Simulink Report: Valve_Comp_

Christian Müller

2008-03-06

Model - Valve_Comp_

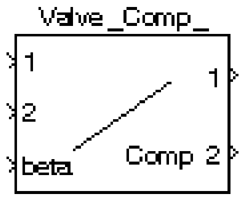


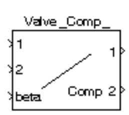
Tabelle 1.1. Valve_Comp_ Simulation Parameters

Solver ode14x	*ZeroCross* on	*StartTime* 0.0*StopTime* 10.0
RelTol 1e-3	*AbsTol* auto	*Refine* 1
InitialStep auto	*FixedStep* auto	*MaxStep* auto

Tabelle 1.2. Valve_Comp_ Summary Information

NumModelInputs	N/A	*NumModelOutputs*	N/A
NumVirtualSubsystems	N/A	*NumNonvirtSubsystems*	N/A
NumNonVirtBlocksInModel	N/A	*NumBlockTypeCounts*	N/A
NumBlockSignals	N/A	*NumBlockParams*	N/A
NumZCEvents	N/A	*NumNonsampledZCs*	N/A

Systems

Name	**Parent**	**Snapshot**	**Blocks**	**Signals**
Valve_Comp_	<root>		Valve_Comp_	Valve_Comp_<1> Valve_Comp_<2>

Blocks

Tabelle 1.3. Block Type Count

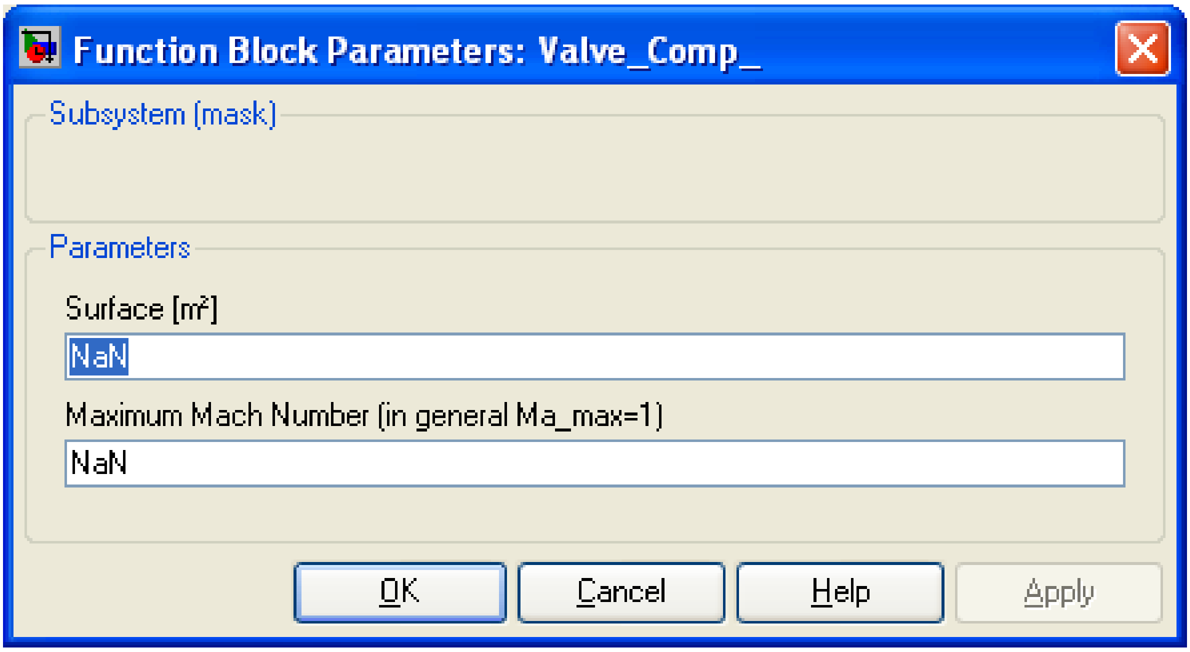
BlockType	**Count**	**Block Names**
Inport	18	In_1, In_2, beta, A, beta, Ma_max, p_in_1, rho_in_1, T_in_1, x_H2O_gas_in_1, x_CO2_in_1, x_H2O_liq_in_1, p_in_2, rho_in_2, T_in_2, x_H2O_gas_in_2, x_CO2_in_2, x_H2O_liq_in_2
Outport	11	rho, T, x_H2O_gas, x_CO2, x_H2O_liq, p_1, m_dot_air_1, p_2, m_dot_air_2, Out_1, Out_2
Constant	3	A, Ma_max, m_air

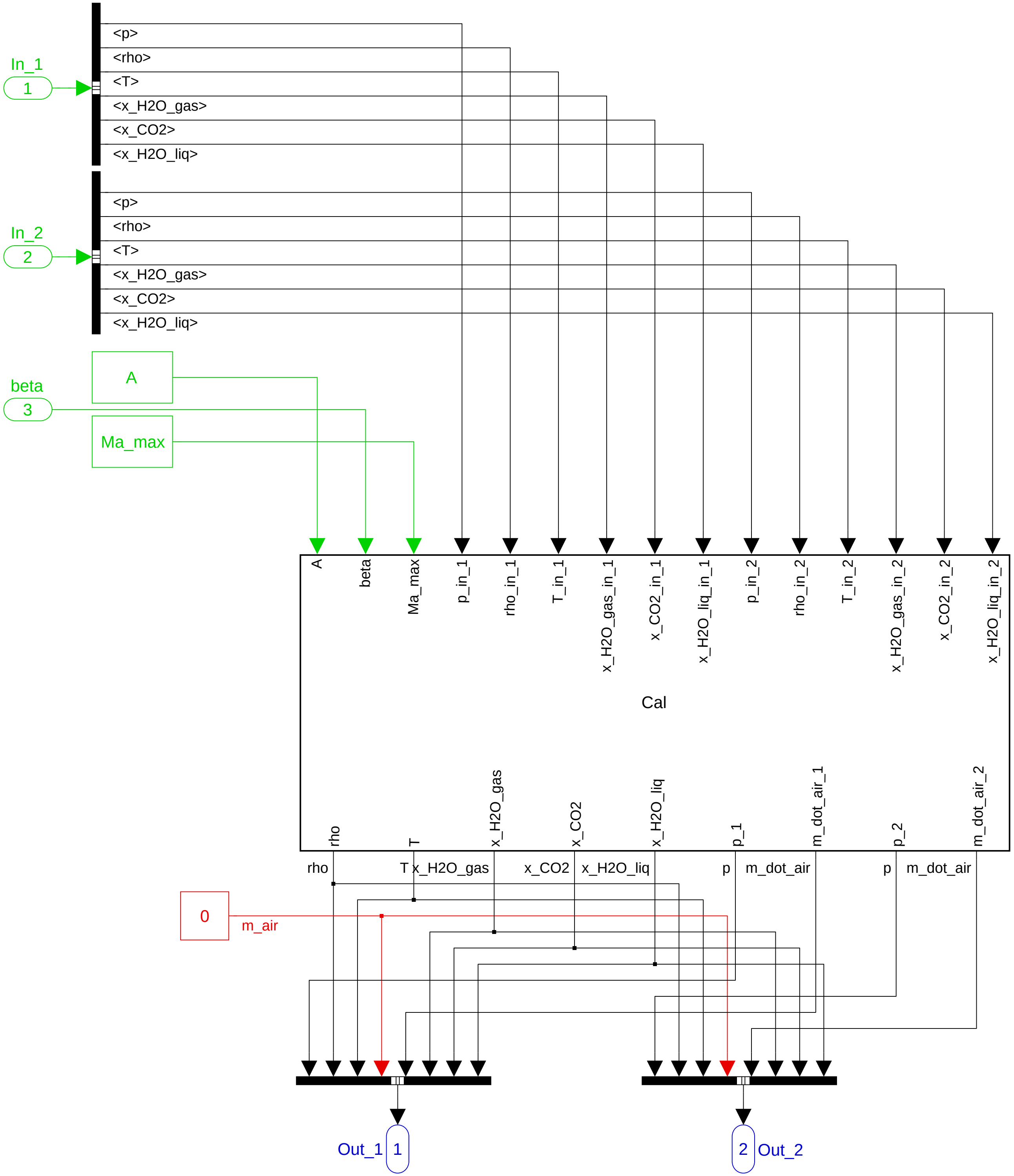
BlockType	Count	Block Names
BusSelector	2	Bus Selector3, Bus Selector5
BusCreator	2	Bus Creator3, Bus Creator4
Terminator	1	Terminator
SubSystem	1	Valve_Comp_
Stateflow (m)	1	Embedded MATLA Function1
S-Function	1	SFunction
Demux	1	Demux

Data and Functions

Tabelle 1.4. Model Functions

Function Name	Parent Blocks	Calling string
NaN	 Valve_Comp_ Valve_Comp_	 `NaN` `NaN`


Function Block Parameters: Valve_Comp_
Subsystem (mask)
Parameters
Surface [m²]
NaN
Maximum Mach Number (in general Ma_max=1)
NaN
OK
Cancel
Help
Apply



In_1
1
<p>
<rho>
<T>
<x_H2O_gas>
<x_CO2>
<x_H2O_liq>
In_2
2
<p>
<rho>
<T>
<x_H2O_gas>
<x_CO2>
<x_H2O_liq>
A
beta
3
Ma_max
A
beta
Ma_max
p_in_1
rho_in_1
T_in_1
x_H2O_gas_in_1
x_CO2_in_1
x_H2O_liq_in_1
p_in_2
rho_in_2
T_in_2
x_H2O_gas_in_2
x_CO2_in_2
x_H2O_liq_in_2
Cal
rho
T
x_H2O_gas
x_CO2
x_H2O_liq
p_1
m_dot_air_1
p_2
m_dot_air_2
rho
T
x_H2O_gas
x_CO2
x_H2O_liq
p
m_dot_air
p
m_dot_air
0
m_air
Out_1
1
2
Out_2

```
function [rho,T,x_H2O_gas,x_CO2,x_H2O_liq,p_1,m_dot_air_1,p_2,m_dot_air_2]=Cal(A,beta,↙
Ma_max,p_in_1,rho_in_1,T_in_1,x_H2O_gas_in_1,x_CO2_in_1,x_H2O_liq_in_1,p_in_2,rho_in_2,↙
T_in_2,x_H2O_gas_in_2,x_CO2_in_2,x_H2O_liq_in_2)

% *************************************************************************
% * Definition of a valve with compressible flow properties
% *
% * Number of inputs:                  2
% *
% * Parameter: Surface:                A
% *            opening factor:         OF
% *            Minor loss coefficient: zeta
% *
% *
% * Relevant input variables of Valve_Comp
% *
% * Pressure :                         p_in
% * Density :                          rho_in
% * Temperature :                      T_in
% * Content water vapor :              x_H2O_gas_in
% * Content CO2 :                      x_CO2_in
% * Content water :                    x_H2O_liq_in
% *
% *
% * Relevant output variables of Valve_Comp
% *
% * Temperature :                      T
% * Mass flow dry air :                m_dot_air
% * Content water vapor :              x_H2O_gas
% * Content CO2 :                      x_CO2
% * Content water :                    x_H2O_liq
% *
% ************************************************************************
% * Embedded Matlab Function Cal:
% *
% * Calculations:
% * 1. Calculation parameter.
% * 2. Definition of the specific gas constants.
% * 3. Calculation of the state variables, Mach number and mass flow.
% *
% *
% * Assumptions:
% * 3. Total values = Input variables higher pressure
% *
% *    Ma    = sqrt(2/(gamma-1))*sqrt(((p_total/p)^((gamma-1)/gamma))-1), gamma =↙
c_p/c_v
% *    rho   = rho_total*(1+((gamma_avg-1)/2)*Ma^2)^(-1/(gamma_avg-1));
% *    T     = T_total*(1+((gamma_avg-1)/2)*Ma^2)^(-1);
% *
% *    The mass flow is calculated with a compressible mass flow equation.
% *    This equation is applicable for the whole range of possible flow velocities.
% *
% *    m_dot = (A_eff*p_total/sqrt(T_total))*sqrt(gamma/R)*M*((1+((gamma-1)/2)*(M^2))^↙
(-(gamma+1)/(2*(gamma-1))));
% *
% *
```

```
% * Last modification : 15.03.2008
% * Author : Christian Müller(HAW)
% *
% ************************************************************************

% * 1. Calculation parameter
A_eff              = sin(pi*beta/180)*A;
% ************************************************************************

% * 2. Definition of the specific gas constants
R_air              = 287.058;
R_H2O_gas          = 461.523;
R_CO2              = 188.924;
c_p_air            = 1005;
c_p_H2O_gas        = 1870;
c_p_CO2            = 830;
% ************************************************************************

% * 3. Calculation of the state variables, Mach number and mass flow
rho                = 0;
T                  = 0;
x_H2O_gas          = 0;
x_CO2              = 0;
x_H2O_liq          = 0;
m_dot_air_1        = 0;
m_dot_air_2        = 0;

p_1                = p_in_2;
p_2                = p_in_1;

if p_in_1 >= p_in_2
   p_total         = p_in_1;
   p               = p_in_2;
   rho_total       = rho_in_1;
   T_total         = T_in_1;
   x_H2O_gas       = x_H2O_gas_in_1;
   x_CO2           = x_CO2_in_1;
   x_H2O_liq       = x_H2O_liq_in_1;

   R_avg           = (R_air+x_H2O_gas*R_H2O_gas+x_CO2*R_CO2)/(1+x_H2O_gas+x_CO2);
   c_p_avg         = (c_p_air+x_H2O_gas*c_p_H2O_gas+x_CO2*c_p_CO2)/(1+x_H2O_gas+x_CO2);
   c_v_avg         = c_p_avg-R_avg;
   gamma_avg       = c_p_avg/c_v_avg;

   Ma              = sqrt(2/(gamma_avg-1))*sqrt(((p_total/p)^((gamma_avg-1)/gamma_avg))↙
-1);

   if Ma > Ma_max
      Ma           = Ma_max;
   end

   rho             = rho_total*(1+((gamma_avg-1)/2)*Ma^2)^(-1/(gamma_avg-1));
   T               = T_total*(1+((gamma_avg-1)/2)*Ma^2)^(-1);

   m_dot           = (A_eff*p_total/sqrt(T_total))*sqrt(gamma_avg/R_avg)*Ma*((1+↙
((gamma_avg-1)/2)*(Ma^2))^(-(gamma_avg+1)/(2*(gamma_avg-1))));
```

```
   m_dot_air_1    = -m_dot/(1+x_H2O_gas+x_CO2);
   m_dot_air_2    =  m_dot/(1+x_H2O_gas+x_CO2);
else
   p_total        = p_in_2;
   p              = p_in_1;
   rho_total      = rho_in_2;
   T_total        = T_in_2;
   x_H2O_gas      = x_H2O_gas_in_2;
   x_CO2          = x_CO2_in_2;
   x_H2O_liq      = x_H2O_liq_in_2;

   R_avg          = (R_air+x_H2O_gas*R_H2O_gas+x_CO2*R_CO2)/(1+x_H2O_gas+x_CO2);
   c_p_avg        = (c_p_air+x_H2O_gas*c_p_H2O_gas+x_CO2*c_p_CO2)/(1+x_H2O_gas+x_CO2);
   c_v_avg        = c_p_avg-R_avg;
   gamma_avg      = c_p_avg/c_v_avg;

   Ma             = sqrt(2/(gamma_avg-1))*sqrt(((p_total/p)^((gamma_avg-1)/gamma_avg))↙
-1);

   if Ma > Ma_max
      Ma          = Ma_max;
   end

   rho            = rho*(1+((gamma_avg-1)/2)*Ma^2)^(-1/(gamma_avg-1));
   T              = T*(1+((gamma_avg-1)/2)*Ma^2)^(-1);

   m_dot          = (A_eff*p_total/sqrt(T_total))*sqrt(gamma_avg/R_avg)*Ma*((1+↙
((gamma_avg-1)/2)*(Ma^2))^(-(gamma_avg+1)/(2*(gamma_avg-1))));
   m_dot_air_1    =  m_dot/(1+x_H2O_gas+x_CO2);
   m_dot_air_2    = -m_dot/(1+x_H2O_gas+x_CO2);
end
% **************************************************************************
```