure1***.
  - Fill the property **Type** with the name of the state machine type; in that case ***Pressure*** (case not sensitive).

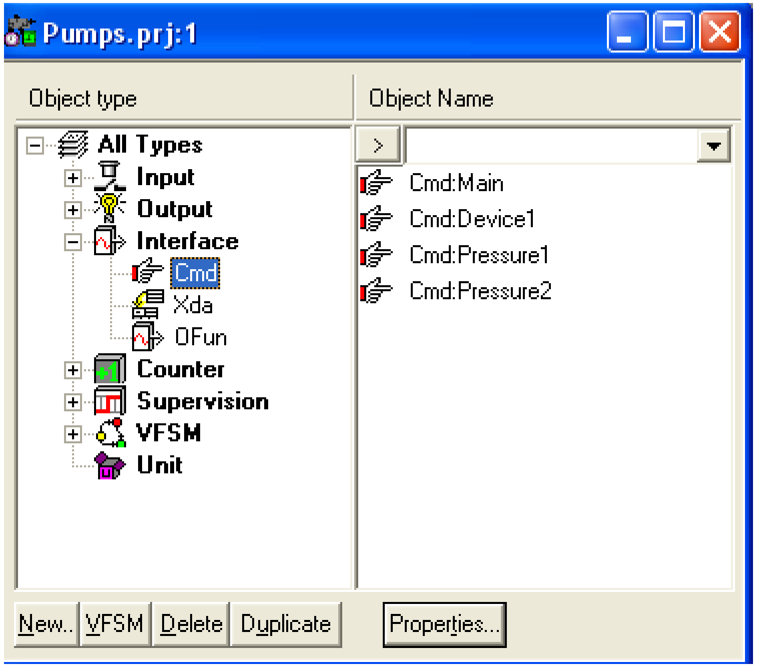




Type	**CMD**
Name	Cmd:Pressure1
Description	
Text	
Link	
Type	pressure

Creating a CMD object

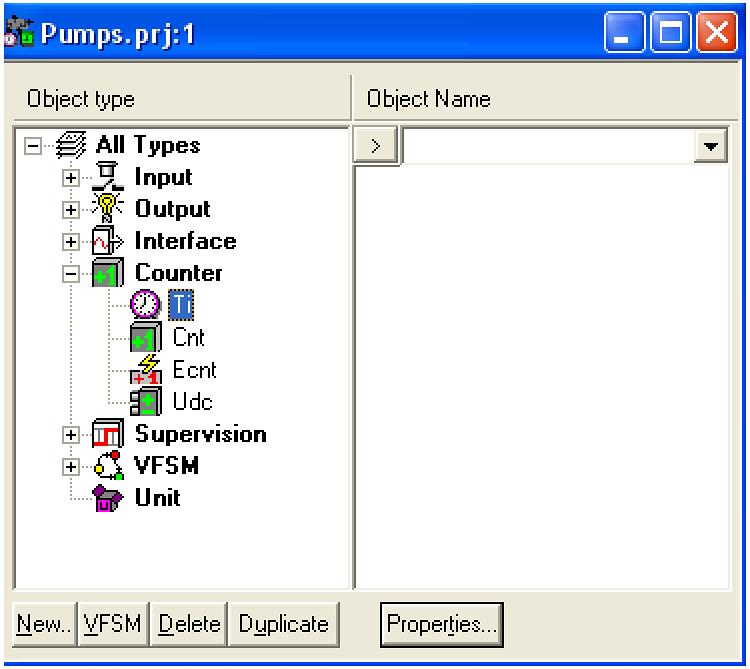
- Similarly, create command objects for all state machines.
- You may use the button **New..** or **Duplicate**.
- You may **Delete** at any time an existing object if it is not used by other objects (try commands ***Where used*** and ***Mark not used*** in the menu opened by the right mouse click in the pane **Object Name**.



Cmd:Main
Cmd:Device1
Cmd:Pressure1
Cmd:Pressure2

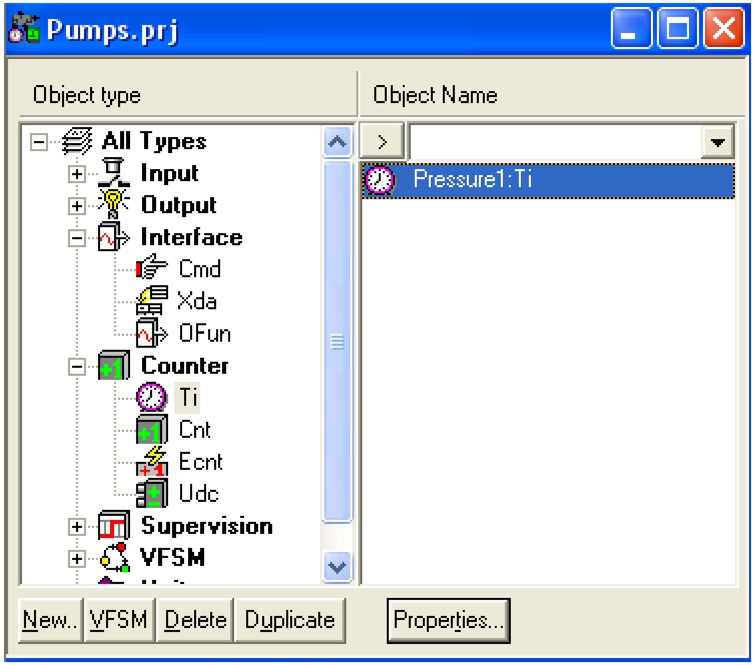
Creating a TI object

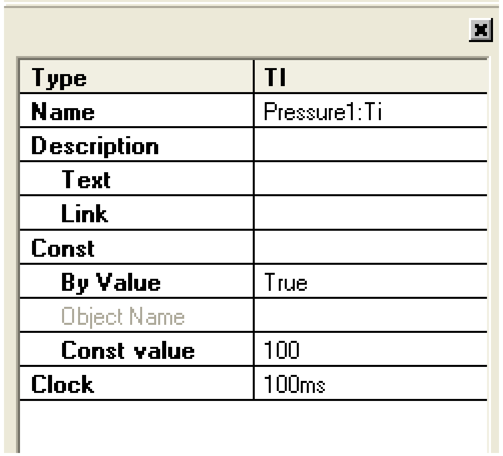
- The ***Pressure1***, ***Pressure2*** and ***Device1*** need timers; thus you have to create 3 objects of type **Ti**.
- Select the type **Counter** / **Ti** in the pane **Object type**.



Creating a TI object

- Create the object **Ti:01** in the pane **Object Name** and edit its properties in the **Property** window.
- Accepting for instance the **By Value**=***True*** and defining the timeout (**Const value=*100***) and the **Clock=*100ms*** you get a timer ***Pressure1:Ti*** for the state machine ***Pressure1***.

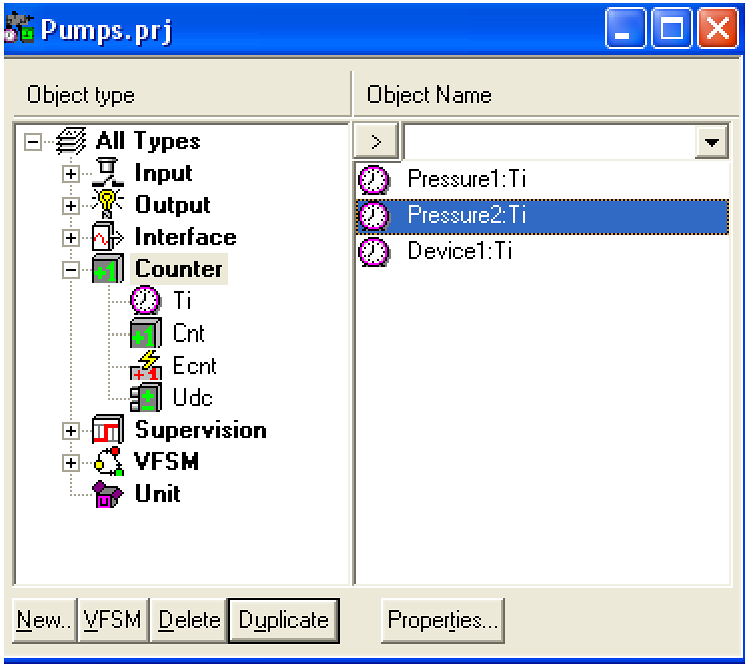


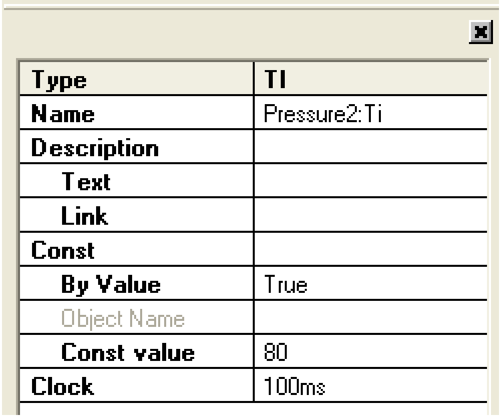


Type	**TI**
Name	Pressure1:Ti
Description	
Text	
Link	
Const	
By Value	True
Object Name	
Const value	100
Clock	100ms

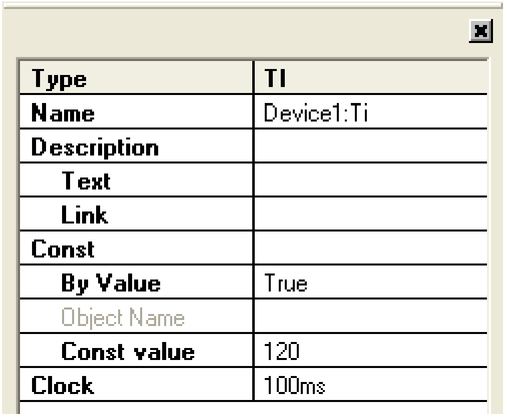
Creating a TI object

- Similarly you define a timer for the state machine ***Pressure2*** with the **Const value**=***80*** and a timer for the state machine ***Device1*** with the **Const value**=***120***: other properties are the same.





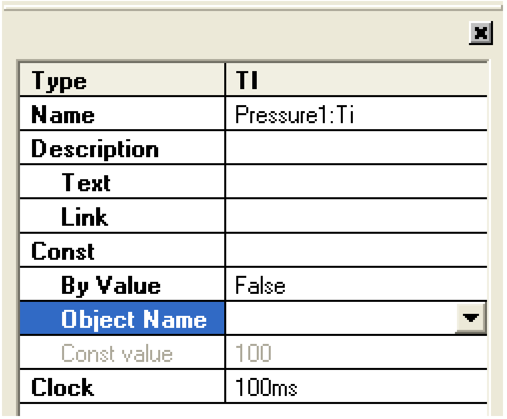
Type	TI
Name	Pressure2:Ti
Description	
Text	
Link	
Const	
By Value	True
Object Name	
Const value	80
Clock	100ms



Type	TI
Name	Device1:Ti
Description	
Text	
Link	
Const	
By Value	True
Object Name	
Const value	120
Clock	100ms

Creating a TI object

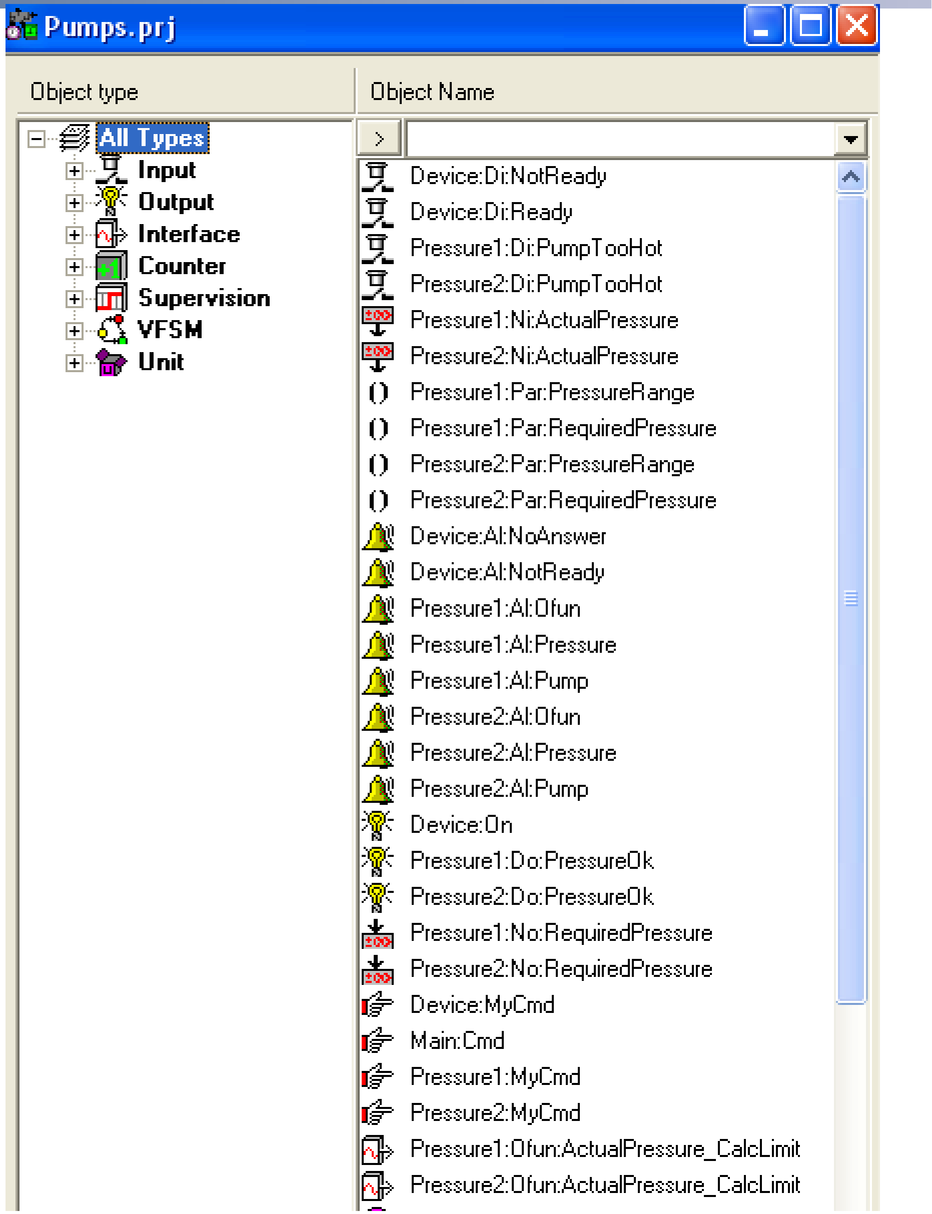
- Choosing **Const value**=***False*** you have to define a source of the timeout value (**Object Name**); it may be an object of type PAR, NI or DAT. As a rule such an object is not owned by a state machine; it is just a parameter of another object.
- Create a required object, for instance PAR and complete the Ti properties.
- The same procedure must be applied to other objects like for instance SWIP, CNT, etc.



Type	TI
Name	Pressure1:Ti
Description	
Text	
Link	
Const	
By Value	False
Object Name	
Const value	100
Clock	100ms

Creating RTDB objects

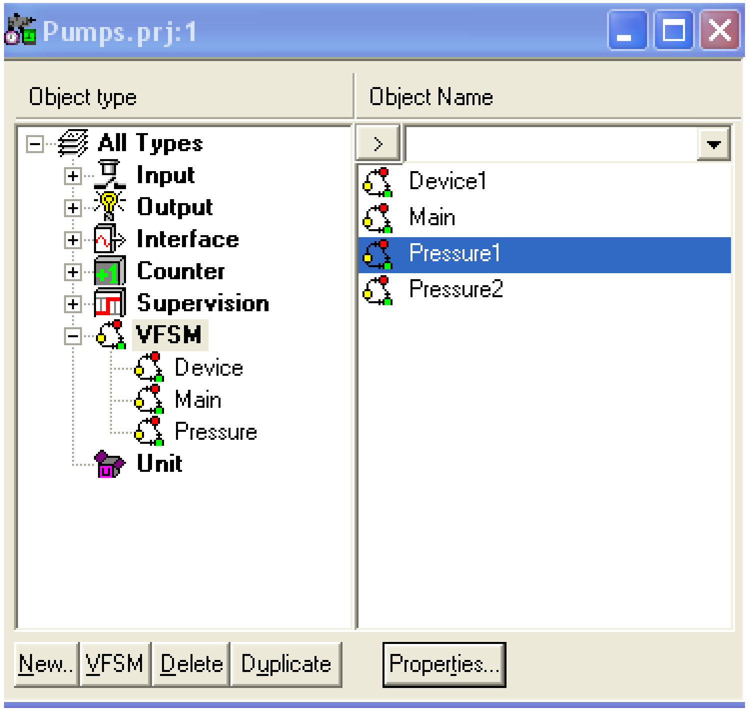
- creation of RTDB objects until you define all objects required by the application.

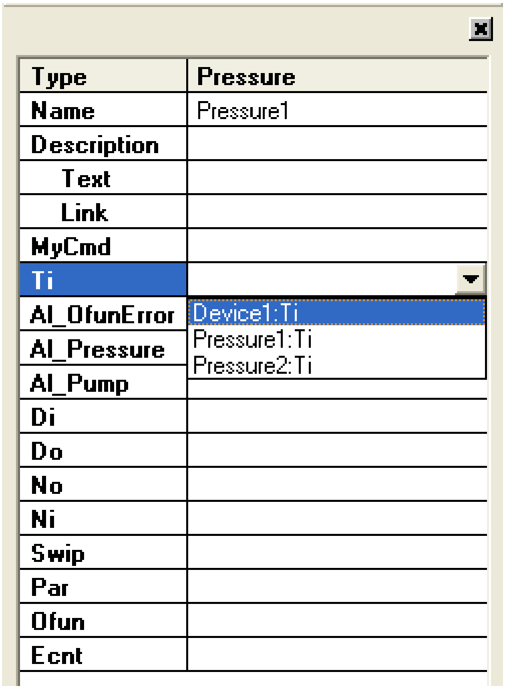


Device:Di:NotReady
Device:Di:Ready
Pressure1:Di:PumpTooHot
Pressure2:Di:PumpTooHot
Pressure1:Ni:ActualPressure
Pressure2:Ni:ActualPressure
Pressure1:Par:PressureRange
Pressure1:Par:RequiredPressure
Pressure2:Par:PressureRange
Pressure2:Par:RequiredPressure
Device:Al:NoAnswer
Device:Al:NotReady
Pressure1:Al:Ofun
Pressure1:Al:Pressure
Pressure1:Al:Pump
Pressure2:Al:Ofun
Pressure2:Al:Pressure
Pressure2:Al:Pump
Device:On
Pressure1:Do:PressureOk
Pressure2:Do:PressureOk
Pressure1:No:RequiredPressure
Pressure2:No:RequiredPressure
Device:MyCmd
Main:Cmd
Pressure1:MyCmd
Pressure2:MyCmd
Pressure1:Ofun:ActualPressure_CalcLimit
Pressure2:Ofun:ActualPressure_CalcLimit

Creating RTDB objects

- Now you can finish the definition of state machine properties.
- Select for instance the state machine Pressure1 in the pane **Object Name**, select the property **Ti** in the **Property** window and open the list of timers in the system.
- Clicking on ***Pressure1:Ti*** define a timer for the ***Pressure1***.

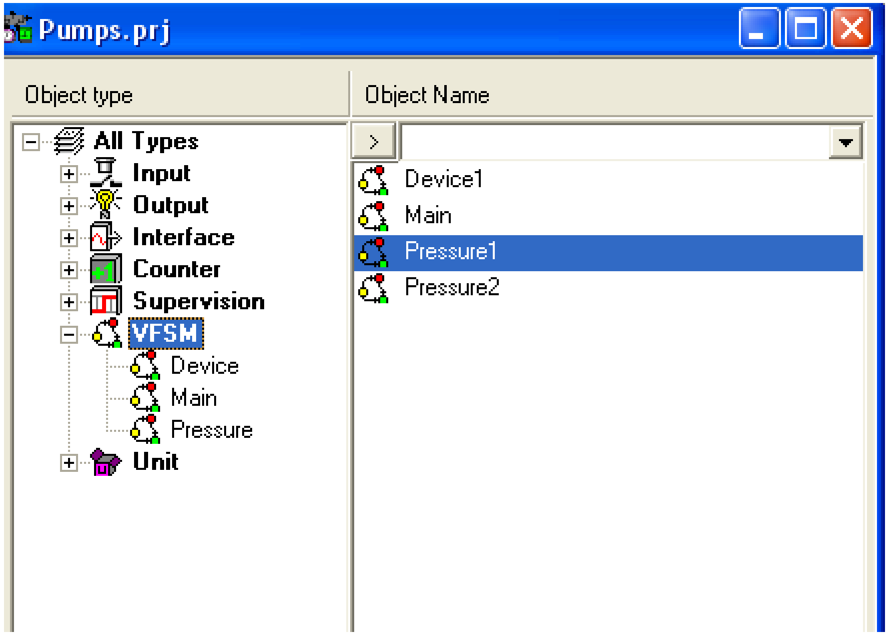


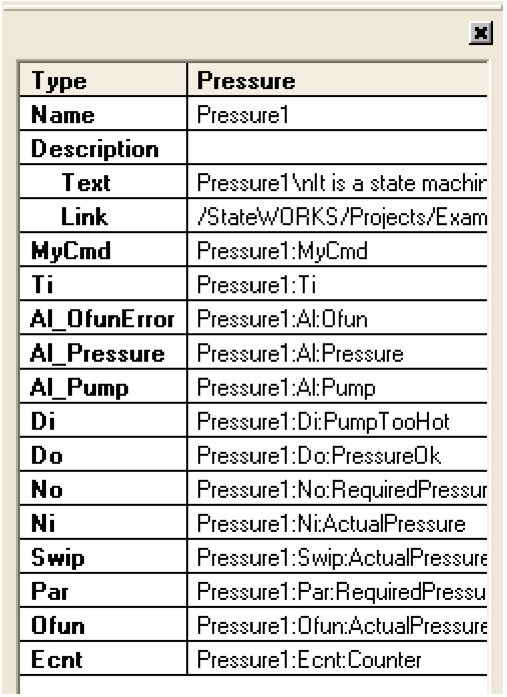


Type	**Pressure**
Name	Pressure1
Description	
Text	
Link	
MyCmd	
Ti	
AI_OfunError	Device1:Ti
AI_Pressure	Pressure1:Ti
AI_Pump	Pressure2:Ti
Di	
Do	
No	
Ni	
Swip	
Par	
Ofun	
Ecnt	

Creating RTDB objects

- Eventually, you get all properties for a state machine ***Pressure1*** as shown below.
- Similarly, you define properties for all state machines in the system.

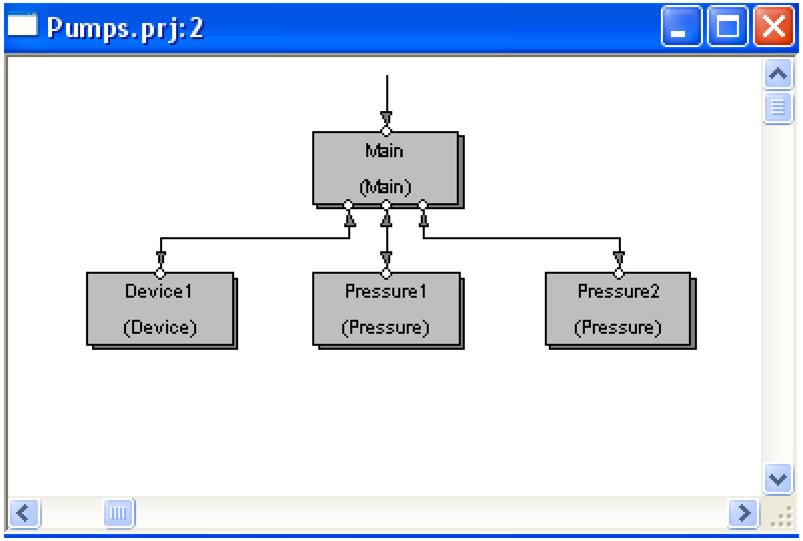




Type	Pressure
Name	Pressure1
Description	
Text	Pressure1\nIt is a state machir
Link	/StateWORKS/Projects/Exam
MyCmd	Pressure1:MyCmd
Ti	Pressure1:Ti
Al_OfunError	Pressure1:Al:Ofun
Al_Pressure	Pressure1:Al:Pressure
Al_Pump	Pressure1:Al:Pump
Di	Pressure1:Di:PumpTooHot
Do	Pressure1:Do:PressureOk
No	Pressure1:No:RequiredPressur
Ni	Pressure1:Ni:ActualPressure
Swip	Pressure1:Swip:ActualPressure
Par	Pressure1:Par:RequiredPressu
Ofun	Pressure1:Ofun:ActualPressure
Ecnt	Pressure1:Ecnt:Counter

SMS diagram

- If you create objects required by all state machines (***Device1***, ***Pressure1***, ***Pressure2***, ***Main***) you get the system of state machines as shown in the **SMS diagram** below.
  Note: load Pumps.prj from the Pumps_Tutorial folder.
- You may test the system using SWLab and StateWORKS monitors but you will notice that it does not work correctly (missing access to output functions).

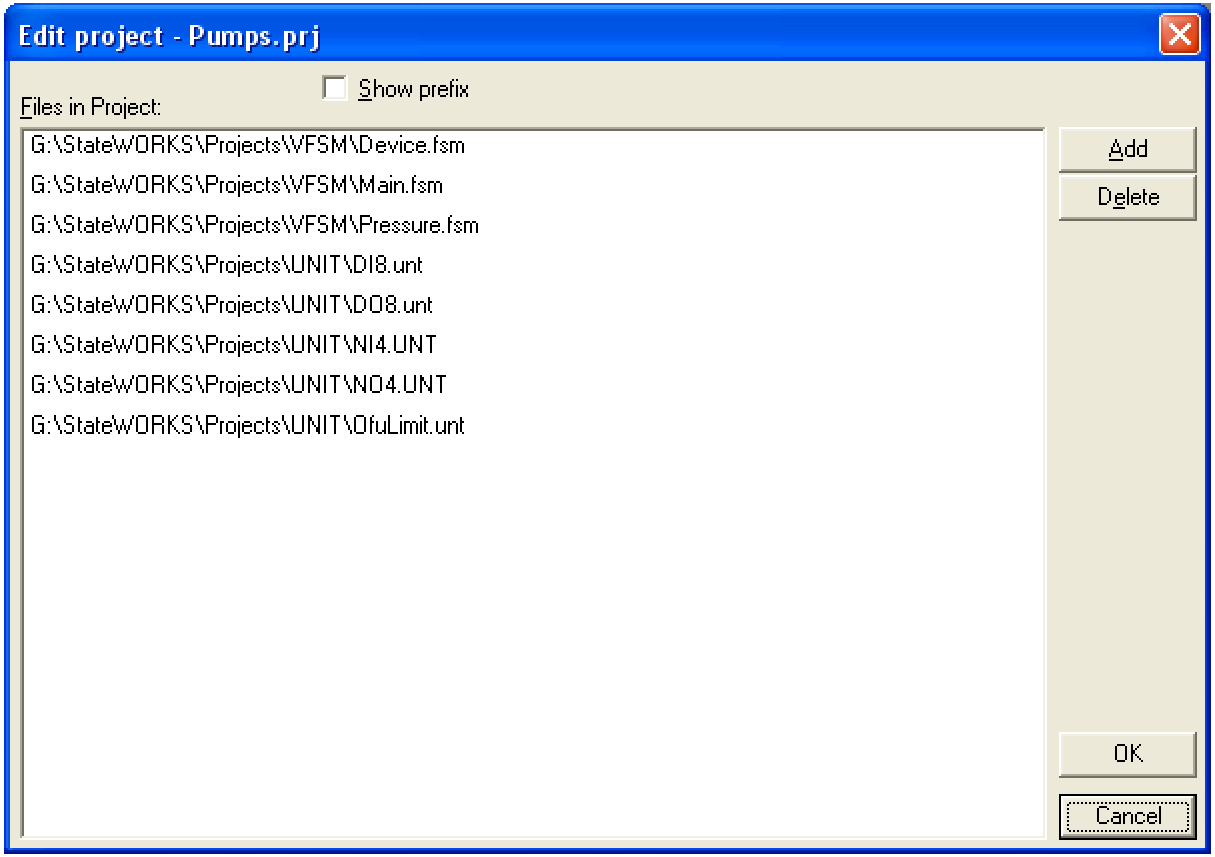


Creating UNIT object

- The RTDB objects created so far can be accessed via TCP/IP; you do it for instance using StateWORKS Monitors.
- The TCP/IP access is mainly intended for a user interface. But you may define an I/O interface which requires to have specific I/O handlers in the run-time system. SWLab has such an I/O Handler to simulate digital / analog inputs and outputs available on the SWLab user interface in the form of switches, LEDs, potentiometers and gauges.
- In the following part you will use predefined UNIT types for creation of UNIT objects required by the SWLab I/O Handler.
- In addition you need a UNIT which represents the interface for the user function called by objects of type OFUN.

Creating UNIT object

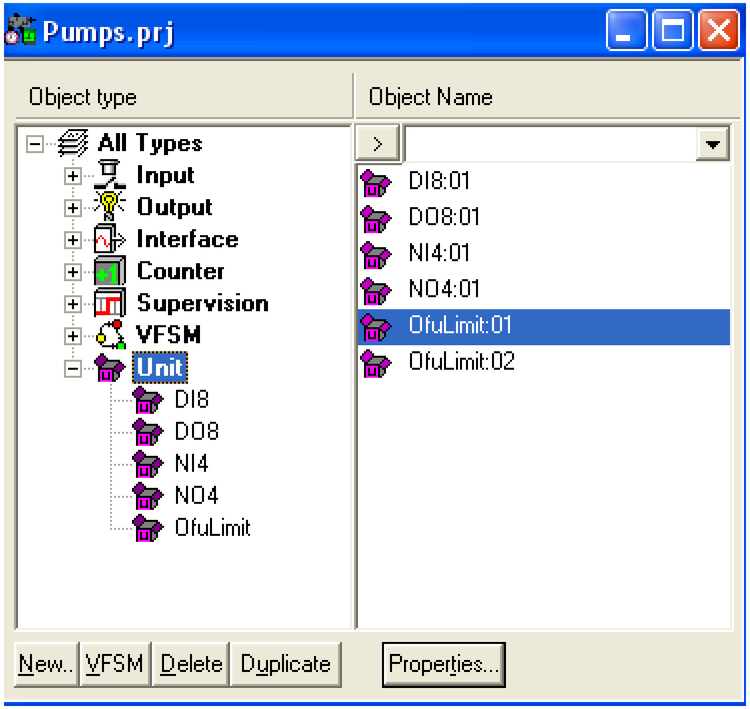
- (Using the button **Add** in the dialog window opened by **Project / Edit**), add UNIT types: ***DI8***, ***DO8***, ***NI4, NO4*** and ***OfuLimit*** to the project



G:\StateWORKS\Projects\VFSM\Device.fsm
G:\StateWORKS\Projects\VFSM\Main.fsm
G:\StateWORKS\Projects\VFSM\Pressure.fsm
G:\StateWORKS\Projects\UNIT\DI8.unt
G:\StateWORKS\Projects\UNIT\DO8.unt
G:\StateWORKS\Projects\UNIT\NI4.UNT
G:\StateWORKS\Projects\UNIT\NO4.UNT
G:\StateWORKS\Projects\UNIT\OfuLimit.unt

Creating UNIT object

- Create UNIT objects for each I/O type.
- Create 2 objects of type OfuLimit: one for each state machine of type Pressure.

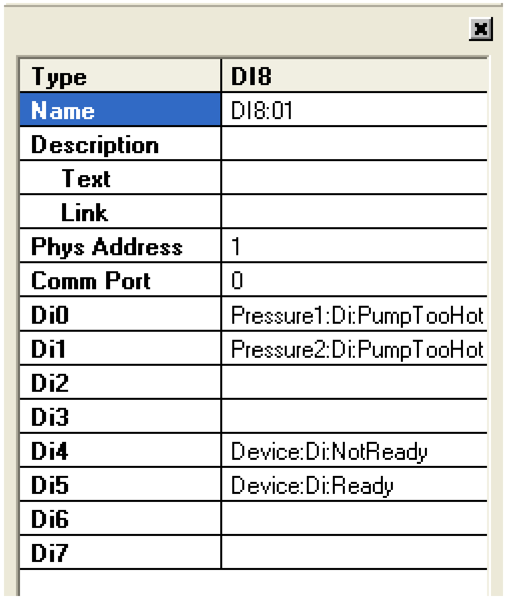


Creating UNIT object

- In principle, a UNIT is a list of objects. In addition to standard properties (**Name**, **Description**), it has also two specific properties: **Phys Address** and **Comm Port**.
- Specifying the UNIT object list you have to decide which elements of SWLab will be used for the project ***Pumps***.

Creating UNIT object

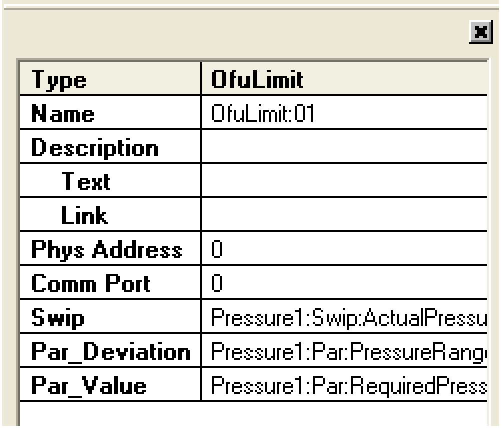
- For instance for the DI-UNIT, you may choose object as shown below.
- Note the value ***1*** chosen for the **Phys Address**. It is required by the run-time system. Similarly, the DO-UNIT requires the value ***3***, NI – the value ***5*** and NO – the value ***7***.



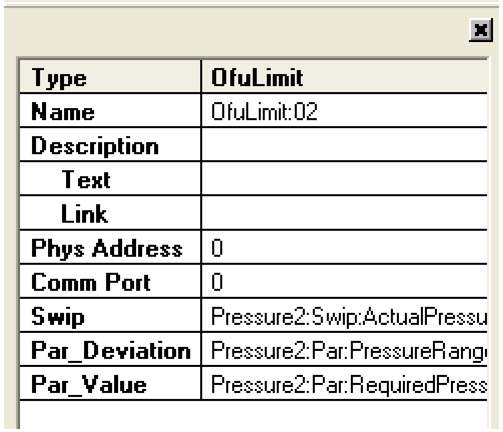
Type	DI8
Name	DI8:01
Description	
Text	
Link	
Phys Address	1
Comm Port	0
Di0	Pressure1:Di:PumpTooHot
Di1	Pressure2:Di:PumpTooHot
Di2	
Di3	
Di4	Device:Di:NotReady
Di5	Device:Di:Ready
Di6	
Di7	

Creating UNIT object

- Below you see values chosen for UNITs of type ***OfuLimit***. To interpret them you have to know the requirements of the OfuLimit function used by the state machines of type ***Pressure***.
- The values of properties **Phys Address** and **Comm Port** are irrelevant in that case.



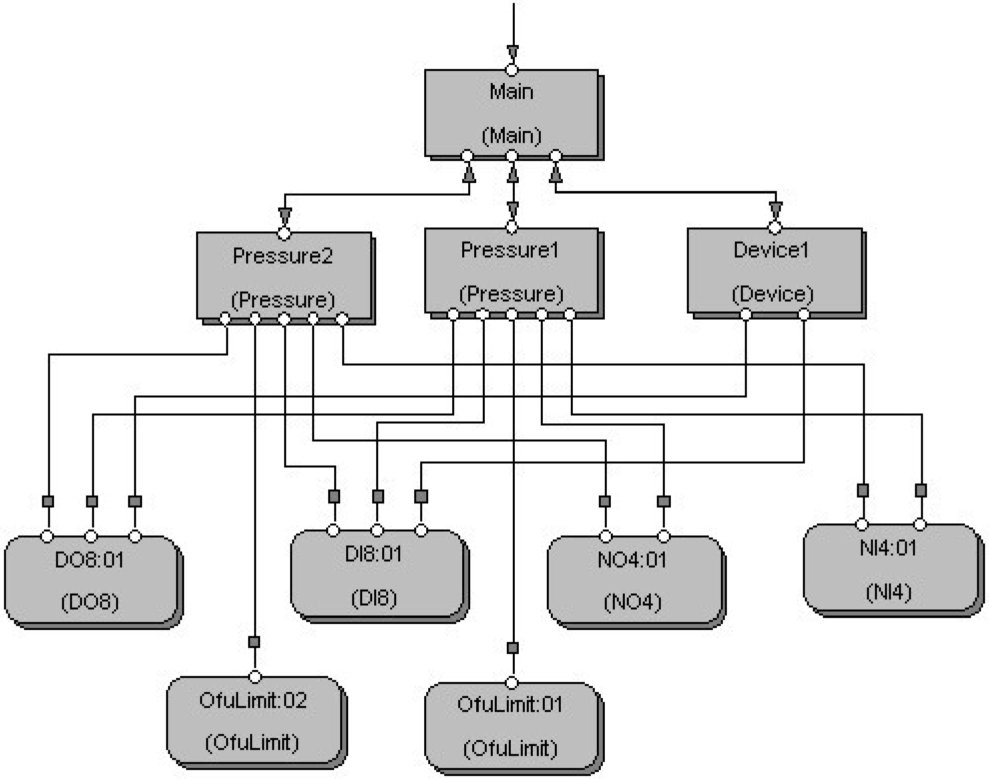
Type	**OfuLimit**
Name	OfuLimit:01
Description	
Text	
Link	
Phys Address	0
Comm Port	0
Swip	Pressure1:Swip:ActualPressu
Par_Deviation	Pressure1:Par:PressureRang
Par_Value	Pressure1:Par:RequiredPress



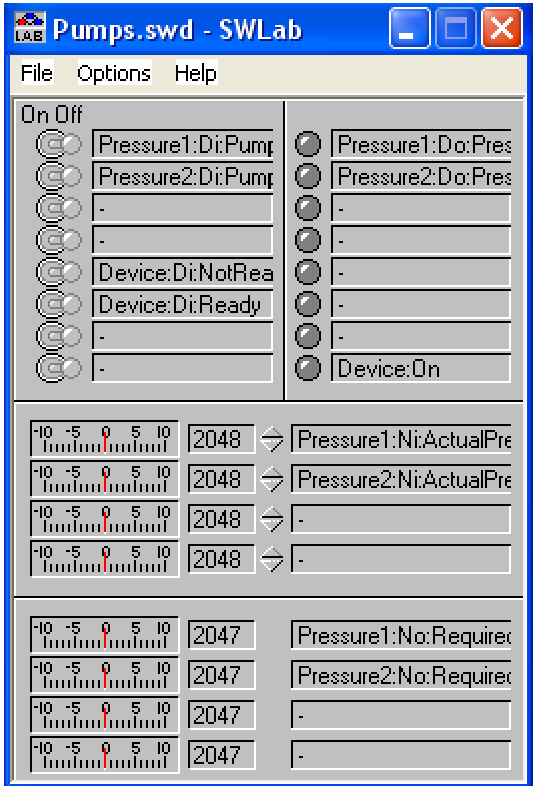
Type	**OfuLimit**
Name	OfuLimit:02
Description	
Text	
Link	
Phys Address	0
Comm Port	0
Swip	Pressure2:Swip:ActualPressu
Par_Deviation	Pressure2:Par:PressureRang
Par_Value	Pressure2:Par:RequiredPress

System complete

- Note: load Pumps.prj from the Pumps folder.
- Now you have a complete system ***Pumps*** which is able to use the User functions for calculating of SWIP limits as well as SWLab I/O interface. The **SMS diagram** of the ***Pumps*** system is shown below and the **SWLab** on the next slide.



System complete



Pressure1:Di:Pump
Pressure2:Di:Pump
-
-
Device:Di:NotRea
Device:Di:Ready
-
-

Pressure1:Do:Pres
Pressure2:Do:Pres
-
-
-
-
-
Device:On

2048 Pressure1:Ni:ActualPre
2048 Pressure2:Ni:ActualPre
2048 -
2048 -

2047 Pressure1:No:Requirec
2047 Pressure2:No:Requirec
2047 -
2047 -

References

[1] Wagner F., et al., *Modeling Software with Finite State Machines: A Practical Approach*. Taylor & Francis CRC Press, 2006.

[2] StateWORKS Studio Help.

[3] StateWORKS Development Tools: User’s Guide & Training Manual. SW Software 2005.

[4] www.stateworks.com - Technical Notes.

[5] StateWORKS: Specifying a state machine – Tutorial.

[6] StateWORKS: Specifying a system of state machines - Tutorial.