

## 8. BOOST HYDSIM – MATLAB® INTERFACE

---

This chapter guides the user through the modeling of BOOST Hydsim -MATLAB® Interface, which is accessible through MATLAB® elements from the Element tree.

There are three options to incorporate MATLAB® model into a BOOST Hydsim model:

- MATLAB® API : Simulink model
- MATLAB® API : m-function
- MATLAB® DLL model

To call MATLAB® PATH from BOOST Hydsim on Linux platforms, the user has to define the location of the API shared libraries and set the path of executable `matlab.exe`. It has to be defined in the shell where BOOST Hydsim is started.

- In C shell, the command to set the library path is:

```
setenv LD_LIBRARY_PATH=<MATLAB>/extern/lib/$Arch:$LD_LIBRARY_PATH
```

- In C shell, the command to set MATLAB® path is:

```
set path = (<MATLAB>/bin $path)
```

- In Bourne or bash shell, the commands to set the library path are:

```
LD_LIBRARY_PATH=<MATLAB>/extern/lib/$Arch:LD_LIBRARY_PATH
export LD_LIBRARY_PATH
```

- In Bourne or bash shell, the command to set MATLAB® path is:

```
PATH=<MATLAB>/bin:PATH
export PATH
```

where `<MATLAB>` is the MATLAB® root directory and `$Arch` is your system architecture.

The environment variable name `LD_LIBRARY_PATH` may vary on different platforms.

It is convenient to place the above commands in a startup script such as `~/ .cshrc` for C shell or `~/ .profile` for Bourne/bash shell.


### 8.1. Model with MATLAB® Simulink

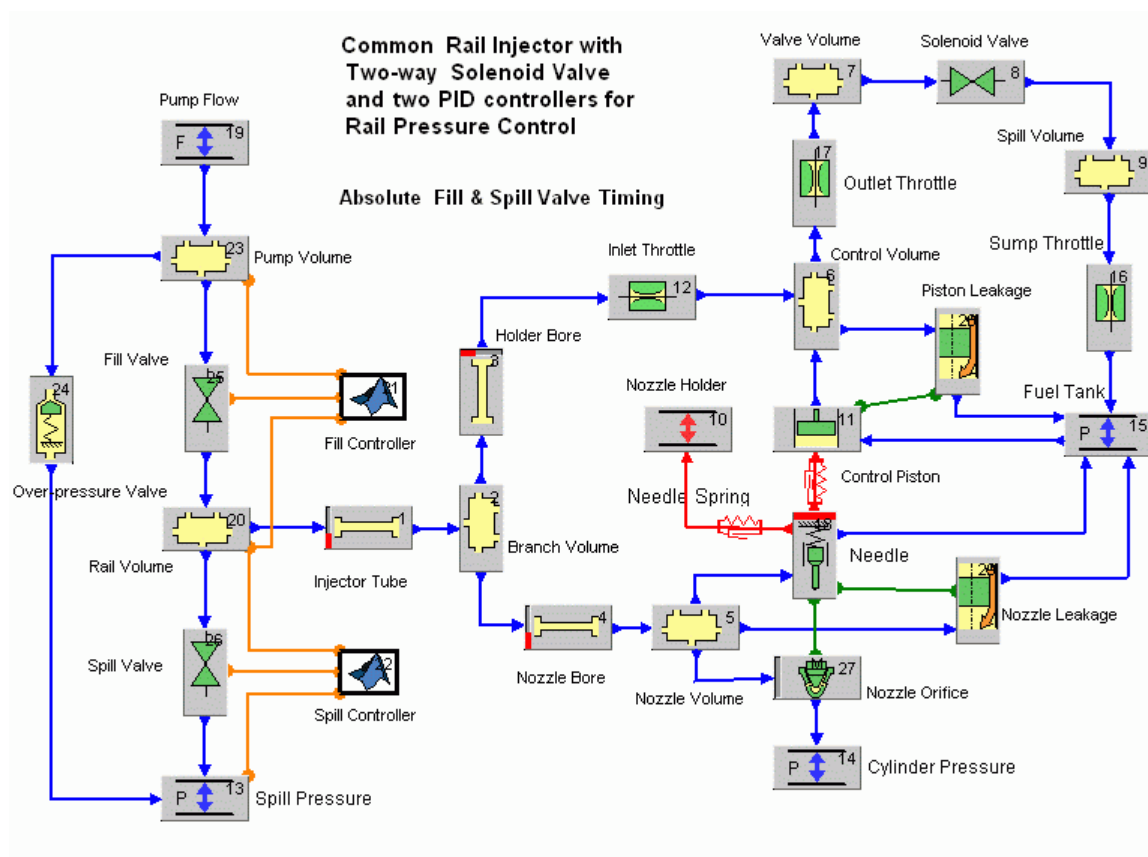
---

In *Chapter 5.1*, a model of the common rail injector with a two-way solenoid valve is shown. This model will be used as a basis for creating a new model with MATLAB® elements which are aimed to model the pressure control in the common rail. For this purpose, two new Volume elements - Pump and Rail Volume - and two additional Switch valves – Fill and Spill Valves – are added to the system. These valves are controlled by PID Controllers represented by MATLAB® elements.

The system layout is shown in *Figure 8-1*. Compared to the common rail injector model in *Section 5.1*, the following elements are added: Pump Flow (No. 19<sup>8</sup>), Pump Volume (No. 23), Rail Volume (No. 20), Fill Valve (No. 25), Spill Valve (No. 26), Spill Pressure (No. 13), Over-pressure Valve (No. 24) and two MATLAB® elements: Fill Controller (No. 21) and Spill Controller (No. 22). The principle of the control system operation is based on the opening and closing of Fill and/or Spill Valves according to the difference between the target pressure (1500 bar) and actual pressure in Rail Volume. If pressure in the Rail gets above target pressure, Spill Controller has to open Spill Valve and, with Fill Valve closed, the pressure in the rail would decrease. In the opposite case Fill Controller has to open Fill Valve and pressure in the rail would increase, assuming that pressure in Pump Volume is high enough to feed the rail. Both Valves have opening and closing times as well as opened and closed delay time.

For safety, an Over-pressure Valve with the opening pressure of 1800 bar is included into the system. It should never open at normal system operation.

All elements belonging to the control system are connected via wire connections (  ) with appropriate MATLAB® elements. It is necessary for the data exchange through the input and output channels of MATLAB® elements.



**Figure 8-1: BOOST Hydsim Model of Common Rail Injector with MATLAB® Elements as PID Controllers**

<sup>8</sup> Numbers given for elements may differ from those in your model. Reference Figure 8-1 for numbered items.

### 8.1.1. Input Data

In this section only the input dialogs of Rail Volume, Spill/Fill Valves and Fill/Spill Controllers will be shown. Other elements have the same input data as the previous model from *Section 5.1*. Additionally, parameters in the **Calculation Control** input dialog - **Calculation end time, Time step and Number of values to store** – are changed. These parameters are defined as global variables and set to 0.015 s, 3e-7 s and 500, respectively.

**Figure 8-2: Input Dialog of Rail Volume**

The Rail Volume size is set to 10000 mm<sup>3</sup>. Pump Volume has 10% of Rail Volume size, i.e. 1000 mm<sup>3</sup>. Initial pressure in all high-pressure volumes is equal to target rail pressure, in our case 1500 bar.

The input dialogs of Fill and Spill Valves are shown in Figure 8-3 and Figure 8-4, respectively. Opening and closing time instant of Fill/Spill Valves are defined in absolute time domain. Here they are formally set to dummy values (0.01 and 0.003 s) because PID Controllers will redefine them during BOOST Hydsim-MATLAB<sup>®</sup> co-simulation.



**Note:** As the interface between BOOST Hydsim and MATLAB<sup>®</sup> starts from second calculation step (when BOOST Hydsim calls MATLAB<sup>®</sup> first time), these dummy values have to be greater than second calculation time, i.e. greater than double time step from **Calculation Control**.

Opening/closing times and flow areas of the valves are assigned to global variables (parameters) which are defined in **Parameters** dialog:

Parameter name	Parameter value (m2)
miu_A_fill	1e-6
miu_A_Spill	6e-9

**Fill Valve 25 - Flow Area controlled by Time/Crank Angle**

Element Name:

Valve timing (start of switching)

Initial state: ☒ Closed ☐ Open

Timing data: ☒ Absolute (T) ☐ Relative (delta T)

Reference: ☒ Calculation start ☐ BTDC (firing)

Time/Crank Angle (CA)

Opening Start s	Closing Start s
0.001	0.003

Data File:

☐ Automatic Reload ☐ Use Data File Path Parameter

---

Effective cross-sectional flow area my\*A

☒ Scaling factor for 2nd column (my)

my\*A of valve seat at opening

	Time/CA s	Area m^2
1	0	0
2	=open_time	=miu_A_fill
3		
4		
5		
6		

Data File:

☐ Automatic Reload ☐ Use Data File Path Parameter

---

☒ Scaling factor for 2nd column (my)

my\*A of valve seat at closing

	Time/CA s	Area m^2
1	0	=miu_A_fill
2	=close_time	0
3		
4		
5		
6		

Data File:

☐ Automatic Reload ☐ Use Data File Path Parameter

### Figure 8-3: Input Dialog of Fill Valve

**Spill Valve 26 - Flow Area controlled by Time/Crank Angle**

---

Element Name  Spill Valve

Valve timing (start of switching)

Initial state:   ☒ Closed             ☐ Open

Timing data:   ☒ Absolute (T)       ☐ Relative (delta T)

Reference:      ☒ Calculation start   ☐ BTDC (firing)

Time/Crank Angle (CA)

Opening Start <small>s</small>	Closing Start <small>s</small>
0.001	0.003

Data File

☐ Automatic Reload   ☐ Use Data File Path Parameter

---

Effective cross-sectional flow area my\*A

☒ Scaling factor for 2nd column (my)  1

my\*A of valve seat at opening

	Time/CA <small>s</small>	Area <small>m²</small>
1	0	0
2	=open_time	=miu_A_spill
3		
4		
5		
6		

Data File

☐ Automatic Reload   ☐ Use Data File Path Parameter

---

my\*A of valve seat at closing

	Time/CA <small>s</small>	Area <small>m²</small>
1	0	=miu_A_spill
2	=close_time	0
3		
4		
5		
6		

Data File

☐ Automatic Reload   ☐ Use Data File Path Parameter

**Figure 8-4: Input Dialog of Spill Valve**


Create three calculation cases (refer to **Case Explorer** description in *Section 4.8*) and assign the following values to `close_time` and `open_time` parameters:

Case No.	Close_time (s)	open_time (s)
1	1e-6	3e-6
2	1e-5	2e-5
3	5e-5	5e-5

### 8.1.1.1. MATLAB® Simulink: General Input Data

In this example, **MATLAB® API: Simulink model** is used. Open input dialog box of **Fill Controller** shown in Figure 8-5. On the left side the dialog element tree is located. In this tree you have three possible choices: **General**, **Input Channels** and **Output Channels**.

Inside **General** input dialog, the complete path (refer to *Section 8.1.2*) and state of an existing Simulink model have to be defined.

Simulink model path can be defined either by typing in the full path or pressing  button at the right end of the input field and selecting Simulink file (\*.mdl).

**Note:** When entering Simulink path, use backslash (\) on PC platforms and slash (/) on Linux/Unix platforms, otherwise BOOST Hydsim will not recognize the path where Simulink model is stored.

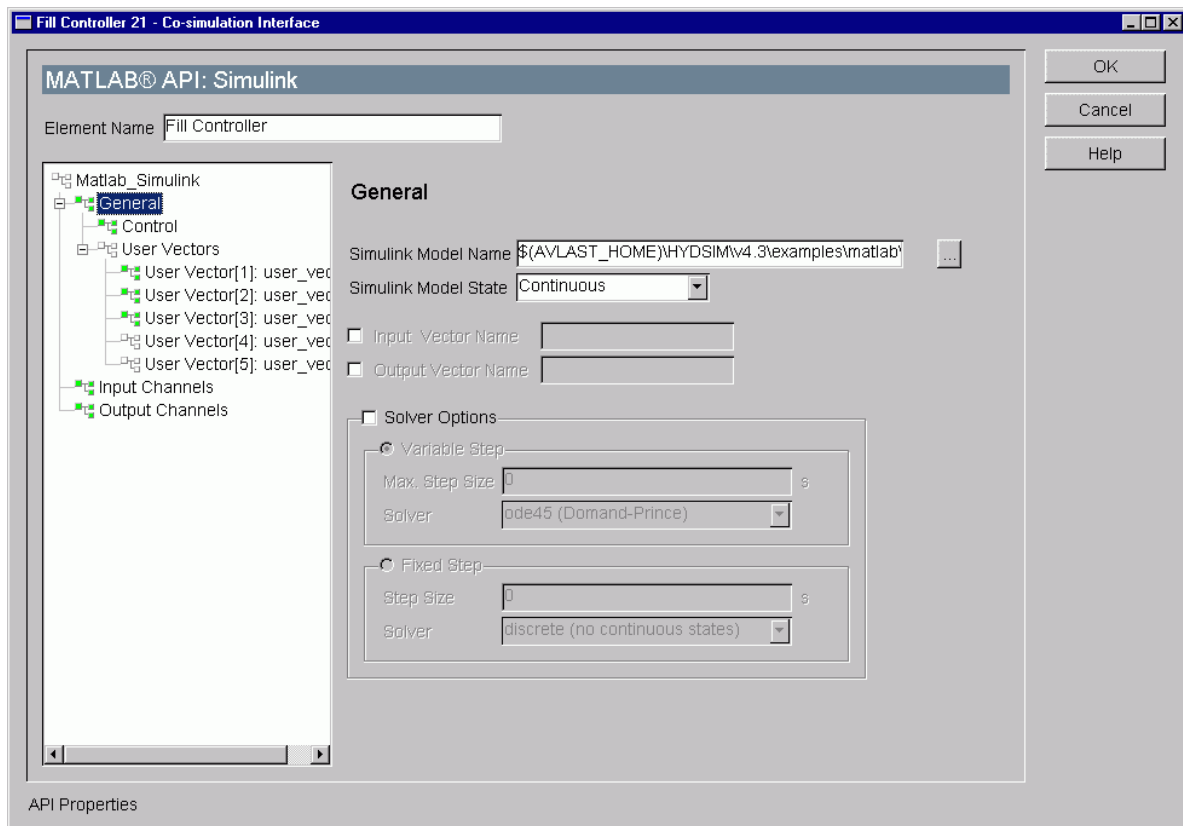


**Note:** It is possible use environmental variables within the Simulink path, e.g. \$AVLAST\_HOME or \$(AVLAST\_HOME). If undefined environmental variable is specified, GUI will issue a warning message. If variable name is incorrect, BOOST Hydsim kernel will issue an error message.

State of Simulink model is set to `Continuous` by default. It can be changed in the Combo-box to `Non-continuous` state.

In our example, we specified the path of the Simulink model (refer to *Section 8.1.2*) using the environmental variable \$(AVLAST\_HOME) and continuous state of this model. If Simulink model is created in **MATLAB®** with **Constant value** element as input and **To Workspace** element as output, then **Input/Output vector** names have to be defined in **General** input dialog.

In the same way, specify Simulink model name and path for **Spill Controller** element.



**Figure 8-5: Input Dialog of Fill Controller**

In **General** input dialog there is a possibility to control the Simulink solver options. For this, activate **Solver Options** button. Default selection is **Variable Step** solvers, 0 for **Max. Step Size** (max. step size will be determined from the start and stop times) and ode45 (Dormand-Prince) for **Solver type**. For the detailed information refer to *Section 8.1.2.1*.

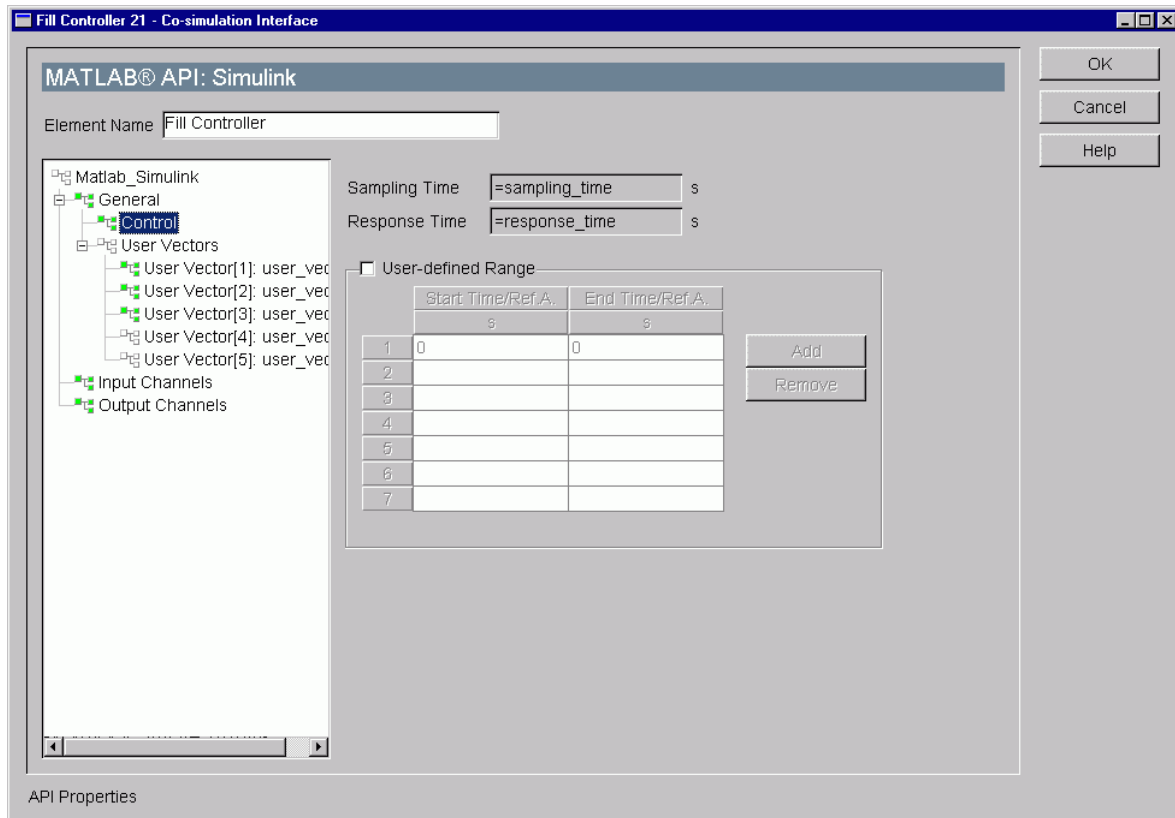


**Note:** Depending on the Simulink model state, an appropriate Solver type has to be selected. For example, Simulink model with continuous states is not compatible with discrete solver type. In this case BOOST Hydsim will display an error message generated by MATLAB®.

**Note:** When BOOST Hydsim calculation is completed, SIMULINK™ model will be closed **without saving** it, i.e. all changes from the BOOST Hydsim interface will be neglected.

### 8.1.1.2. MATLAB® Simulink: Control Input Data

**Control** subdialog box is used for setting control parameters for BOOST Hydsim - MATLAB® co-simulation. Click the Control button and assign the global parameters `sampling_time` and `response_time` to the respective input data.



**Figure 8-6: MATLAB® API Element Control Dialog**

Set these two parameters to the following values in **Case Explorer** dialog:

Case no.	sampling_time (s)	Response_time (s)
1	5e-4	1e-5
2	0.001	1e-4
3	0.0011	1e-4

Specify the same global parameters for **Spill Controller** element.

**Sampling time** is typically the scanning time at which MATLAB® Input Channel (Sensor) reads information from the connected BOOST Hydsim element.

**Response time** is time delay (from sampling event) after which MATLAB® Output Channel (Actuator) sends information to the connected BOOST Hydsim element. Typically it is a reaction time of a physical system represented by the MATLAB® model (due to inertia, wire resistance etc.).

Both parameters **sampling time** and **response time** are set to 0 by default. This implies that data exchange between BOOST Hydsim and MATLAB® will be performed in each calculation step.



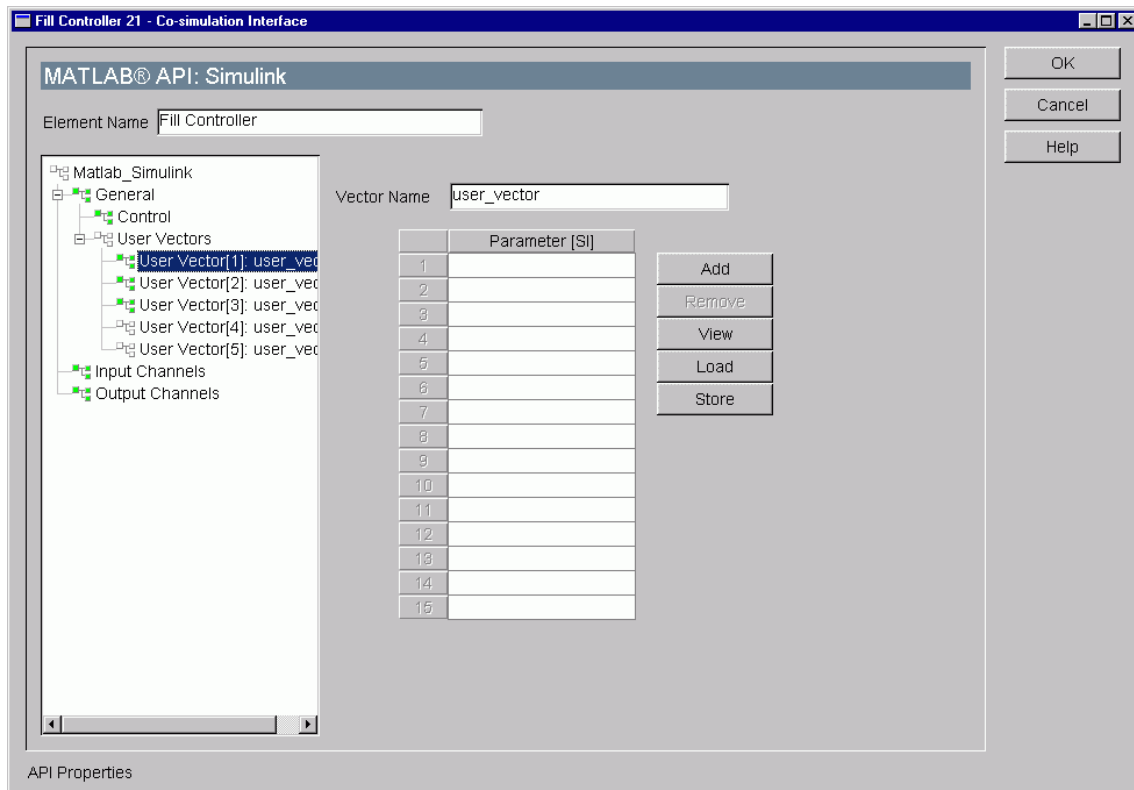
**Note:** If **Sampling/Response time** is smaller than BOOST Hydsim calculation time step, it will be reset to BOOST Hydsim time step. If **Sampling/Response time** is not a multiple of BOOST Hydsim time step, it will be adjusted to the first higher multiple value of time step.

**Note:** **Response time** must be smaller than or equal to the **Sampling time**. Otherwise BOOST Hydsim will issue an error message.

**User-defined Range** serves to define independent BOOST Hydsim -MATLAB® will co-simulation intervals. If not active (default), the co-simulation will be performed within the entire BOOST Hydsim calculation interval. Refer to the [BOOST Hydsim Users Guide](#) for more information.

### 8.1.1.3. MATLAB® Simulink: User Vector Input Data

BOOST Hydsim supports passing of user vectors from BOOST Hydsim into MATLAB® Workspace. In this way, user vectors can be used in the Simulink or m-function model. To define a user vector, open e.g. User vector [1]. The following dialog box will appear:



**Figure 8-7: Input Dialog of User Vector**

Within this dialog box, specify the name in **Vector Name** input field and its entries (parameters) by clicking on **Add**. Click on **View** to view all entries.





**Note:** All entries of user vector must be specified in SI units. Maximum 5 user vectors can be defined. Each vector can have up to 100 entries.

**Note:** It is not necessary to define user vectors in the successive order because BOOST Hydsim will pass to MATLAB® only the vectors with defined names. If e.g. the 1<sup>st</sup>, 3<sup>rd</sup> and 5<sup>th</sup> user vectors are defined, then only these three vectors (with their names) will be passed to MATLAB® Workspace.

**Note:** User vector is set into MATLAB® Workspace as one-dimensional matrix. Thus Simulink (or m-function) can use its entries for simulation.

**Note:** Never use same names for user vectors within one or different MATLAB® elements. Otherwise the vector entries will be overwritten.

In our example, three user vectors are defined: `miu_A_fill`, `Valve_time` and `K_f`. First vector possesses only one entry, effective cross-sectional area of **Fill Valve**, which will be used in the Simulink calculation. Second vector contains 7 entries denoting diverse time constants (refer to *Figure 8-14* for Valve timing). These entries are defined as global parameters. Most of them are specified in **Parameters** dialog:

Vector entry no.	Parameter name	Parameter value
1	<code>Sampling_time1</code>	<code>=sampling_time</code>
2	<code>Calculation_step1</code>	<code>=calculation_step</code>
3	<code>Open_time1</code>	<code>=open_time</code>
4	<code>Close_time1</code>	<code>=close_time</code>
7	<code>Response_time1</code>	<code>=response_time</code>

Double definition of same parameters (with two different names) is necessary due to unit system: user vector requires a SI unit, so a parameter with an arbitrary unit cannot be used there. All parameters are already defined, except `calculation_step` assigned in **Calculation Control** dialog box and set to  $3e-7$  s.

Remaining two parameters are defined in the **Case Explorer** as follows:

Vector entry no.	Parameter name	Parameter values (s)		
		Case 1	Case 2	Case 3
5	<code>open_delay</code>	1e-5	4e-5	5e-5
6	<code>close_delay</code>	1e-5	4e-5	5e-5

The third user vector contains PID controller parameters  $K_p$ ,  $K_i$  and  $K_D$  (refer to *Section 8.1.2* for calculation of their values).

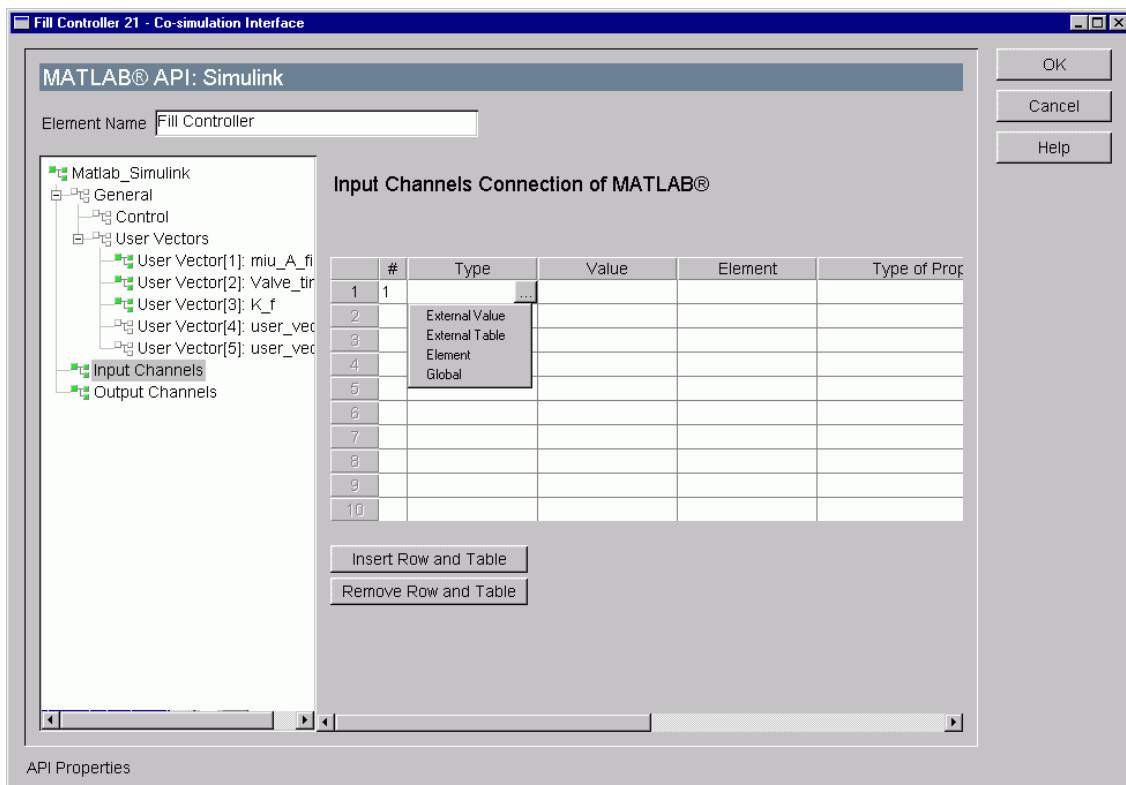
For **Spill controller** the following user vectors are used:

User vector name	Vector entry no.	Parameter name	Parameter value
miu_A_Spill	1	miu_A_spill1	=miu_A_spill

User vector name	Vector entry no.	Vector entry value
K_s	1	78.4
	2	2.4e+9
	3	6.533e-7

#### 8.1.1.4. MATLAB® Simulink: Input Channels Data

Select **Input Channels** in the dialog element tree to open the following:



**Figure 8-8: Input Dialog of Input Channels**

Select **Insert Row and Table** to enter the input parameters. To select the appropriate parameter for the input channel, click inside the input field and select a type of input channel from the submenu.

For the first channel select **External Value**, then enter a constant value in the **Value** column. Set it to 1500e5 Pa in SI unit (1500 bar) which is our reference pressure in Rail Volume.



**Note:** **External Value** and all values in **External Table** must be specified in SI units.

For the second channel select **Element**, then open submenu in the Element column and select **Rail Volume** element. Click into **Type of Properties** column and select there **El. output parameter**. Then click into Property column and select **pressure in volume [Pa]**. In this way, BOOST Hydsim will send pressure (in Pa) of the Rail Volume in the second input channel to the Simulink model.



**Note:** In **Element** column, BOOST Hydsim lists only those elements which are connected to the corresponding MATLAB® element by wire connection.

In the same manner select other input channels as follows:

Channel #	Type	Value	Element	Type of Property	Property
3	Element		Rail Volume (20)	El. local fluid properties	bulk modulus [N/m <sup>2</sup> ]
4	Element		Rail Volume (20)	El. output parameter	actual volume [m <sup>3</sup> ]
5	Element		Fill Valve (25)	El. local fluid properties	fluid density [kg/m <sup>3</sup> ]
6	Element		Pump Volume (23)	El. output parameter	pressure in volume [m <sup>3</sup> ]
7	Global				time [s]

In the # column the numbers are the Port numbers for the Simulink model (refer to *Section 8.1.2*).



**Note:** Changing numbers in # column would be necessary if e.g. the user modifies the numbering of input and/or output channels in Simulink model.

For **Spill Controller** set the following input channels:

Channel #	Type	Value	Element	Type of Property	Property
1	External value	1500e5			
2	Element		Rail Volume (20)	El. output parameter	pressure in volume [Pa]
3	Element		Rail Volume (20)	El. local fluid properties	bulk modulus [N/m <sup>2</sup> ]

4	Element		Rail Volume (20)	El. output parameter	actual volume [m <sup>3</sup> ]
5	Element		Spill Valve (26)	El. local fluid properties	fluid density [kg/m <sup>3</sup> ]
6	Element		Spill Pressure (13)	El. output parameter	pressure [Pa]
7	Global				time [s]

### 8.1.1.5. MATLAB® Simulink: Output Channels Data

Select **Output Channels** in the dialog element tree and specify output channels in the same way as input channels. Select the following properties:

- 1<sup>st</sup> channel – Fill Valve (25) as Element, El. input parameters as Type of Property and switching time T1/delta T1 [s] as Property
- 2<sup>nd</sup> channel – Fill Valve (25) as Element, El. input parameters as Type of Property and switching time T2/delta T2 [s] as Property



**Note:** Number of selected input/output channels has to be the same as the number of input/output channels in the Simulink model. Otherwise, BOOST Hydsim will issue an error message. However, if input/output vector names are used, BOOST Hydsim will not control number of selected input and output channels.

For **Spill Controller** set the following output channels:

- 1<sup>st</sup> channel – Spill Valve (26) as Element, El. input parameters as Type of Property and switching time T1/delta T1 [s] as Property
- 2<sup>nd</sup> channel – Spill Valve (26) as Element, El. input parameters as Type of Property and switching time T2/delta T2 [s] as Property

### 8.1.2. Creating Simulink model

Creating of a Simulink model requires the SIMULINK™ installation of MATLAB® software package. Here we assume that the user is familiar with the SIMULINK™ application.

In our example we use PID controllers for regulating the pressure in the Rail Volume around constant pressure of 1500 bar. As input variable, it uses pressure difference between actual pressure in Rail Volume and rated pressure. Based on this difference, the algorithm calculates the opening time of Fill Valve. Opening time is calculated from the continuity equation (8.1.1) and Bernoulli equation (8.1.2). Opening time has to be large enough to cover the minimum opening and closing time delays and ramps (refer to *Figure 8-14*).

$$\Delta p = \frac{E}{V} \cdot \dot{Q}, \quad (8.1.1)$$

where:

$\Delta p$  ..... pressure difference

$E$  ..... bulk modulus of the fluid

$V$  ..... actual volume of Rail Volume

$\dot{Q}$  ..... cumulative volumetric flow rate

$$\dot{Q} = \mu A \cdot \sqrt{\frac{2 \cdot \Delta p}{\rho}} \cdot \Delta T, \quad (8.1.2)$$

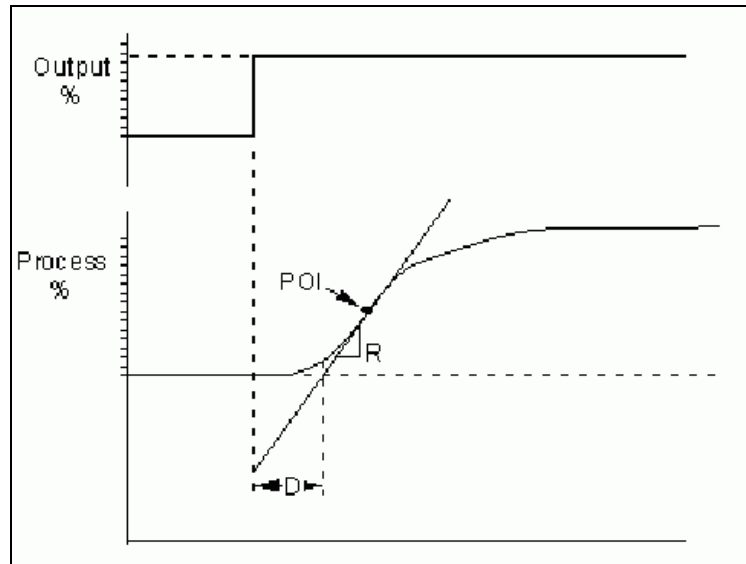
where:

$\mu A$  ..... effective cross-sectional area of Fill Valve

$\rho$  ..... fluid density

$\Delta T$  ..... opening time

To determine all three PID controller elements (Proportional, Integral and Derivative), Ziegler Nichols tuning method (open loop reaction rate<sup>9</sup>) is used. This method, known as the "reaction curve" method, is illustrated in Figure 8-9.



**Figure 8-9: Controlled Process Graph**

Here the following notations are used:

$X (\%)$  ..... change of output (linear relationship between output and process is assumed)

$R (\%/min)$  ..... rate of change at the point of inflection (POI)

<sup>9</sup> John Shaw, The PID Control Algorithm: How it Works and how to Tune It

$D$  (min) .....time till interception of tangent line and original process value

In our example, we use the following values:

$$X = \frac{\Delta p}{p_{set}} \cdot 100(\%) = 0,6667\%$$

where:

$\Delta p$ (assumed) ..... pressure difference (10 bar),

$p_{set}$  ..... rated pressure (1500 bar).

$$R = \frac{1}{\Delta T} = 612245 \frac{\%}{\text{min}},$$

where:

$\Delta T$  ..... reaction time (Interaction Time-2\*D=1.6333e-6 min)

$D = 1,667 \cdot 10^{-8}$  min(assumed)

The Proportional, Integral and Derivative constants are calculated by:

$$\text{Proportional: } K_p = 1,2 \cdot \frac{X}{D \cdot R} = 78,4.$$

$$\text{Integral: } K_i = \text{Gain} \cdot \text{Re set} = 2,4 \cdot 10^9,$$

where:

$$\text{Gain} = K_p$$

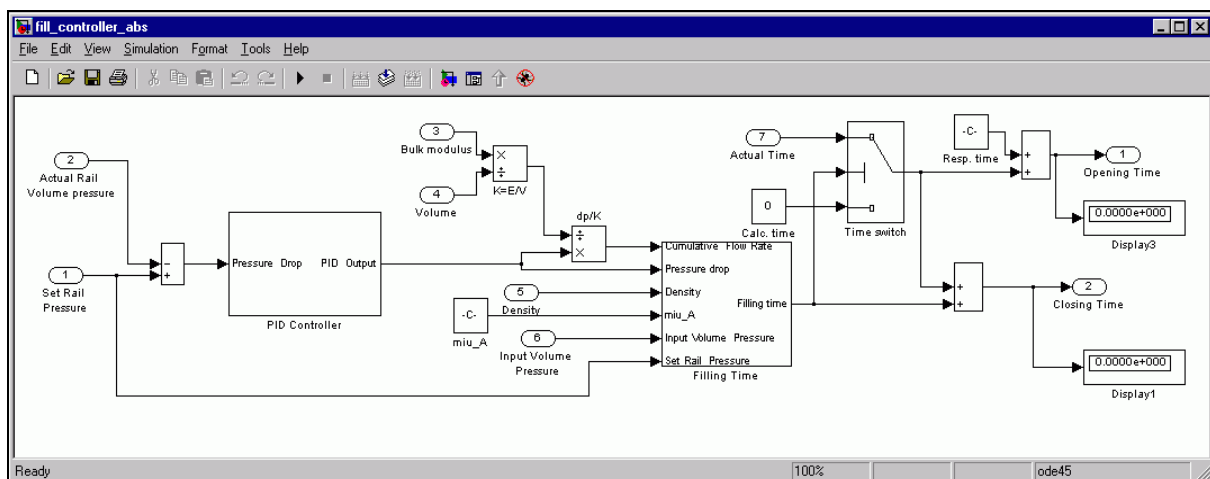
$$\text{Re set} = \frac{0,5}{D}$$

$$\text{Derivative: } K_d = \text{Gain} \cdot \text{Derivative} = 6,533 \cdot 10^{-7},$$

where

$$\text{Derivative} = 0,5 \cdot D.$$

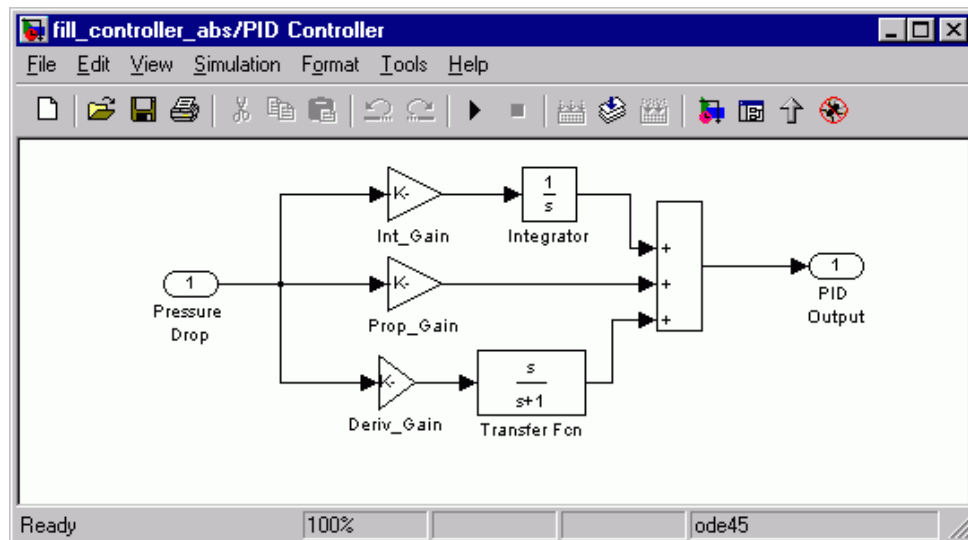
Figure 8-10 shows the Simulink model of **Fill Controller** based on the above definitions.



**Figure 8-10: Simulink Model of Fill Controller**

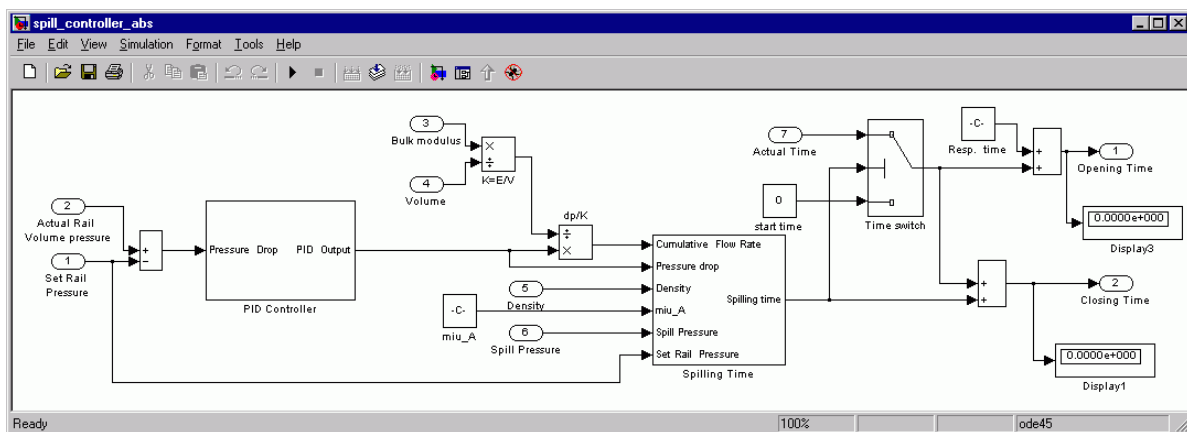
Within this Simulink model (file `fill_controller_abs.mdl`), three user vectors (refer to *Section 8.1.1.3*) are used for PID coefficients. They are defined in BOOST Hydsim model as vectors  $K_f$ , effective cross-sectional Area  $\mu_{iu\_A}$  and  $Valve\_time$ .

PID controller itself is defined by the subsystem `PID Controller` (refer to *Figure 8-11*) with  $K_f(1)$  as proportional coefficient,  $K_f(2)$  as integral coefficient and  $K_f(3)$  as derivative coefficient.



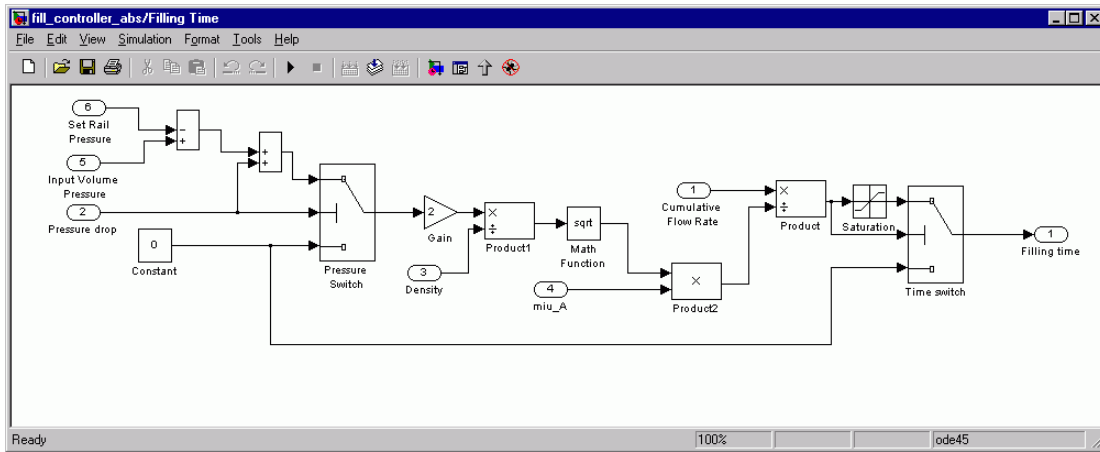
**Figure 8-11: Simulink Model of PID Controller**

In the same way, the Simulink model of **Spill Controller** shown in *Figure 8-12* is built up. It contains analogous PID components, but with the opposite calculation algorithm for pressure difference

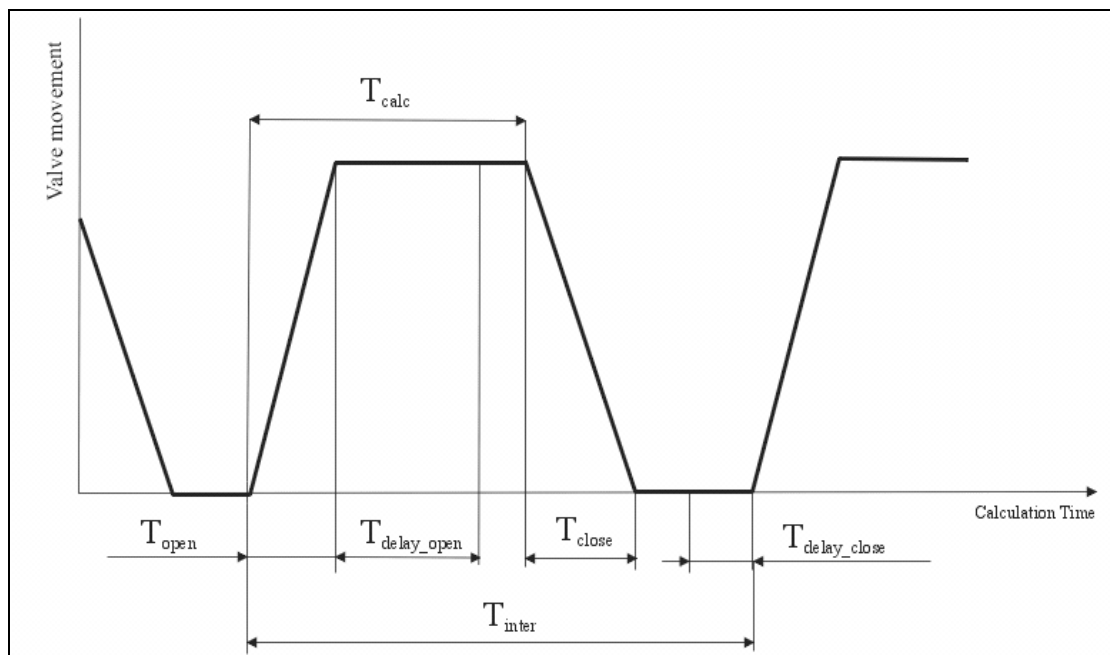


**Figure 8-12: Simulink Model of Spill Controller**

Calculation of opening time of **Fill Valve** is performed by `Filling Time` subsystem shown in *Figure 8-13*. This subsystem calculates respective opening time  $T_{calc}$  (refer to *Figure 8-14*) according to the Bernoulli equation (8.1.2) and predefined timing characteristics of the valve movement. All timing data are defined in the user vector  $Valve\_time$ .



**Figure 8-13: Simulink Model for Filling Time Calculation**



**Figure 8-14: Timing of Valve Movement**

Calculated opening (filling) time has to satisfy the following bounds:

$$(T_{calc})_{min} = T_{open} + T_{delay\_open},$$

$$(T_{calc})_{max} = T_{inter} - (T_{close} + T_{delay\_close}),$$

where:

$T_{calc}$  ..... calculated filling time

$(T_{calc})_{min}$  ..... minimum possible filling time

$(T_{calc})_{max}$  ..... maximum possible filling time

$T_{open}$  ..... opening time (from 0 to max. lift)

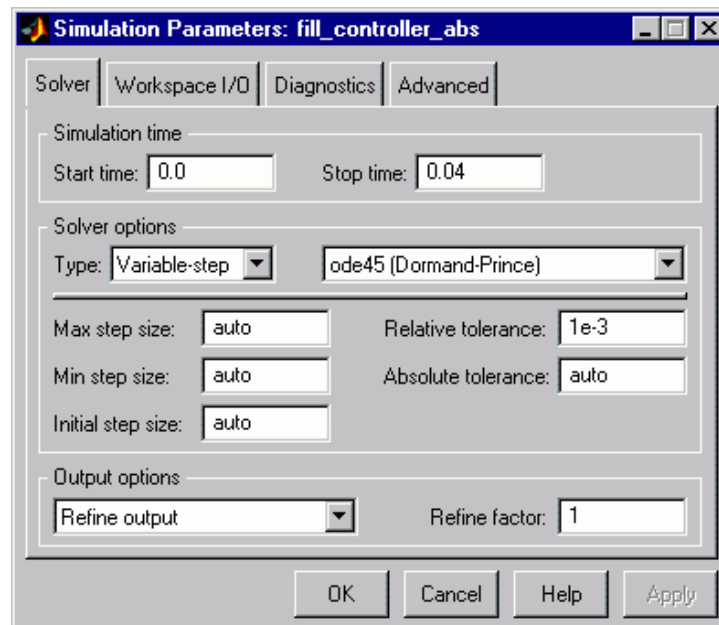
$T_{delay\_open}$  ..... min. opening delay (specified)



$T_{inter}$  ..... sampling (interception) time  
 $T_{close}$  ..... closing time (from max. lift to 0)  
 $T_{delay\_close}$  ..... min. closing delay (specified)

### 8.1.2.1. Solver options

After having created the Simulink model, set the calculation parameters: Solver and Workspace I/O options. Open it from **Simulation | Simulation parameters** PullDown menu in the Simulink model of **Fill Controller**.



**Figure 8-15: Solver Options**

In the **Solver** window shown in Figure 8-15, **Simulation time** (Start and Stop) has no meaning because BOOST Hydsim will reset it during calculation. The entries for **Solver Options** can be either set there to the desired values or changed in BOOST Hydsim during pre-calculation (refer to *Section 8.1.1.1*).

User can choose incrementation between **Variable-step** (modified step size during simulation) and **Fixed-step** solver (constant step size during simulation) and select type of Solver. The default option is **ode45** for the variable-step Solvers and **discrete** for the fixed-step Solvers.

For variable-step solvers, the user can set the maximum and initial step size parameters. By default, indicated by the value **auto**, these parameters are determined automatically. The **Max. step size** parameter controls the largest time step the solver can take. The default step size is calculated from the Start and Stop times as:

$$\text{Max. step size} = \frac{t_{stop} - t_{start}}{50}.$$

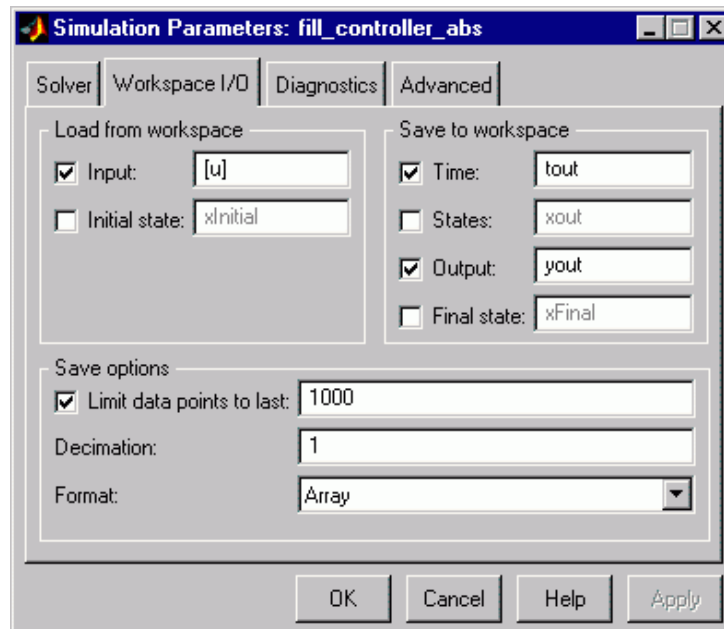
The **Initial step size** parameter is a suggested first step size.

For fixed-step solvers, the user can set the **Fixed step size** which is set to `auto` by default (default value is determined according to the same equation as default **Max. step size** for variable-step solvers).

BOOST Hydsim will set **Output options** to `Refine` output with **Refine factor** 1 during co-simulation because it gets only the last values from the output vectors as simulation results.

### 8.1.2.2. Workspace options

Select the **Workspace I/O** tab in **Simulation parameters** dialog box.



**Figure 8-16: Workspace I/O Options**

It is not necessary to select any parameter in this dialog box because BOOST Hydsim will set all required parameters. If the Simulink model is created with input port elements (no **Input vector** name - refer to *Figure 8-5*), it will set **Load from workspace** as **Input vector** `[u]`, otherwise this parameter will be disabled. External Input vector `u`, will have one row with  $n+1$  columns, where  $n$  is the total number of the model input ports (first entry is the simulation start time).

If `Continuous` model state is selected (refer to *Figure 8-5*), **Final state** and **Initial state** will be set in MATLAB® Workspace as `FinalState_m` during co-simulation, where  $m$  is the order number of MATLAB® element in BOOST Hydsim model. It means that BOOST Hydsim will take `Final State` from previous simulation as `Initial State` for the actual co-simulation (as `Initial State` at co-simulation start is 0).

**Time, States and Output** vectors are set to `[time]`, `[state]` and `[out]` vectors, respectively.

### 8.1.3. Output Parameters

BOOST Hydsim supports the viewing of all input and output channel parameters of MATLAB® API element in Impress Chart post-processor. To perform this, select desired output parameters from the **Element | Store Results** menu. These output results may be used to control the BOOST Hydsim -MATLAB® co-simulation.

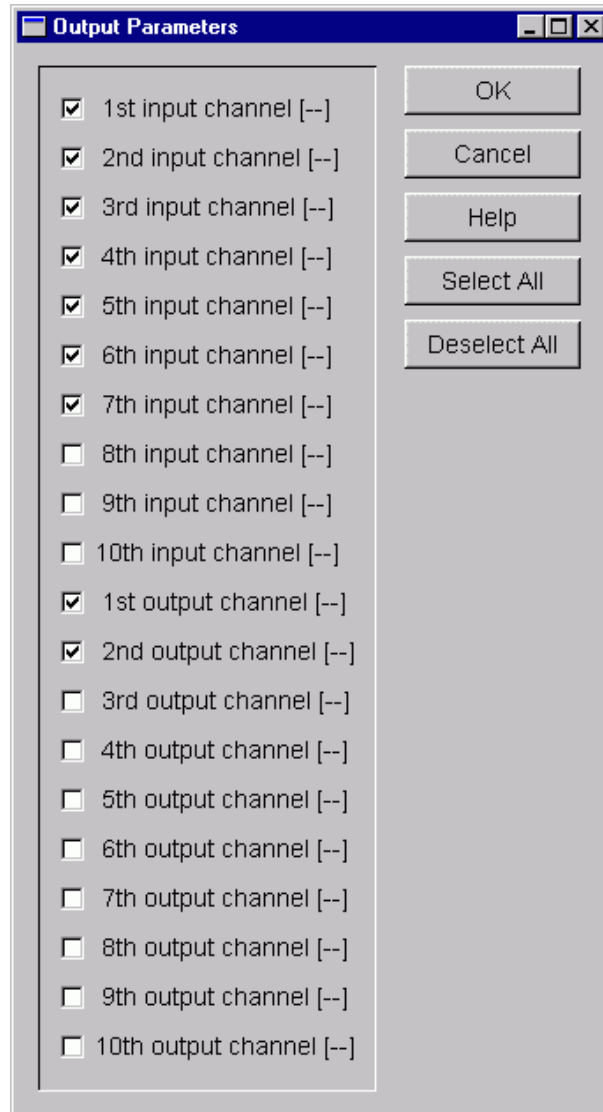


Figure 8-17: Output Data of MATLAB API Element

### 8.1.4. Running the Calculation

There are no **Initial Conditions** in MATLAB® API: Simulink element.

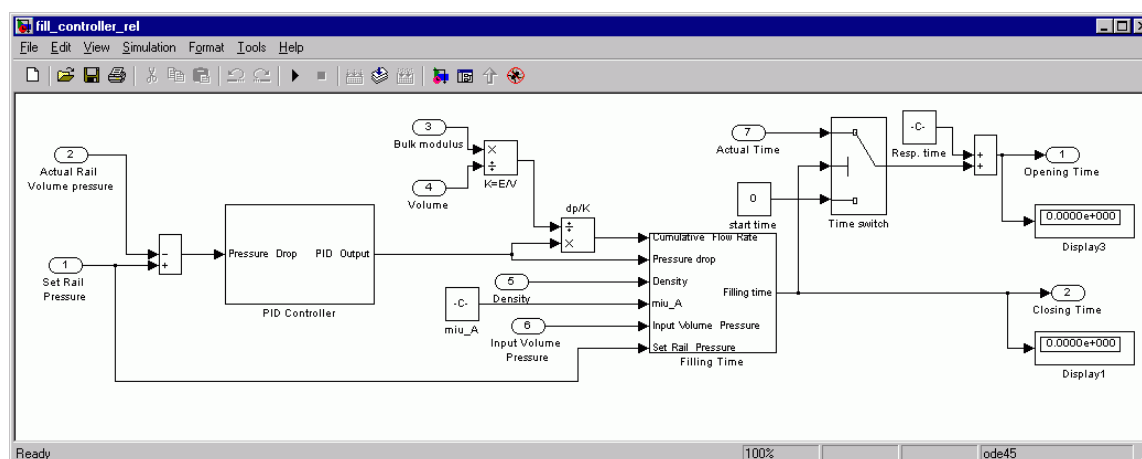
For performing the calculation, refer to *Section 4.4*.

In this example, Opening/Closing times of Fill and Spill Valves are defined in absolute time domain. Alternatively, it is possible to specify them in the relative time domain. For this, switch **Timing data** button in the input dialog of Fill and Spill Valve to **Relative (delta T)** as shown in Figure 8-18. Adjust the Simulink model accordingly (refer to *Figure 8-19*).

Save the adjusted Simulink model and BOOST Hydsim model under different names, e.g. `fill_controller_rel.mdl` and `comrail2wv_rel_md1_1.hyd` respectively.

Figure 8-18 shows the input dialog box for the Fill Valve. The 'Valve timing (start of switching)' section is configured with 'Initial state' set to 'Closed', 'Timing data' set to 'Relative (deltaT)', and 'Reference' set to 'Calculation start'. The 'Time/Crank Angle (CA)' section shows a table for 'Opening Start' and 'Closing Start' times. The 'Effective cross-sectional flow area my\*A' section shows two tables: 'my\*A of valve seat at opening' and 'my\*A of valve seat at closing'. The 'Opening' table has a row for 'open\_time' and 'miu\_A\_fill'. The 'Closing' table has a row for 'close\_time' and '0'. There are also 'Data File' fields and checkboxes for 'Automatic Reload' and 'Use Data File Path Parameter'.

**Figure 8-18: Input Dialog Box of Fill Valve with Relative Timing Data**



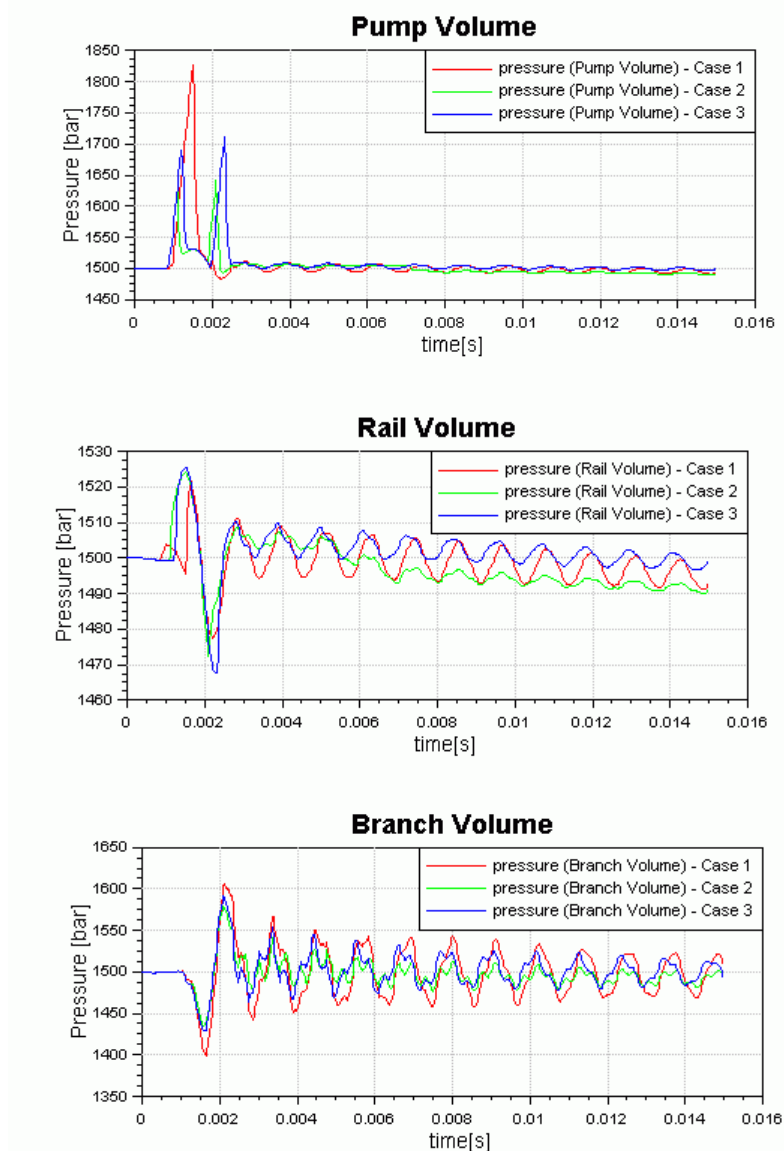
**Figure 8-19: Simulink Model of Fill Valve with Relative Timing**

### 8.1.5. Calculation Results

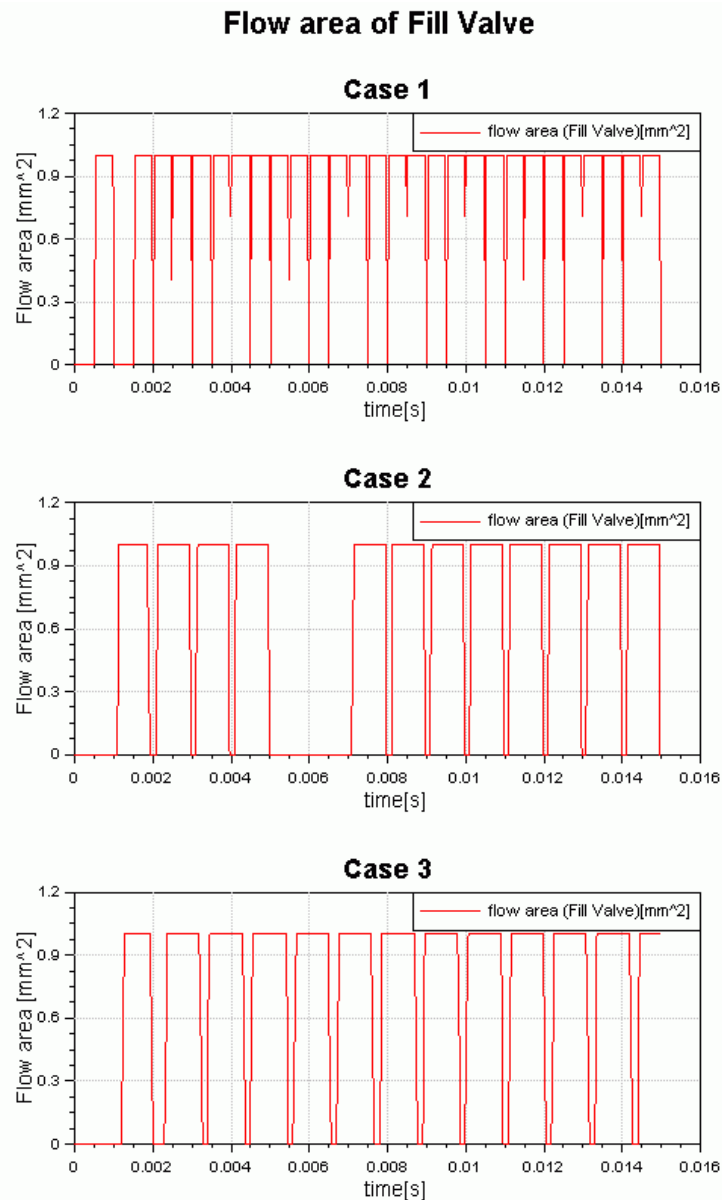
In this section, the calculation results of three cases defined in **Case Explorer** are shown, for the BOOST Hydsim model with absolute timing values. Remember, each case contains individual sampling and response times and different timing constants of the Fill and Spill Valves.

To view the calculation results, select **Show Results** from **Simulation** Pulldown menu. The new Impress Chart window with the **Result** tree of the active BOOST Hydsim model will pop up.

Figure 8-20 shows the pressures in Rail Volume, Pump Volume and Branch Volume. Figure 8-21 depicts effective flow area of Fill Valve, for each case separately. Analysis of the calculation results is left here for the user as an exercise.



**Figure 8-20: Pressure in Pump, Rail and Branch Volumes**



**Figure 8-21: Flow Area of Controlled Fill Valve**

### 8.1.6. Running the Calculation with Case Sets

We use the basic model `comrail2wv_abs_mdl_1.hyd` for extending it with three **Case Sets**. Assume we need to run the model with three different rail pressures, and within each pressure case we have to vary the volume of fuel rail. An intelligent way to set up and perform such a task is to use **Case Sets**.

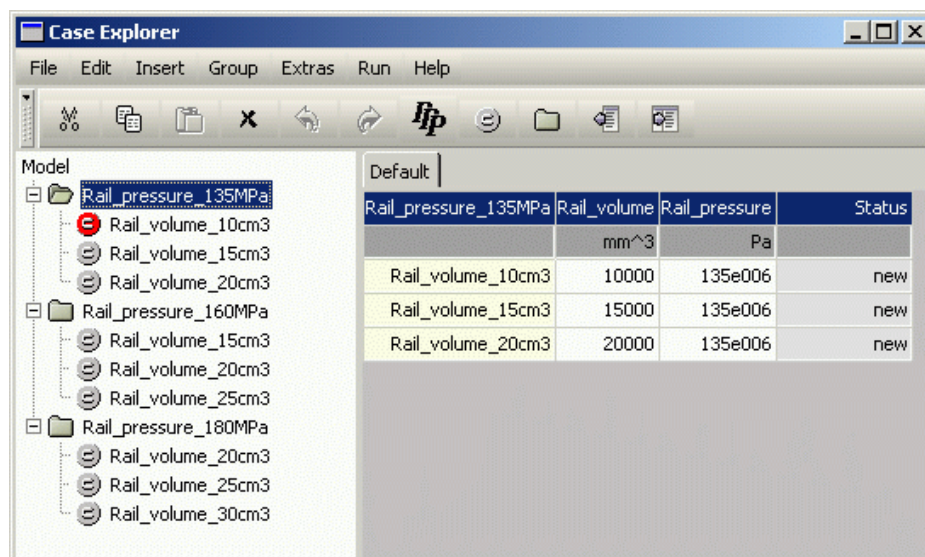
Rail pressure is used in the Pump Volume (element No. 23), Rail Volume (No. 20), Branch Volume (No. 2), Control Volume (No. 6) and Valve Volume (No. 7). Define initial pressure in these volumes to global parameter `Rail_pressure`.

Assign global parameter `Pump_volume` to the Initial volume of Pump Volume, `Rail_volume` to the Initial volume of Rail Volume. Next, assign local parameter `Over_pressure` to the Opening/closing pressure of Over-pressure Valve (No. 24).

Open input dialog boxes of Fill and Spill Controller and assign External Value in the first Input Channel/Value column to global parameter Rail\_pressure1.

Create three **Case Sets** and set the parameter Rail\_pressure to 1350, 1600 and 1800 bar, respectively. Rename the **Case Sets** to Rail\_pressure\_135MPa, Rail\_pressure\_160MPa and Rail\_pressure\_180MPa, respectively. Under each **Case Set** create three **Cases** with different values for Rail\_volume. Supply appropriate name for each case as shown in table below.

Case Set	Case name	Rail_volume (mm <sup>3</sup> )	Rail_pressure (Pa)
Rail_pressure_135MPa	Rail_volume_10cm3	10000	135e6
	Rail_volume_15cm3	15000	
	Rail_volume_20cm3	20000	
Rail_pressure_160MPa	Rail_volume_15cm3	15000	160e6
	Rail_volume_20cm3	20000	
	Rail_volume_25cm3	25000	
Rail_pressure_180MPa	Rail_volume_20cm3	20000	180e6
	Rail_volume_25cm3	25000	
	Rail_volume_30cm3	30000	



**Figure 8-22: Three Case Sets**

Within **Model Parameters** dialog set global parameter Pump\_volume to Rail\_volume/10, global parameter Rail\_Pressure1 to Rail\_pressure and local parameter Over-pressure to 180 MPa.

Set parameters sampling\_time to 0.001 s and response\_time to 1e-4 s.

Save the model under different name, e.g. comrail2wv\_abs\_3case\_sets.hyd. Run the calculation for all three **Case Sets**, i.e. nine cases altogether.

Figure 8-23 shows the results for the Rail Volume pressure. Each layer represents one **Case Set** and contains three curves corresponding to different Rail Volume sizes.

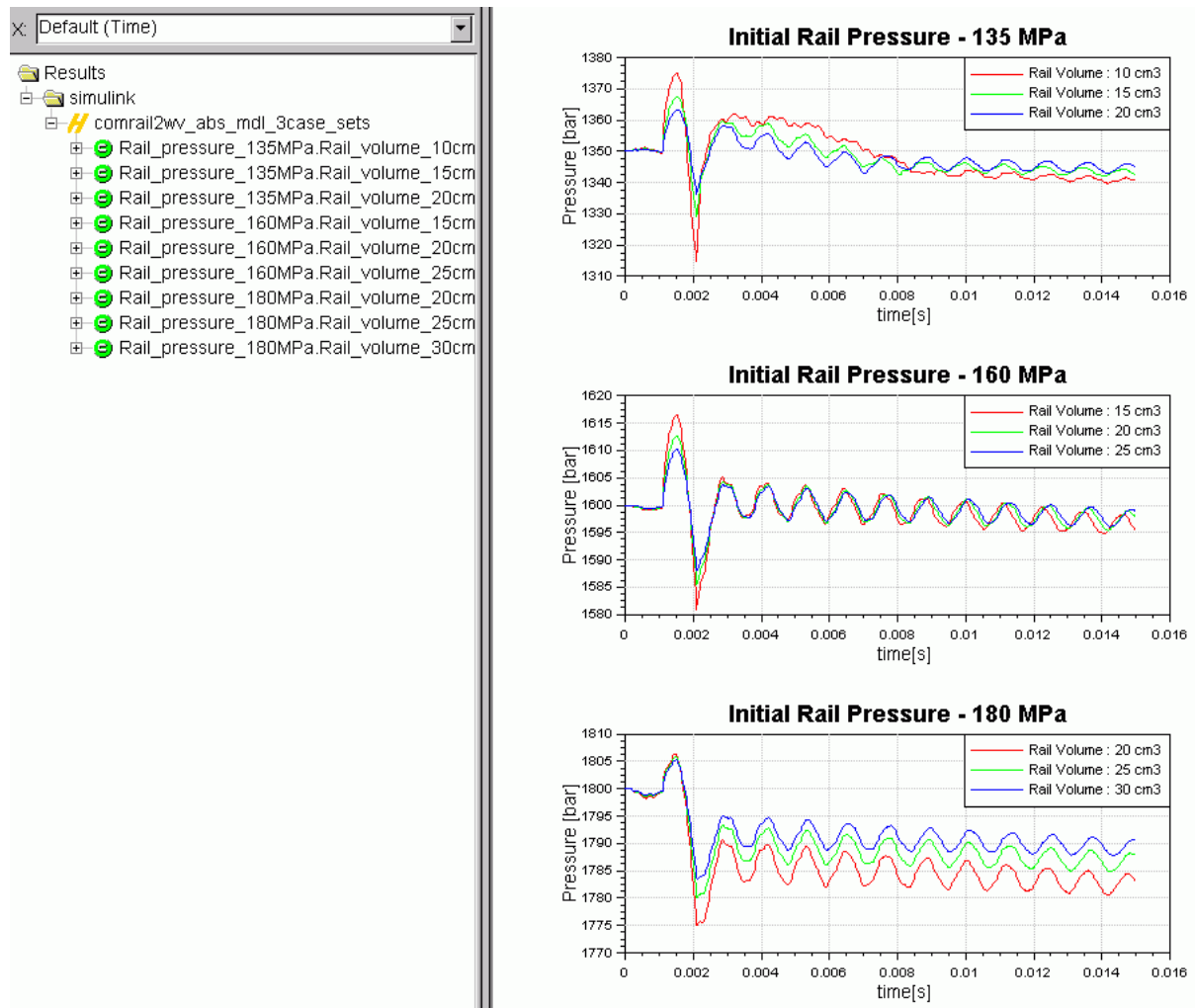


Figure 8-23: Pressure in Rail Volume for Three Case Sets

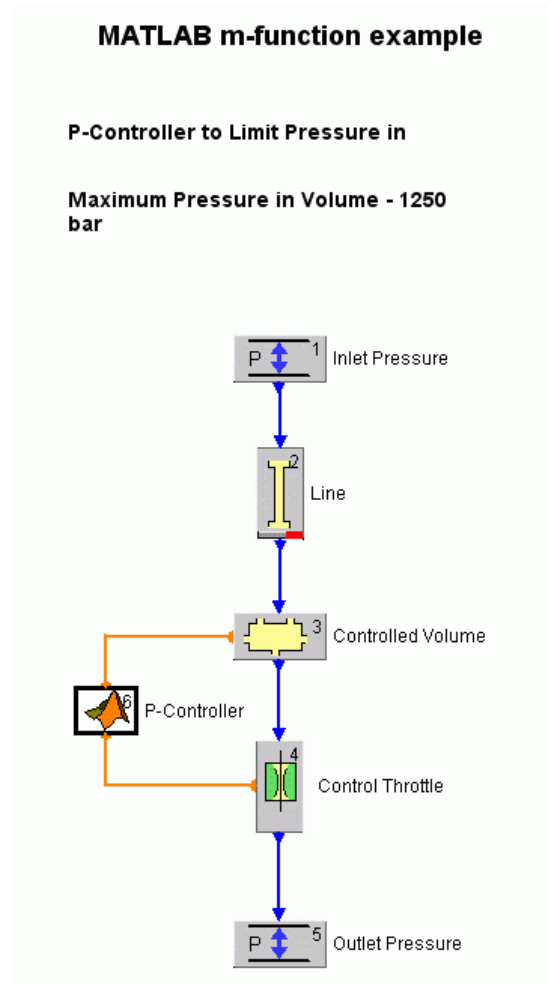
## 8.2. Example with MATLAB® m-function

In this example, an application of **MATLAB® m-function** element is shown. It represents a P-controller in BOOST Hydsim model of a simple pipe-volume-orifice system.

The model of the system is shown in *Figure 8-24*. It consists of Input Pressure (boundary condition) with variable pressure, Line, Controlled Volume, Control Throttle and Leak Pressure (1 bar). P-Controller (**m-function** element) has to regulate pressure in the Controlled Volume so that it does not exceed the limiting value (1250 bar in our case). Control parameter is the diameter of the Control Throttle.



If the pressure in the system increases beyond the limiting value, P-Controller has to open Control Throttle, thus initiating the fluid flow from Controlled Volume to Leak Pressure. As soon as the pressure drops below the limiting value, P-Controller has to close the Control Throttle again.

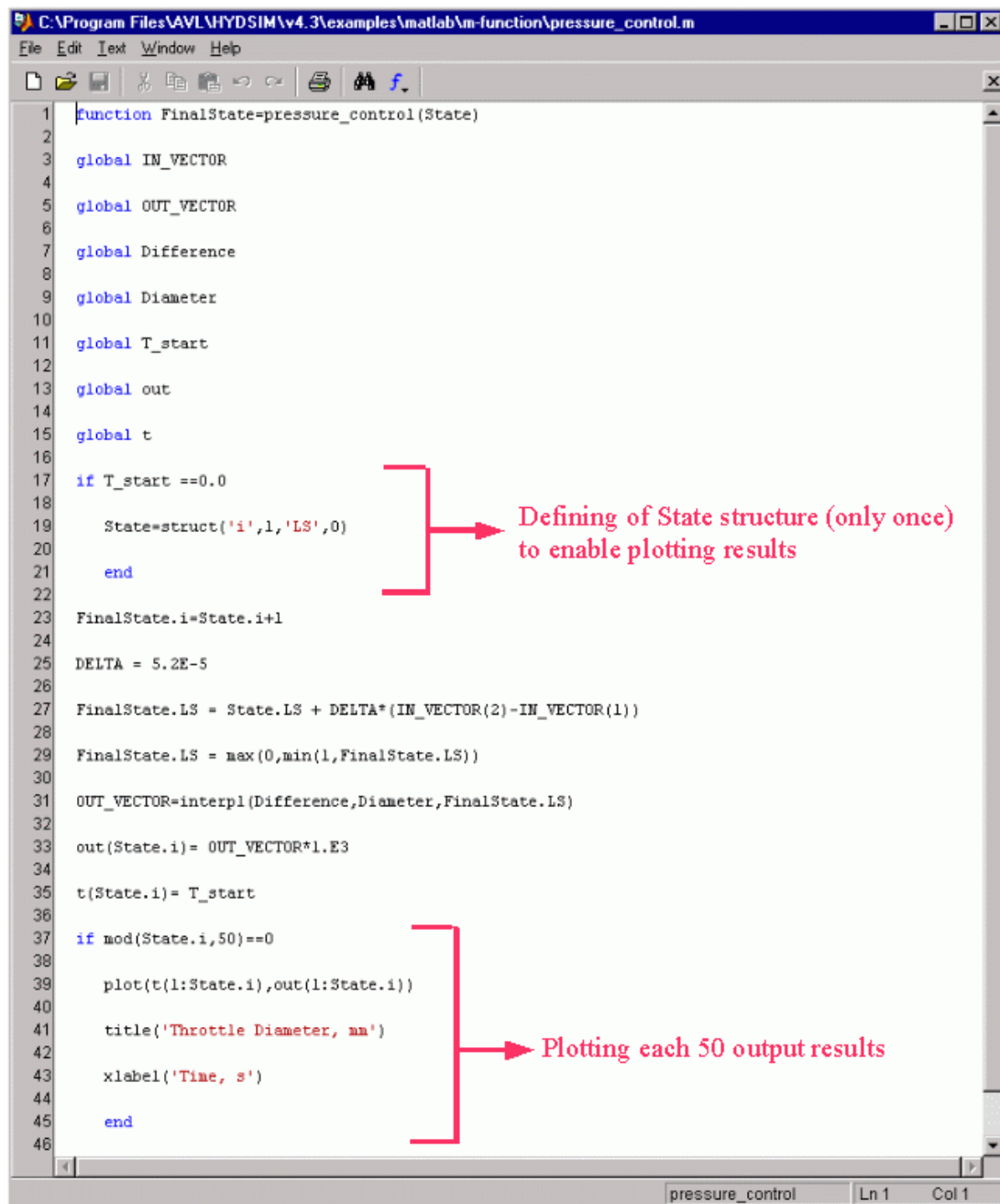


**Figure 8-24: BOOST Hydsim Model with P-Controller as MATLAB® m-function**

### 8.2.1. Creating m-function

The m-function is created with the M-file Editor of MATLAB® software. Here we assume that the user is familiar with the M-file Editor.

As stated earlier, in our example we use P-controller to regulate pressure in Controlled Volume. As Input variable, the controller uses pressure difference between actual pressure of Controlled Volume and rated (limiting) pressure. According to this difference the cross-section diameter of Control Throttle is calculated.



**Figure 8-25: M-function for P-Controller**

The screen shot of the m-function is shown in Figure 8-25. The m-function has to be declared as `function FinalState=function name(State)` because BOOST Hydsim uses this statement while calling m-function. For each calculation time step, program will assign `FinalState` variable to `State` variable, which will be used as initial state for the next calculation step.

Input and output vectors must be declared as global variables. These vectors must be defined in BOOST Hydsim with the same names (refer to *Section 8.2.2.1*).

In this example, two additional vectors `Difference` and `Diameter` are used. These vectors define the functional relationship: effective diameter of Controlled Throttle vs. pressure difference.

According to it and calculated `FinalState` variable, m-function interpolates the value for output vector (cross-section diameter of Control Throttle). The above vectors are defined in BOOST Hydsim model as user vectors (refer to *Section 8.2.2.1*).

The last part of the m-function serves to plot the calculated diameter of Control Throttle vs. time. For visualization purpose each 50<sup>th</sup> step is plotted during co-simulation (refer to *Figure 8-30*).

Save the m-function on file under the name `pressure_control.m`.

## 8.2.2. Input Data

In this section, the input data of Control Throttle and P-Controller will be discussed.

Control Throttle 4 - Flow with User-defined Resistance

Element Name: Control Throttle

Fluid Properties...

☐ Cross-section Area ☒ Cross-section Diameter

		tube	throttle
Diameter	mm	2	0

☒ Constant flow resistance in throttle

☒ Flow Resistance coef. ☐ Flow Rate

Flow resistance coef. (Z>1): 1 [-]

☐ Variable flow resistance in throttle

	Pressure ratio [-]	Resistance coef. [-]
1	1	1
2		
3		
4		
5		
6		
7		

Buttons: Insert row, Remove row, View..., Load..., Store...

Data File:

☐ Automatic Reload ☐ Use Data File Path Parameter

☐ Expansions/Contractions at Connections

		input end	output end
Cross-sect. area	m <sup>2</sup>		
Posit. flow resist.	[-]	0.5	1
Negat. flow resist.	[-]	1	0.5

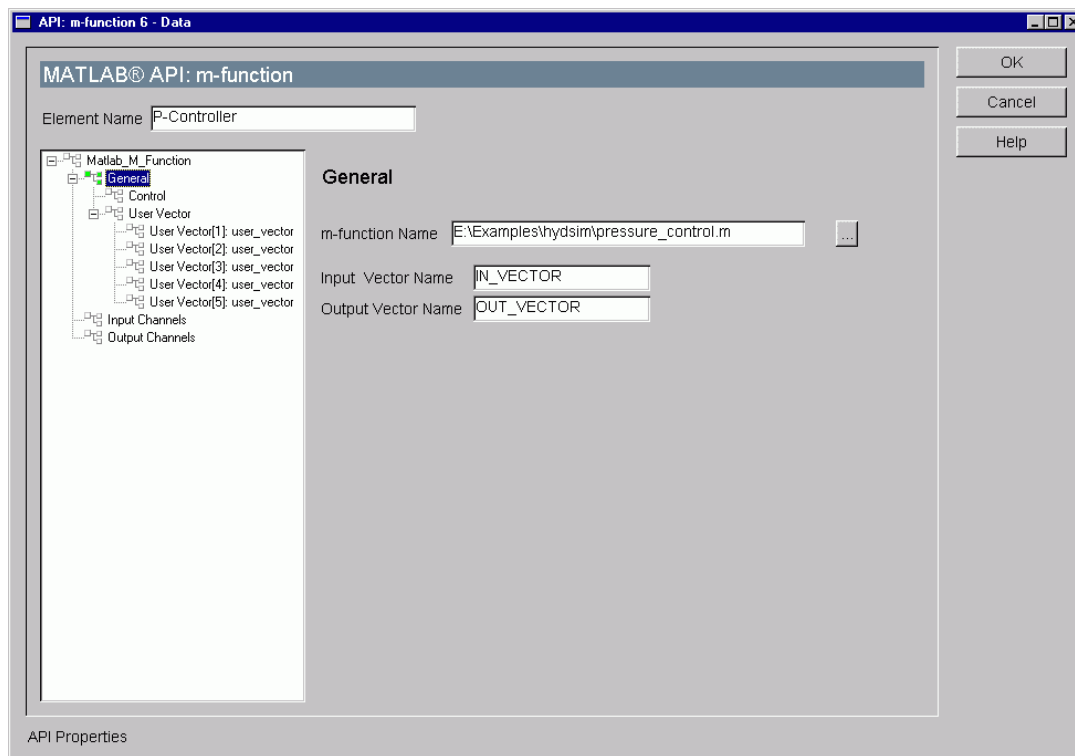
Buttons: OK, Cancel, Help

**Figure 8-26: Input Data of Control Throttle**

Input dialog box of Control Throttle is shown in Figure 8-26. At calculation start Control Throttle is closed (its cross-section diameter is set to 0). For simplicity, the flow resistance coefficient is set to 1, i.e. the calculated throttle diameter is not a geometric but an effective diameter. If the pressure in Controlled Volume increases beyond 1250 bar, P-Controller will calculate the new effective diameter of Control Throttle.

### 8.2.2.1. MATLAB® m-function: Input Data

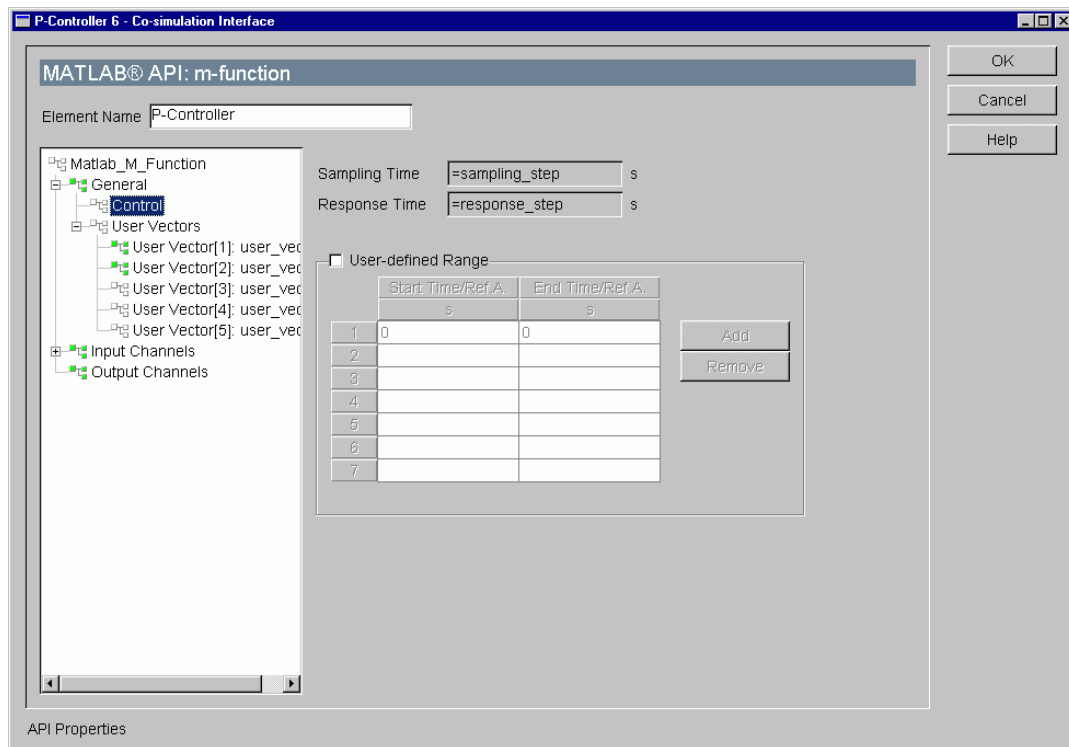
Open the input dialog box of P-controller. In General subdialog shown in Figure 8-27, define the complete path name of m-function and the names of input/output vectors. These vectors have to correspond to the respective vectors in the m-function (refer to *Section 8.2.1*).



**Figure 8-27: General Input Dialog Box of P-Controller**

In Control subdialog (Figure 8-28), assign global parameters `sampling_step` and `response_step` to the corresponding input data (refer to *Section 8.1.1.2* for their definition). Create three cases and set the above parameters to the following values:

Case no.	response_step (s)	sampling_step (s)
1	2e-7	5e-7
2	4e-7	1e-6
3	5e-7	2e-6



**Figure 8-28: Input Dialog Box of P-Controller**

Define a user vector by click on the `User Vector[1]` in the dialog element tree (refer to Section 8.1.1.3 for creating the user vectors). As already mentioned, two user vectors `Difference` and `Diameter` are used in our example:

`Difference`: 0, 0.1, 0.25, 0.5, 0.75, 1

`Diameter`: 0, 0.0005, 0.0008, 0.001, 0.0012, 0.0013

Now open Input and Output Channels dialog boxes to define input and output channels. Specify two input and one output channels (refer to Section 8.1.1.4 and Section 8.1.1.5 for selecting input and output channels, respectively).

Select `External Value` in the first input channel and assign it to global parameter `max_pressure` in **Value** column. Set its value to 1250e5 in **Parameters** dialog. In the second input channel select pressure in Controlled Volume as follows: Controlled Volume (3) in **Element** column, El. output parameters in **Type of Property** column and pressure in volume [Pa] in **Property** column.

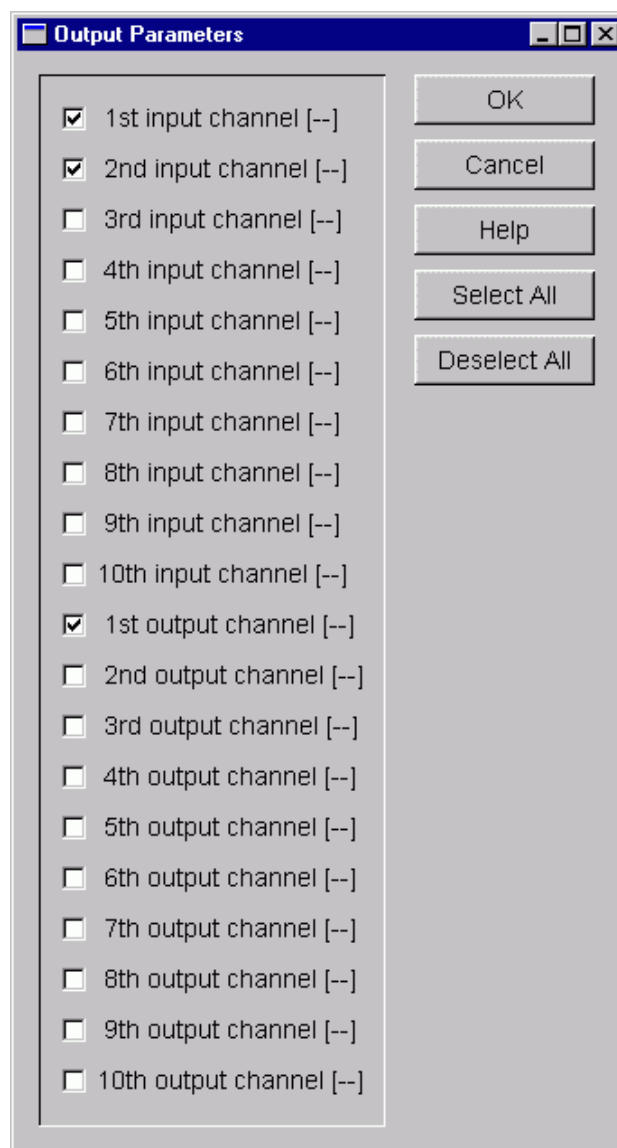
In Output channel, select the diameter of Control Throttle in the following way: Control Throttle (4) in **Element** column, El. input parameters in **Type of Property** column and throttle cross-section diameter [m] in **Property** column).

Save the model under the name `pressure_control_m_1.hyd`.

### 8.2.3. Output Parameters

Dialog Output Parameters of **MATLAB® m-function** element is the same as of **MATLAB® Simulink** element (refer to Section 8.1.3).

Select the first and second input and first output channel as shown in Figure 8-29.



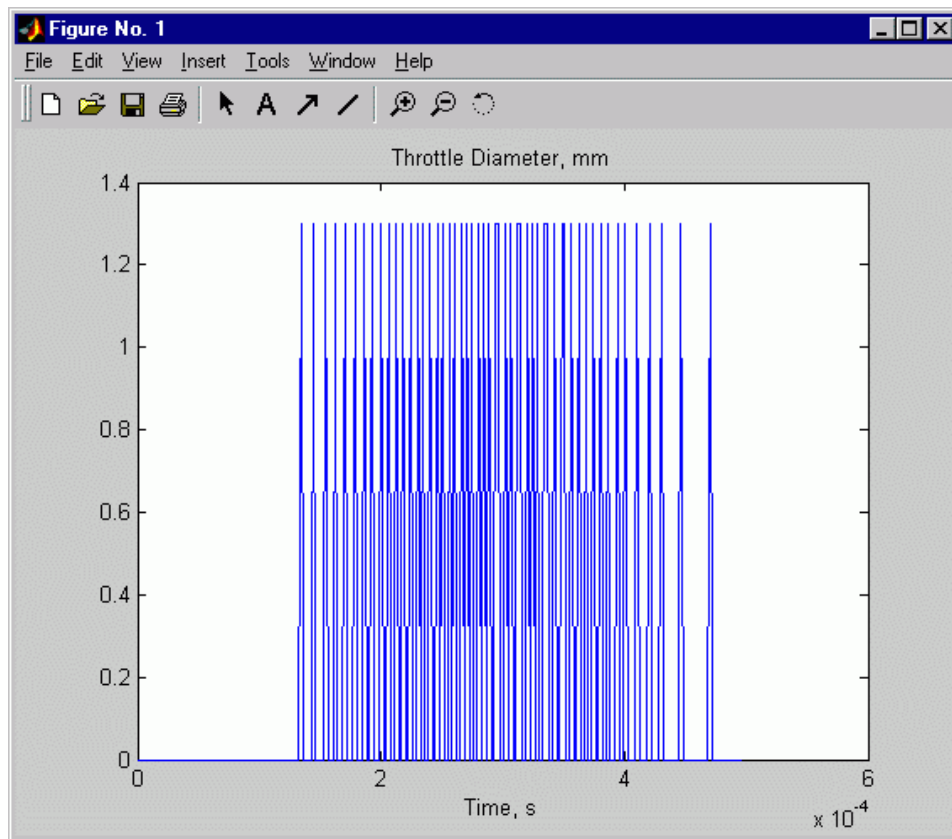
**Figure 8-29: Output Data of MATLAB m-function Element**

### **8.2.4. Running the Calculation**

There are no **Initial Conditions** in MATLAB® API: m-function element.

Perform the calculation with all three cases (refer to *Section 4.4*).

Note that during calculation the plot of m-function output results appears on the screen as shown in Figure 8-30. It depicts the actual value of the effective diameter of Control Throttle after each 50<sup>th</sup> calculation step.



**Figure 8-30: Output Results of m-function at Co-simulation**

### 8.2.5. Calculation Results

Figure 8-31 shows the calculation results for all three cases. Upper graph depicts the pre-defined boundary pressure in Input Pressure element and calculated pressures in Controlled Volume element. Middle graph shows the cumulative flow rates through Control Throttle resulting from the P-Controller performance. Bottom graph displays the input parameters of P-Controller: rated (limiting) pressure and actual pressure. The 1<sup>st</sup> case has the smallest sampling and response steps ( $5e-7$  and  $2e-7$  s, respectively). Naturally it appears to be the best theoretical controller because it regulates the pressure very smoothly. However, such fast controller is practically impossible to realize. The 2<sup>nd</sup> case has larger sampling and response steps ( $1e-6$  and  $4e-7$  s, respectively). The control efficiency in this case is still very good but the pressure trace becomes somewhat noisy. The 3<sup>rd</sup> case has more realistic sampling and response steps ( $2e-6$  and  $5e-7$  s, respectively). In this case the controller reacts much slower to the pressure changes, therefore it cannot prevent short pressure excursions above the limit (1250 bar). Nevertheless, even in this case the controller operation is satisfactory.

Note that further increase of sampling and response steps will lead to the situation when the controller fails, i.e. it is not capable anymore to keep the pressure under the limiting value. This we leave for the user as an exercise.

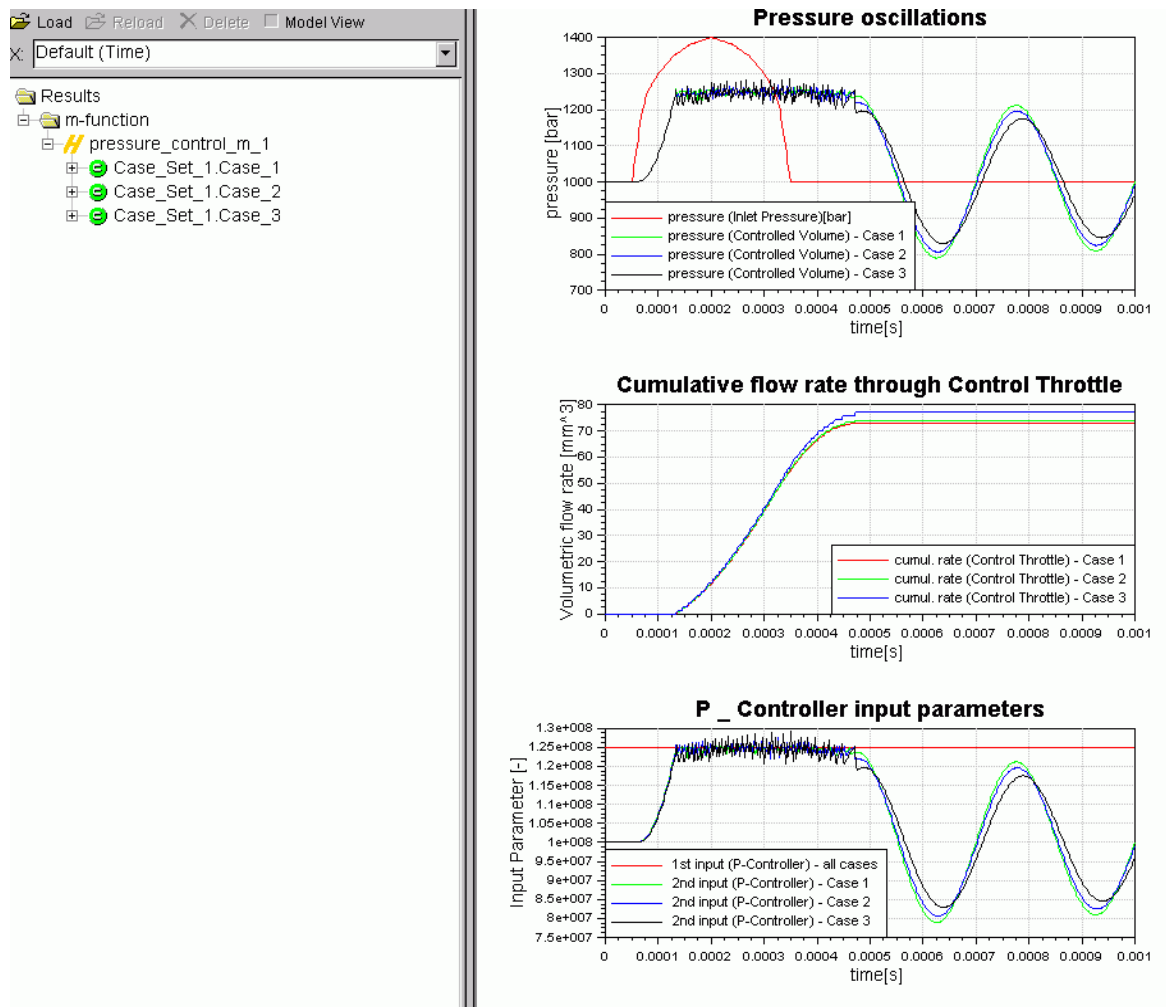


Figure 8-31: Calculation Results for Different P-controller Parameters (3 cases)

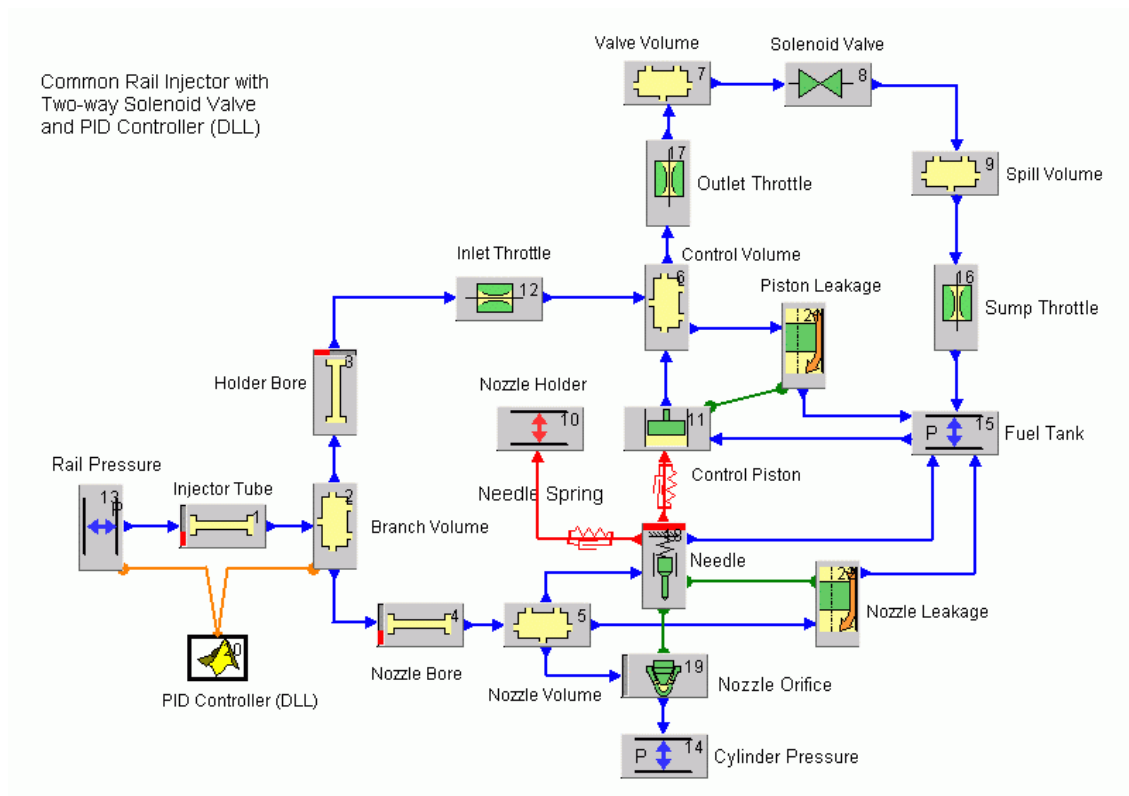
## 8.3. Example with MATLAB® DLL

### 8.3.1. Overview

In this chapter, BOOST Hydsim example with a MATLAB® DLL element will be shown. As a basis for it, the model of the common rail injector from *Chapter 5.1* is used. MATLAB® DLL element will be a Hydsim applied for the modeling of a PID Controller. The controller is aimed to suppress pressure oscillations in Branch Volume caused by injection process. These oscillations are clearly visible in *Figure 5-17*. In this case, the MATLAB® DLL element formally can be treated as a ‘variable orifice’ switched in parallel with Injection Tube. This ‘variable orifice’ has to generate additional flow rate between Rail and Branch Volume in the right direction.



The layout of the injector is shown in *Figure 8-32*. The model consists of the same elements as the model in the *Chapter 5.1*, only one new element (No. 20<sup>10</sup>) - PID Controller (DLL) - is added. PID Controller regulates pressure in Branch Volume according to the pressure difference between Rail Pressure and Branch Volume.

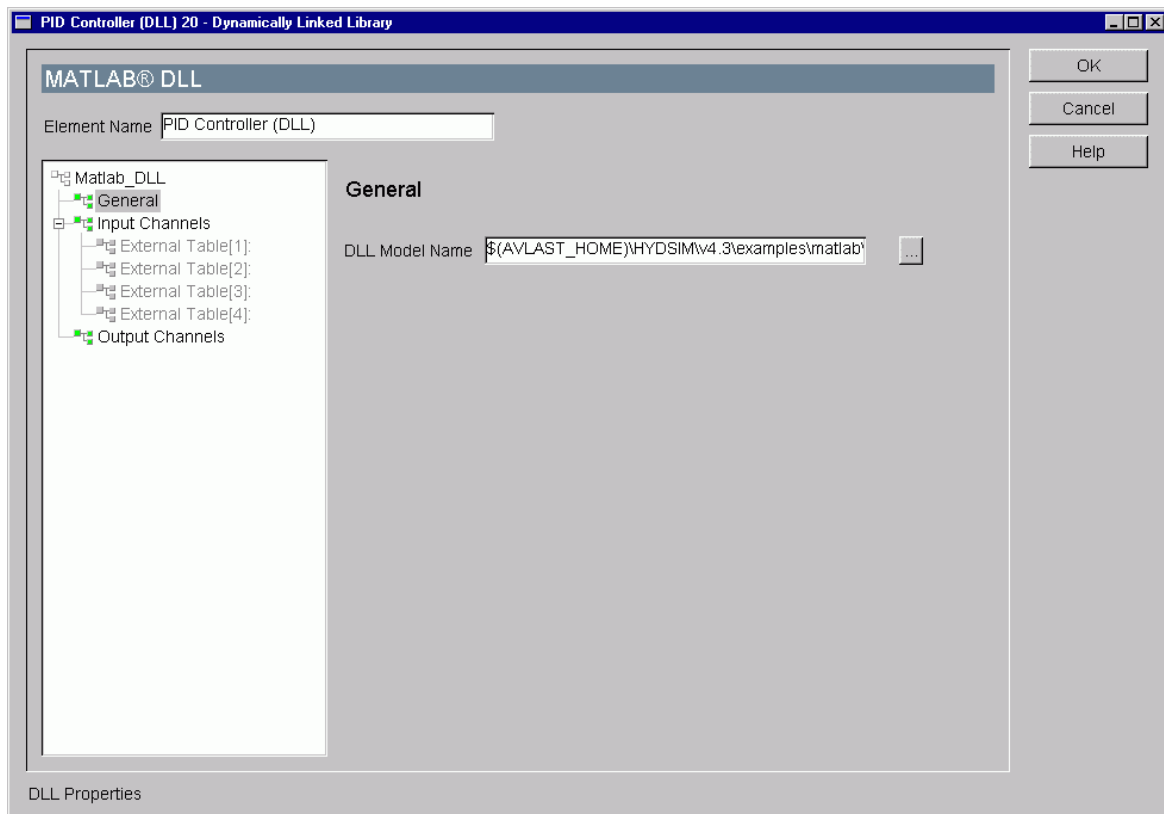


**Figure 8-32: BOOST HydSim Model with MATLAB® DLL Element**

### 8.3.2. Input Data

Figure 8-33 shows the input dialog box of PID Controller (DLL).

<sup>10</sup> Numbers given for elements may differ from those in your model. Reference Figure 8-32 for numbered items.



**Figure 8-33: Input Dialog Box of PID Controller (MATLAB® DLL)**

General part of input dialog of MATLAB® DLL element contains a single input: full path of a dll file. Input and Output Channel dialog boxes are completely same as those of MATLAB® API (Simulink and m-function) elements.

Contrary to MATLAB® API elements, MATLAB® DLL element has no Control and User vector input dialogs.

Refer to *Section 8.3.3* for creating a MATLAB® dll file.

For our model, define Input Channels as follows:

- 1<sup>st</sup>: Element, Rail Pressure (13), El. output parameters, pressure [Pa]
- 2<sup>nd</sup>: Element, Branch Volume (2), El. output parameters, pressure in volume [Pa]
- 3<sup>rd</sup>: Element, Branch Volume (2), El. local fluid properties, bulk modulus [N/m<sup>2</sup>]
- 4<sup>th</sup>: Element, Branch Volume (2), El. output parameters, actual volume [m<sup>3</sup>]

Next, specify one Output Channel:

- Branch Volume (2), El. output parameters, additional flow rate [m<sup>3</sup>/s]

Set Time step in **Calculation Control** dialog to 2e-7 s and Calculation end time to 0.008 s.

### 8.3.3. Creating MATLAB® DLL file

Creating of MATLAB® DLL file requires the installation of SIMULINK™ and Real-Time Workshop™ tools of MATLAB® software package.

Refer to the *Section 8.1.2* for creating the Simulink model of the PID controller. In our case it calculates additional flow rate from the continuity equation:

$$\dot{q} = \frac{\Delta p \cdot V}{E},$$

where:

$\dot{q}$  ..... additional flow rate

$\Delta p$  ..... pressure difference

$V$  ..... actual volume of Branch Volume

$E$  ..... bulk modulus of fluid

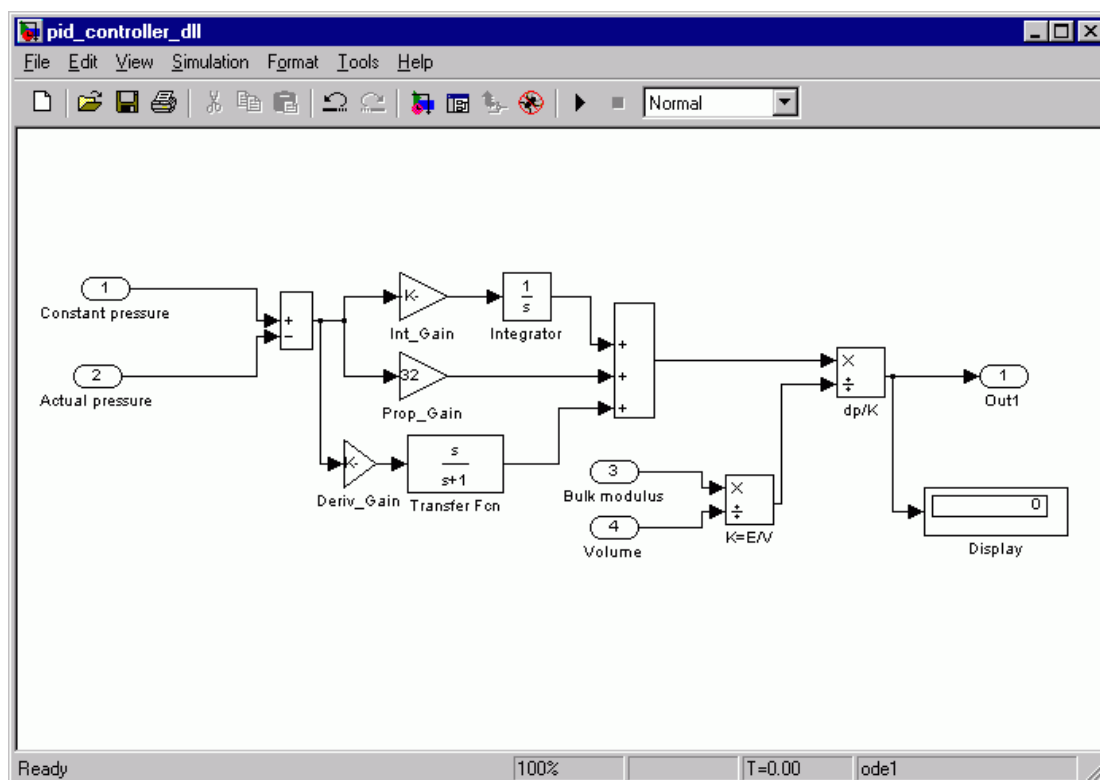
PID components are calculated according to the Ziegler Nichols tuning method: open loop reaction rate (refer to *Section 8.1.2*). Assuming pressure drop of 50 bar and reaction time of 1.6e-8 min we obtain the following values:

$$K_p = 32,$$

$$K_I = 8 \cdot 10^9,$$

$$K_D = 3.2 \cdot 10^{-8}.$$

Simulink model of the PID controller is shown in *Figure 8-34*.



**Figure 8-34: Simulink Model of PID Controller (DLL)**

The following procedure may be used for generating of the DLL by MATLAB® Real Time Workshop software. This procedure works with MATLAB® versions v5.3, v6.0, v6.1, v6.5 and higher.

By typing in `mex -setup` in the MATLAB® command line, a menu pops up where the `c++` compiler for DLL generation has to be selected.

Depending on the MATLAB® version, the following path has to be added to the MATLAB®/Simulink path. It is necessary to point to the `avl_grt_dll_nt.tmf` (or `avl_grt_dll_unix.tmf`) and `avl_grt_dll.c` files:

**MATLAB® v5.3:** `BOOST Hydsim_HOME.. \ .. \matlab\v5.3`

executing the MATLAB® command `'addpath'('path_to_add')` **Path**

**MATLAB® v6.0 and higher:** `BOOST Hydsim_HOME.. \ .. \matlab\v6.x`

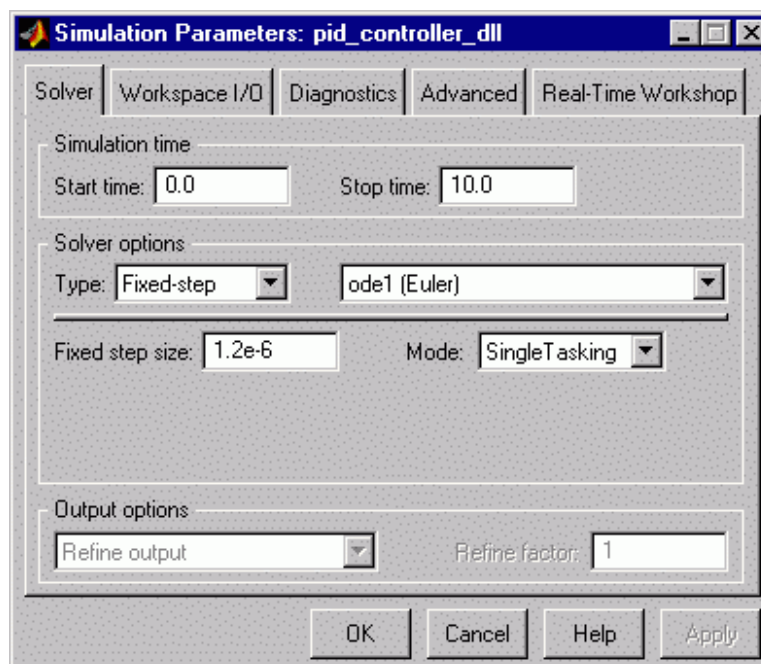
via MATLAB® GUI (**File | Set Path** select **Add Folder Path** and **SAVE**)

**Note:** For a DLL generation, use file `avl_grt_dll_nt.tmf` from the appropriate subdirectory (v5.3, v6.x) depending on MATLAB® version.



**Note:** It is useful to define an environmental variable `$(BOOST Hydsim_HOME)` pointing to the location directory of BOOST Hydsim executable `BOOST Hydsim.exe`, i.e. `$(AVLAST_HOME)\BOOST Hydsim\v4.5\bin\PLATFORM`.

Open **Solver** window from **Simulation|Simulation Parameters** PullDown menu and set its parameters as shown in *Figure 8-35*.

**Figure 8-35: SIMULINK Solver Settings**

In the **Solver** window, the incrementation and Solver type have to be defined. The incrementation must be constant (Fixed-step) and adjusted to the BOOST Hydsim time step ( $2e-7$  s for Time step in the **Calculation Control**). BOOST Hydsim will reference this parameter to its own time step, i.e. send input parameters to the DLL model and receive results back after the DLL incrementation time. This principle is similar to the response time in MATLAB® API element (refer to *Section 8.1.1.2*).



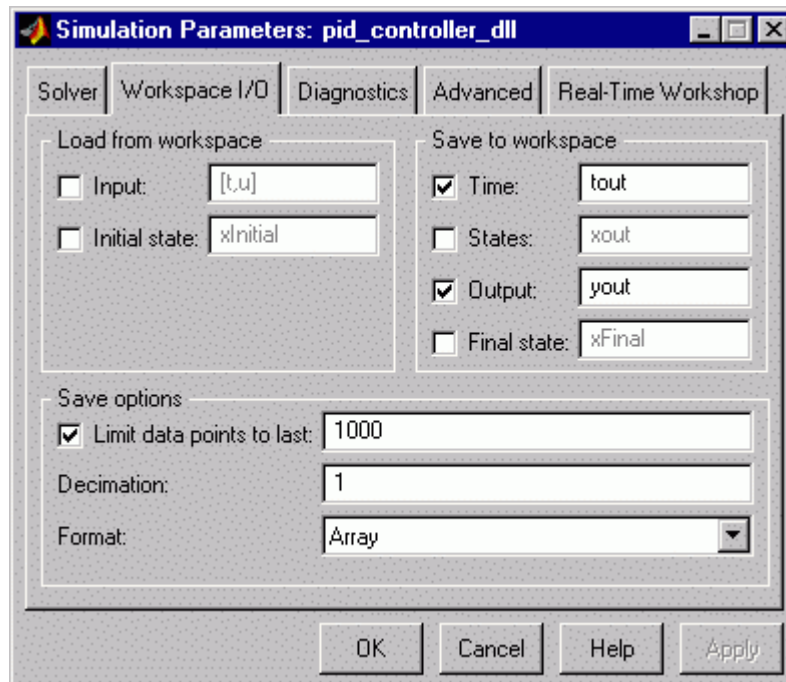
**Note:** The incrementation must be greater than or equal to the BOOST Hydsim time step and smaller than BOOST Hydsim calculation end time. Otherwise, program will issue an error message.

The values for **Start** and **Stop Time** have no meaning because BOOST Hydsim will reset them during calculation.



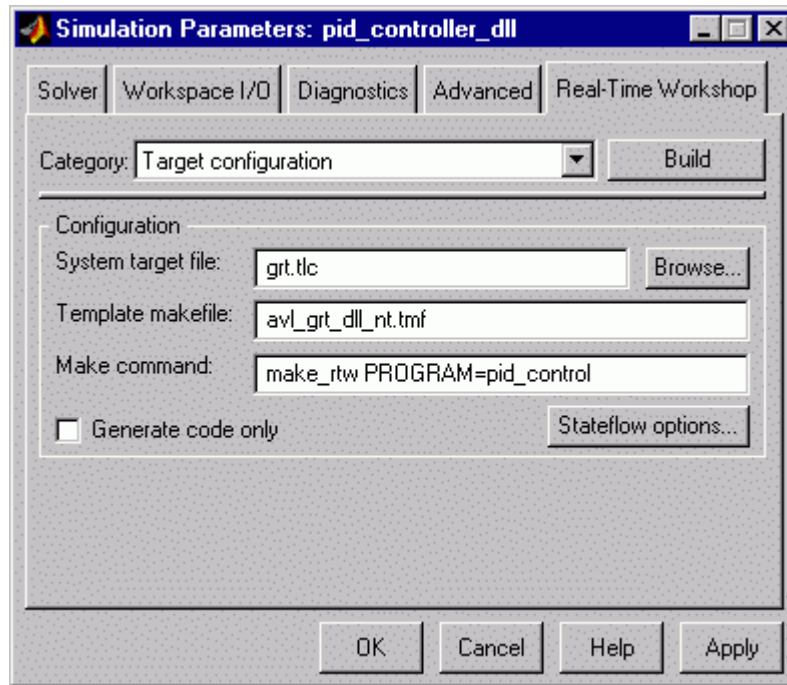
**Note:** If no integrators, memory blocks etc. are used in the Simulink model, the value of the Fixed-step size will be ignored and automatically set to the BOOST Hydsim time step. If DLL incrementation is not a multiple of the BOOST Hydsim time step, program will use the first higher multiple as DLL incrementation.

In the **Workspace I/O** window shown in *Figure 8-36*, **Input** and **Initial state** parameters must be inactive.



**Figure 8-36: SIMULINK Workspace Settings**

In the **Real-Time Workshop** window, shown in *Figure 8-37*, the settings for the DLL generation have to be defined.



**Figure 8-37: SIMULINK Real-Time Workshop Settings**

**System target file** window defines the type of the code which can be produced. Enter `grt.tlc` (general real-time).

**Template makefile** window defines the name of the template file from which the new Makefile is generated. For the generating of the desired DLL file using the C++ V6.0 compiler on Windows, enter the template file `avl_grt_dll_nt.tmf`. On Linux/Unix platforms, enter the template file `avl_grt_dll_unix.tmf`. This file and `avl_grt_dll.c` file must be stored either in the directory `%MATLAB_ROOT%\rtw\c\grt` or in the current working directory (if **Set Path** is not specified). Otherwise the complete path name must be entered.

Enter `make_rtw PROGRAM=pid_control` in **Make command** field.

`PROGRAM=pid_control` defines the file name of the DLL. Alternatively the model name is used and an underscore is placed in front of it. If the DLL file name has to be same as the MDL file (Simulink model) name, an underscore must be placed in front (refer to the table below). Otherwise no valid C-MEX DLL will be generated, as the MDL file in MATLAB® could be not opened any longer, because MATLAB® always checks first whether a DLL file with the same name is existing or not.

Make command	Simulink model	Created DLL
<code>make_rtw</code>	<code>example.mdl</code>	<code>_example.dll</code>
<code>make_rtw PROGRAM=example</code>	<code>example.mdl</code>	<code>_example.dll</code>
<code>make_rtw PROGRAM=test</code>	<code>example.mdl</code>	<code>Test.dll</code>

Click **Build** button to generate a C-code. The Makefile will be produced, executable `nmake.exe` will be called from this Makefile, and the object files will be linked to the DLL.



**Note:** For MATLAB® v6.0 and higher, the DLL and other necessary files will be created in the new directory named model\_name\_grt\_rtw. This directory will be placed in the current working directory.

### 8.3.4. Output Parameters

BOOST Hydsim supports the post-processing of all input and output channel parameters of MATLAB® DLL element and DLL incrementation. To perform this, select desired output parameters from the **Element | Store Results** menu. These output results may be used to control the interaction between BOOST Hydsim and MATLAB® DLL.

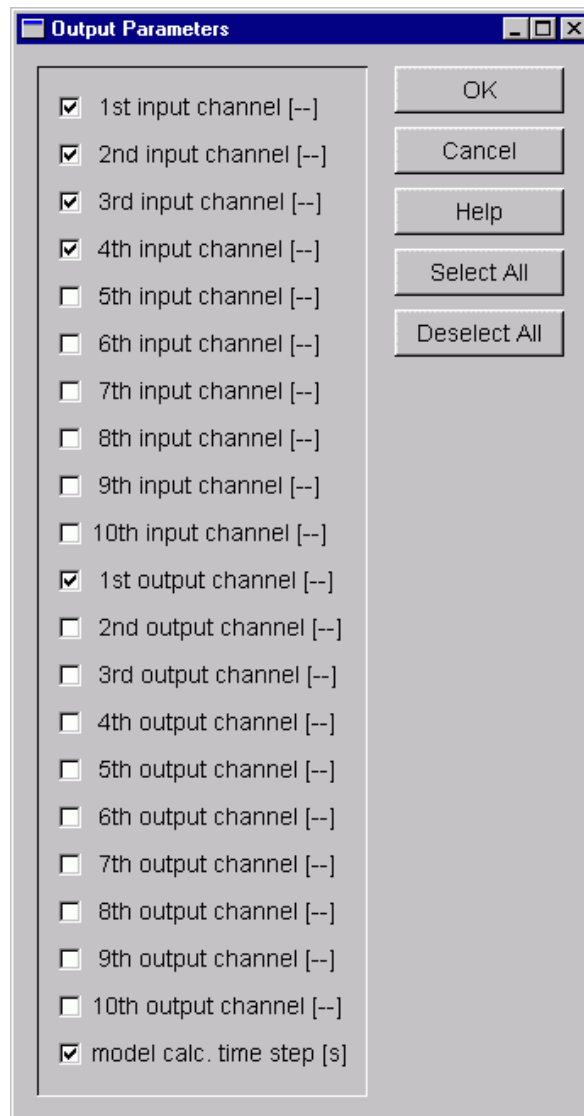


Figure 8-38: Output Data of MATLAB DLL Element

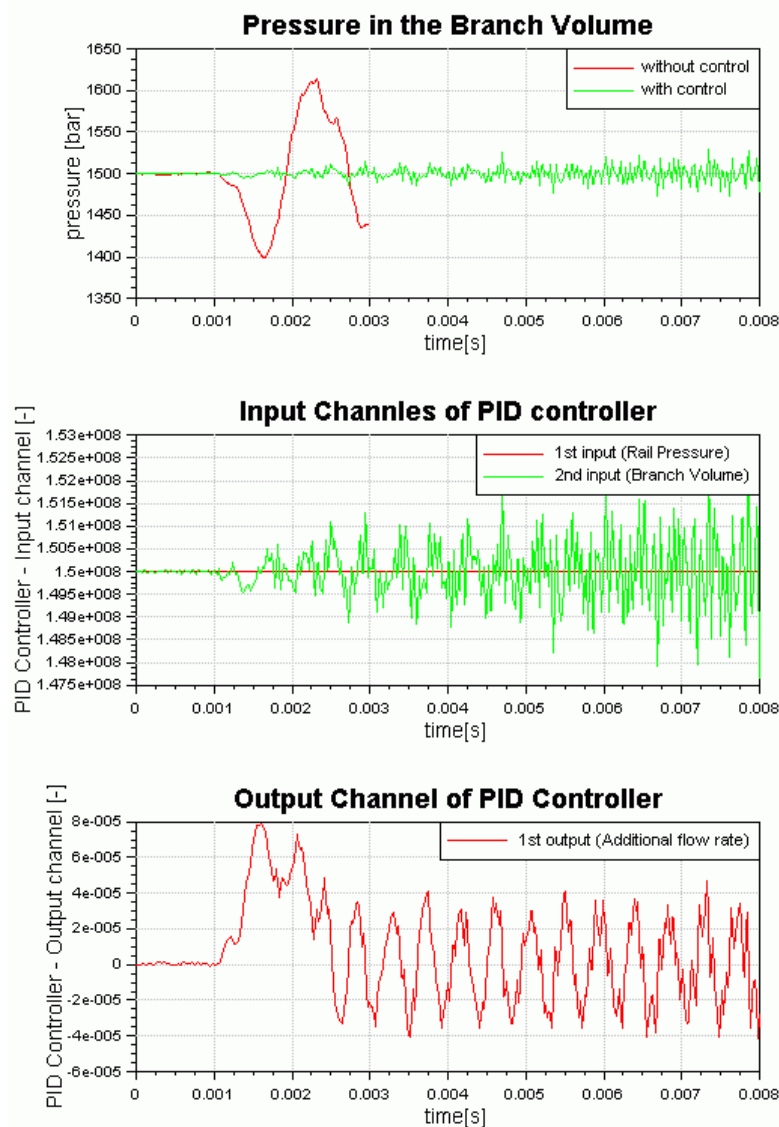
### 8.3.5. Running the Calculation

For performing the calculation, refer to *Section 4.4*.

### 8.3.6. Calculation Results

To view the calculation results, select **Show Results** from **Simulation** Pulldown menu. The new Impress Chart window with the **Result** tree of the active BOOST Hydsim model will pop up. Next, load the results of the original model `common_rail2wv.hyd` from *Chapter 5.1* for comparison.

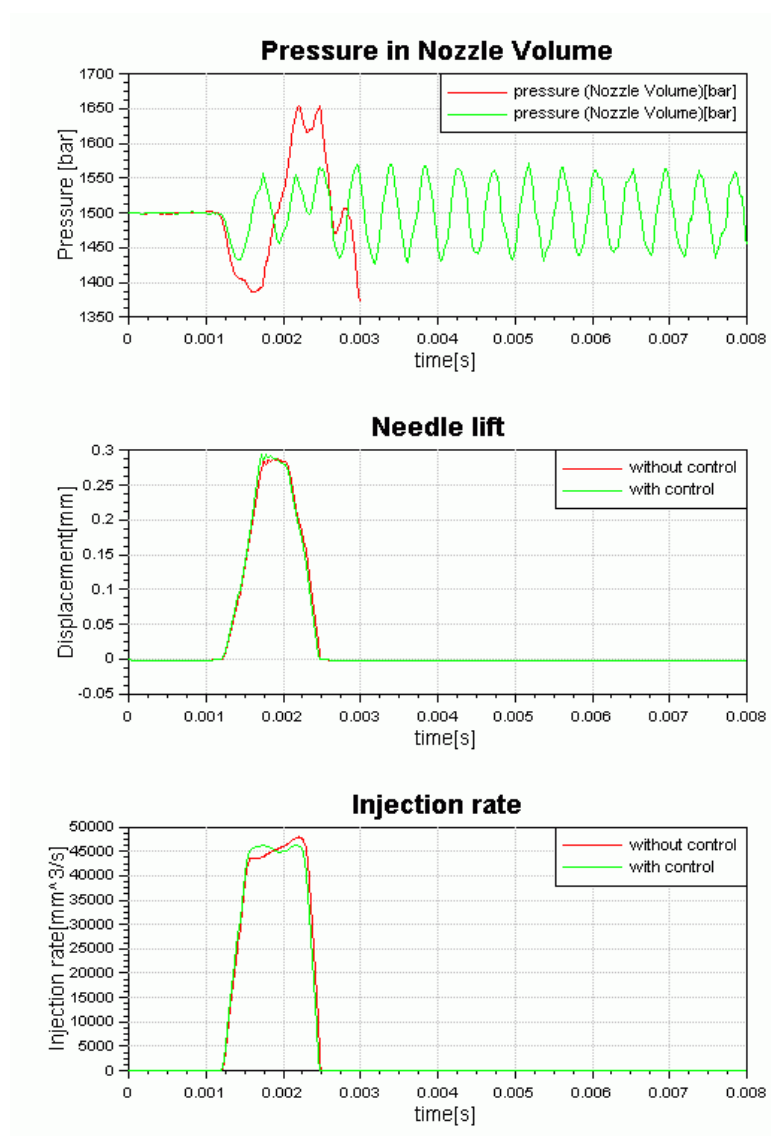
Figure 8-39, upper graph shows the pressure in Branch Volume, for the models with and without controller, respectively. Obviously, the model with the PID Controller exhibits very small pressure oscillations compared to the original model. Middle and bottom graphs depict the parameters of Input and Output Channels of the controller, respectively.



**Figure 8-39: Pressure in Branch Volume and PID Controller Data (DLL)**

Figure 8-40 shows the comparison of the pressure in Nozzle Volume, Needle motion and injection rate. In the model with PID Controller, the pressure oscillations in Nozzle Volume are considerably smaller. The needle lift and injection rate remain very close to the original model results. The user should not search for deep physical meaning in these results. It is not a model of a real control system but just a demonstration example.





**Figure 8-40: Pressure in Nozzle Volume, Needle Lift and Injection Rate**

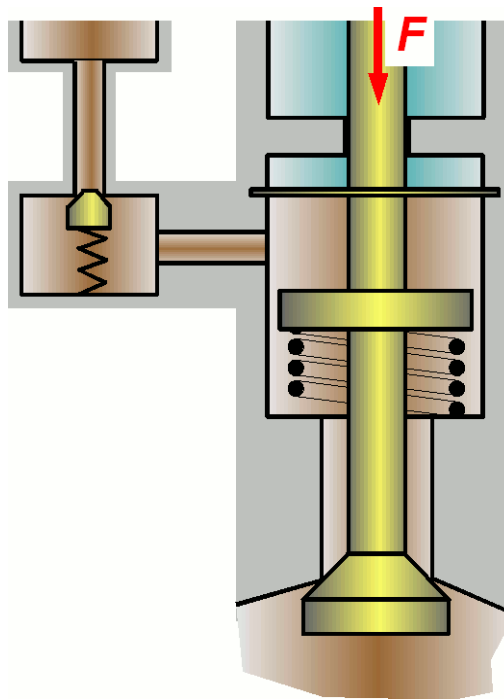


## 9. BOOST HYDSIM – BOOST LINK

### 9.1. Introduction

BOOST-Hydsim-BOOST Link is used for the co-simulation of liquid-gaseous (hydraulic-pneumatic), mechanical-gaseous or combined hydromechanical-gaseous systems by BOOST Hydsim and BOOST Cycle Simulation. There are two BOOST link elements in **External** group of BOOST Hydsim element tree: **Boost Link (Plenum)** and **Boost Link (Valve)**. **Boost Link (Plenum)** element in BOOST Hydsim represents the interface to the **Plenum** element in BOOST while **Boost Link (Valve)** element represents the interface to the **Check Valve** element in BOOST. Both link elements have to be connected to a **Standard Piston** element. Data exchange between BOOST Hydsim and BOOST Cycle simulation kernels is carried out via ACCI interface.

A typical application example of BOOST Hydsim-BOOST co-simulation is the gas injector (or gas valve) controlled by a hydraulic fluid. The principal scheme of such injector is shown in Figure 9-1. The injector consists of the gas metering valve, needle with spring, outwards opening nozzle and hydraulic control system. Gas elements are depicted in brown color, liquid elements – in cyan color. The needle, metering valve body and control liquid part is simulated by BOOST Hydsim while entire gas part is modeled in BOOST. Figure 9-2 shows the division of the injector into BOOST and BOOST Hydsim parts for the further sub-model creation.

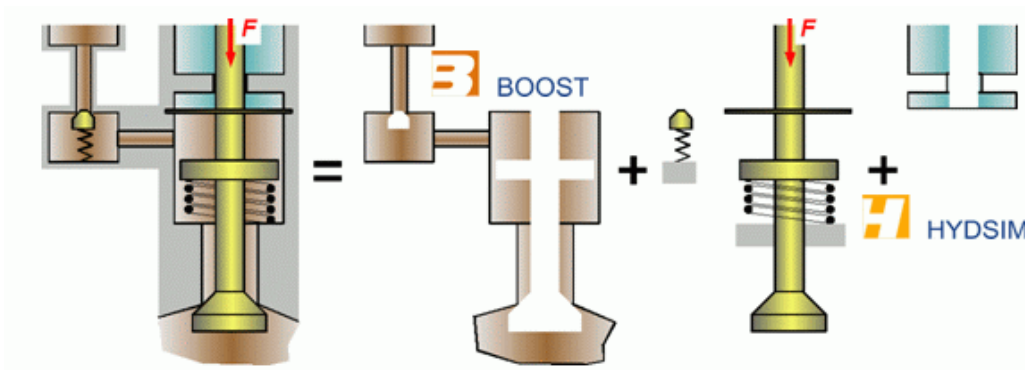


**Figure 9-1: Schematic of outwards opening gas valve with hydraulic control**

Model of the gas injector (valve) consists of three different types of physical elements:

- gas elements (plenums, pipes and throttle)
- hydraulic elements (volumes and orifices)

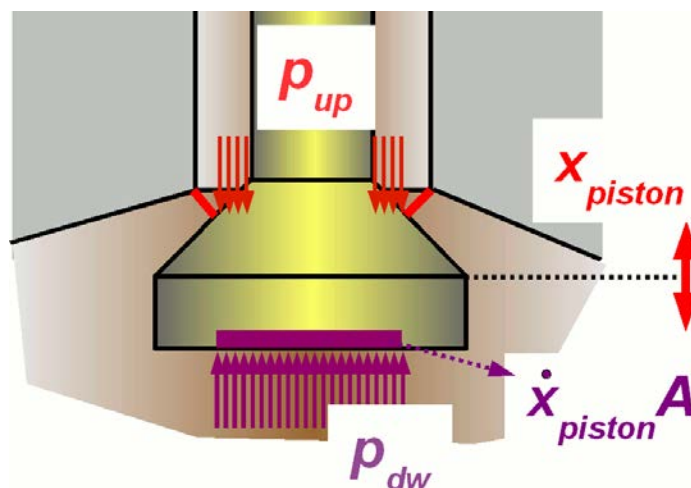
- mechanical elements (needle, piston and springs)



**Figure 9-2: Gas injector division into BOOST and BOOST Hydsim sub-models**

As stated earlier, BOOST Hydsim covers the simulation not only of the control liquid flow but also the motion of mechanical elements (multi-body dynamics). Interface between BOOST Hydsim and BOOST simulation is based on the **Standard Piston** element from BOOST Hydsim side and **Plenum** or **Check Valve** element from BOOST side. Gas pressures in the gas system are calculated by BOOST. These are typically stagnation pressures in **Plenum** elements or dynamic pressures upstream and downstream of **Check Valve** elements. **Piston** areas pressurized by gas are defined and controlled in BOOST Hydsim sub-model.

For each piston-type element BOOST Hydsim calculates coordinate and velocity. For a **Piston** element connected to a **Boost Link (Plenum)** element, the product of piston velocity and pressurized area yields the flow rate (called also displacement flow). This displacement flow is transferred to BOOST **Plenum** element at every data exchange step. For a **Piston** element connected to a **Boost Link (Valve)** element, piston input/output coordinate and velocity are transmitted to BOOST **Check Valve** element. In return, BOOST transfers to BOOST Hydsim the gas pressures and their gradients acting on the respective **Piston** elements. The data exchange between BOOST Hydsim and BOOST at the needle valve seat is illustrated in Figure 9-3.



**Figure 9-3: Data exchange between BOOST and BOOST Hydsim at needle seat**

## 9.2. CNG Injector Model with BOOST Link

This chapter guides the user through the BOOST Hydsim-BOOST Link example based on the simplified model of the Compressed Natural Gas (CNG) injector. The injector schematic is same as in the previous chapter (refer to Figure 9-1), only the liquid part is omitted for simplification and the needle motion is directly controlled by a solenoid valve.

### 9.2.1. Creating the Boost Model

The whole gas part of the injector is modeled using BOOST software. Schematic and BOOST model of the gas system is shown Figure 9-4. It consists of the following elements:

- Supply Pressure (**SB1**)
- Metering Valve [**CV1**] / linked to BOOST Hydsim /
- Inlet Volume (**PL1**) / linked to BOOST Hydsim /
- Pipe (**3**)
- Nozzle Volume (**PL2**)
- Nozzle Valve (**CV2**) / linked to BOOST Hydsim /
- Cylinder Pressure (**SB2**)

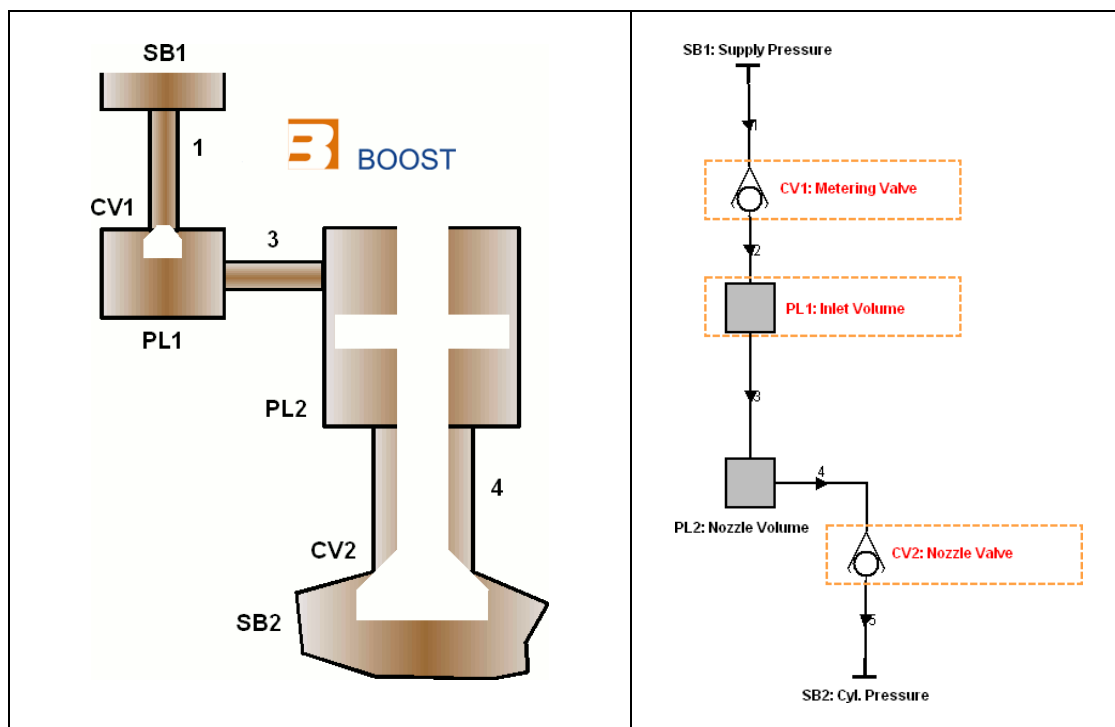


Figure 9-4: Schematic and BOOST sub-model of CNG injector

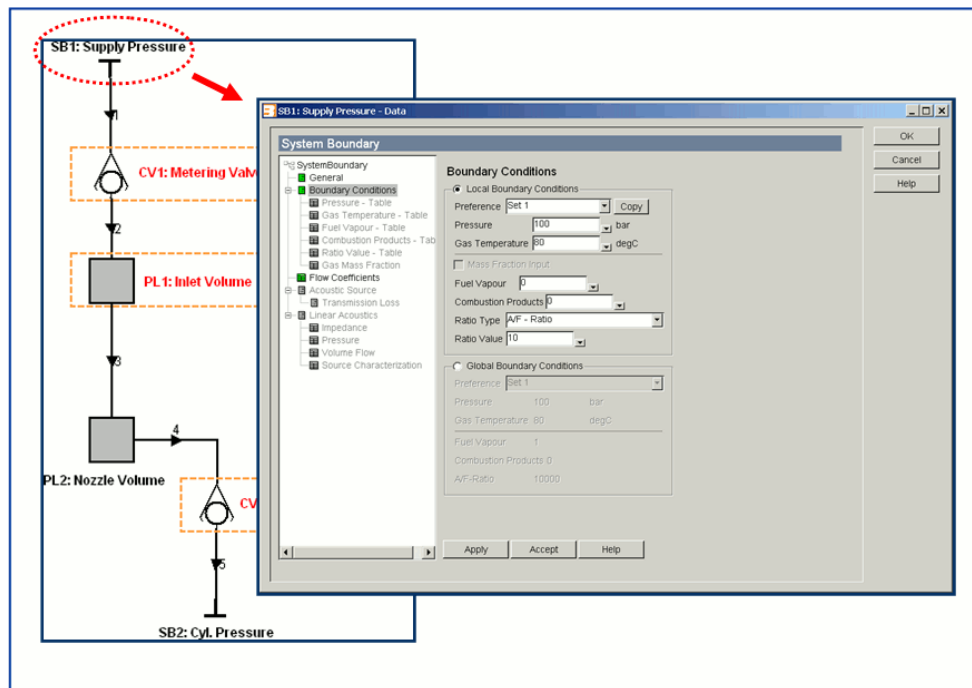
#### 9.2.1.1. Supply Pressure

Compressed natural gas (CNG) is supplied to the injector inlet with constant pressure and temperature (system boundary SB1). Boundary SB1 is connected via pipe 1 to the metering valve CV1. Input dialog of the boundary SB1 is shown in Figure 9-5.

##### Supply Pressure /

Pressure: 100 bar

Temperature: 80 degC



**Figure 9-5: Input dialog of Supply Pressure (SB1)**

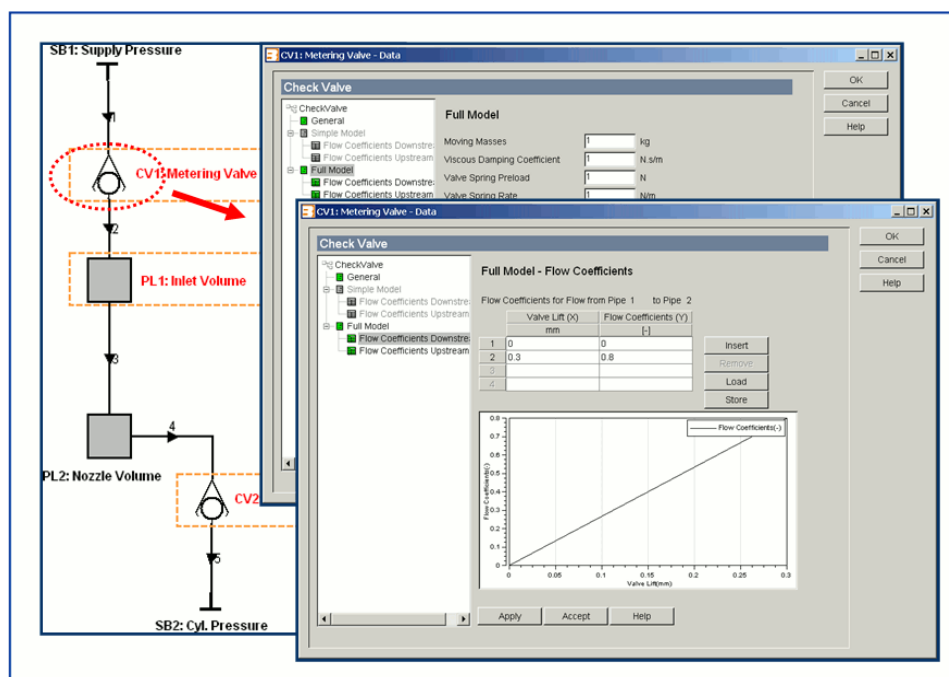
Fuel vapour value is set to zero because the fuel is not a liquid but natural gas (methane).

### 9.2.1.2. Metering Valve

#### Metering Valve /

Full Model Tab: ignored – valve lift and velocity transferred from BOOST Hydsim

Flow Model Tab: linear function, at valve lift of 0.3 mm flow coefficient is 0.8



**Figure 9-6: Input dialog of Metering Valve (CV1)**

Next important element in BOOST sub-model is the inlet metering valve (CV1). It is just a spring-controlled mechanical valve. Its input dialog is shown in Figure 9-6. The flow characteristic of the metering valve is defined by flow coefficients as a function of the valve body lift. As Metering Valve CV1 is linked to respective BOOST Link (Valve) in BOOST Hydsim sub-model, its inertia and stiffness data will be ignored, i.e. BOOST Hydsim data will be used. This means that the valve body lift will not be calculated by BOOST but will be received from BOOST Hydsim. BOOST calculates only the gas flow through the valve including upstream and downstream pressures acting on the valve body. These pressures are transferred to BOOST Hydsim. Note that downstream flow direction in BOOST is always the flow from the pipe element with the lower number into the pipe with the higher ID number.

The list of Plenum and Check Valve elements linked to BOOST Hydsim is defined in BOOST **Control dialog** box at **User Defined Parameters** tab (refer to Chapter 9.2.1.8).

### 9.2.1.3. Inlet Volume

Next element is Inlet Volume-Plenum PL1. It is connected via Pipe 2 to Metering Valve CV1 and via Pipe 3 to Nozzle Volume-Plenum PL2. The input dialog of Inlet Volume is shown in Figure 9-7. This element is linked to the respective BOOST Link (Plenum) element in BOOST Hydsim sub-model. It calculated the pressure acting on the bottom area of Metering Valve body and sends it to BOOST Hydsim. In return actual it receives from BOOST Hydsim the displacement flow of the Metering Valve body.

#### Inlet Volume /

Volume size: 30 mm<sup>2</sup>

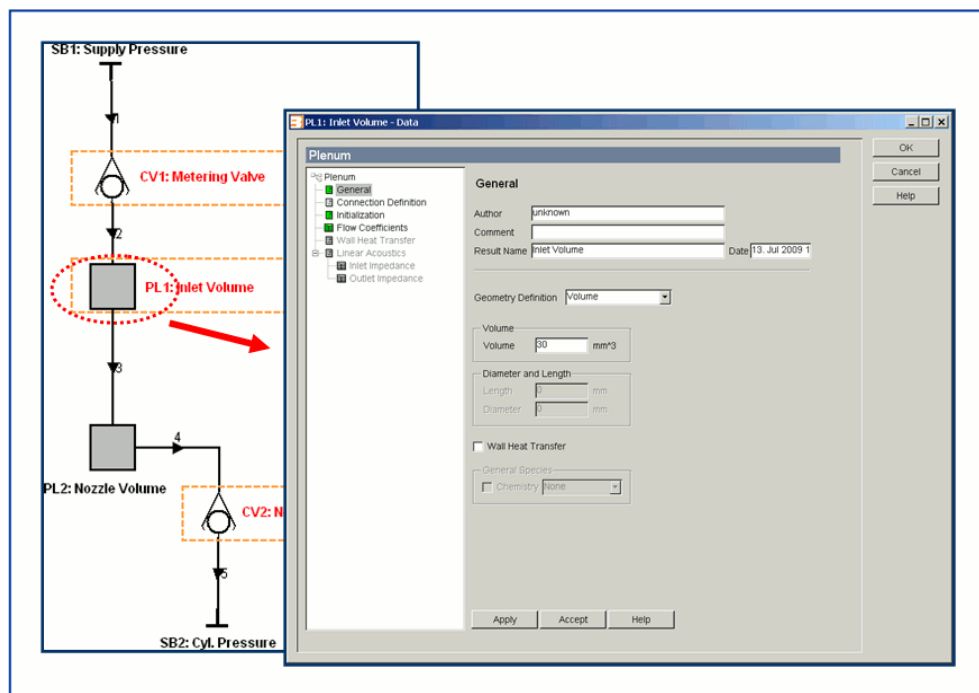


Figure 9-7: Input dialog of Inlet Volume (PL1)

### 9.2.1.4. Nozzle Bore

Nozzle Bore (Pipe 3) connects Inlet Volume-Plenum PL2 and Nozzle Volume-Plenum PL3. Input dialog of Nozzle Bore is shown in Figure 9-8. There laminar and turbulent friction is considered.

#### Nozzle Bore /

Pipe Length: 25 mm

Diameter: 1.5 mm

Laminar Friction Coefficient: 64

Friction Coefficient: 0.019

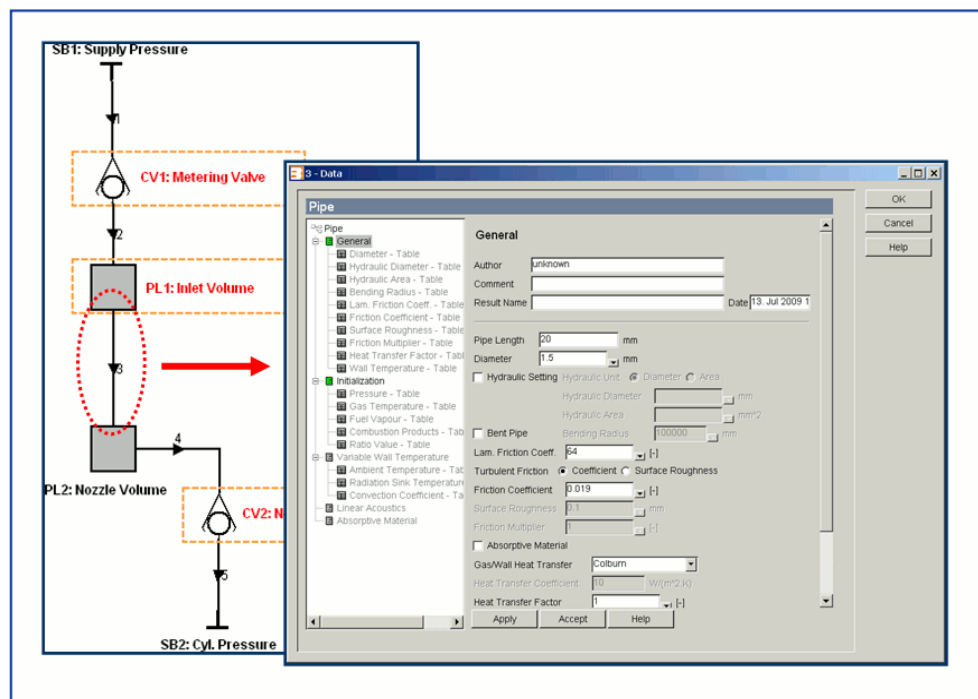


Figure 9-8: Input dialog of Nozzle Bore

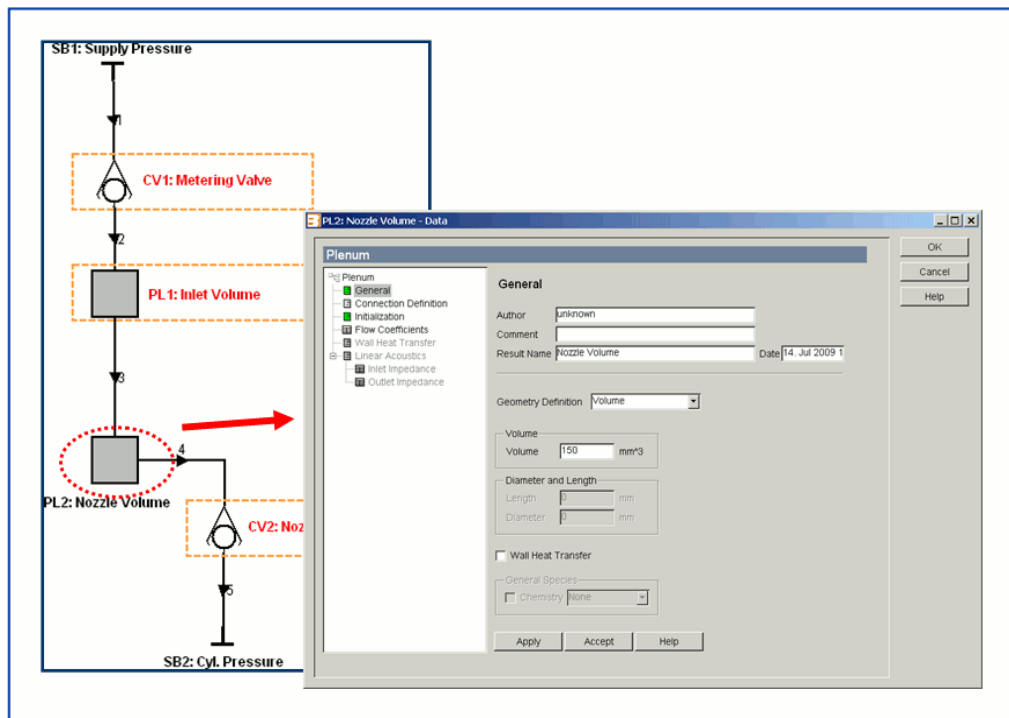
### 9.2.1.5. Nozzle Volume

Input dialog of Nozzle Volume-Plenum PL2 is shown Figure 9-9.

#### Nozzle Volume /

Volume size: 30 mm<sup>2</sup>





**Figure 9-9: Input dialog of Nozzle Volume PL2**

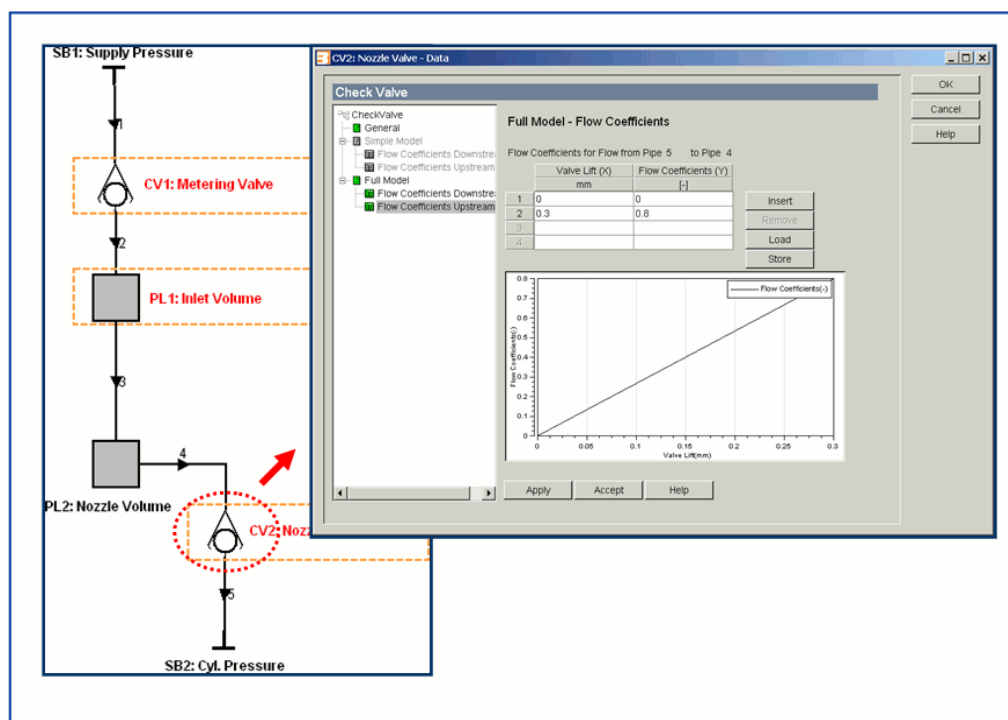
In our simplified injector model Nozzle Volume is simulated as one element. Volumes of all parts along the gas flow passage from Nozzle Bore to Needle Seat are summarized into one entity. For the more accurate simulation of gas dynamics inside this passage, it may be split into several volumes connected via pipes and restrictions.

### 9.2.1.6. Nozzle Valve

Nozzle Valve CV2 is a BOOST Check Valve element simulating the gas flow from Nozzle Volume PL2 into the cylinder. It is linked to the respective BOOST Link (Valve) element in BOOST Hydsim sub-model. Through this link the position of the valve body (Needle) is transferred to Nozzle Valve element at each data exchange step. Using this Needle lift, BOOST determines the actual values of the flow coefficients for the calculation of downstream or upstream gas flow. Input dialog of Nozzle valve is shown in Figure 9-10.

#### Nozzle Valve /

- |                 |  |
|-----------------|--|
| Full Model Tab: | ignored – valve lift and velocity transferred from BOOST Hydsim  |
| Flow Model Tab: | linear function, at valve lift of 0.3 mm flow coefficient is 0.8 |



**Figure 9-10: Input dialog of Nozzle Valve CV2**

### 9.2.1.7. Cylinder Pressure

The last element of the BOOST sub-model is the Cylinder Pressure boundary SB2. Input dialog of this boundary is shown in Figure 9-11. There pressure and temperature in the cylinder for simplicity are assumed to be constant. This is justifiable over a short injection event (24 deg CA). In a real cylinder chamber pressure and temperature may vary, of course.

#### Cylinder Pressure /

Pressure: 10 bar

Temperature: 100 degC

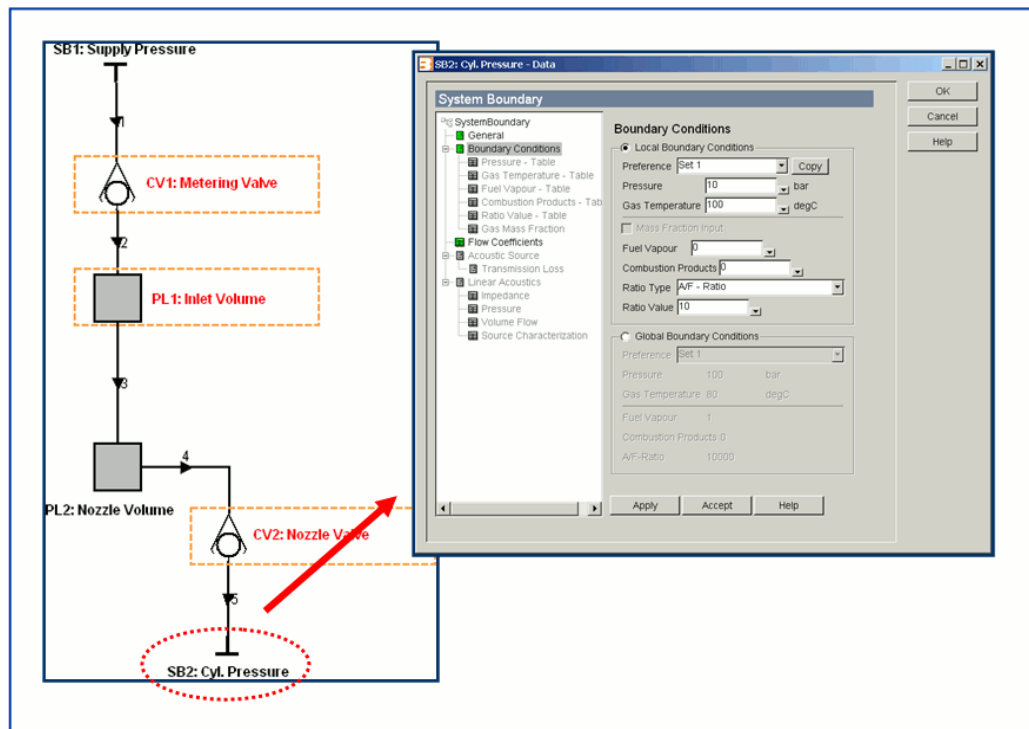


Figure 9-11: Input dialog of Cylinder Pressure boundary SB2

### 9.2.1.8. User Defined Parameters

As stated earlier, three elements in the BOOST sub-model (Metering Valve CV1, Inlet Volume PL1 and Nozzle Valve CV2) are linked with their respective counterparts in the BOOST Hydsim sub-model. These links in BOOST are specified via **User Defined Parameters** under **Simulation | Control** dialog shown in Figure 9-12. There one has to define whether the co-simulation with BOOST Hydsim is active or not (ON/OFF), specify the name of host computer for BOOST Hydsim co-simulation (e.g. localhost), port number for ACCI interface (e.g. 7777) and names of elements which are constrained with the communication channels (CHANNEL1= CHECKVALVE\*1, CHANNEL2= CHECKVALVE\*2 and CHANNEL3= PLENUM\*1).

#### User Defined Parameters / Simulation / Control /

1<sup>st</sup> / Parameter Key: HDSM\_LNK\_SIMULATION

1<sup>st</sup> / Value: ON

2<sup>nd</sup> / Parameter Key: HDSM\_LNK\_HOST

2<sup>nd</sup> / Value: localhost

3<sup>rd</sup> / Parameter Key: HDSM\_LNK\_PORT

3<sup>rd</sup> / Value: 7777

4<sup>th</sup> / Parameter Key: HDSM\_LNK\_CHANNEL\*1

4<sup>th</sup> / Value: CHECKVALVE\*1

5<sup>th</sup> / Parameter Key: HDSM\_LNK\_CHANNEL\*2

5<sup>th</sup> / Value: CHECKVALVE\*2

6<sup>th</sup> / Parameter Key: HDSM\_LNK\_CHANNEL\*3

6<sup>th</sup> / Value: PLENUM\*1

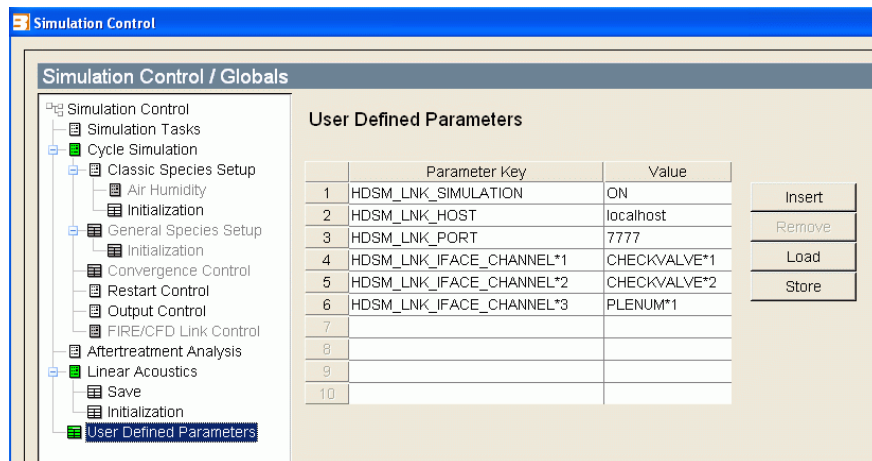


Figure 9-12: Simulation Control - User Defined Parameters Dialog

### 9.2.1.9. Simulation Control

Select **Cycle Simulation** in the tree to open the following window and enter the data:

**Reference Speed:** 2000 rpm

**End of Simulation:** 24 deg

**Simulation Step Size:** 1 deg

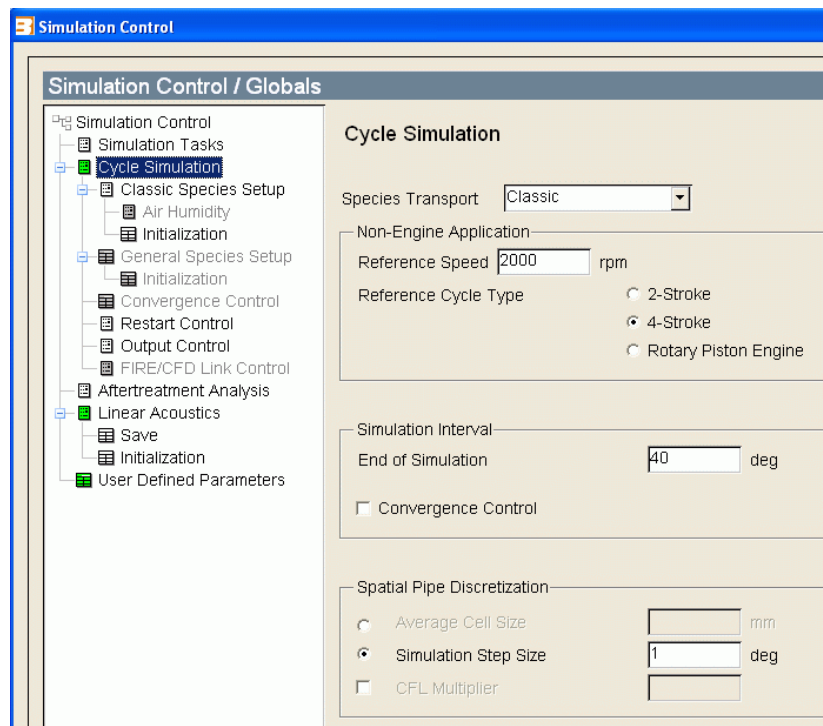


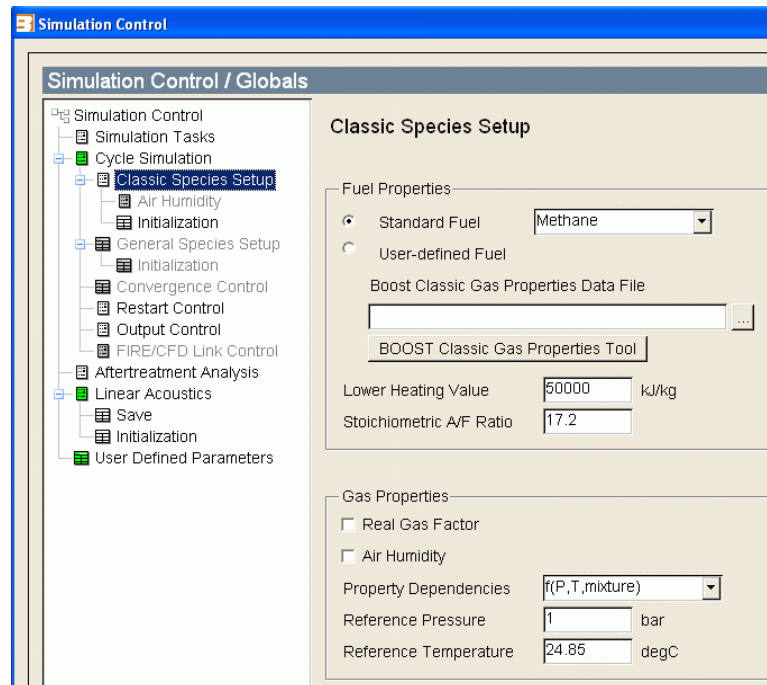
Figure 9-13: Simulation Control - Cycle Simulation Dialog

Certain control data like **Reference Speed**, **End of Simulation** and **Simulation Step Size** are defined in BOOST Hydsim which is the master software in this case. This data, if

defined in the BOOST **Simulation Control** dialog, will be ignored, i.e. they will be used only as initial values.

Select **Classic Species Setup** in the tree to open the following window and enter the data shown:

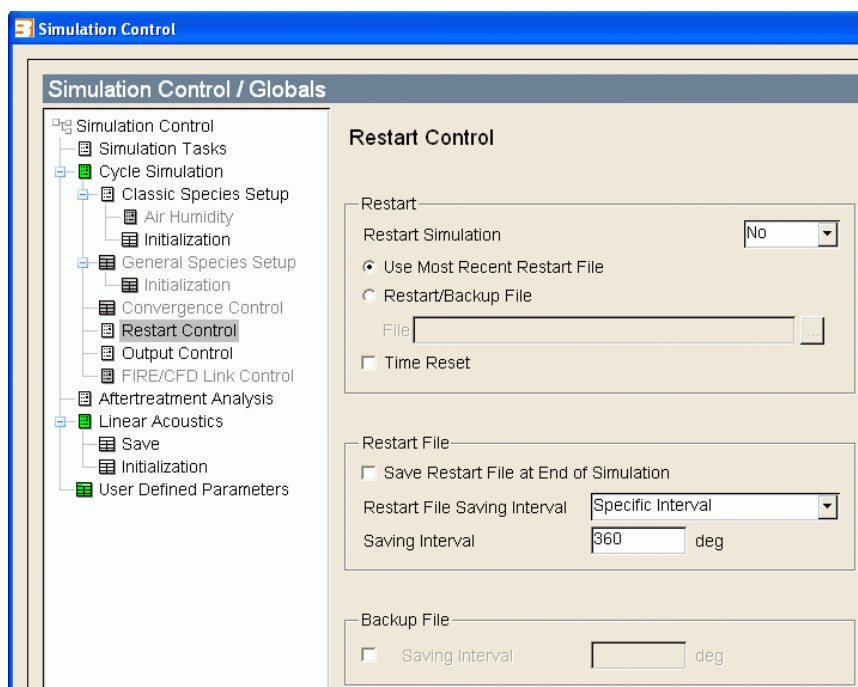
**Standard Fuel:** Methane  
**Lower Heating Value:** 50000 kJ/kg  
**Stoichiometric A/F Ratio:** 0.1 deg



**Figure 9-14: Simulation Control - Classic Species Setup Dialog**

Select **Restart Control** to open the following window and enter the data shown:

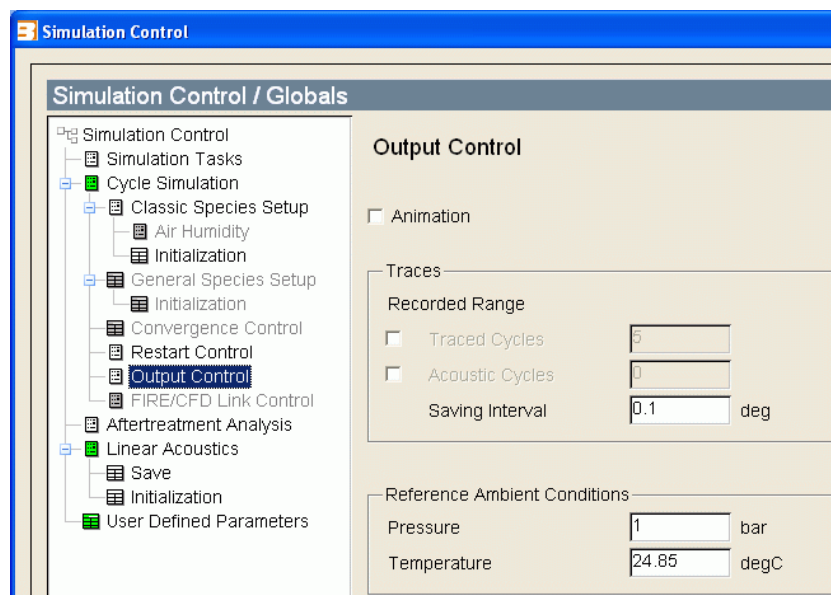
**Restart File Saving Interval:** Specific Interval  
**Saving Interval:** 360 deg



**Figure 9-15: Simulation Control – Restart Control Dialog**

Select **Output Control** to open the following window and enter the data shown:

**Saving Interval:** 0.1 deg

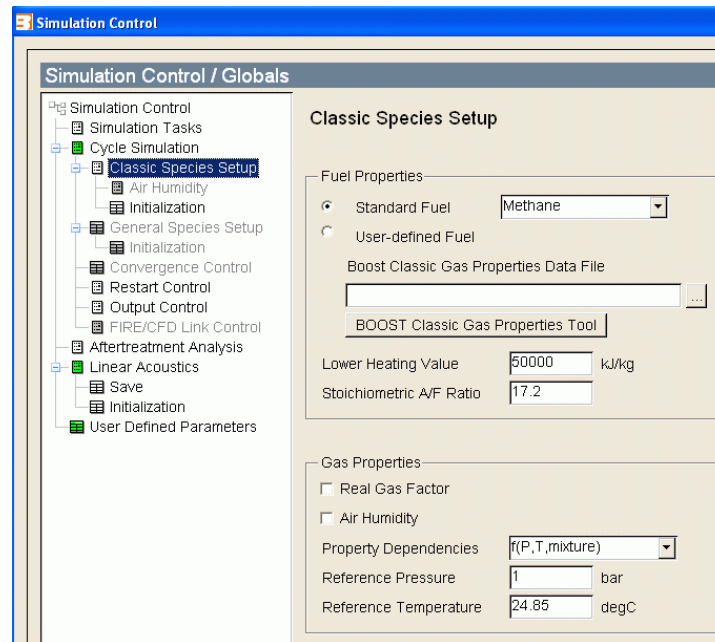


**Figure 9-16: Simulation Control – Output Control Dialog**

### 9.2.1.10. Running Boost model

BOOST calculation has to be started in a standard way by clicking **Cycle Simulation** in the **Run** dialog. In case BOOST Hydsim -BOOST co-simulation is activated in **User Defined Parameters** dialog (parameter key HDSM\_LNK\_SIMULATION set to ON), BOOST will automatically start ACCI server process which will ensure communication between the two

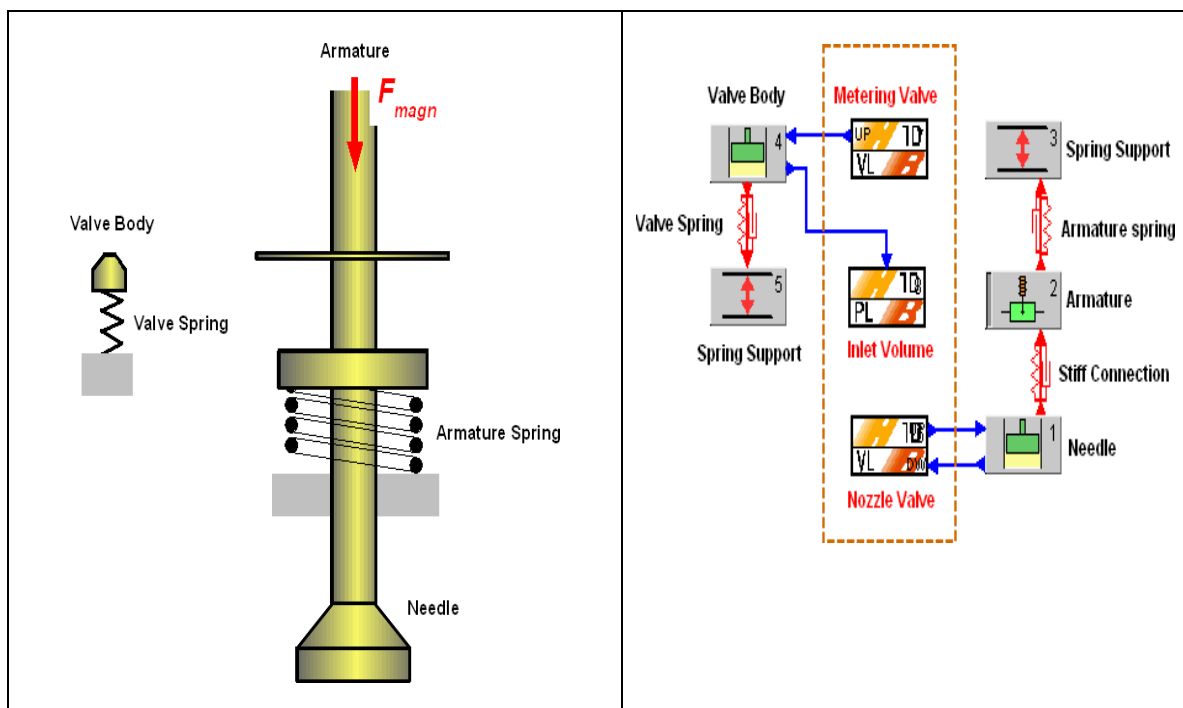
kernels. The flow chart of BOOST calculation can be viewed in **Simulation Logfile** window shown in Figure 9-17.



**Figure 9-17: Simulation Logfile dialog**

### 9.2.2. Creating the BOOST Hydsim Model

All mechanical injector parts are modeled using BOOST Hydsim software. Schematic and BOOST Hydsim sub-model of the mechanical system is shown in Figure 9-18.



**Figure 9-18: Schematic and BOOST Hydsim sub-model of CNG injector**

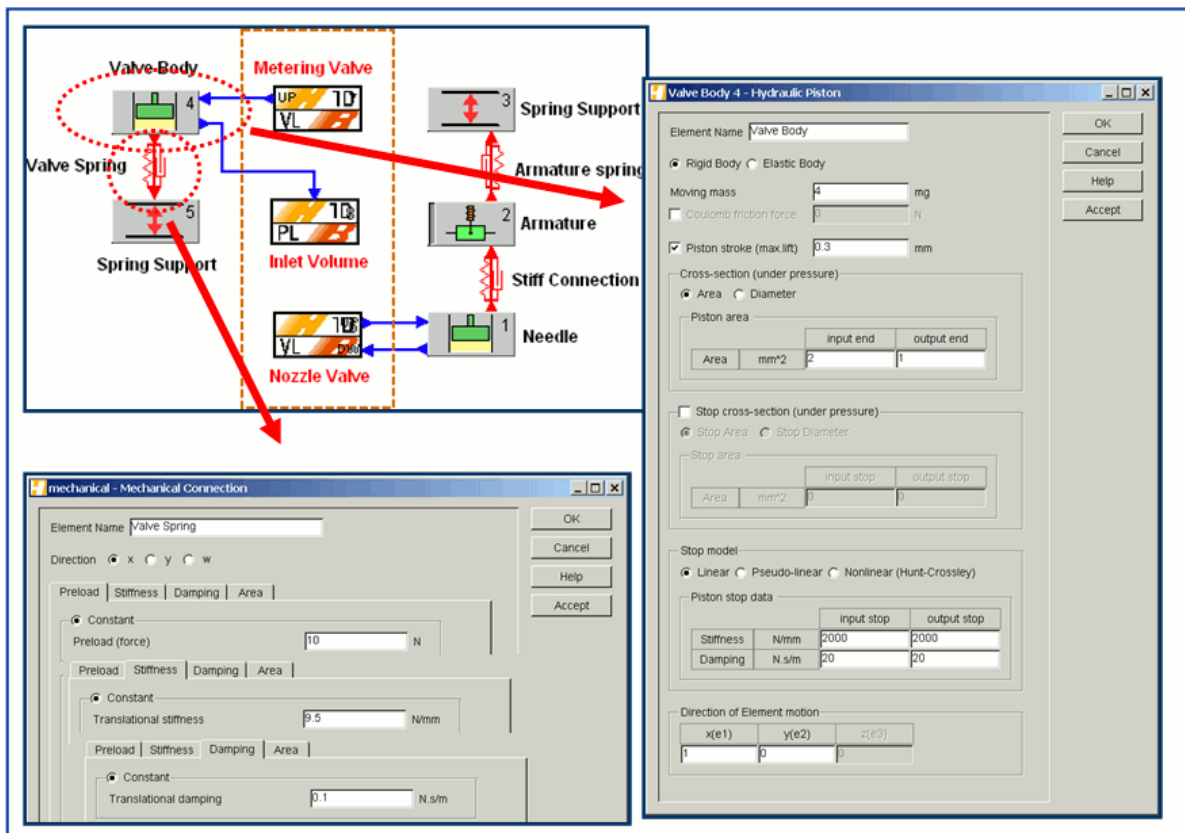


BOOST HydSim sub-model consists of the following elements:

- Spring Support (Valve) (**BOUNDARY/Mechanical**)
- Valve Body [**PISTON/Standard**]
- Spring Support (Armature) (**BOUNDARY/Mechanical**)
- Armature (**SOLENOID/Armature (Basic Model)**)
- Needle (**PISTON/Standard**)
- Metering Valve (**EXTERNAL/Boost Link (Valve)**)
- Inlet Volume (**EXTERNAL / Boost Link (P)**)
- Nozzle Valve (**EXTERNAL / Boost Link (Valve)**)

### 9.2.2.1. Valve Body (linked to Boost)

Valve Body is simulated using Standard Piston element connected via mechanical connection (valve spring) to a Mechanical Boundary element. Gas pressures acting on input and output areas of Piston are received via respective BOOST Link elements (connected to Valve Body with hydraulic connections). Input dialogs of Valve Body and its spring are shown in Figure 9-19.



**Figure 9-19: Input dialogs of Valve Body and its spring**

#### Valve Body /

Moving mass: 4 mg

Piston stroke: 0.3 mm

Input piston area: 2 mm<sup>2</sup>



Output piston area: 1 mm<sup>2</sup>

Input stop stiffness: 2000 N/mm

Input stop damping: 20 Ns/m

Output stop stiffness: 2000 N/mm

Output stop damping: 20 Ns/m

#### **Valve Spring /**

Preload: 10 N

Stiffness: 9.5 N/mm

Damping: 0.1 Ns/m

### **9.2.2.2. Needle (linked to Boost)**

Needle is simulated using Standard Piston element rigidly connected to Armature of the solenoid valve. In this way needle lift is directly controlled by the armature motion.

Pressurized areas are defined at needle seat position. At this position BOOST calculates gas flow through Nozzle Valve (BOOST Link element) using Needle lift received from BOOST Hydsim.

BOOST Link (Valve) element has two connection anchors: upstream (UP) and downstream (DW). The symbols UP and DW are placed at the connection anchor in the icon of BOOST Link (Valve). In our model, input hydraulic connection of Needle-Piston element is attached to the upstream (UP) anchor and output hydraulic connection - to the downstream (DW) anchor of BOOST Link (Valve). This implies that input end area of Needle-Piston is subjected to upstream gas pressure and output end area of Needle-Piston - to downstream gas pressure. Both pressures are calculated by BOOST.

#### **Needle /**

Moving mass: 6 mg

Input end diameter: 2 mm

Output end diameter: 2 mm

Input stop stiffness: 2000 N/mm

Input stop damping: 20 Ns/m

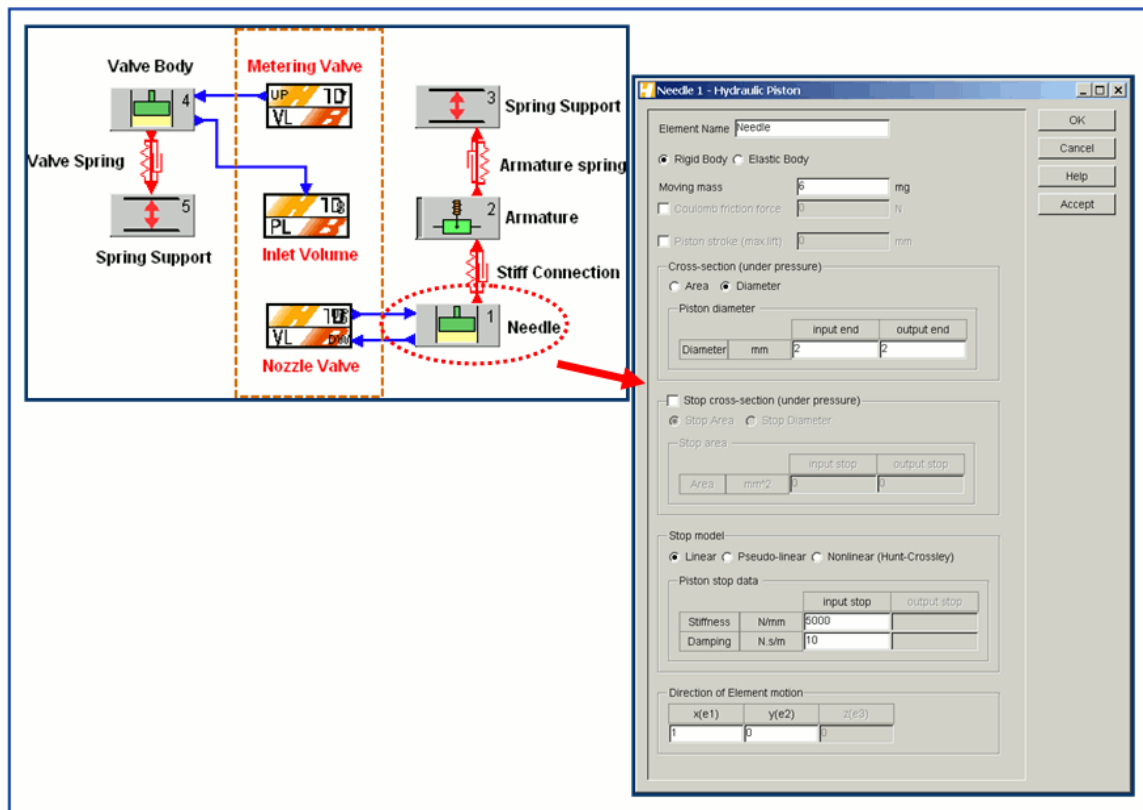


Figure 9-20: Input dialog of Needle-Piston

### 9.2.2.3. Armature

Solenoid Armature is an electromechanical element driven by a magnetic force. For simplicity, in our model magnetic force is chosen to be constant (boundary condition). Switch ON/OFF events of magnetic force are specified in the solenoid timing table. Force magnitude has to be high enough to open the valve against the armature spring. When solenoid is switched off (magnetic force set to zero), armature spring returns the armature and needle to the seat position and thus closes the nozzle. Input dialogs of Armature and its spring are shown in Figure 9-21.

#### Armature /

Moving mass: 4 mg

Armature stroke: 0.3 mm

Magnetic force: 70 N

Switch ON event: 0.1 ms

Switch OFF event: 0.8 ms

Body diameter on input end: 3 mm

Body diameter on output side: 3 mm

Input stop stiffness: 50000 N/mm

Input stop damping: 10 Ns/m

Output stop stiffness: 50000 N/mm

Output stop damping: 10 Ns/m

#### Armature Spring /

Preload: 50 N

Stiffness: 50 N/mm

Damping: 0.3 Ns/m

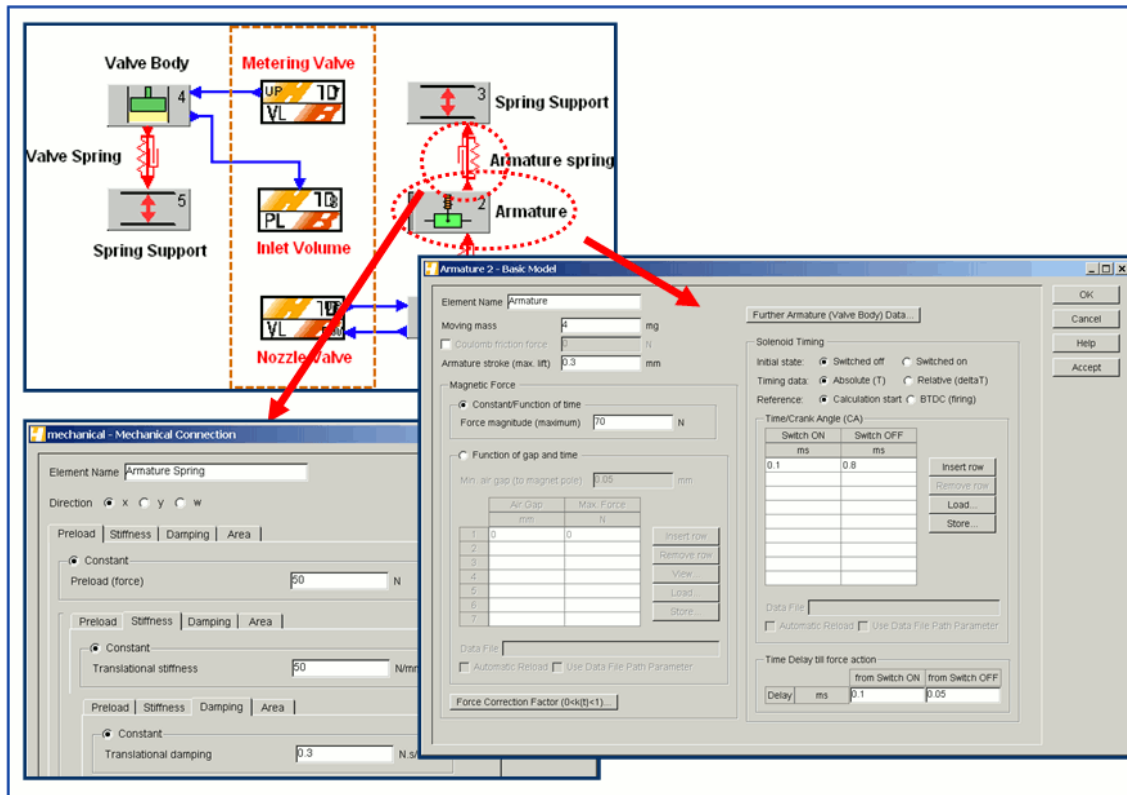


Figure 9-21: Input dialog of Solenoid Armature

#### 9.2.2.4. Metering Valve (Boost Link)

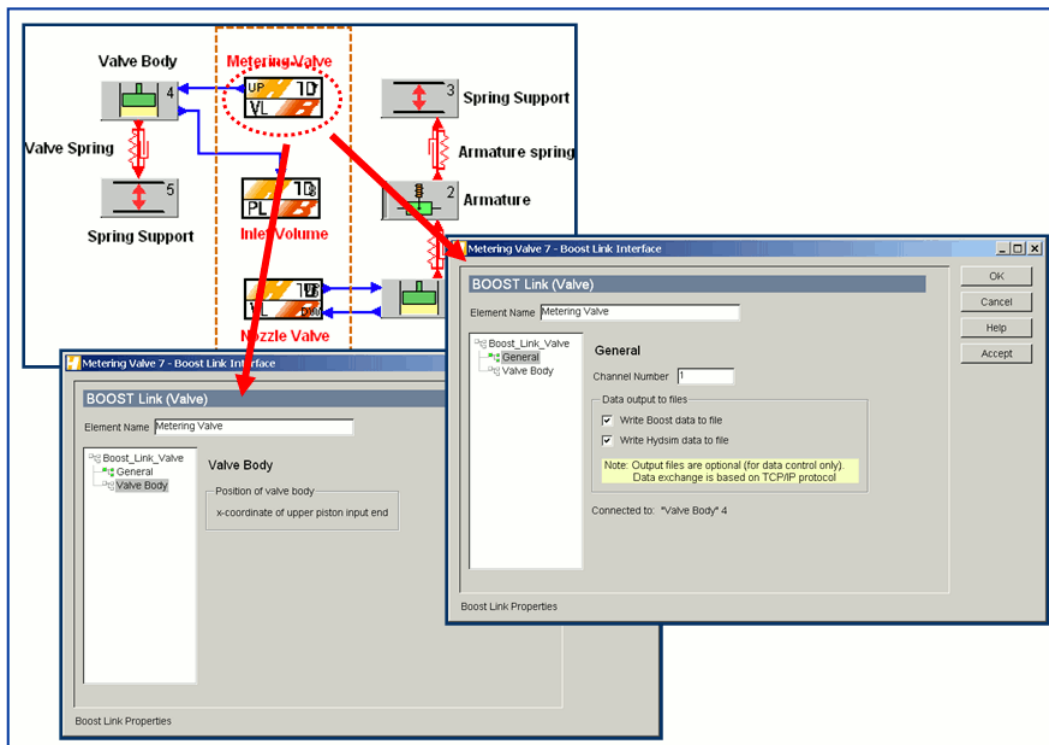
Metering Valve is an interface element of BOOST Link (Valve) type. It is connected by the hydraulic connection to the input end of Valve Body (piston). In this way the input end of Valve Body is subjected to the upstream (UP) pressure from its counterpart element CV1 in the BOOST sub-model. Displacement (x-coordinate) of input end of Valve Body is communicated to BOOST. Communication takes place through 1<sup>st</sup> channel. Input dialog of Metering valve is shown in Figure 9-22.

Number of channels in the communication between BOOST and BOOST Hydsim is defined by the number of BOOST Link elements in BOOST Hydsim. The user can decide himself which channel to assign to which counterpart element in the BOOST sub-model.

#### Metering Valve /

Channel Number: 1

Position of valve body: x-coordinate of upper piston input end (sent to BOOST)



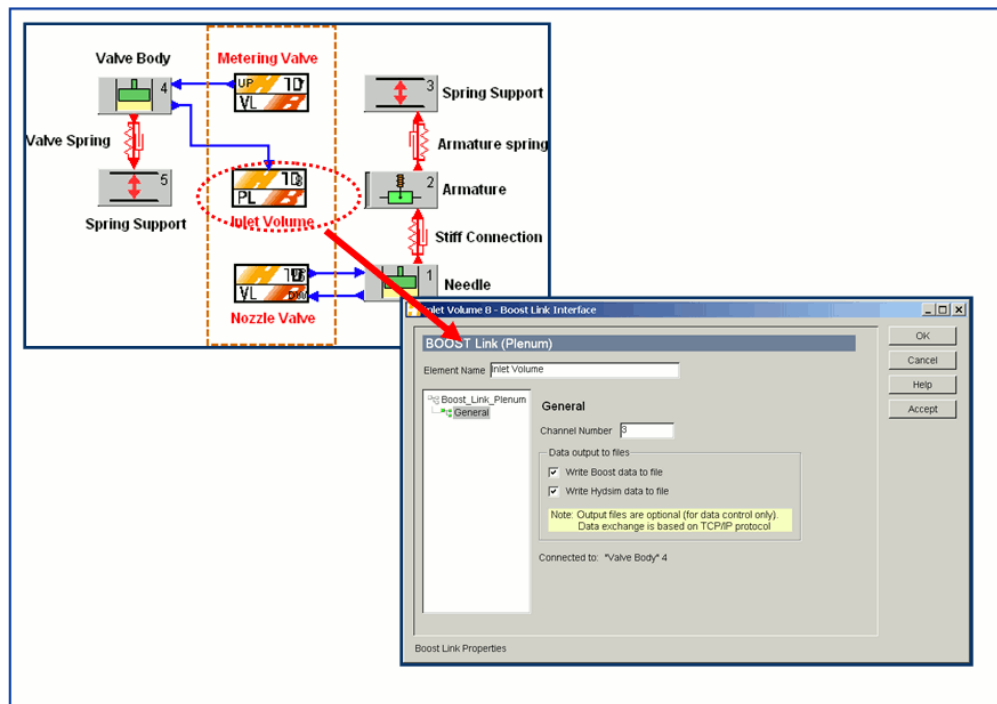
**Figure 9-22: Input dialog of Metering Valve (interface element)**

#### 9.2.2.5. Inlet Volume (Boost Link)

Inlet Volume is an interface element of BOOST Link (Plenum) type. It is connected by the hydraulic connection to the output end of Valve Body (piston). BOOST Link (Plenum) element can be linked only with Plenum elements in BOOST sub-model. Generally BOOST Link (Plenum) element in BOOST Hydsim has same characteristics like any other volume element. It may have up to 10 hydraulic connections. Each Piston element connected to ST Link (Plenum) will generate displacement flow. BOOST Hydsim will sum up these displacements and send the total sum to BOOST. Gas pressure received from BOOST will be applied (through hydraulic connection) on the respective area of connected Piston element. Input dialog of Inlet Volume is shown in Figure 9-23.

##### **Inlet Volume /**

Channel Number: 3



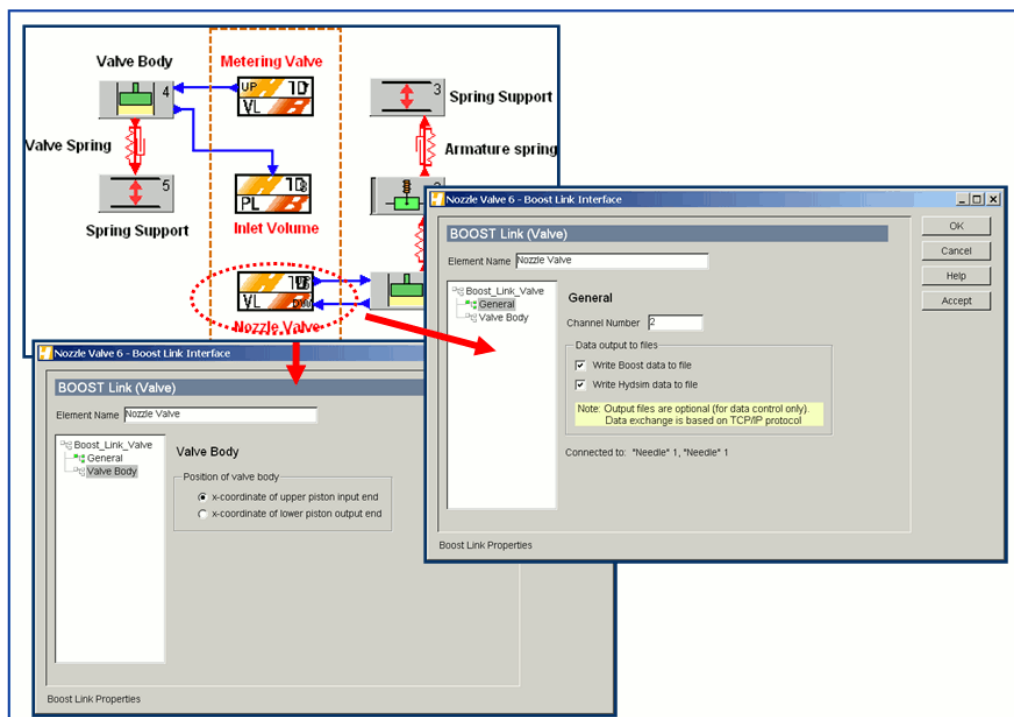
**Figure 9-23: Input dialog of Inlet Volume (interface element)**

### 9.2.2.6. Nozzle Valve (Boost Link)

#### Nozzle Valve /

Channel Number: 2

Position of valve body: x-coordinate of upper piston input end (sent to BOOST)



**Figure 9-24: Input dialog of Nozzle Valve (interface element)**

Nozzle Valve is an interface element of BOOST Link (Valve) type. It is connected by two hydraulic connections to Needle element. Input end of Needle (arrow of the hydraulic connection pointing into Needle icon) is connected to upstream (UP) anchor while output end of Needle (arrow of the hydraulic connection pointing out of Needle icon) is connected to downstream (DW) anchor of Needle Valve. This implies that the input area of Needle is subjected to the upstream (UP) pressure from Nozzle Valve CV2 element in the BOOST sub-model. Analogously, the output area of Needle is subjected to the downstream (DW) pressure from Nozzle Valve CV2. BOOST Hydsim sends to BOOST the coordinate of the Needle input end because this coordinate is activated in the input dialog shown in Figure 9-24. The decision which coordinate should be send to BOOST has to be made by the user. In our case it may be the needle input or output coordinate. However, BOOST can receive only one coordinate of one Piston element. If the Piston is elastic (i.e. it has two different coordinates at input and output end) or if two Piston elements are connected to BOOST Link (Valve), then the user has to choose the appropriate coordinate of one Piston for the transfer to BOOST.

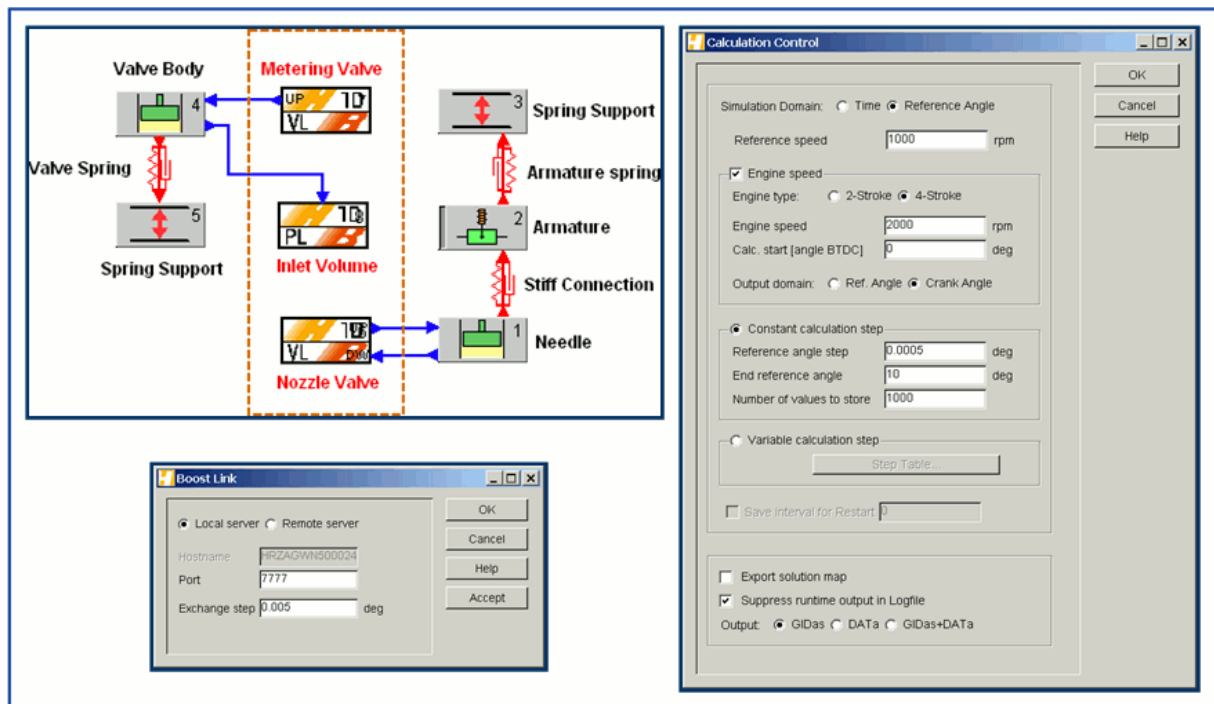
### 9.2.2.7. Calculation Control

#### Calculation Control / Simulation /

Reference Speed: 1000 rpm  
Engine Speed: 2000 rpm  
Output Domain: Crank Angle  
Reference angle step: 0.0006 deg  
End Reference Angle: 12 deg  
Number of values to store: 400

#### Boost Link / Simulation / ACCI Interface /

Hostname: localhost  
Port number: 7777  
Exchange step: 0.012 deg

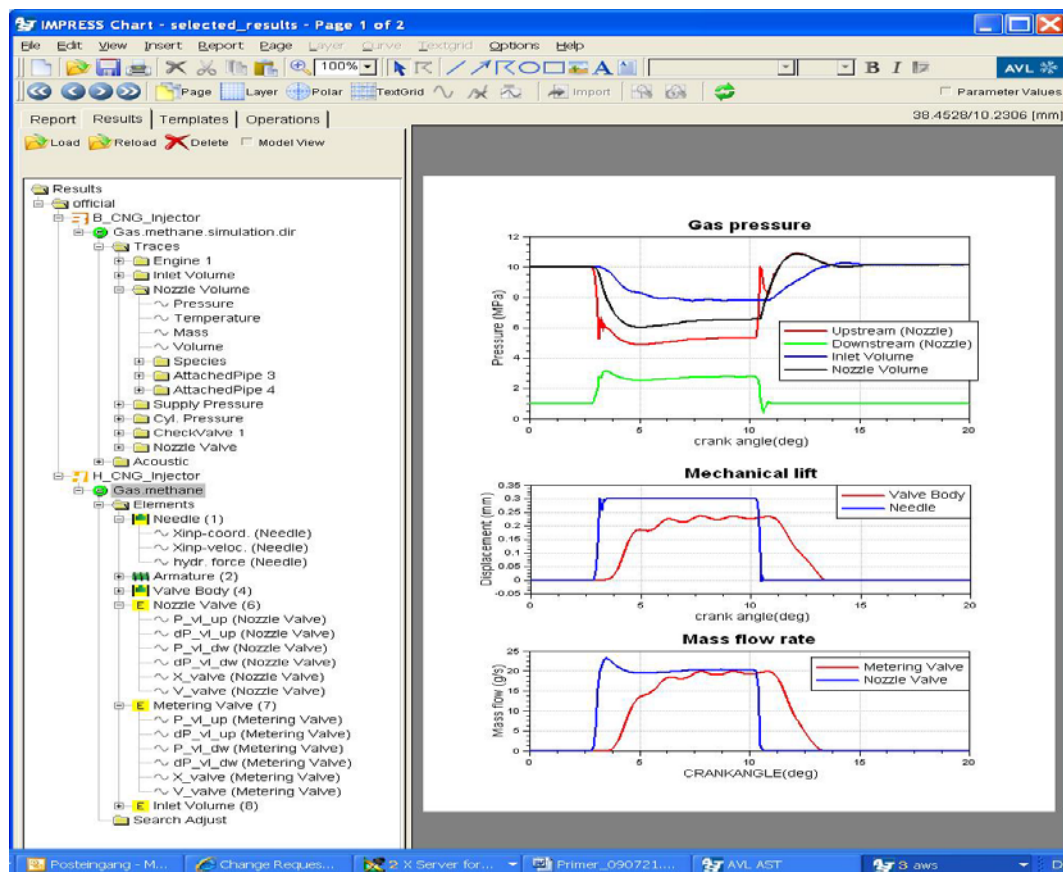


**Figure 9-25: Calculation Control and ACCI Interface dialogs**

**Calculation Control** and **ACCI Interface** dialogs are shown in Figure 9-25. There all relevant data necessary for BOOST Hydram simulation and communication with BOOST are defined. As BOOST uses the crank angle domain for calculation, in BOOST Hydram the **Crank angle** output domain is selected, too. Note that the data exchange step with BOOST is chosen 10 times larger than the BOOST Hydram calculation step. To speed up the data exchange via ACCI interface (which is the most time-consuming calculation part), the exchange step can be chosen even up to 20 times larger but the result accuracy has to be controlled. The reason behind it is that gas dynamics is numerically much more “softer” problem than multi-body or fluid dynamics. Start and end of calculation is determined by BOOST Hydram as co-simulation master.

### 9.2.3. Co-simulation Results

Selected calculation results in the crank angle domain are shown in Figure 9-26. Note that both BOOST and BOOST Hydram results are easily postprocessed in the same IMPRESS Chart window.



**Figure 9-26: Selected results of BOOST Hydsim -BOOST co-simulation**



# 10. BOOST HYDSIM – FIRE LINK

---

## 10.1. Introduction

---

BOOST Hydsim-FIRE Link is an on-line coupling between the BOOST Hydsim injector model and FIRE nozzle flow model. It is designed exclusively for the coupled 1D-3D nozzle flow simulation. For this purpose, there is a **Fire Link (Nozzle)** element in **External** group of BOOST Hydsim element tree. It can be connected via a wire link to two **Nozzle** group elements (only): **Extended SAC Nozzle** and **Extended VCO Nozzle**. Data exchange between BOOST Hydsim and FIRE kernels is carried out via ACCI interface.

The coupled simulation technique works in the following way: BOOST Hydsim controls the co-simulation process and calls the FIRE multiphase solver at the time the needle lift reaches a prescribed tolerance (option 1) or at user-defined data exchange intervals (option 2). Between these intervals, e.g. pilot and main injection, when the needle is closed, the CFD simulation is carried out with a much larger or even one single time step. In this way the computational effort for the simulation of entire injection cycle is minimized. Concerning the physical data exchange, BOOST Hydsim provides to FIRE the current lift of the needle tip and the fuel pressure and temperature at the interface. Based on these data, FIRE moves the computational mesh, adjusts the boundary conditions, computes a time step and sends back to BOOST Hydsim the hydraulic force acting on the needle tip, mass flow rate through the needle seat and spray holes and spatially averaged pressure and temperature in the nozzle sac.

## 10.2. Common Rail Injector Model with FIRE Link

---

This chapter guides the user through the BOOST Hydsim-FIRE Link example based on the diesel common rail injector model with Sac nozzle.

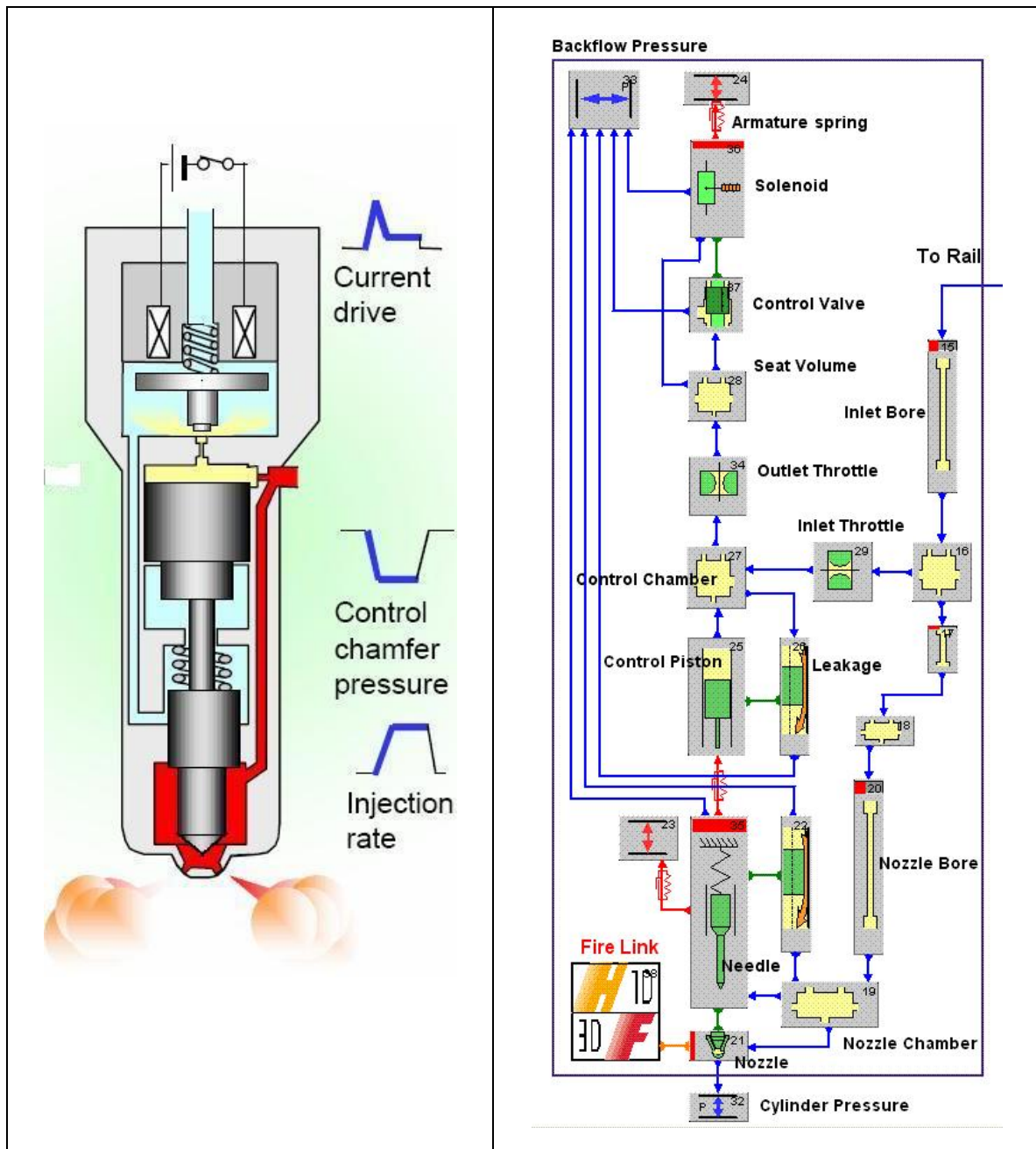
### 10.2.1. BOOST Hydsim Model of Injector

The injector in question is a classical solenoid-controlled injector with 2/2 way valve. Schematic of the injector is shown in Figure 10-1. The injector consists of the rail inlet, high and low pressure bores and volumes, sac-type nozzle with needle, control piston moving in the control chamber, inlet and outlet orifices and control valve actuated by the solenoid armature. Hydraulic boundary conditions for the injector model are the high pressure connection to rail, backflow, leakage pressure and cylinder pressure. Electrical boundary condition is the actuation current of the solenoid coil.

FIRE Link element in BOOST Hydsim is a symbolic icon connected to the Nozzle element by the so-called wire connection. Within its dialog the exact interface position, needle lift tolerance, shaft diameter, time delay till the co-simulation start, exchange time step and other control parameters are defined. FIRE and BOOST Hydsim calculation domains with exchange variables are shown in Figure 10-2. On each exchange step FIRE transfers to BOOST Hydsim the following data:

- $F_{Nx}$  – x-component of needle tip force vector
- $q_{seat}$  – flow rate through needle seat
- $q_{holes}$  – flow rate through spray holes (outlet)

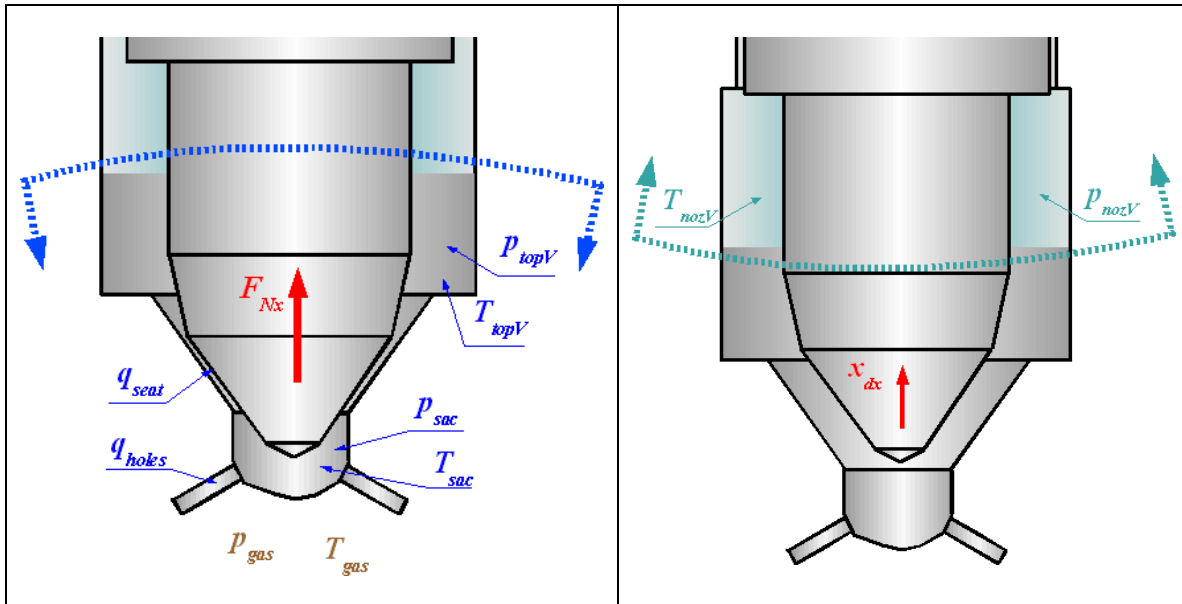
- $p_{topV}$  – boundary pressure below interface line
- $T_{topV}$  – boundary temperature below interface line
- $p_{sac}$  – average pressure in sac volume
- $T_{sac}$  – average temperature in sac volume
- $p_{gas}$  – boundary pressure in spray chamber
- $T_{gas}$  – boundary temperature in spray chamber



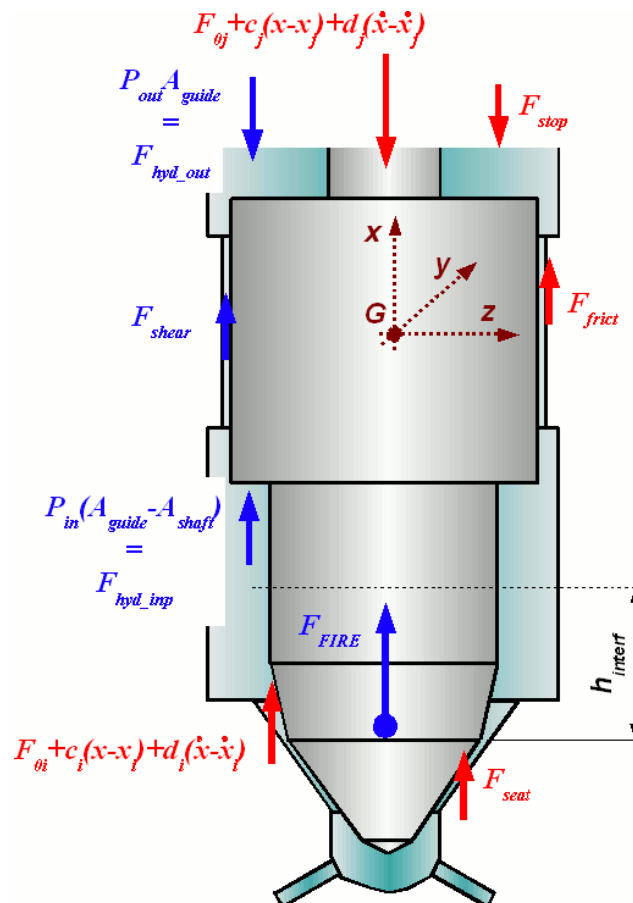
**Figure 10-1: Schematic and BOOST Hydsim model of Common Rail injector**

In return at the end of the exchange time step BOOST Hydsim transfers to FIRE the following data:

- $X_{dx}$  – x-component of needle tip displacement vector
- $p_{nozV}$  – boundary pressure above interface line
- $T_{nozV}$  – boundary temperature above interface line



**Figure 10-2: FIRE and BOOST Hydsim Calculation domains with exchange variables**



**Figure 10-3: Forces acting on Needle**

In the actual version of the coupling only the longitudinal x-component of the needle tip force and displacement vectors are used. Needle lift tolerance for the FIRE solver call is defined in this example by 4  $\mu\text{m}$ . BOOST Hydsim calculation step is set to 10-7 s, exchange step with FIRE is 20 time larger (2x10-6 s).

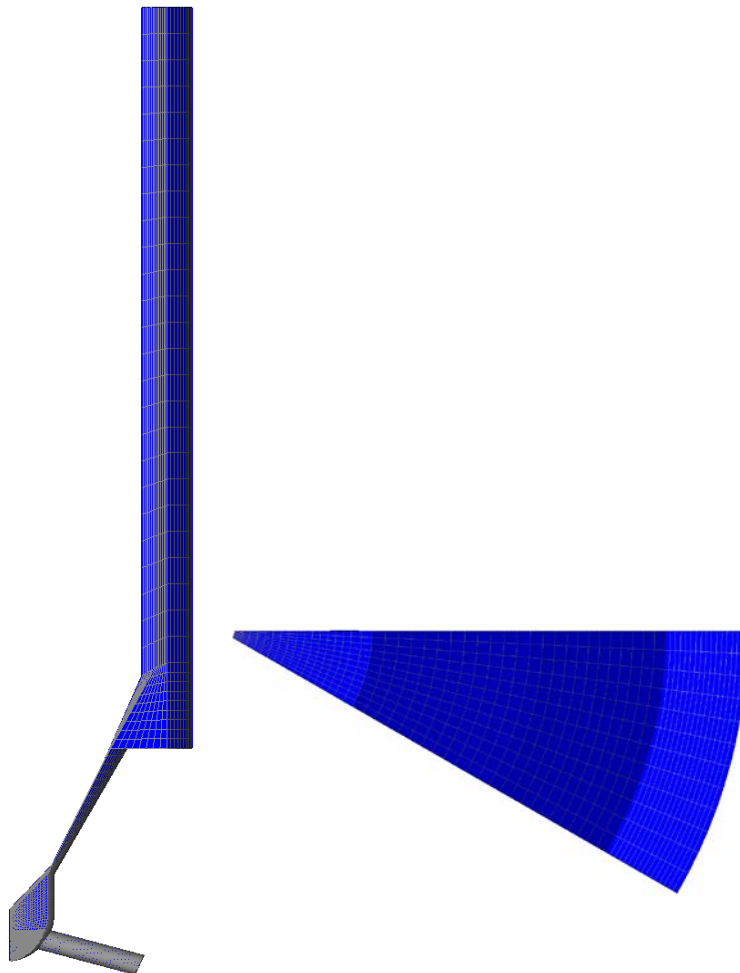
This step is applied only for the intervals where needle lift exceeds the tolerance. Outside of these intervals (i.e. practically between the injection events) 5 times smaller or even one single time step is used.

Schematic of injector needle with hydraulic (blue color) and mechanical forces (red color) acting in longitudinal direction is shown in Figure 10-3. These forces can be classified as follows:

- $F_{FIRE}$  – force on needle tip imported from FIRE
- $F_{hyd\_inp}$  – input pressure force acting on needle guide from nozzle volume
- $F_{shear}$  – viscous shear force at guide due to leakage
- $F_{hyd\_out}$  – output pressure force on needle back side from back volume
- $F_{seat}$  – needle reaction force at contact with seat
- $F_{stop}$  – needle reaction force at contact with stop
- $F_{frict}$  – Coulomb friction force at needle guide
- $F_0 + cx + dx'$  – mechanical connection force from the neighboring element

### 10.2.2. FIRE Nozzle mesh

Figure 10-4 shows the computational grid used in the CFD simulation. The whole domain consists of the hollow cylinder around the needle shaft, needle seat area with nozzle sac and six holes. To reduce the computational effort, only one of twelve periodic 3D-degree-segments is modeled. The cake-mesh of the nozzle sac and hole is shown in Figure 10-5.



**Figure 10-4: SAC-nozzle cake-mesh, front view (left) and top view (right)**



**Figure 10-5: SAC-nozzle cake-mesh, nozzle sac region with nozzle hole**

The mesh consists of 33620 hexahedral cells with 12 cell layers in the circumferential direction so that one layer corresponds to 2.5 degrees. The outlet bore is meshed separately and connected to the sac-volume by arbitrary-interfaces. The nozzle gap itself contains 31 cell rows in the radial direction (which is the main flow direction) and 8 rows in vertical direction (normal to the main flow direction). These cells are from 10 (at the seat edge) to 90  $\mu\text{m}$  long and 12.5  $\mu\text{m}$  high at needle lift of 100  $\mu\text{m}$ . This mesh topology is chosen according to the AVL experience with the 3D nozzle flow simulation.

The needle movement is accomplished by the mesh-deformation at solver run-time. Only one mesh is created in a reference position. The mesh-deformation-function shifts the needle surface at each time step according to the needle displacement received from BOOST Hydsim. After that, the position of the internal nodes is updated by the Laplace interpolation scheme. To prevent collapsing cells at zero or very small needle lift, a minimal gap size (5  $\mu\text{m}$  in the actual case, measured normal to the needle seat) is maintained in the mesh. The needle movement below this minimal gap is simulated by assigning a very high flow resistance to all those cells that are actually located within the solid needle.

For more details of FIRE nozzle mesh please consult Verification Report – FIRE-BOOST Hydsim Online Coupling, SAC Nozzles.

### **10.2.3. FIRE Nozzle Interface Data**

FIRE Nozzle interface is represented by a symbolic **FIRE Link** element in BOOST Hydsim. The co-simulation data have to be defined in its **Properties** dialog. This dialog consists of **General** and **Control** sheets shown in Figure 10-6 and Figure 10-7, respectively. In **General** sheet the user has to define the interface border height from needle seat (any position between seat and guide can be chosen), needle shaft diameter at interface and (optionally) names of data output files with FIRE and BOOST Hydsim exchange data (very useful for cross-checking). In **Control** sheet the exchange time/angle step (at open needle) between BOOST Hydsim and FIRE has to be specified. Usually, it should be 10 to 20 times higher than BOOST Hydsim calculation step.

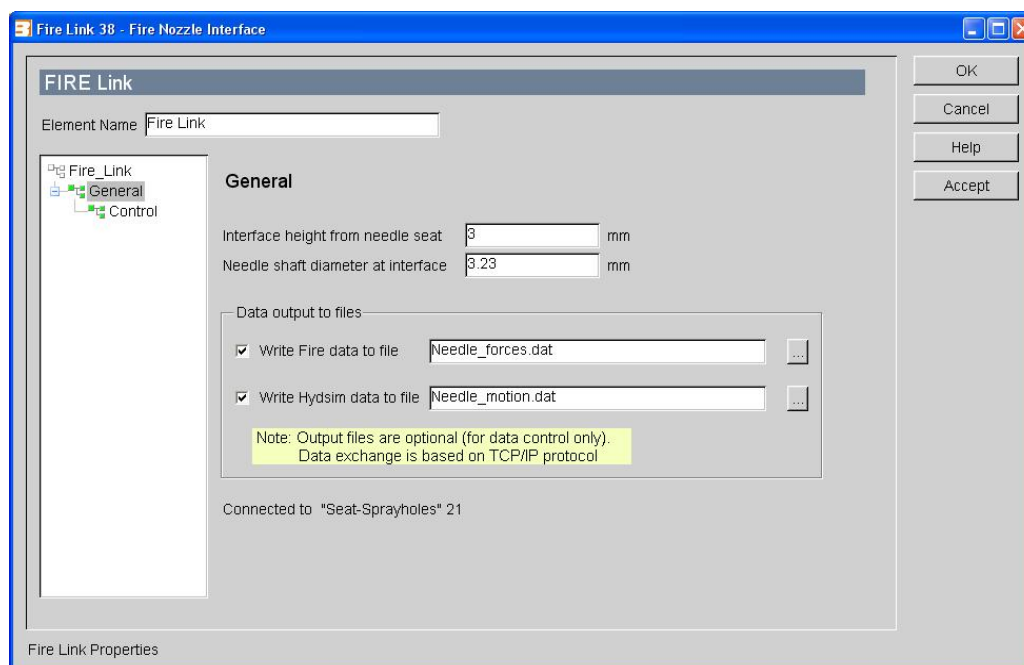


Figure 10-6: FIRE Link General Dialog

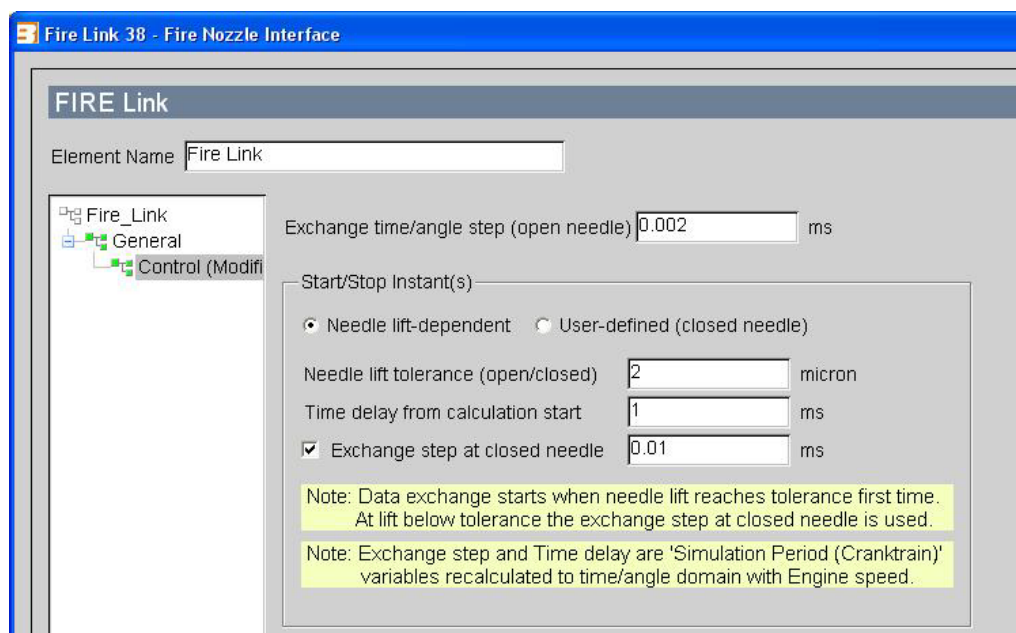


Figure 10-7: FIRE Link Control Dialog

Within **Control** sheet two options of co-simulation with FIRE solver can be chosen. First, **Needle-lift dependent**, option implies initial call of the FIRE solver (start of data exchange) at user-defined needle tolerance (2  $\mu\text{m}$  in our example). At needle lift below tolerance the exchange step at closed needle (optional parameter) is used. If not defined, it is set by the program to the exchange step at open needle. Additionally parameter **Time delay from calculation start** has to be specified (1 ms in our example). BOOST Hydsim-FIRE co-simulation cannot start earlier than this time interval expires. This parameter is required to prevent the unexpected co-simulation begin at start where BOOST Hydsim model may not be in the equilibrium position (i.e. needle may have initial lift due to wrong

initials conditions). Second, **User-defined (closed needle)**, option simply implies calling of the FIRE solver at user-defined time/angle intervals.

Between the intervals a single (i.e. usually very high) exchange time step is used. In this way, the CFD calculation is skipped from the end of the previous interval till the beginning of the next interval. This allows speeding up the simulation procedure considerably. However, the user has to take care that the specified intervals fully cover the injection events. Otherwise, if e.g. the needle is already open at the interval start or not yet closed at the interval end, an error message will be produced.

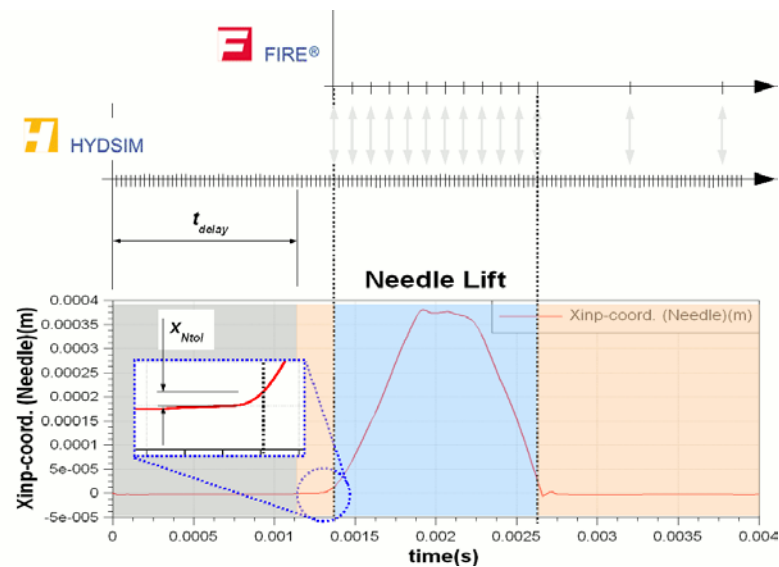


Figure 10-8: Co-simulation with Needle-lift dependent option

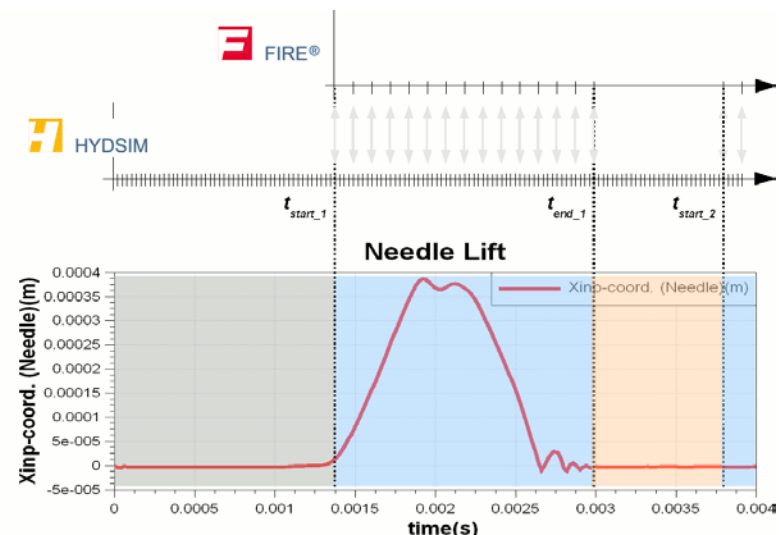
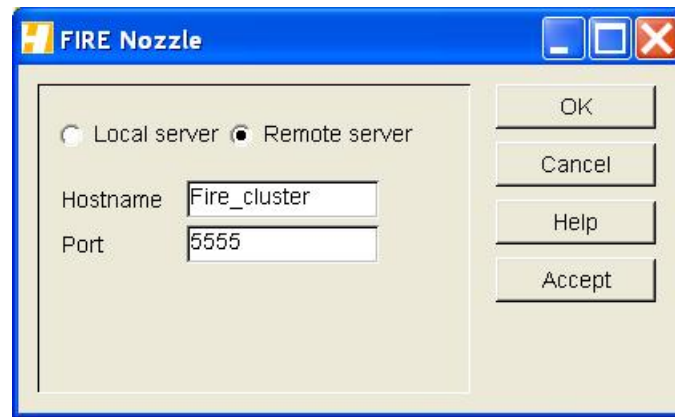


Figure 10-9: Co-simulation within User-defined time interval(s)

### 10.2.4. Running Co-simulation

Before starting the BOOST Hydsim- FIRE co-simulation, the name and port number of the ACCI server have to be specified within the ACCI Interface/FIRE Nozzle dialogs shown in Figure 10-10.





**Figure 10-10: ACCI interface FIRE Nozzle dialog**

The same server name and port number have to be defined in the FIRE case, too. Then standard BOOST Hydsim simulation with Run command can be performed. Restart calculation is also supported.

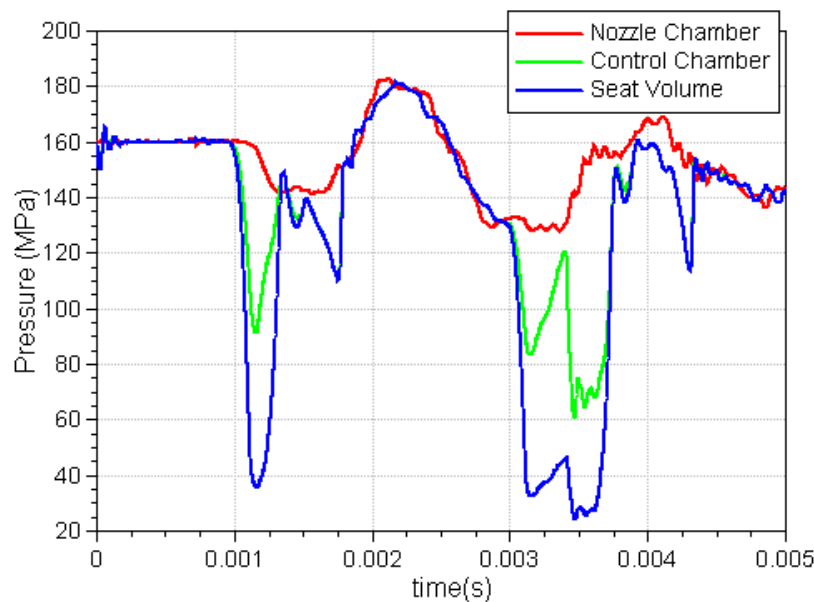
### 10.2.5. Coupled Simulation Results

BOOST Hydsim calculates the injector flow till interface border and mechanical motion of the needle, control piston and solenoid armature. For this, hydraulic, mechanical, electromagnetic and shear forces are calculated first within the entire system. The flow rate through needle seat and spray holes is obtained from FIRE. Depending on the flow regime (laminar or turbulent) this flow rate is compared with the simplified 1D flow theory. Basically, discharge coefficient at the narrowest cross-sectional area of the needle seat is estimated and checked for limiting range. If for the turbulent flow regime this coefficient exceeds maximal value (1), a warning message is produced in **View Logfile** window (actually the 5% error is formally allowed).

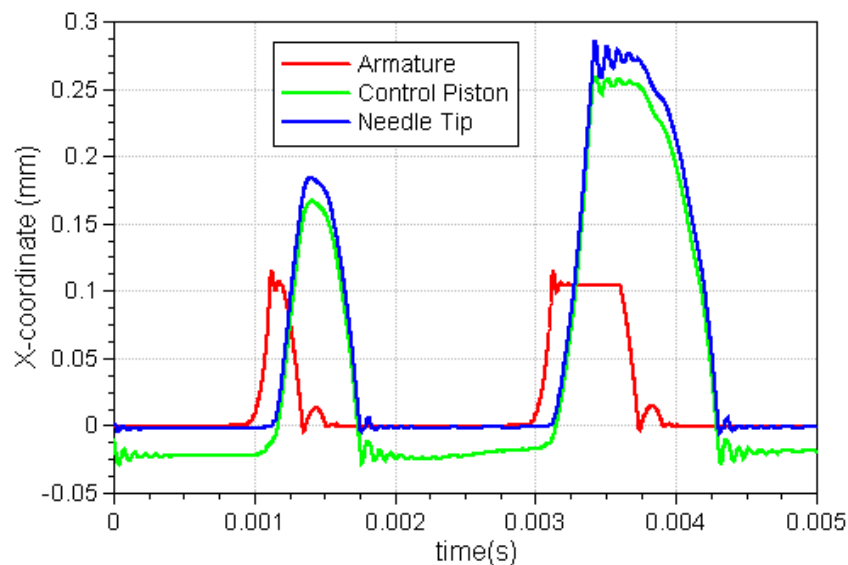
Within the actual example, two injection events (pilot and main) are modeled. Pressures in the nozzle and control chambers and seat volume are depicted in Figure 10-11. Lift of the solenoid armature, control piston and needle is shown in Figure 10-12. One can observe that the delay between the solenoid actuation (armature motion) and needle lift start is 0.2 ms. At pilot injection the needle lift is always ballistic. The control piston is modeled as an elastic body, therefore its initial lift is negative because the piston is compressed by the rail pressure in the control chamber. As soon as the control valve opens, the pressure in the control chamber is gradually released. This causes the elastic relaxation of the control piston and the delayed opening of the needle tip. Hydraulic forces acting on needle tip (from FIRE) and guide (from BOOST Hydsim) are depicted in Figure 10-13. Flow through the inlet and outlet orifices and nozzle (injection rate) is plotted in Figure 10-14. Note that by performing BOOST Hydsim calculation standalone, flow discharge coefficients at needle seat and spray holes have to be specified on input. These coefficients are unknown and usually can be obtained only by the specific experiment. With BOOST Hydsim-FIRE co-simulation, these coefficients are not required: they are estimated automatically as the by-product of the calculation. This is a big advantage of the coupled 1D-3D simulation over the standard 1D simulation alone.



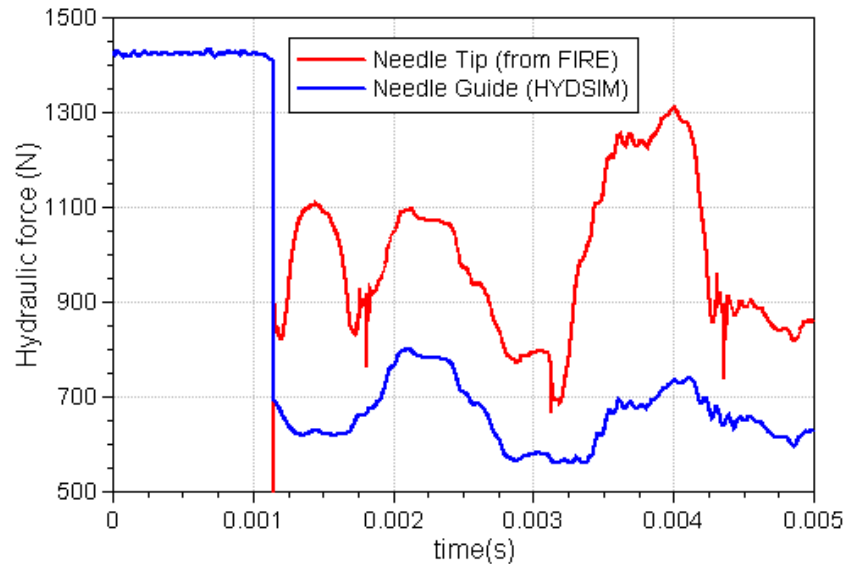
Valuable information on the nozzle cavitation can be extracted from the effective cross-sectional flow areas at needle seat and nozzle holes. These flow areas (geometric and effective) are depicted in Figure 10-15. The total geometric area of six nozzle holes is  $0.24 \text{ mm}^2$  (constant value). Total effective area of holes varies from  $0.161 \text{ mm}^2$  at turbulent non-cavitating flow to  $0.146 \text{ mm}^2$  at cavitating flow. Effective flow area at needle seat is a function of nozzle geometry and needle lift. It reaches the effective holes area of  $0.146 \text{ mm}^2$  at  $90 \text{ }\mu\text{m}$  needle lift for both pilot and main injections. At this lift the cavitation transition from the needle seat to nozzle holes gradually occurs. Note that FIRE time counter is shifted from the BOOST Hydsim time by  $1.14 \text{ ms}$  because FIRE calculation is started at needle lift of  $5 \text{ }\mu\text{m}$  (user-defined tolerance). In other words, BOOST Hydsim time of  $1.14 \text{ ms}$  implies zero time in FIRE.



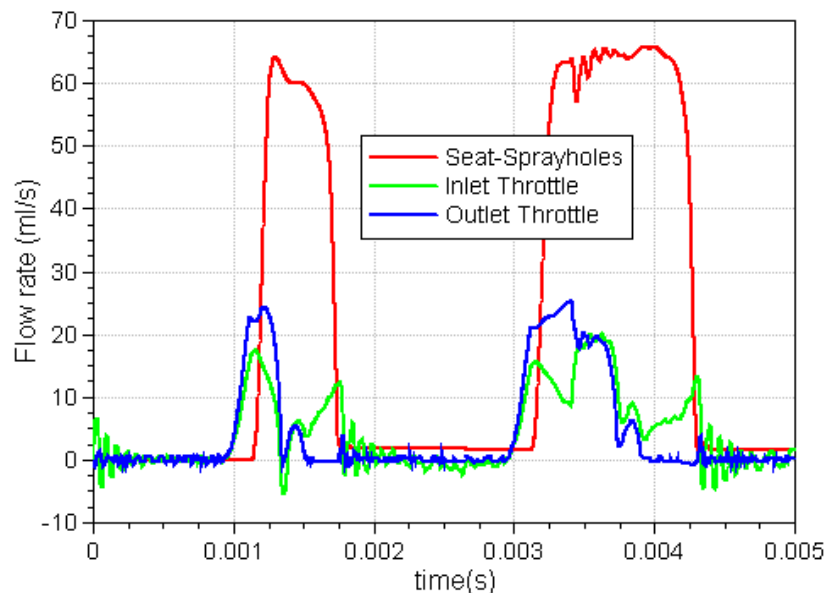
**Figure 10-11: Pressure in nozzle and control chambers and seat volume**



**Figure 10-12: Motion of solenoid armature, control piston and needle tip**



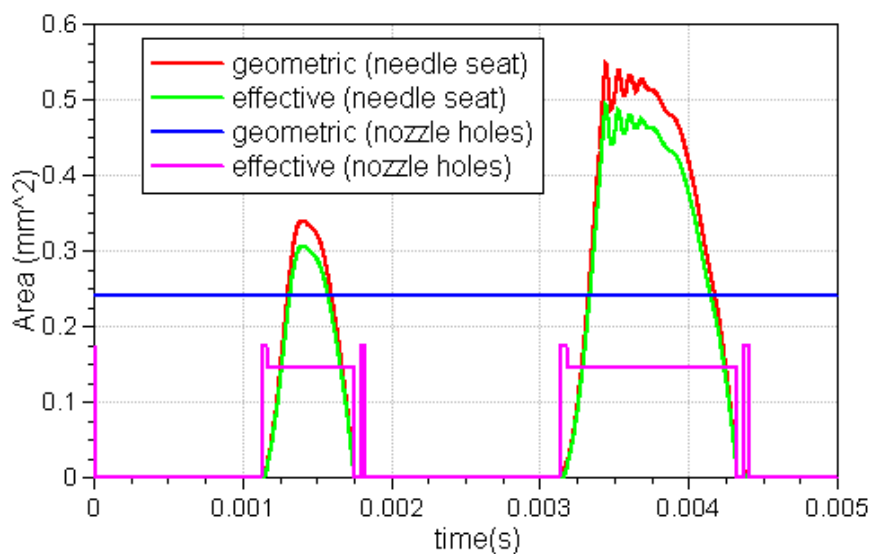
**Figure 10-13: Hydraulic force on needle tip (FIRE) and guide (BOOST Hydsim)**



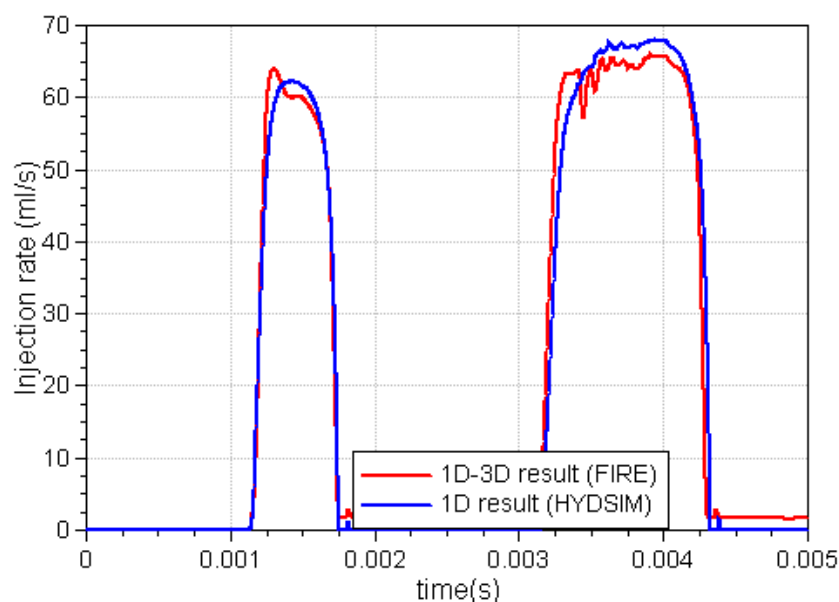
**Figure 10-14: Flow rate through nozzle (seat and holes) and inlet/outlet throttles**

From the FIRE calculation we can derive the average flow discharge coefficient values at needle seat and spray holes. For this the classical Bernoulli equation is used, both for cavitating and non-cavitating turbulent flow regime. In this way, the needle seat discharge coefficient value is estimated as 0.9. Nozzle hole discharge coefficients are identified to be 0.67 (at non-cavitating turbulent flow) and 0.61 (at cavitating flow), respectively. Using the identified values, the BOOST Hydsim standalone calculation with the 1D nozzle flow model is performed and good correlation with the results of 1D-3D coupled simulation is attained. Needle lift in both pilot and main injection is almost identical for both 1D and 1D-3D simulation cases. Injection rates also fit together reasonably well as shown in Figure 10-16. This leads to the important conclusion that the flow discharge coefficients, derived from coupled 1D-3D simulation, can be successfully used for the further 1D nozzle calculation with BOOST Hydsim alone.

Obviously this implies a substantial saving of the computational time. For comparison, coupled 1D-3D simulation of the actual injector with the optimized calling procedure of the CFD solver requires 7 to 8 hours CPU time on a typical PC Intel Core 2 while 1D simulation with BOOST Hydsim alone takes only 15 to 30 s.



**Figure 10-15: Geometric and effective flow area of needle seat and nozzle holes**



**Figure 10-16: Injection rate from 1D-3D co-simulation and 1D simulation alone with adjusted flow discharge coefficients**



# 11. CONCLUDING REMARKS

---

This Primer is intended as an extended introduction for a BOOST Hydsim user. It describes the modeling and analysis of different types of fuel injection and hydraulic valve train systems, which are commonly used in automotive, marine propulsion, and other combustion engines. However, the Primer does not include the advanced models of complete injection systems, e.g. multiple injectors, complete high-pressure fuel supply systems, peripheral devices, etc. These systems, of course, can also be modeled with BOOST Hydsim, but may require certain prior experience and substantial computational effort. Their handling is explained in the BOOST Hydsim training courses.

The capabilities of BOOST Hydsim are not limited to modeling of fuel injection systems and hydraulic valve trains. In principle, BOOST Hydsim can be applied to the dynamic analysis of any type of hydraulic and hydromechanical system, as long as one-dimensional flow modeling is sufficient. Such types of systems are e.g. fuel storage, supply or filling/spill systems, hydraulic power transmission and control units etc. This Primer does not cover the modeling of hydraulic or hydro-mechanical systems other than fuel injection systems and hydraulic valve trains.

Till recent time BOOST Hydsim was usually used as stand-alone software for the simulation of hydraulic systems. However, now it became an even more powerful tool for engine analysis when combined with other AVL products: FIRE Multi-phase Flow and Spray modules, BOOST for engine cycle simulation and EXCITE for engine vibration and acoustic analysis. Recently the coupling interfaces between BOOST Hydsim and FIRE for the nozzle flow simulation and BOOST Hydsim and BOOST for the hydromechanical-gaseous system simulation have been developed. Data-exchange interfaces to other AVL products can be developed upon the user request.

The delivery of BOOST Hydsim software contains all examples described in this Primer: simple hydraulic system, common-rail injectors with two-way and three-way solenoid valves, unit injector, mechanical and electronically-controlled in-line pump systems, electronic unit pump, electro-hydraulic valve trains and common rail models with PID controllers. These and many other examples are located in the directory `<install_dir>/examples/BOOST Hydsim`. They can be directly loaded and executed in AVL Workspace/BOOST Hydsim as soon as it is installed.

The list of main examples delivered with the BOOST Hydsim installation is provided in the Appendix.



## 12. APPENDIX

---

Below you can find a list of typical examples delivered with the BOOST Hydsim installation. Examples are located in the directory `<install_dir>/examples/BOOST Hydsim/vxxxxx/<examples_group>`. The list of BOOST Hydsim examples is being constantly extended. Therefore a number of similar additional examples (not included into the below table) can be found in the installation.

**BOOST HYDSIM Examples****825\_Aftertreatment**

Urea\_dosingsystem\_1.hyd

Urea\_dosingsystem\_2.hyd

**870\_BOOST Link**

B\_CNG\_Injector.bwf

Primer

Boost file

B\_Piston\_Plenum.bwf

Boost file

B\_Valve\_Piston.bwf

Boost file

H\_CNG\_Injector.hyd

Primer

Hydsim file

ACCI link

H\_Piston\_Plenum.hyd

Hydsim file

ACCI link

H\_Valve\_Piston.hyd

Hydsim file

ACCI link

**845\_Control Valve**

Check\_valve1\_std.hyd

Check\_valve2\_std.hyd

Check\_valve.hyd

Control\_valve1.hyd

Control\_valve2.hyd

Spring\_valve\_std.hyd

**835\_Engine brake**

engine\_brake\_6cyl.hyd

engine\_brake\_valve\_act.hyd

engine\_brake\_valve\_gas.hyd

**801\_Fuel Injection – diesel**

Comrail\_piezo1\_mrc.hyd

Primer

Comrail\_piezo1\_mrc\_therm.hyd

Comrail\_piezo1\_svd.hyd

Primer

Comrail\_piezo2\_mrc.hyd

Comrail\_piezo2\_svd.hyd

CR13\_piezo1\_Btube\_ap.hyd

CR13\_piezo2\_Btube.hyd

Primer

CR13\_piezo2\_mctip.hyd

HP\_pump\_shaft\_2.hyd

HP\_pump\_shaft\_3.hyd

common\_rail2wv.hyd

Primer

common\_rail2wv\_ap.hyd

common\_rail2wv\_ap\_2.hyd

common\_rail2wv\_therm.hyd

common\_rail3wv.hyd

Primer

common\_rail3wv\_2.hyd



common_rail3wv_ap.hyd			
common_rail32sw.hyd			
comrail2wv_3cases.hyd			
CRail2wv_spray_nozzle.hyd			
CRail2wv_spray_orifice.hyd			
CRail_Injector1.hyd			
CRail_Injector1_therm.hyd			
CRail_Injector2.hyd			
CRail_Injector2_fdb.hyd			
CRail_Injector3.hyd			
CRail_Injector3_ap.hyd			
CRI_Btube_Fm_ext.hyd			
CRI_Btube_Fm_int.hyd			
CRI_RSN_leakages.hyd			
CRIP2_injector_Fm_el_int.hyd			
CRIP2_injector_Fm_el_int_pd.hyd			
CRIP2_injector_Fm_el_int_th.hyd			
CRIP2_injector_Fm_int.hyd			
CRIP2_injector_Fm_int_pd.hyd			
RSN_SAC_nozzle.hyd			
RSN_VCO_nozzle.hyd			
CR_system4_bends.hyd			
CR_system6_bends.hyd			
CR_system_V12.hyd			
CR13_piezo2_4inj.hyd			
CR13_piezo3_4inj.hyd			
distributor_pump.hyd			
distributor_pump_obsn.hyd			
distributor_pump_therm.hyd			
radial_pump.hyd			
radial_pump_inv_con.hyd			
radial_pump_obsn.hyd			
radial_pump_therm.hyd			
VE_pump_fire.hyd			
VP44_pump.hyd			
hpi_tp_injector1.hyd			
hpi_tp_injector1_therm.hyd			
hpi_tp_injector2.hyd			
hpi_tp_injector2_therm.hyd			
cam_contact_loss.hyd			

electronic_pump.hyd	Primer		
electronic_pump_cases.hyd			
electronic_pump_inv_con.hyd			
electronic_pump_nozzle.hyd			
Inline_pump1.hyd	Primer		
Inline_pump1_ap.hyd			
Inline_pump2.hyd			
Inline_pump.hyd			
Inline_pump_ap.hyd			
PLD_GRV.hyd			
plunger_basic_inline.hyd			
plunger_head_groove.hyd			
plunger_relief_grinding.hyd			
pde_awk_4cases_1.hyd			
pde_awk_4cases_1fdb.hyd			
pde_awk_4cases_2.hyd			
pde_awk_4cases_2fdb.hyd			
timing_unit_inj_4cyl.hyd			
unit_inj_shaft_rocker.hyd			
unit_inj_sid_piston1.hyd			
unit_inj_sid_piston2.hyd			
unit_injector.hyd	Primer		
unit_injector_E31.hyd			
unit_injector_obsn.hyd			
unit_injector_restart.hyd	Primer		
unit_injector_rocker1.hyd			
unit_injector_rocker.hyd			
unit_injector_shaft.hyd			
unit_injectors_4.hyd			
<b>805_Fuel Injection – dimethyl_ether</b>			
common_rail_4dme.hyd			
common_rail_4dme_2.hyd			
DME_injector_3sets.hyd			
DME_injector_inv_con.hyd			
<b>810_Fuel Injection – FIRE Link</b>			
SAC_CRI2_lift_ca.hyd		Hydsim file	ACCI link
SAC_CRI2_lift_dt.hyd	Primer	Hydsim file	ACCI link
SAC_CRI2_userdef_ca.hyd		Hydsim file	ACCI link
SAC_CRI2_userdef_ca.hyd		Hydsim file	ACCI link

common_rail2wv_Fire.hyd			Off line
CRIP2_injector_Fm_int.hyd			Off line
VE_pump_Fire.hyd			Off line

### 815\_Fuel Injection – Gas

gas_injector4.hyd			
gas_injector4_opt.hyd			

### 811\_Fuel Injection – Gasoline

GDI_injector1.hyd			
GDI_injector1_therm.hyd			
GDI_simple.hyd			
GDI_simple_therm.hyd			
GDI_system.hyd			
GDI_system_full.hyd			
GDI_system_simple.hyd			
HDEV5_injector.hyd			
TGDI_rail.hyd			
V8_ported_rail1.hyd			
V8_ported_rail2.hyd			

### 802\_Fuel Injection – Heavy Oil

fuelsystem_2inj.hyd			
Hydraulic_Cylinder_Unit_1.hyd			
injection_valve1.hyd			
injection_valves3.hyd			
Mech_pump.hyd			

### 820\_Fuel Injection – Other

Flow_meter_hp.hyd			
Nozzle_flow_coef.hyd			
Sac_nozzle_multicont.hyd			
Sac_nozzle_standard.hyd			
Sac_volume1.hyd			
Sac_volume1_RK5.hyd			
Sac_volume2.hyd			
Sac_volume1_RK5.hyd			
Sac_volume_gas.hyd			
Sac_volume_obsn1.hyd			
Sac_volume_obsn2.hyd			

### 850\_General

abs_roughness_const.hyd			
abs_roughness_var.hyd			

Bended_lines.hyd			
Contact_test.hyd			
fixed_friction_factor.hyd			
Leakage_2pistons.hyd			
Leakage_gap_comp.hyd			
Leakage_userdef_2Dtable.hyd			
Leakage_userdef_3Dtable_1.hyd			
Leakage_userdef_3Dtable_2.hyd			
local_coordinates_Cam.hyd			
local_coordinates_Rocker.hyd			
mcormack_line_1ph.hyd			
mcormack_line_1ph_fr.hyd			
mcormack_line_2ph.hyd			
mcormack_line_2ph_fr.hyd			
Orifice_std.hyd			
Orifice_std_therm.hyd			
simple_line1_ch.hyd			
simple_line1_dA.hyd	Primer		
simple_line1_gd.hyd			
simple_line1_lp.hyd			
simple_line1_mc.hyd			
simple_line2_ch.hyd			
simple_line2_dA.hyd			
simple_line2_gd.hyd			
simple_line2_lp.hyd			
simple_line2_lp_thermal.hyd			
simple_line2_mc.hyd			
simple_line_4cases.hyd			
Stop_piston_linear.hyd			
Stop_piston_nonlinear.hyd			
Stop_pseudo_linear.hyd			
Twin_piston_contact.hyd			
<b>860_MATLAB</b>			
common_rail2wv_dll_1.hyd	Primer		
common_rail2wv_dll_1ap.hyd			
common_rail2wv_dll_2.hyd			
common_rail2wv_dll_2ap.hyd			
comrail_4injectors_p_dll.hyd			
comrail_4injectors_pid_dll.hyd			
TGDI_rail_4inj_pid_dll.hyd			

TGDI\_rail\_4inj\_pid\_dll\_2.hyd

pressure\_control\_m\_1.hyd

Primer

pressure\_control\_m\_1ap.hyd

pressure\_control\_m\_2.hyd

pressure\_control\_m\_2ap.hyd

temperature\_control\_m.hyd

Comrail2wv\_abs\_mdl\_1.hyd

Primer

Comrail2wv\_abs\_mdl\_1ap.hyd

Comrail2wv\_abs\_mdl\_2.hyd

Comrail2wv\_abs\_mdl\_2ap.hyd

Comrail2wv\_abs\_mdl\_3case\_sets.hyd

Comrail2wv\_rel\_mdl\_1.hyd

Comrail2wv\_rel\_mdl\_1ap.hyd

Comrail2wv\_rel\_mdl\_2.hyd

Comrail2wv\_rel\_mdl\_2ap.hyd

Comrail\_4injectors\_mdl.hyd

**840\_Mech. Drive**

camshaft\_torsion.hyd

camshaft\_unit\_inj4.hyd

Elastic\_shaft\_1.hyd

Elastic\_shaft\_2.hyd

Lift\_amplifier.hyd

Piezo\_actuator\_lift.hyd

Timing\_inj\_drive\_4cyl.hyd

Timing\_inj\_drive\_4cyl\_2.hyd

**830\_Valve Train**

Accumulator\_1.hyd

EHVA\_fingerfol\_1.hyd

Primer

EHVA\_fingerfol\_1el.hyd

EHVA\_fingerfol\_1el\_gas.hyd

EHVA\_fingerfol\_2.hyd

Primer

EHVA\_fingerfol\_2el.hyd

EHVA\_fingerfol\_3.hyd

Primer

EHVA\_fingerfol\_3el.hyd

EHVA\_fingerfol\_3el\_gas.hyd

Timing\_inj\_drive\_4cyl.hyd

Valve\_actuator.hyd

Var\_valve\_train.hyd