Libreria Modelica EEB

Documentazione

a cura di Alberto Leva
DEIB, Politecnico di Milano

Sommario

Questo documento descrive la libreria Modelica sviluppata nel contesto del progetto EEB. per la simulazione computazionalmente efficiente e a livello di dettaglio scalabile di edifici e dei sistemi in essi installati, al fine di migliorarne l'efficienza energetica. La libreria contiene circa 1600 modelli, la grande maggioranza di primo principio, suddivisi per categoria secondo un approccio adatto a un utente esperto del dominio ma non di metodi e strumenti per la simulazione dinamica. Particolare cura è stata posta sull'aspetto dell'efficienza, arrivando a raggiungere velocità di simulazione oltre 100 volte il tempo reale per modelli aggregati composti da alcune decine di migliaia di equazioni, anche su un comune laptop. Sono anche rappresentati i sistemi di controllo, in modo da poter provare facilmente soluzioni alternative anche sotto questo importante aspetto. La libreria è infine concepita e strutturata in modo da essere facilmente estendibile, così da potersi adattare rapidamente e agilmente alle numerose evoluzioni che si osservano nel settore.

Informazioni principali

I componenti di EEB sono stati descritti ovunque possibile con modelli di primo principio, così da permettere una validazione componente per componente e da consentire anche la simulazione di sistemi non ancora fisicamente esistenti.

I modelli sono stati scritti in modo tale da ridurre al minimo indispensabile l'informazione necessaria per parametrizzarli, a vantaggio dell'uso della libreria in situazioni in cui non molti dettagli sono disponibili (caso abbastanza frequente quando si affronta il già costruito).

Ove ciò ha matematicamente senso, per la stessa ragione è stata prevista anche la possibilità di parametrizzare i modelli in base a dati misurati (tipicamente in regime stazionario).

Il livello di dettaglio scalabile è stato ottenuto anzitutto rendendo parametriche tutte le discretizzazioni spaziali e anche sfruttando una tecnica non usuale (ovvero con un certo carattere d'innovatività) che consiste nel calcolare l'andamento di opportune variabili di stato sulla base non delle equazioni fisiche (come avviene ai livelli di dettaglio più fini) ma della descrizione dinamica dei sistemi di controllo presenti nell'oggetto modellato.

La libreria contiene anche modelli delle proprietà delle sostanza coinvolte, sia fluide (per esempio l'aria umida o i gas per le intercapedini delle finestre) che solide.

Sono rappresentati, come anticipato e sempre a dettaglio scalabile, anche i componenti usati per comporre i sistemi di controllo: le rappresentazioni a tempo continuo consentono di sfruttare l'efficienza dei solutori a passo variabile, mentre quelle digitali replicano in modo fedele il comportamento dell'effettivo codice di controllo.

Per ogni dettaglio si rimanda alla documentazione dei singoli package, subpackage e singoli componenti riportata nel seguito.

Documentazione
dei singoli componenti

EEB

Package Content

Name	Description
Icons	
Types	Package with type definitions
Settings	Package with models settings
Functions	Package with functions
Utilities	Package with utilities
BoundaryConditions	Package with boundary conditions
Interfaces	Package with interfaces
Media	Package with medium models
Components	
Appliances	
Controllers	
InternalTests	
CaseStudies	

EEB.Icons

Information

Extends from Modelica.Icons.IconsPackage (Icon for packages containing icons).

Package Content

Name	Description
AmbientConditions	
Envelope	
TestModel	
CaseStudyModel	

EEB.Icons.AmbientConditions

Modelica definition

```
model AmbientConditions
end AmbientConditions;
```

EEB.Icons.Envelope

Modelica definition

```
model Envelope
end Envelope;
```

EEB.Icons.TestModel

Modelica definition

```
model TestModel
equation

end TestModel;
```

EEB.Icons.CaseStudyModel

Modelica definition

```
model CaseStudyModel
equation

end CaseStudyModel;
```

EEB.Types

Package with type definitions

Information

This package contains type definitions.

Extends from Modelica.Icons.TypesPackage (Icon for packages containing type definitions).

Package Content

Name	Description
AmbCondVariabilty	Enumeration for variability of ambient conditions

Types and constants

```
type AmbCondVariabilty = enumeration(
  ACV_constant   "All conditions constant and set to avg values",
  ACV_variable   "Conditions vary according to provided ranges") "Enumeration for variability of ambient conditi
```

EEB.Settings

Package with models settings

Information

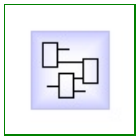
This package contains models settings.

Extends from Modelica.Icons.UtilitiesPackage (Icon for utility packages).

Package Content

Name	Description
DigitalControlSettings	Digital control settings

EEB.Settings.DigitalControlSettings



Digital control settings

Information

Define settings as far as the digital control.

Parameters

Type	Name	Default	Description
Time	Ts	0.1	[s]

Modelica definition

```
model DigitalControlSettings " Digital control settings"
  parameter Time Ts = 0.1;
end DigitalControlSettings;
```

EEB.Utilities

Package with utilities

Information

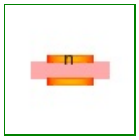
This package contains utility components such as for input data.

Extends from Modelica.Icons.UtilitiesPackage (Icon for utility packages).

Package Content

Name	Description
HPvector_VectOfHPs	HeatPortVector from vector of HeatPort

EEB.Utilities.HPvector_VectOfHPs



HeatPortVector from vector of HeatPort

Information

This model creates a heat port vector from n heat port.

It is used in WaterPipeExchanging_Nvols.

Parameters

Type	Name	Default	Description
Integer	n	2	No. of elements

Connectors

Type	Name	Description
HeatPortVec	vecHP	
HeatPort	vecOfHPs[n]	

Modelica definition

```
model HPvector_VectOfHPs "HeatPortVector from vector of HeatPort"
  parameter Integer n = 2 "No. of elements";
  Interfaces.Thermal.HeatPortVec vecHP(n = n);
  Interfaces.Thermal.HeatPort vecOfHPs[n];
equation
  for i in 1:n loop
    vecOfHPs[i].T = vecHP.T[i];
    vecOfHPs[i].Q_flow + vecHP.Q_flow[i] = 0;
  end for;
end HPvector_VectOfHPs;
```

EEB.Functions

Package with functions

Information

This package contains all the functions used in the library, which are not present in the MSL.

Extends from Modelica.Icons.FunctionsPackage (Icon for packages containing functions).

Package Content

Name	Description
sqrtReg	Symmetric square root approximation with finite derivative in zero
sqrtReg_der	Derivative of sqrtReg
set_layers_thicknesses	Set the thickness in each layer
set_layers_conductivities	Set the conductivity in each layer
range_conv_linear	

EEB.Functions.sqrtReg

Symmetric square root approximation with finite derivative in zero

Information

This function approximates square root, such that the derivative is finite in zero.

Extends from Modelica.Icons.Function (Icon for functions).

Inputs

Type	Name	Default	Description
Real	x		
Real	delta	0.01	Range of significant deviation from sqrt(x)

Outputs

Type	Name	Description
Real	y	

Modelica definition

```
function sqrtReg "Symmetric square root approximation with finite derivative in zero"
  extends Modelica.Icons.Function;
  input Real x;
  input Real delta = 0.01 "Range of significant deviation from sqrt(x)";
  output Real y;
algorithm
  y := x / sqrt(sqrt(x * x + delta * delta));
end sqrtReg;
```

EEB.Functions.sqrtReg_der

Derivative of sqrtReg

Information

Extends from Modelica.Icons.Function (Icon for functions).

Inputs

Type	Name	Default	Description
Real	x		
Real	delta	0.01	Range of significant deviation from sqrt(x)
Real	dx		Derivative of x

Outputs

Type	Name	Description
Real	dy	

Modelica definition

```
function sqrtReg_der "Derivative of sqrtReg"
  extends Modelica.Icons.Function;
  input Real x;
  input Real delta = 0.01 "Range of significant deviation from sqrt(x)";
  input Real dx "Derivative of x";
  output Real dy;
algorithm
  dy := dx * 0.5 * (x * x + 2 * delta * delta) / (x * x + delta * delta) ^ 1.25;
end sqrtReg_der;
```

EEB.Functions.set_layers_thicknesses

Set the thickness in each layer

Information

This function that sets the thickness in each layer component, given as input the number of layers and each thickness.

It is used in SolidMultilayer_NonHomogeneous.

Extends from Modelica.Icons.Function (Icon for functions).

Inputs

Type	Name	Default	Description
Integer	n		number of layers
Real	s[n]		layer thicknesses

Outputs

Type	Name	Description
Real	d[n + 1]	

Modelica definition

```
function set_layers_thicknesses "Set the thickness in each layer"
  extends Modelica.Icons.Function;
  input Integer n "number of layers";
  input Real s[n] "layer thicknesses";
  output Real d[n + 1];
algorithm
  d[1] := s[1] / 2;
  d[n + 1] := s[n] / 2;
  for i in 2:n loop
    d[i] := s[i - 1] / 2 + s[i] / 2;
  end for;

end set_layers_thicknesses;
```

EEB.Functions.set_layers_conductivities

Set the conductivity in each layer

Information

This function that sets the conductivities in each layer component, given as input the number of layers , each thickness and thermal conductivities.

It is used in SolidMultilayer_NonHomogeneous.

Extends from Modelica.Icons.Function (Icon for functions).

Inputs

Type	Name	Default	Description
Integer	n		
Real	s[n]		
Real	lambda[n]		

Outputs

Type	Name	Description
Real	L[n + 1]	

Modelica definition

```
function set_layers_conductivities "Set the conductivity in each layer"
  extends Modelica.Icons.Function;
  input Integer n;
  input Real s[n];
  input Real lambda[n];
  output Real L[n + 1];
algorithm
  L[1] := lambda[1];
  L[n + 1] := lambda[n];
  for i in 2:n loop
    L[i] := lambda[i - 1] * lambda[i] / (s[i - 1] * lambda[i] + s[i] * lambda[i - 1]) * (s[i - 1] + s[i]);
  end for;

end set_layers_conductivities;
```

EEB.Functions.range_conv_linear

Information

Extends from Modelica.Icons.Function (Icon for functions).

Inputs

Type	Name	Default	Description
Real	u		
Real	umin		
Real	umax		
Real	ymin		
Real	ymax		

Outputs

Type	Name	Description
Real	y	

Modelica definition

```
function range_conv_linear
  extends Modelica.Icons.Function;
  input Real u;
  input Real umin;
  input Real umax;
  input Real ymin;
  input Real ymax;
  output Real y;
algorithm
  // solve([a*umin+b=ymin,a*umax+b=ymax],[a,b]);
  // a=(ymin-ymax)/(umin-umax);
  // b=(umin*ymax-umax*ymin)/(umin-umax);
  y := max(ymin, min(ymax, (ymin - ymax) / (umin - umax) * u + (umin * ymax - umax * ymin) / (umin - umax)));
end range_conv_linear;
```

EEB.Interfaces

Package with interfaces

Information

This package provides interfaces for models, in particular:

- Air interfaces: contains moist air flange;

- Water interfaces: contains subcooled water flange;

- Electrical interfaces: contains pashor pin;

- Thermal interfaces: contains thermal prot.

Extends from Modelica.Icons.InterfacesPackage (Icon for packages containing interfaces).

Package Content

Name	Description
Air	Package with moist air interfaces
Electrical	Package with electrical ports
Thermal	Package with thermal ports
Water	Package with water interfaces

EEB.Interfaces.Air

Package with moist air interfaces

Information

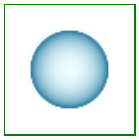
This package provides interfaces for moist air.

Extends from Modelica.Icons.InterfacesPackage (Icon for packages containing interfaces).

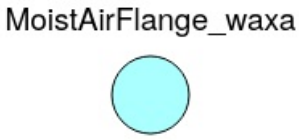
Package Content

Name	Description
MoistAirFlange_waxa	Interface for moist air (w,AU)
MoistAirFlange_wawvQd_waPart	Interface for moist air (wa,wv) - wa part
MoistAirFlange_wawvQd_wvPart	Interface for moist air (wa,wv) - wv part
MoistAirFlange_wawvQd	Interface for moist air (wa,wv)
PartialTwoPort_waxa	Partial two port waxa
sensor_Tphi	

EEB.Interfaces.Air.MoistAirFlange_waxa



Interface for moist air (w,AU)



Information

This connector is composed of:

- a potential variable "pa" which is the pressure [Pa];

- a flow variable "wa" which is the dry air mass flowrate [kg/s];

- two stream variables "ha" and "xa" which are respectively specifc enthalpy [J/kg] and absolute humidity [Kg vap/Kg dry air].

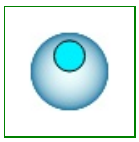
Contents

Type	Name	Description
AbsolutePressure	pa	Pressure [Pa]
flow MassFlowRate	wa	Dry air mass flowrate [kg/s]
stream SpecificEnthalpy	ha	Specifc enthalpy [J/kg]
stream MassFraction	xa	Absolute humidity [Kg vap/Kg dry air] [1]

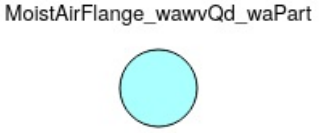
Modelica definition

```
connector MoistAirFlange_waxa "Interface for moist air (w,AU)"
  AbsolutePressure pa "Pressure";
  flow MassFlowRate wa "Dry air mass flowrate";
  stream SpecificEnthalpy ha "Specifc enthalpy";
  stream MassFraction xa "Absolute humidity [Kg vap/Kg dry air]";
end MoistAirFlange_waxa;
```

EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart



Interface for moist air (wa,wv) - wa part



Information

This connector is composed of:

- a potential variable "pa" which is the total pressure [Pa];

- a flow variable "wa" which is the dry air mass flowrate [kg/s];

- a stream variable "ha" which is the dry air enthalpy [J/kg].

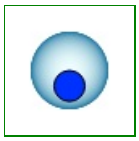
Contents

Type	Name	Description
AbsolutePressure	pa	Total pressure [Pa]
flow MassFlowRate	wa	Dry air mass flowrate [kg/s]
stream SpecificEnthalpy	ha	Dry air enthalpy [J/kg]

Modelica definition

```
connector MoistAirFlange_wawvQd_waPart "Interface for moist air (wa,wv) - wa part"
  AbsolutePressure pa "Total pressure";
  flow MassFlowRate wa "Dry air mass flowrate";
  stream SpecificEnthalpy ha "Dry air enthalpy";
end MoistAirFlange_wawvQd_waPart;
```

EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart



Interface for moist air (wa,wv) - wv part

Information

This connector is composed of:

- a potential variable "pv" which is the vapour pressure [Pa];

- a flow variable "wv" which is the vapour air mass flowrate [kg/s];

- two stream variables "hv" which is vapour specifc enthalpy [J/kg].

Contents

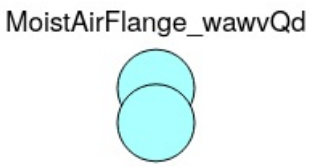
Type	Name	Description
AbsolutePressure	pv	Vapour pressure [Pa]
flow MassFlowRate	wv	Vapour mass flowrate [kg/s]
stream SpecificEnthalpy	hv	Vapour enthalpy [J/kg]

Modelica definition

```
connector MoistAirFlange_wawvQd_wvPart "Interface for moist air (wa,wv) - wv part"
  AbsolutePressure pv "Vapour pressure";
  flow MassFlowRate wv "Vapour mass flowrate";
  stream SpecificEnthalpy hv "Vapour enthalpy";
end MoistAirFlange_wawvQd_wvPart;
```

EEB.Interfaces.Air.MoistAirFlange_wawvQd

Interface for moist air (wa,wv)



Information

This is an hierichal connectors, composed of:

- dry air connector;

- vapour connector;

- an heat port.

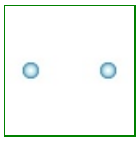
Contents

Type	Name	Description
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
connector MoistAirFlange_wawvQd "Interface for moist air (wa,wv)"
  MoistAirFlange_wawvQd_waPart dryair;
  MoistAirFlange_wawvQd_wvPart vapour;
  Interfaces.Thermal.HeatPort diffuse;
end MoistAirFlange_wawvQd;
```

EEB.Interfaces.Air.PartialTwoPort_waxa



Partial two port waxa

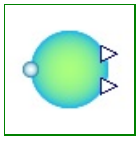
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	

Modelica definition

```
partial model PartialTwoPort_waxa "Partial two port waxa"
  AbsolutePressure pa1 "Pressure";
  AbsolutePressure pa2 "Pressure";
  MassFlowRate wa1 "Dry air mass flowrate";
  MassFlowRate wa2 "Dry air mass flowrate";
  SpecificEnthalpy haout1 "Specifc enthalpy";
  SpecificEnthalpy haout2 "Specifc enthalpy";
  MassFraction xaout1 "Absolute humidity [Kg vap/Kg dry air]";
  MassFraction xaout2 "Absolute humidity [Kg vap/Kg dry air]";
  EEB.Interfaces.Air.MoistAirFlange_waxa air_flange1;
  EEB.Interfaces.Air.MoistAirFlange_waxa air_flange2;
equation
  pa1 = air_flange1.pa;
  wa1 = air_flange1.wa;
  haout1 = air_flange1.ha;
  xaout1 = air_flange1.xa;
  pa2 = air_flange2.pa;
  wa2 = air_flange2.wa;
  haout2 = air_flange2.ha;
  xaout2 = air_flange2.xa;
end PartialTwoPort_waxa;
```

EEB.Interfaces.Air.sensor_Tphi



phi

Connectors

Type	Name	Description

MoistAirFlange_waxa	airSense	
output RealOutput	T	
output RealOutput	phi	

Modelica definition

```
model sensor_Tphi
  EEB.Media.Substances.MoistAir air;
  Interfaces.Air.MoistAirFlange_waxa airSense;
  Modelica.Blocks.Interfaces.RealOutput T;
  Modelica.Blocks.Interfaces.RealOutput phi;
equation
  airSense.wa = 0;
  airSense.xa = inStream(airSense.xa);
  airSense.ha = inStream(airSense.ha);
  air.p = airSense.pa;
  air.X = inStream(airSense.xa);
  air.h = inStream(airSense.ha);
  T = air.T;
  phi = air.phi;
end sensor_Tphi;
```

EEB.Interfaces.Water

Package with water interfaces

Information

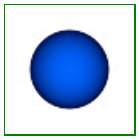
This package provides water interface.

Extends from Modelica.Icons.InterfacesPackage (Icon for packages containing interfaces).

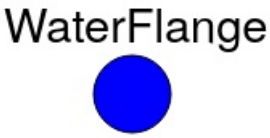
Package Content

Name	Description
WaterFlange	Interface for water
PartialTwoPort_water	Partial component with two ports

EEB.Interfaces.Water.WaterFlange



Interface for water



Information

This connector is composed of:

- a potential variable "p" which is the pressure [Pa];

- a flow variable "w" which is the water mass flowrate [kg/s];

- two stream variables "h" which is specifc enthalpy [J/kg].

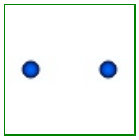
Contents

Type	Name	Description
AbsolutePressure	p	Pressure [Pa]
flow MassFlowRate	w	Mass flowrate [kg/s]
stream SpecificEnthalpy	h	Specifc enthalpy [J/kg]

Modelica definition

```
connector WaterFlange "Interface for water"
  AbsolutePressure p "Pressure";
  flow MassFlowRate w "Mass flowrate";
  stream SpecificEnthalpy h "Specifc enthalpy";
end WaterFlange;
```

EEB.Interfaces.Water.PartialTwoPort_water



Partial component with two ports

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	

Modelica definition

```
partial model PartialTwoPort_water "Partial component with two ports"
  AbsolutePressure p1 "Pressure at falnge1";
  AbsolutePressure p2 "Pressure at falnge2";
```

```
  MassFlowRate w1 "Mass flowrate at falnge1";
  MassFlowRate w2 "Mass flowrate at falnge2";
  SpecificEnthalpy hout1 "Specifc enthalpy at falnge1";
  SpecificEnthalpy hout2 "Specifc enthalpy at falnge2";
  EEB.Interfaces.Water.WaterFlange water_flange2;
  EEB.Interfaces.Water.WaterFlange water_flange1;
equation
  p1 = water_flange1.p;
  w1 = water_flange1.w;
  hout1 = water_flange1.h;
  p2 = water_flange2.p;
  w2 = water_flange2.w;
  hout2 = water_flange2.h;
end PartialTwoPort_water;
```

EEB.Interfaces.Thermal

Package with thermal ports

Information

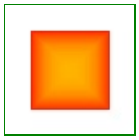
This package provides thermal ports.

Extends from Modelica.Icons.InterfacesPackage (Icon for packages containing interfaces).

Package Content

Name	Description
HeatPort	Heat port
HeatPortVec	Vector of heat port
HeatPortRad	

EEB.Interfaces.Thermal.HeatPort



Heat port

Information

MSL thermal port

Extends from Modelica.Thermal.HeatTransfer.Interfaces.HeatPort (Thermal port for 1-dim. heat transfer).

Contents

Type	Name	Description
Temperature	T	Port temperature [K]
flow HeatFlowRate	Q_flow	Heat flow rate (positive if flowing from outside into the component) [W]

Modelica definition

```
connector HeatPort "Heat port"
  extends Modelica.Thermal.HeatTransfer.Interfaces.HeatPort;
end HeatPort;
```

EEB.Interfaces.Thermal.HeatPortVec

Vector of heat port

Information

Vector of n heat port

Parameters

Type	Name	Default	Description
Integer	n	2	No. of elements

Contents

Type	Name	Description
Integer	n	No. of elements
Temperature	T[n]	temperature [K]
flow HeatFlowRate	Q_flow[n]	heat rate [W]

Modelica definition

```
connector HeatPortVec "Vector of heat port"
  parameter Integer n(min = 1) = 2 "No. of elements";
  Temperature T[n] "temperature";
  flow HeatFlowRate Q_flow[n] "heat rate";
end HeatPortVec;
```

EEB.Interfaces.Thermal.HeatPortRad

Information

Extends from Modelica.Thermal.HeatTransfer.Interfaces.HeatPort (Thermal port for 1-dim. heat transfer).

Contents

Type	Name	Description
Temperature	T	Port temperature [K]
flow HeatFlowRate	Q_flow	Heat flow rate (positive if flowing from outside into the component) [W]

Modelica definition

```
connector HeatPortRad
  extends Modelica.Thermal.HeatTransfer.Interfaces.HeatPort;
end HeatPortRad;
```

EEB.Interfaces.Electrical

Package with electrical ports

Information

This package provides electrical pots using phasor pins.

Extends from Modelica.Icons.InterfacesPackage (Icon for packages containing interfaces).

Package Content

Name	Description
PhasorPin	Phasor pin
PositivePhasorPin	Positive phasor pin
NegativePhasorPin	Negative phasor pin
PosNegPhasorPins	Positive-negative phasor pins
PosNegPins	Positive-negative electircal pins
PhasorOnePort	Component with two electrical pins p and n and current i from p to n
PhasorTwoPin	Phasor two pins

EEB.Interfaces.Electrical.PhasorPin

Phasor pin

Information

This connector is composed of:

- two potential variable "vre" and "vim" which are respectively real and immaginary part of the voltage phasor [V];

- two flow variables "ire" and "iim" which are respectively real and immaginary part of the current phasor [A].

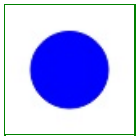
Contents

Type	Name	Description
Voltage	vre	[V]
Voltage	vim	[V]
flow Current	ire	[A]
flow Current	iim	[A]

Modelica definition

```
connector PhasorPin "Phasor pin"
  Voltage vre, vim;
  flow Current ire, iim;
end PhasorPin;
```

EEB.Interfaces.Electrical.PositivePhasorPin



Positive phasor pin

Information

Extends from PhasorPin (Phasor pin).

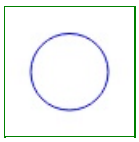
Contents

Type	Name	Description
Voltage	vre	[V]
Voltage	vim	[V]
flow Current	ire	[A]
flow Current	iim	[A]

Modelica definition

```
connector PositivePhasorPin "Positive phasor pin"
  extends PhasorPin;
end PositivePhasorPin;
```

EEB.Interfaces.Electrical.NegativePhasorPin



Negative phasor pin

Information

Extends from PhasorPin (Phasor pin).

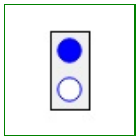
Contents

Type	Name	Description
Voltage	vre	[V]
Voltage	vim	[V]
flow Current	ire	[A]
flow Current	iim	[A]

Modelica definition

```
connector NegativePhasorPin "Negative phasor pin"
  extends PhasorPin;
end NegativePhasorPin;
```

EEB.Interfaces.Electrical.PosNegPhasorPins



Positive-negative phasor pins

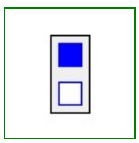
Contents

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
connector PosNegPhasorPins "Positive-negative phasor pins"
  PositivePhasorPin p;
  NegativePhasorPin n;
end PosNegPhasorPins;
```

EEB.Interfaces.Electrical.PosNegPins



Positive-negative electircal pins

Contents

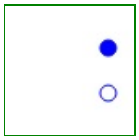
Type	Name	Description
PositivePin	p	
NegativePin	n	

Modelica definition

```
connector PosNegPins "Positive-negative electircal pins"
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
```

```
end PosNegPins;
```

EEB.Interfaces.Electrical.PhasorOnePort



Component with two electrical pins p and n and current i from p to n

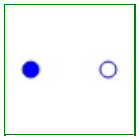
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
partial model PhasorOnePort "Component with two electrical pins p and n and current i from p to n"
  PositivePhasorPin p;
  EEB.Interfaces.Electrical.NegativePhasorPin n;
end PhasorOnePort;
```

EEB.Interfaces.Electrical.PhasorTwoPin



Phasor two pins

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
partial model PhasorTwoPin "Phasor two pins"

  EEB.Interfaces.Electrical.PositivePhasorPin p;
  EEB.Interfaces.Electrical.NegativePhasorPin n;
  Voltage Vre, Vim;
  Current Ire, Iim;
equation
  p.ire + n.ire = 0;
  p.iim + n.iim = 0;
  Vre = p.vre - n.vre;
  Vim = p.vim - n.vim;
  Ire = p.ire;
  Iim = p.iim;
end PhasorTwoPin;
```

EEB.BoundaryConditions

Package with boundary conditions

Information

This package contains boundary conditions components such as for ambient conditions.

Package Content

Name	Description
AmbientConditions	

EEB.BoundaryConditions.AmbientConditions

Information

Extends from EEB.Icons.AmbientConditions.

Parameters

Type	Name	Default	Description
AmbCondVariabilty	acv	Types.AmbCondVariabilty.ACV_...	Variability of ambient conditions
Integer	doy_start	1	Day-of-year (1-365) of simulation start
Integer	hod_start	0	hour-of-day (0-23) of simulation start
Integer	moh_start	0	minute-of-hour (0-59) of simulation start
Ambient air and wind			
Pressure	pa_avg	101325	Pressure (fixed) [Pa]
MassFraction	Xa_avg	0.001	Absolute humidity (fixed) [1]
Temperature	Ta_avg	273.15 - 5	Average temperature [K]
TemperatureDifference	Ta_Dex	4	Daily T excursion around day average [K]
TemperatureDifference	Ta_Yex	4	Yearly excursion of day T average around Ta_avg [K]
Temperature	Tref_DD	273.15 + 20	Ref T for degree day computation [K]
Velocity	wv	0	Wind velocity (fixed) [m/s]
Angle	wdir	0	Wind direction (0?=N, 90?=E) [rad]
Sun, sky, ground			
HeatFlux	Phimax	1000	Max radiative flux [W/m2]
Real	azi0	90	Constant or yearly avg azimuth (0?=N, 90?=E)
Real	azi_Yex	20	Yearly excursion of azimuth
Real	zen0	30	Constant or avg daily max zenith (heigth on horizon)
Real	zen_Yex	10	Yearly excursion of daily max zenith
Integer	esrh	5	earliest sunrise hour (0-23) for varying zen/azi
Integer	esrm	10	earliest sunrise min (0-59) for varying zen/azi
Integer	lsrh	8	latest sunrise hour (0-23) for varying zen/azi
Integer	lsrm	30	latest sunrise min (0-59) for varying zen/azi
Integer	essh	17	earliest sunset hour (0-23) for varying zen/azi
Integer	essm	30	earliest sunset min (0-59) for varying zen/azi
Integer	lssh	22	latest sunset hour (0-23) for varying zen/azi
Integer	lssm	15	latest sunset min (0-59) for varying zen/azi
Temperature	Tsky	273.15 + 10	Sky temperature (fixed) [K]
Temperature	Tgnd	273.15 + 12	Ground temperature (fixed) [K]

Modelica definition

```
model AmbientConditions
  extends EEB.Icons.AmbientConditions;
  // Constants
  constant Integer SPM = 60 "seconds per minute";
  constant Integer MPH = 60 "minutes per hour";
  constant Integer SPH = SPM * MPH "seconds per minute";
```

```
  constant Integer HPD = 24 "hours per day";
  constant Integer DPY = 365 "days per year";
  constant Integer SPD = SPH * HPD "seconds per day";
  constant Integer SPY = SPD * DPY "seconds per year";
  constant Integer JUN21 = 172 "June 21 is day 172 of year";
  constant Integer DEC21 = 355 "December 21 is day 355 of year";
  // PARAMETERS ---------------------------------------------------------
  // *** General
  parameter Types.AmbCondVariabilty acv = Types.AmbCondVariabilty.ACV_constant "Variability of ambient conditions";
  parameter Integer doy_start(min = 1, max = DPY) = 1 "Day-of-year (1-365) of simulation start";
  parameter Integer hod_start(min = 0, max = HPD - 1) = 0 "hour-of-day (0-23) of simulation start";
  parameter Integer moh_start(min = 0, max = MPH - 1) = 0 "minute-of-hour (0-59) of simulation start";
  // *** Ambient air and wind properties
  parameter Pressure pa_avg = 101325 "Pressure (fixed)";
  parameter MassFraction Xa_avg = 0.001 "Absolute humidity (fixed)";
  parameter Temperature Ta_avg = 273.15 - 5 "Average temperature";
  parameter TemperatureDifference Ta_Dex = 4 "Daily T excursion around day average";
  parameter TemperatureDifference Ta_Yex = 4 "Yearly excursion of day T average around Ta_avg";
  parameter Temperature Tref_DD = 273.15 + 20 "Ref T for degree day computation";
  parameter Velocity wv = 0 "Wind velocity (fixed)";
  parameter Angle wdir(min = 0, max = 360) = 0 "Wind direction (0?=N, 90?=E)";
  // *** Sun, sky and ground properties
  parameter HeatFlux Phimax = 1000 "Max radiative flux";
  parameter Real azi0(min = 0, max = 360) = 90 "Constant or yearly avg azimuth (0?=N, 90?=E)";
  parameter Real azi_Yex = 20 "Yearly excursion of azimuth";
  parameter Real zen0(min = 0, max = 180) = 30 "Constant or avg daily max zenith (heigth on horizon)";
  parameter Real zen_Yex(min = 0, max = 180) = 10 "Yearly excursion of daily max zenith";
  parameter Integer esrh(min = 0, max = 23) = 5 "earliest sunrise hour (0-23) for varying zen/azi";
  parameter Integer esrm(min = 0, max = 59) = 10 "earliest sunrise min (0-59) for varying zen/azi";
  parameter Integer lsrh(min = 0, max = 23) = 8 "latest sunrise hour (0-23) for varying zen/azi";
  parameter Integer lsrm(min = 0, max = 59) = 30 "latest sunrise min (0-59) for varying zen/azi";
  parameter Integer essh(min = 0, max = 23) = 17 "earliest sunset hour (0-23) for varying zen/azi";
  parameter Integer essm(min = 0, max = 59) = 30 "earliest sunset min (0-59) for varying zen/azi";
  parameter Integer lssh(min = 0, max = 23) = 22 "latest sunset hour (0-23) for varying zen/azi";
  parameter Integer lssm(min = 0, max = 59) = 15 "latest sunset min (0-59) for varying zen/azi";
  parameter Temperature Tsky = 273.15 + 10 "Sky temperature (fixed)";
  parameter Temperature Tgnd = 273.15 + 12 "Ground temperature (fixed)";
  // Time-related quantities
  Real hours "hours from sim start";
  Real days "days from sim start";
  Real hod "hour of day (0-24, real-valued, reset every day)";
  Time sod "second of day (0-86400, real-valued, reset every day)";
  Real doy "day of year (0-365, real-valued, reset every year)";
  Real soy "second of year (0-31536000, real-valued, reset every year)";
  Real yusw "yearly unit sine wave, max on JUN21 (thus approx min on DEC21)";
  // Ambient air properties
  Pressure pamb "Ambient air p";
  Temperature Tamb "Ambient air T";
  // Sun-related quantities
  Real azimuth(min = 0, max = 360) "Current ";
  Real zenith(min = 0, max = 180) "Current ";
  Real solarRad "Current ";
  Real DD "Degree day";
protected
  parameter Time soy_start = (doy_start - 1) * SPD + hod_start * SPH + moh_start * SPM;
  parameter Real trAvg = (esrh * MPH + esrm + lsrh * MPH + lsrm) / MPH / 2 "mean sunrise time in hours";
  parameter Real tsAvg = (essh * MPH + essm + lssh * MPH + lssm) / MPH / 2 "mean sunset time in hours";
  parameter Real Dtr = (lsrh * MPH + lsrm - esrh * MPH - esrm) / MPH / 2 "sunrise excursion in hours";
  parameter Real Dts = (lssh * MPH + lssm - essh * MPH - essm) / MPH / 2 "sunset excursion in hours";
  Real zenithMax, tr, ts, az, bz, cz, ar, br, cr;
equation
// Degree day
  der(DD) = if Tref_DD > Tamb then (Tref_DD - Tamb) / SPD else 0;
// Time-related quantities
  hours = time / SPH;
  days = hours / HPD;
  soy = time + soy_start - SPY * floor((time + soy_start) / SPY);
  sod = soy - SPD * floor(soy / SPD);
  hod = sod / SPH;
  doy = (soy - SPY * floor(soy / SPY)) / SPD;
  yusw = sin((soy - JUN21 * SPD) / SPY * 2 * pi + pi / 2);
// sunrise and sunset time (in hours)
  tr = trAvg - Dtr * yusw;
//sin(pi*(1/(3600*24*182)*time));
  ts = tsAvg + Dts * yusw;
//sin(pi*(1/(3600*24*182)*time));
// Ambient air properties
  pamb = pa_avg;
  zenithMax = zen0 + zen_Yex * yusw;
// parabola coeffs for zenith angle
```

```
  az = -4 * zenithMax / (ts - tr) ^ 2;
  bz = 4 * zenithMax * (ts + tr) / (ts - tr) ^ 2;
  cz = -4 * zenithMax * ts * tr / (ts - tr) ^ 2;
// parabola coeffs for solar radiation
  ar = -4 * Phimax / (ts - tr) ^ 2;
  br = 4 * Phimax * (ts + tr) / (ts - tr) ^ 2;
  cr = -4 * Phimax * ts * tr / (ts - tr) ^ 2;
  if acv == Types.AmbCondVariabilty.ACV_constant then
    Tamb = Ta_avg;
    zenith = zen0;
    azimuth = azi0;
    solarRad = Phimax;
  else
    Tamb = Ta_avg + Ta_Yex * yusw + Ta_Dex * sin(pi * (1 / (24 * 3600)) * time - 2 * 3.14 / 3);
    zenith = smooth(0, max(0, az * hod ^ 2 + bz * hod + cz));
    azimuth = azi0 + azi_Yex * yusw;
    solarRad = smooth(0, max(0, ar * hod ^ 2 + br * hod + cr));
  end if;
end AmbientConditions;
```

EEB.Media

Package with medium models

Information

This package contain media models for moist air and water. These models are simpler than the ones present in Modelica.Media, therefore often with a more efficient simulation. The leak of accuracy is not so relevant thanks to the range temperature in building HVAC applications.

There is also a Materials package containing records with properties of many materials.

Extends from Modelica.Icons.MaterialPropertiesPackage (Icon for package containing property classes).

Package Content

Name	Description
Constants	Pakage with constants
Substances	Package with substances
Materials	Package with materials

EEB.Media.Constants

Pakage with constants

Information

Extends from Modelica.Icons.UtilitiesPackage (Icon for utility packages).

Package Content

Name	Description
cp_a=1005	Specific heat capacity of dry air
cp_v=1870	Specific heat capacity of water vapour
cp_l=4186	Specific heat capacity of liquid water
h_v_3pt=2501*1e3	Specific enthalpy of water vapour at triple point (0?C) [J/Kg]
Ra=286.9	Gas constant of dry air
Rv=461.5	Gas constant of water vapour in moist air
d_l=1000	density of subcooled water
MMa=24	molar mass of dry air
MMw=14	molar mass of water
patm=101325	Atmospheric pressure

Types and constants

```
final constant SpecificHeatCapacity cp_a = 1005 "Specific heat capacity of dry air";
final constant SpecificHeatCapacity cp_v = 1870 "Specific heat capacity of water vapour";
final constant SpecificHeatCapacity cp_l = 4186 "Specific heat capacity of liquid water";
final constant SpecificEnthalpy h_v_3pt = 2501 * 1e3 "Specific enthalpy of water vapour at triple point (0?C) [J/K
final constant SpecificHeatCapacity Ra = 286.9 "Gas constant of dry air";
final constant SpecificHeatCapacity Rv = 461.5 "Gas constant of water vapour in moist air";
final constant Density d_l = 1000 "density of subcooled water";
final constant MolarMass MMa = 24 "molar mass of dry air";
final constant MolarMass MMw = 14 "molar mass of water";
final constant Pressure patm = 101325 "Atmospheric pressure";
```

EEB.Media.Substances

Package with substances

Information

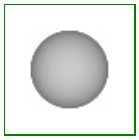
This package provides the model of moist air and subcooled water.

Extends from Modelica.Icons.MaterialPropertiesPackage (Icon for package containing property classes).

Package Content

Name	Description
MoistAir	Moist air
SubcooledWater	Subcooled water

EEB.Media.Substances.MoistAir



Moist air

Information

This is the moist air model that is based on Mollier Diagram.

In the HVAC systems, the air is generally considered as a binary mixture of dry air and water vapor, i.e. moist air. State properties of moist air are calculated by using the Dalton's law on perfect gas mixtures.

In order to find out the thermodynamic state of the moist air, we use the Gibbs' phase rule

$$F = C - P + 2$$

where C is the number of components (in this case, vapor and dry air), P the number of phases in thermodynamic equilibrium (in this case the only phase is the gas one), and F the number of degrees of freedom.

Thus, dealing with moist air, C is two (vapor and dry air), P is one (gas phase only) and therefore F is equal to three. So we need to know the total pressure, the specific enthalpy h (using convetion that h=0 if T=0°C and no water vapour is present), and the mass fraction X, which is the vapor and dry air mass ratio.

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Parameters

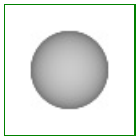
Type	Name	Default	Description
Boolean	is4cap	false	set true to suggest pTX as states when used in a capacity

Modelica definition

```
model MoistAir "Moist air"
  extends Modelica.Icons.MaterialProperty;
  import MAC = EEB.Media.Constants;
  parameter Boolean is4cap=false "set true to suggest pTX as states when used in a capacity";
  connector Input_p = input Pressure "pressure";
  connector Input_T = input Temperature "temperature";
  connector Input_X = input MassFraction "absolute humidity [kg vap/kg da]";
  Input_p p(
   stateSelect = if is4cap then StateSelect.prefer else StateSelect.default)
  "pressure";
  Input_T T(
   stateSelect = if is4cap then StateSelect.prefer else StateSelect.default)
  "temperature";
  Input_X X(
   stateSelect = if is4cap then StateSelect.prefer else StateSelect.default)
  "absolute humidity [kg vap/kg da]";
  //-------------------------------------------------------------------------------
  MassFraction x "vapour mass fraction [kg vap/kg tot]";
  Pressure pv "vapour partial pressure";
  Pressure pvs "vapour saturation pressure";
  Real phi "relative humidity";
  Temperature Twb "wet bulb temperature";
  MassFraction Xs "absolute humidity [kg vap/kg da] at saturation (phi=1)";
  SpecificEnthalpy ha "specific enthalpy of dry air [J/kg da]";
  SpecificEnthalpy hv "specific enthalpy of vapour [J/kg vap]";
```

```
  SpecificEnthalpy h "specific enthalpy of moist air [J/kg da]";
  SpecificEnthalpy hl "specific enthalpy of liquid water at T [J/kg liq]";
  Density d "density of moist air";
  SpecificHeatCapacity cp "moist air specific heat capacity [J/kg da K]";
  SpecificHeatCapacity R "equivalent gas constant of moist air";
equation
  x = X / (1 + X);
  pv = p * (X / 0.622) / (1 + X / 0.622);
  pvs = 610.78 * exp((T - 273.15) / (T - 273.15 + 238.3) * 17.2694);
  pv = pvs * phi;
  pv = 610.78 * exp((Twb - 273.15) / (Twb - 273.15 + 238.3) * 17.2694);
  // for Twb
  pvs = p * (Xs / 0.622) / (1 + Xs / 0.622);
  // for Xs
  ha = MAC.cp_a * (T - 273.15);
  hv = MAC.h_v_3pt + MAC.cp_v * (T - 273.15);
  h = ha + X * hv;
  hl = MAC.cp_l * (T - 273.15);
  d = (p - pv) / (MAC.Ra * T) + pv / (MAC.Rv * T);
  cp = MAC.cp_a + X * MAC.cp_v;
  R = MAC.Ra + MAC.Rv * X;
end MoistAir;
```

EEB.Media.Substances.SubcooledWater



Subcooled water

Information

This is the subcooled water model.

The state equations for the substances in liquid phase are generally very complex. Therefore it is preferred to study the properties of the liquids according to simplifying assumptions. The most important one is to assume an incompressible substance behavior which means that the volume does not change for every transformation. It occurs that the assumption of incompressible substance behavior is lawful, for all substances, at thermodynamic states not too much close to the critical state. Thus, we assume the subcooled water as an incompressible substance far from its critical state. These two assumptions lead to the following conclusions:

- the specific heat at constant pressure and volume have the same value cp=cv=c;

- the variation of the specific enthalpy is the same of the specific internal energy.

In fact, because we deal with incompressible substance behavior, the specific volume of the fluid is about 10−3m3/kg. Therefore, we can neglect the vdp term:

$$dh=cdT+vdp \approx cdT=du$$

- the specific heat, the specific enthalpy and the specific internal energy are functions of the same temperature only:

$$c = c\ (T);$$

$$h = h\ (T);$$

$$u = u\ (T).$$

We can finally assume that the specific heat is constant and equal to c = 4.186kJ/kgK, because its variation with the temperature is almost negligible.

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Modelica definition

```
model SubcooledWater "Subcooled water"
  extends Modelica.Icons.MaterialProperty;
  import MAC = EEB.Media.Constants;
  connector Input_p = input SI.Pressure "pressure";
  connector Input_T = input SI.Temperature "temperature";
  Input_p p "pressure";
  Input_T T "temperature";
  //------------------------------------------------------------------------
  SpecificEnthalpy h "specific enthalpy";
  Density d "density";
  SpecificHeatCapacity cp "specific heat capacity per [J/kg da K]";
equation
  h = MAC.cp_l * (T - 273.15);
  d = MAC.d_l;
  cp = MAC.cp_l;
```

```
end SubcooledWater;
```

EEB.Media.Substances.MoistAir.Input_p

pressure

Modelica definition

```
connector Input_p = input Pressure "pressure";
```

EEB.Media.Substances.MoistAir.Input_T

temperature

Modelica definition

```
connector Input_T = input Temperature "temperature";
```

EEB.Media.Substances.MoistAir.Input_X

absolute humidity [kg vap/kg da]

Modelica definition

```
connector Input_X = input MassFraction "absolute humidity [kg vap/kg da]";
```

EEB.Media.Substances.SubcooledWater.Input_p

pressure

Modelica definition

```
connector Input_p = input SI.Pressure "pressure";
```

EEB.Media.Substances.SubcooledWater.Input_T

temperature

Modelica definition

```
connector Input_T = input SI.Temperature "temperature";
```

EEB.Media.Materials

Package with materials

Information

This package provides records, containing the properties of the elements belonging to the following groups :

- wall materials;

- door materials;

- glasses;

- envelpe gases.

Extends from Modelica.Icons.MaterialPropertiesPackage (Icon for package containing property classes).

Package Content

Name	Description
WallMaterials	Package with wall materials
DoorMaterials	Package with door materials
Glasses	Package with glasses
EnvelopeGases	Package with envelope gases

EEB.Media.Materials.WallMaterials

Package with wall materials

Information

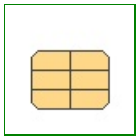
This package provides records for wall materials.

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Package Content

Name	Description
BaseWallMaterial	
Brick	
Concrete	
GenericInsulation	
Gypsum	
Plywood	

EEB.Media.Materials.WallMaterials.BaseWallMaterial



Information

Extends from Modelica.Icons.Record (Icon for records).

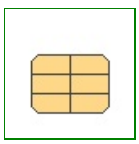
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda		Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp		Specific heat capacity [J/(kg.K)]
Density	ro		Mass density [kg/m3]

Modelica definition

```
record BaseWallMaterial
  extends Modelica.Icons.Record;
  parameter Modelica.SIunits.ThermalConductivity lambda "Thermal conductivity";
  parameter Modelica.SIunits.SpecificHeatCapacity cp "Specific heat capacity";
  parameter Modelica.SIunits.Density ro "Mass density";
  /*
      parameter Real R(unit="m2.K/W")
        "Thermal resistance of a unit area of material";

      parameter Integer nStaRef(min=0) = 3
        "Number of state variables in a reference material of 0.2 m concrete";
      parameter Integer nSta(min=1)=max(1, integer(ceil(nStaReal)))
        "Actual number of state variables in material"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Boolean steadyState= (c == 0 or d == 0)
        "Flag, if true, then material is computed using steady-state heat conduction"
        annotation(Evaluate=true);
      parameter Real piRef=331.4
        "Ratio x/sqrt(alpha) for reference material of 0.2 m concrete"
        annotation (Dialog(tab="Advanced"));
      parameter Real piMat=if steadyState then piRef else x*sqrt(c*d)/sqrt(k)
        "Ratio x/sqrt(alpha)"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Real nStaReal(min=0) = nStaRef*piMat/piRef
        "Number of states as a real number"
        annotation (Dialog(tab="Advanced"));
    */
end BaseWallMaterial;
```

EEB.Media.Materials.WallMaterials.Brick



Information

Extends from EEB.Media.Materials.WallMaterials.BaseWallMaterial.

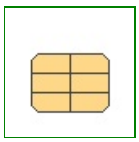
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.89	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	790	Specific heat capacity [J/(kg.K)]
Density	ro	1920	Mass density [kg/m3]

Modelica definition

```
record Brick
  extends EEB.Media.Materials.WallMaterials.BaseWallMaterial(lambda = 0.89, ro = 1920, cp = 790);
end Brick;
```

EEB.Media.Materials.WallMaterials.Concrete



Information

Extends from EEB.Media.Materials.WallMaterials.BaseWallMaterial.

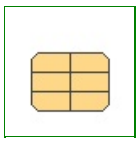
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	1.4	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	840	Specific heat capacity [J/(kg.K)]
Density	ro	2240	Mass density [kg/m3]

Modelica definition

```
record Concrete
  extends EEB.Media.Materials.WallMaterials.BaseWallMaterial(lambda = 1.4, ro = 2240, cp = 840);
end Concrete;
```

EEB.Media.Materials.WallMaterials.GenericInsulation



Information

Extends from EEB.Media.Materials.WallMaterials.BaseWallMaterial.

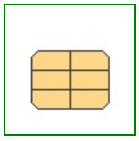
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.03	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	1200	Specific heat capacity [J/(kg.K)]
Density	ro	40	Mass density [kg/m3]

Modelica definition

```
record GenericInsulation
  extends EEB.Media.Materials.WallMaterials.BaseWallMaterial(lambda = 0.03, ro = 40, cp = 1200);
end GenericInsulation;
```

EEB.Media.Materials.WallMaterials.Gypsum



Information

Extends from EEB.Media.Materials.WallMaterials.BaseWallMaterial.

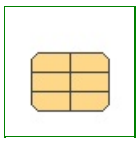
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.16	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	1090	Specific heat capacity [J/(kg.K)]
Density	ro	800	Mass density [kg/m3]

Modelica definition

```
record Gypsum
  extends EEB.Media.Materials.WallMaterials.BaseWallMaterial(lambda = 0.16, ro = 800, cp = 1090);
end Gypsum;
```

EEB.Media.Materials.WallMaterials.Plywood



Information

Extends from EEB.Media.Materials.WallMaterials.BaseWallMaterial.

Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.12	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	1280	Specific heat capacity [J/(kg.K)]
Density	ro	540	Mass density [kg/m3]

Modelica definition

```
record Plywood
  extends EEB.Media.Materials.WallMaterials.BaseWallMaterial(lambda = 0.12, ro = 540, cp = 1280);
end Plywood;
```

EEB.Media.Materials.DoorMaterials

Package with door materials

Information

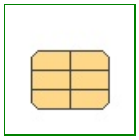
This package provides records door materials.

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Package Content

Name	Description
BaseDoorMaterial	
GenericWood	GenericWood

EEB.Media.Materials.DoorMaterials.BaseDoorMaterial



Information

Extends from Modelica.Icons.Record (Icon for records).

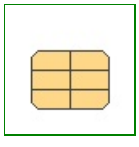
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda		Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp		Specific heat capacity [J/(kg.K)]
Density	rho		Mass density [kg/m3]

Modelica definition

```
record BaseDoorMaterial
  extends Modelica.Icons.Record;
  parameter Modelica.SIunits.ThermalConductivity lambda "Thermal conductivity";
  parameter Modelica.SIunits.SpecificHeatCapacity cp "Specific heat capacity";
  parameter Modelica.SIunits.Density rho "Mass density";
  /*
      parameter Real R(unit="m2.K/W")
        "Thermal resistance of a unit area of material";

      parameter Integer nStaRef(min=0) = 3
        "Number of state variables in a reference material of 0.2 m concrete";
      parameter Integer nSta(min=1)=max(1, integer(ceil(nStaReal)))
        "Actual number of state variables in material"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Boolean steadyState= (c == 0 or d == 0)
        "Flag, if true, then material is computed using steady-state heat conduction"
        annotation(Evaluate=true);
      parameter Real piRef=331.4
        "Ratio x/sqrt(alpha) for reference material of 0.2 m concrete"
        annotation (Dialog(tab="Advanced"));
      parameter Real piMat=if steadyState then piRef else x*sqrt(c*d)/sqrt(k)
        "Ratio x/sqrt(alpha)"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Real nStaReal(min=0) = nStaRef*piMat/piRef
        "Number of states as a real number"
        annotation (Dialog(tab="Advanced"));
    */
end BaseDoorMaterial;
```

EEB.Media.Materials.DoorMaterials.GenericWood



GenericWood

Information

Extends from BaseDoorMaterial.

Parameters

Type	Name	Default	Description

ThermalConductivity	lambda	0.15	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	2000	Specific heat capacity [J/(kg.K)]
Density	rho	540	Mass density [kg/m3]

Modelica definition

```
record GenericWood "GenericWood"
  extends BaseDoorMaterial(lambda = 0.15, cp = 2000, rho = 540);
end GenericWood;
```

EEB.Media.Materials.Glasses

Package with glasses

Information

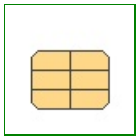
This package provides records for glasses.

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Package Content

Name	Description
BaseGlass	
Glass	Glass(lambda=2,rho=2200,cp=795)

EEB.Media.Materials.Glasses.BaseGlass



Information

Extends from Modelica.Icons.Record (Icon for records).

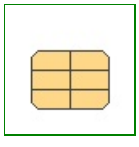
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda		Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp		Specific heat capacity [J/(kg.K)]
Density	ro		Mass density [kg/m3]

Modelica definition

```
record BaseGlass
  extends Modelica.Icons.Record;
  parameter Modelica.SIunits.ThermalConductivity lambda "Thermal conductivity";
  parameter Modelica.SIunits.SpecificHeatCapacity cp "Specific heat capacity";
  parameter Modelica.SIunits.Density ro "Mass density";
  /*
      parameter Real R(unit="m2.K/W")
        "Thermal resistance of a unit area of material";

      parameter Integer nStaRef(min=0) = 3
        "Number of state variables in a reference material of 0.2 m concrete";
      parameter Integer nSta(min=1)=max(1, integer(ceil(nStaReal)))
        "Actual number of state variables in material"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Boolean steadyState= (c == 0 or d == 0)
        "Flag, if true, then material is computed using steady-state heat conduction"
        annotation(Evaluate=true);
      parameter Real piRef=331.4
        "Ratio x/sqrt(alpha) for reference material of 0.2 m concrete"
        annotation (Dialog(tab="Advanced"));
      parameter Real piMat=if steadyState then piRef else x*sqrt(c*d)/sqrt(k)
        "Ratio x/sqrt(alpha)"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Real nStaReal(min=0) = nStaRef*piMat/piRef
        "Number of states as a real number"
        annotation (Dialog(tab="Advanced"));
    */
end BaseGlass;
```

EEB.Media.Materials.Glasses.Glass



Glass(lambda=2,rho=2200,cp=795)

Information

Extends from BaseGlass.

Parameters

Type	Name	Default	Description

ThermalConductivity	lambda	2	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	795	Specific heat capacity [J/(kg.K)]
Density	ro	2200	Mass density [kg/m3]

Modelica definition

```
record Glass "Glass(lambda=2,rho=2200,cp=795)"
  extends BaseGlass(lambda = 2, ro = 2200, cp = 795);
end Glass;
```

EEB.Media.Materials.EnvelopeGases

Package with envelope gases

Information

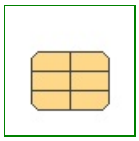
This package provides records for envelope gases

Extends from Modelica.Icons.MaterialProperty (Icon for property classes).

Package Content

Name	Description
BaseEnvelopeGas	
Air	
Argon	
Kripton	
SF6	

EEB.Media.Materials.EnvelopeGases.BaseEnvelopeGas



Information

Extends from Modelica.Icons.Record (Icon for records).

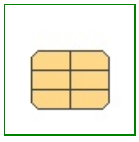
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda		Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp		Specific heat capacity [J/(kg.K)]
Density	ro		Mass density [kg/m3]

Modelica definition

```
record BaseEnvelopeGas
  extends Modelica.Icons.Record;
  parameter Modelica.SIunits.ThermalConductivity lambda "Thermal conductivity";
  parameter Modelica.SIunits.SpecificHeatCapacity cp "Specific heat capacity";
  parameter Modelica.SIunits.Density ro "Mass density";
  /*
      parameter Real R(unit="m2.K/W")
        "Thermal resistance of a unit area of material";

      parameter Integer nStaRef(min=0) = 3
        "Number of state variables in a reference material of 0.2 m concrete";
      parameter Integer nSta(min=1)=max(1, integer(ceil(nStaReal)))
        "Actual number of state variables in material"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Boolean steadyState= (c == 0 or d == 0)
        "Flag, if true, then material is computed using steady-state heat conduction"
        annotation(Evaluate=true);
      parameter Real piRef=331.4
        "Ratio x/sqrt(alpha) for reference material of 0.2 m concrete"
        annotation (Dialog(tab="Advanced"));
      parameter Real piMat=if steadyState then piRef else x*sqrt(c*d)/sqrt(k)
        "Ratio x/sqrt(alpha)"
        annotation(Evaluate=true, Dialog(tab="Advanced"));
      parameter Real nStaReal(min=0) = nStaRef*piMat/piRef
        "Number of states as a real number"
        annotation (Dialog(tab="Advanced"));
    */
end BaseEnvelopeGas;
```

EEB.Media.Materials.EnvelopeGases.Air



Information

Extends from BaseEnvelopeGas.

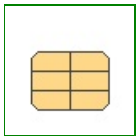
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.02496	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	1008	Specific heat capacity [J/(kg.K)]
Density	ro	1.232	Mass density [kg/m3]

Modelica definition

```
record Air
  extends BaseEnvelopeGas(lambda = 0.02496, cp = 1008, ro = 1.232);
end Air;
```

EEB.Media.Materials.EnvelopeGases.Argon



Information

Extends from BaseEnvelopeGas.

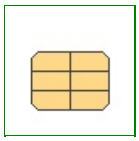
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.01772	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	520	Specific heat capacity [J/(kg.K)]
Density	ro	1.784	Mass density [kg/m3]

Modelica definition

```
record Argon
  extends BaseEnvelopeGas(lambda = 0.01772, cp = 520, ro = 1.784);
end Argon;
```

EEB.Media.Materials.EnvelopeGases.Kripton



Information

Extends from BaseEnvelopeGas.

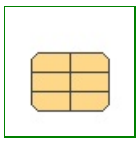
Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.00949	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	248	Specific heat capacity [J/(kg.K)]
Density	ro	3.708	Mass density [kg/m3]

Modelica definition

```
record Kripton
  extends BaseEnvelopeGas(lambda = 0.00949, cp = 248, ro = 3.708);
end Kripton;
```

EEB.Media.Materials.EnvelopeGases.SF6



Information

Extends from BaseEnvelopeGas.

Parameters

Type	Name	Default	Description
ThermalConductivity	lambda	0.01275	Thermal conductivity [W/(m.K)]
SpecificHeatCapacity	cp	614	Specific heat capacity [J/(kg.K)]
Density	ro	6.36	Mass density [kg/m3]

Modelica definition

```
record SF6
  extends BaseEnvelopeGas(lambda = 0.01275, cp = 614, ro = 6.36);
```

```
end SF6;
```

EEB.Components

Package Content

Name	Description
BaseComponents	Package with base component models
AggregateComponents	

EEB.Components.BaseComponents

Package with base component models

Information

This package contains all the base components of this library that are necessary to build more complex models.

Package Content

Name	Description
Ambient	Package with Ambient components
Water	Package with models of components with water
Air	Package with models of components with moist air
Thermal	Package with heat transfer elements
Electrical	
Envelope	
HVAC	
Test	

EEB.Components.BaseComponents.Ambient

Package with Ambient components

Information

This package contains model for ambient elements. Each model refers to an outer model that is AmbientConditions ,which contains all the infomration needed.

Package Content

Name	Description
GroundTemp	Ground temperature
AmbientAirTemp	Ambient air temperature
AmbientAirTempWithOpenings	Ambient air temperature with openings
Radiaton_SkyGround	Radiation sky-ground
Test	

EEB.Components.BaseComponents.Ambient.GroundTemp

Ground temperature

Information

This model is used to set the ground temperature in the model. It refers to an outer model that is AmbientConditions.

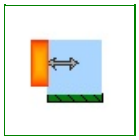
Connectors

Type	Name	Description
HeatPort	port	

Modelica definition

```
model GroundTemp "Ground temperature"
  // match default comp name in AmbientSettings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  Interfaces.Thermal.HeatPort port;
equation
  port.T = ambient_settings.Tgnd;
end GroundTemp;
```

EEB.Components.BaseComponents.Ambient.AmbientAirTemp



Ambient air temperature

Information

This model is used to set the ambient air temperature in the model. It refers to an outer model that is AmbientConditions.

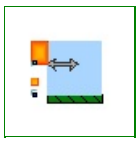
Connectors

Type	Name	Description
HeatPort	port	

Modelica definition

```
model AmbientAirTemp "Ambient air temperature"
  // match default comp name in AmbientSettings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  Interfaces.Thermal.HeatPort port;
```

```
equation
  port.T = ambient_settings.Tamb;
end AmbientAirTemp;
```



EEB.Components.BaseComponents.Ambient.AmbientAirTempWithOpenings

Ambient air temperature with openings

Information

This model is used in order to set ambient air temperature....

It refers to an outer model that is AmbientConditions.

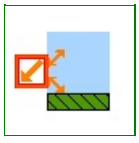
Connectors

Type	Name	Description
HeatPort	port	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
model AmbientAirTempWithOpenings "Ambient air temperature with openings"
  // match default comp name in AmbientSettings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  Media.Substances.MoistAir air;
  Interfaces.Thermal.HeatPort port;
public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart vapour;
  EEB.Interfaces.Thermal.HeatPort diffuse;
equation
  port.T = ambient_settings.Tamb;
  air.p = ambient_settings.pa_avg;
  air.T = ambient_settings.Tamb;
  air.X = ambient_settings.Xa_avg;
  dryair.pa = air.p;
  dryair.ha = air.cp * air.T;
  vapour.pv = air.pv;
  vapour.hv = air.hv;
  diffuse.T = air.T;
end AmbientAirTempWithOpenings;
```

EEB.Components.BaseComponents.Ambient.Radiaton_SkyGround



Radiation sky-ground

Information

This model is used in order to take into account of the sky-ground radiation....

It refers to an outer model that is AmbientConditions.

Parameters

Type	Name	Default	Description
Length	L	5	Surface length [m]

Length	H	3	Surface height [m]
Real	inclination	0	Inclination of the surface: 90° vertical, 180° horizontal
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity

Connectors

Type	Name	Description
HeatPortRad	wall	

Modelica definition

```
model Radiaton_SkyGround "Radiation sky-ground"
  import Modelica.Constants.*;
  // Clarke, 1985
  parameter Length L = 5 "Surface length";
  parameter Length H = 3 "Surface height";
  parameter Real inclination(min = 0, max = 180) = 0 "Inclination of the surface: 90° vertical, 180° horizontal";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  // match default comp name in Ambientambient_settings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  HeatFlowRate Q_sky;
  HeatFlowRate Q_ground;
  EEB.Interfaces.Thermal.HeatPortRad wall;
protected
  parameter Real Fws = (1 - cos(inclination / 180 * pi)) / 2;
  parameter Real Fwg = (1 + cos(inclination / 180 * pi)) / 2;
equation
  wall.Q_flow + Q_sky + Q_ground = 0;
  Q_sky = sigma * Fws * L * H * es * (ambient_settings.Tsky ^ 4 - wall.T ^ 4);
  Q_ground = sigma * Fwg * L * H * es * eg * (ambient_settings.Tgnd ^ 4 - wall.T ^ 4);
end Radiaton_SkyGround;
```

EEB.Components.BaseComponents.Ambient.Test

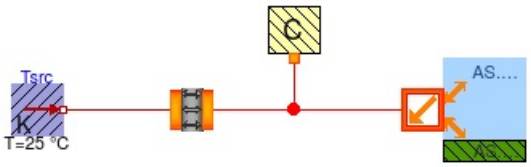
Information

Extends from Modelica.Icons.Package (Icon for standard packages), EEB.Icons.TestModel.

Package Content

Name	Description
TestAmb_SkyGround	

EEB.Components.BaseComponents.Ambient.Test.TestAmb_SkyGround



Information

Extends from EEB.Icons.TestModel.

Modelica definition

```
model TestAmb_SkyGround
  extends EEB.Icons.TestModel;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings;
  EEB.Components.BaseComponents.Ambient.Radiaton_SkyGround radi_sky_ground(inclination = 45);
  EEB.Components.BaseComponents.Thermal.Capacities.ThermalCap cap;
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS conv;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature Tsrc(T = 298.15);
equation
  connect(cap.surf, radi_sky_ground.wall1);
  connect(conv.ss2, cap.surf);
  connect(Tsrc.port, conv.ss1);

end TestAmb_SkyGround;
```

EEB.Components.BaseComponents.Envelope

Information

Extends from EEB.Icons.Envelope.

Package Content

Name	Description
SolidMultilayer_Homogeneous	
SolidMultilayer_NonHomogeneous	
GasLayer	
GlassLayer	
DoubleGlass	

EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous

Parameters

Type	Name	Default	Description
Area	A	10	wall surface [m2]
Length	s	0.4	wall thickness [m]
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

Connectors

Type	Name	Description
HeatPort	side1	
HeatPort	side2	

Modelica definition

```
model SolidMultilayer_Homogeneous
  parameter Area A = 10 "wall surface";
  parameter Length s = 0.4 "wall thickness";
  parameter Density ro = 2400 "wall density";
  parameter SpecificHeatCapacity cp = 880 "wall cp";
  parameter ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter Temperature Tstart = 273.15 + 25 "initial T, all layers";
  Temperature T[n](each start = Tstart);
  ThermalConductance g = lambda * A / (s / n);
  HeatCapacity c = ro * A * s / n * cp;
  Interfaces.Thermal.HeatPort side1;
  Interfaces.Thermal.HeatPort side2;
equation
  0 = c * der(T[1]) - side1.Q_flow + g * (T[1] - T[2]);
  side1.T = T[1];
  for i in 2:n - 1 loop
    c * der(T[i]) = g * (T[i - 1] - T[i]) - g * (T[i] - T[i + 1]);
  end for;
  c * der(T[n]) = g * (T[n - 1] - T[n]) + side2.Q_flow;
  side2.T = T[n];
end SolidMultilayer_Homogeneous;
```

EEB.Components.BaseComponents.Envelope.SolidMultilayer_NonHomogeneous

Parameters

Type	Name	Default	Description
Area	A	10	wall surface [m2]
Length	s[:]	{0.05,0.4,0.05}	layer thicknesses [m]
Density	ro[:]	{1600,2400,1600}	layer densities [kg/m3]
SpecificHeatCapacity	cp[:]	{400,880,400}	layer cps [J/(kg.K)]
ThermalConductivity	lambda[:]	{0.2,1.91,0.2}	layer thermal conds [W/(m.K)]
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

Connectors

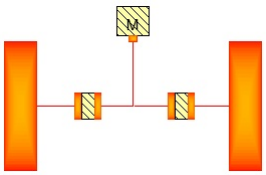
Type	Name	Description
HeatPort	side1	
HeatPort	side2	

Modelica definition

```
model SolidMultilayer_NonHomogeneous
  Components.BaseComponents.Thermal.Capacities.MassT layer[size(s, 1)](each Tstart = Tstart, M = M, cp = cp);
  Components.BaseComponents.Thermal.HeatTransfer.Conduction_SS cond[size(s, 1) + 1](each A = A, d = Functions.set_layers_thicknesses(size(s, 1), s), lambda = Functions.set_layers_conductivitie
  Interfaces.Thermal.HeatPort side1;
  Interfaces.Thermal.HeatPort side2;
  parameter Area A = 10 "wall surface";
  parameter Length s[:] = {0.05, 0.4, 0.05} "layer thicknesses";
  parameter Density ro[:] = {1600, 2400, 1600} "layer densities";
  parameter SpecificHeatCapacity cp[:] = {400, 880, 400} "layer cps";
  parameter ThermalConductivity lambda[:] = {0.2, 1.91, 0.2} "layer thermal conds";
  parameter Temperature Tstart = 273.15 + 25 "initial T, all layers";
  Temperature T[size(s, 1)];
  // parameter Integer n=size(s,1) "number of layers";
protected
  parameter Mass M[size(s, 1)] = array(ro[i] * s[i] * A / size(s, 1) for i in 1:size(s, 1));
equation
```

```
  for i in 1:size(s, 1) loop
    T[i] = layer[i].T;
  end for;
  connect(side1, cond[1].ss1);
  for i in 1:size(s, 1) loop
    connect(layer[i].surf, cond[i].ss2);
    connect(layer[i].surf, cond[i + 1].ss1);
  end for;
  connect(side2, cond[size(s, 1) + 1].ss2);
end SolidMultilayer_NonHomogeneous;
```

EEB.Components.BaseComponents.Envelope.GasLayer



Parameters

Type	Name	Default	Description
Length	L	5	Length of the envelope [m]
Length	H	3	Height of the envelope [m]
Length	d	0.2	Width of the envelope [m]
BaseEnvelopeGas	material		Type of gas
Boolean	static	false	Flag that enable static analysis
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]

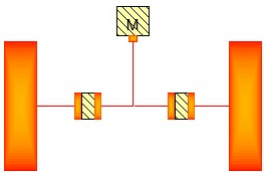
Connectors

Type	Name	Description
HeatPort	side1	
HeatPort	side2	

Modelica definition

```
model GasLayer
  parameter Length L = 5 "Length of the envelope";
  parameter Length H = 3 "Height of the envelope";
  parameter Length d = 0.2 "Width of the envelope";
  parameter Media.Materials.EnvelopeGases.BaseEnvelopeGas material "Type of gas";
  parameter Boolean static = false "Flag that enable static analysis";
  parameter Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  Interfaces.Thermal.HeatPort side1;
  Interfaces.Thermal.HeatPort side2;
  Thermal.Capacities.MassT massT(M = M, cp = cp, Tstart = Tstart);
  Thermal.HeatTransfer.Conduction_SS cs1(A = L * H, d = d / 2, lambda = lambda);
  Thermal.HeatTransfer.Conduction_SS cs2(A = L * H, d = d / 2, lambda = lambda);
protected
  parameter ThermalConductivity lambda = material.lambda "Thermal conductivity";
  parameter SpecificHeatCapacity cp = material.cp "Specific heat capacity";
  parameter Density ro = material.ro "Density";
  parameter Mass M = ro * L * H * d "Mass";
equation
  connect(side1, cs1.ss1);
  connect(cs2.ss2, side2);
  connect(cs1.ss2, cs2.ss1);
  connect(cs1.ss2, massT.surf);
end GasLayer;
```

EEB.Components.BaseComponents.Envelope.GlassLayer



Parameters

Type	Name	Default	Description
Length	L	5	Length of the envelope [m]
Length	H	3	Height of the envelope [m]
Length	d	0.2	Width of the envelope [m]
BaseGlass	material		Type of glass
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]

Connectors

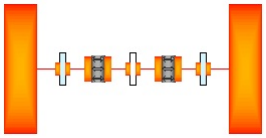
Type	Name	Description
HeatPort	side1	
HeatPort	side2	

Modelica definition

```
model GlassLayer
  parameter Length L = 5 "Length of the envelope";
  parameter Length H = 3 "Height of the envelope";
  parameter Length d = 0.2 "Width of the envelope";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
  parameter Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  Interfaces.Thermal.HeatPort side1;
  Interfaces.Thermal.HeatPort side2;
  Thermal.Capacities.MassT massT(M = M, cp = cp, Tstart = Tstart);
  Thermal.HeatTransfer.Conduction_SS cs1(A = L * H, d = d / 2, lambda = lambda);
  Thermal.HeatTransfer.Conduction_SS cs2(A = L * H, d = d / 2, lambda = lambda);
protected
  parameter ThermalConductivity lambda = material.lambda "Thermal conductivity";
  parameter SpecificHeatCapacity cp = material.cp "Specific heat capacity";
  parameter Density ro = material.ro "Density";
  parameter Mass M = ro * L * H * d "Mass";
equation
```

```
  connect(side1, cs1.ss1);
  connect(cs2.ss2, side2);
  connect(cs1.ss2, cs2.ss1);
  connect(cs1.ss2, massT.surf);
end GlassLayer;
```

EEB.Components.BaseComponents.Envelope.DoubleGlass



Parameters

Type	Name	Default	Description
Length	L	5	Length of the glass [m]
Length	H	3	Height of the glass [m]
Length	d_glass	0.2	Width of the single glass [m]
Length	d_int	0.2	Distance between two glasses [m]
CoefficientOfHeatTransfer	gamma	5	Heat transfer coefficient between the glasses and internal gas [W/(m2.K)]
BaseGlass	material_glass		Type of glass
BaseEnvelopeGas	material_gas		Type of internal gas between the glasses
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]

Connectors

Type	Name	Description
HeatPort	side1	
HeatPort	side2	

Modelica definition

```
model DoubleGlass
  parameter Length L = 5 "Length of the glass";
  parameter Length H = 3 "Height of the glass";
  parameter SI.Length d_glass = 0.2 "Width of the single glass";
  parameter SI.Length d_int = 0.2 "Distance between two glasses";
  parameter SI.CoefficientOfHeatTransfer gamma = 5 "Heat transfer coefficient between the glasses and internal gas";
  parameter EEB.Media.Materials.Glasses.BaseGlass material_glass "Type of glass";
  parameter EEB.Media.Materials.EnvelopeGases.BaseEnvelopeGas material_gas "Type of internal gas between the glasses";
  parameter SI.Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  Interfaces.Thermal.HeatPort side1;
  Interfaces.Thermal.HeatPort side2;
  EEB.Components.BaseComponents.Envelope.GlassLayer glass2(Tstart = Tstart, material = material_glass, d = d_glass, L = L, H = H);
  EEB.Components.BaseComponents.Envelope.GlassLayer glass1(Tstart = Tstart, material = material_glass, d = d_glass, L = L, H = H);
  Thermal.HeatTransfer.Convection_SS conv2(S = L * H, gamma = gamma);
  EEB.Components.BaseComponents.Envelope.GasLayer gas(L = L, H = H, d = d_int, material = material_gas, Tstart = Tstart);
  Thermal.HeatTransfer.Convection_SS conv1(S = L * H, gamma = gamma);
equation
  connect(side1, glass1.side1);
  connect(glass1.side2, conv1.ss1);
  connect(conv1.ss2, gas.side1);
  connect(gas.side2, conv2.ss1);
  connect(conv2.ss2, glass2.side1);
  connect(glass2.side2, side2);
end DoubleGlass;
```

EEB.Components.BaseComponents.Air

Package with models of components with moist air

Information

This package contains models of components with moist air.

Package Content

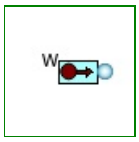
Name	Description
Volumes	
Movers	
Pdrops	
Renovation	
Sources	
Sinks	

EEB.Components.BaseComponents.Air.Sources

Package Content

Name	Description
AirSource_wTX_fixed	
AirSource_pTX_fixed	
AirSource_pTphi_prescribed	

EEB.Components.BaseComponents.Air.Sources.AirSource_wTX_fixed



Parameters

Type	Name	Default	Description
MassFlowRate	w0	1	Dry air mass flow rate [kg/s]
Temperature	T0	273.15 + 20	prescribed temperature [K]
MassFraction	X0	0.001	prescribed absolute humidity [kg_H20/kg_DA] [1]

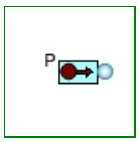
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange	

Modelica definition

```
model AirSource_wTX_fixed
  Media.Substances.MoistAir air;
  parameter MassFlowRate w0 = 1 "Dry air mass flow rate";
  parameter Temperature T0 = 273.15 + 20 "prescribed temperature";
  parameter MassFraction X0 = 0.001 "prescribed absolute humidity [kg_H20/kg_DA]";
  Interfaces.Air.MoistAirFlange_waxa air_flange;
equation
  air_flange.wa = -w0;
  air.p = air_flange.pa;
  air.T = T0;
  air.X = X0;
  // enthalpy boundary condition
  air.h = air_flange.ha;
  air.X = air_flange.xa;
end AirSource_wTX_fixed;
```

EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed



Parameters

Type	Name	Default	Description
Pressure	p0	101325	prescribed pressure [Pa]
Temperature	T0	273.15 + 20	prescribed temperature [K]
MassFraction	X0	0.001	prescribed absolute humidity [kg_H20/kg_DA] [1]

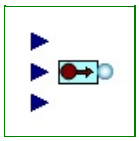
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange	

Modelica definition

```
model AirSource_pTX_fixed
  Media.Substances.MoistAir air;
  parameter Pressure p0 = 101325 "prescribed pressure";
  parameter Temperature T0 = 273.15 + 20 "prescribed temperature";
```

```
  parameter MassFraction X0 = 0.001 "prescribed absolute humidity [kg_H20/kg_DA]";
  Interfaces.Air.MoistAirFlange_waxa air_flange;
equation
  air_flange.pa = p0;
  air.p = air_flange.pa;
  air.T = T0;
  air.X = X0;
  // enthalpy boundary condition
  air.h = air_flange.ha;
  air.X = air_flange.xa;
end AirSource_pTX_fixed;
```



EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange	
input RealInput	iP	
input RealInput	iT	
input RealInput	iphi	

Modelica definition

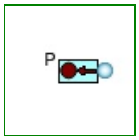
```
model AirSource_pTphi_prescribed
  Media.Substances.MoistAir air;
  Interfaces.Air.MoistAirFlange_waxa air_flange;
  Modelica.Blocks.Interfaces.RealInput iP;
  Modelica.Blocks.Interfaces.RealInput iT;
  Modelica.Blocks.Interfaces.RealInput iphi;
equation
  air_flange.pa = iP;
  air.p = air_flange.pa;
  air.T = iT;
  air.phi = iphi / 100;
// FIXME RH 0-100 as from data files
  air.h = air_flange.ha;
  air.X = air_flange.xa;
end AirSource_pTphi_prescribed;
```

EEB.Components.BaseComponents.Air.Sinks

Package Content

Name	Description
AirSink_P_fixed	

EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed



Parameters

Type	Name	Default	Description
Pressure	p0	101325	prescribed pressure [Pa]

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange	

Modelica definition

```
model AirSink_P_fixed
  Media.Substances.MoistAir air;
  parameter Pressure p0 = 101325 "prescribed pressure";
  Interfaces.Air.MoistAirFlange_waxa air_flange;
equation
  air.p = air_flange.pa;
  air_flange.pa = p0;
  air_flange.ha = inStream(air_flange.ha);
  air_flange.xa = inStream(air_flange.xa);
  // enthalpy boundary condition
  air.h = air_flange.ha;
  air.X = air_flange.xa;
end AirSink_P_fixed;
```

EEB.Components.BaseComponents.Air.Volumes

Package Content

Name	Description
AirVolume	
AirVolume_closed	
AirVolumeWithWall_Condensing	
AirVolume_wawvQdPort	
AirVolume_only_wawvQdport	

EEB.Components.BaseComponents.Air.Volumes.AirVolume

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Volume	V	0.001	volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
Temperature	Tstart	273.15 + 20	Initial moist air temperature [K]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
HeatPort	heatPort	

Modelica definition

```
model AirVolume
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Volume V = 0.001 "volume";
  parameter Pressure Pstart = 101325 "Initial moist air pressure";
  parameter Temperature Tstart = 273.15 + 20 "Initial moist air temperature";
  parameter MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  EEB.Media.Substances.MoistAir air(is4cap=true);
  Mass Ma(stateSelect=StateSelect.avoid) "Total dry air mass";
  Mass Mv(stateSelect=StateSelect.avoid) "Total vapour mass";
  Energy Ea(stateSelect=StateSelect.avoid) "Energy of the moist air";
  Pressure p(start=Pstart);
  Temperature T(start=Tstart);
  MassFraction X(start=Xstart);
  Interfaces.Thermal.HeatPort heatPort;
equation
  air.p = p;
  air.T = T;
  air.X = X;
  // Masses and energy
  Ma + Mv = V * air.d;
  Mv = Ma * air.X;
  Ea = Ma * (air.h - air.p / air.d);
  // Balances
  der(Ma) = wa1 + wa2;
  der(Mv) = wa1 * actualStream(air_flange1.xa) + wa2 * actualStream(air_flange2.xa);
  der(Ea) = wa1 * actualStream(air_flange1.ha) + wa2 * actualStream(air_flange2.ha)
            + heatPort.Q_flow;
  // Connector equations
  heatPort.T = T;
  pa1 = air.p;
  pa2 = air.p;
  xaout1 = air.X;
  xaout2 = air.X;
  haout1 = air.h;
  haout2 = air.h;
end AirVolume;
```

EEB.Components.BaseComponents.Air.Volumes.AirVolume_closed

Parameters

Type	Name	Default	Description
Volume	V	0.001	volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
Temperature	Tstart	273.15 + 20	Initial moist air temperature [K]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]

Connectors

Type	Name	Description
HeatPort	heatPort	

Modelica definition

```
model AirVolume_closed
  EEB.Media.Substances.MoistAir air;
  parameter Volume V = 0.001 "volume";
  parameter Pressure Pstart = 101325 "Initial moist air pressure";
  parameter Temperature Tstart = 273.15 + 20 "Initial moist air temperature";
  parameter MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  Mass Ma "Total dry air mass";
  Mass Mv "Total vapour mass";
  Energy Ea "Energy of the moist air";
  Pressure p(start = Pstart) "Pressure of the air";
```

```
  Interfaces.Thermal.HeatPort heatPort;
initial equation
  air.T = Tstart;
  air.X = Xstart;
  air.p = Pstart;
equation
  air.p = p;
  Ma + Mv = V * air.d;
  Mv = Ma * air.X;
  Ea = Ma * (air.h - air.p / air.d);
  der(Ea) = heatPort.Q_flow;
  der(Ma) = 0;
  der(Mv) = 0;
  air.T = heatPort.T;
end AirVolume_closed;
```

EEB.Components.BaseComponents.Air.Volumes.AirVolumeWithWall_Condensing

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Volume	V	1	volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
Temperature	Tstart	273.15 + 20	Initial moist air temperature [K]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]
HeatCapacity	Cw	50	wall heat capacity [J/K]
ThermalConductance	Gaw	100	air-wall thermal conductance [W/K]

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
HeatPort	heatPort	

Modelica definition

```
model AirVolumeWithWall_Condensing
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Volume V = 1 "volume";
  parameter Pressure Pstart = 101325 "Initial moist air pressure";
  parameter Temperature Tstart = 273.15 + 20 "Initial moist air temperature";
  parameter MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  parameter HeatCapacity Cw = 50 "wall heat capacity";
  parameter ThermalConductance Gaw = 100 "air-wall thermal conductance";
  EEB.Media.Substances.MoistAir air(is4cap=true);
  EEB.Media.Substances.MoistAir wallsat;
  Mass Ma(stateSelect=StateSelect.avoid) "Total dry air mass";
  Mass Mv(stateSelect=StateSelect.avoid) "Total vapour mass";
  Energy Ea(stateSelect=StateSelect.avoid) "Energy of the moist air";
  Pressure p(start=Pstart);
  Temperature T(start=Tstart);
  MassFraction X(start=Xstart);
  Temperature Tw(stateSelect=StateSelect.prefer) "Wall temperature";
  MassFlowRate wc "Condensed water mass flow rate";
  HeatFlowRate Qawsens "Condensation sensible heat";
  HeatFlowRate Qawlat "Condenstion latent heat";
  Interfaces.Thermal.HeatPort heatPort;
equation
  air.p = p;
  air.T = T;
  air.X = X;
// Masses and energy
  Ma + Mv = V * air.d;
  Mv = Ma * air.X;
  Ea = Ma * (air.h - air.p / air.d);
// Balances
  der(Ma) = wa1 + wa2;
  der(Mv) = wa1 * actualStream(air_flange1.xa)
            + wa2 * actualStream(air_flange2.xa) - wc;
  der(Ea) = wa1 * actualStream(air_flange1.ha)
            + wa2 * actualStream(air_flange2.ha)
            - Qawsens - Qawlat;
  Cw * der(Tw) = Qawsens + Qawlat + heatPort.Q_flow;
// Wallsat air conditions
  wallsat.p = p;
  wallsat.T = Tw;
  wallsat.phi = 1.0;
// Condensation and corresponding heat rates
  wc = if air.X > 1e-6 and air.X > wallsat.X then Gaw / air.cp * (air.X - wallsat.X) else 0.0;
  Qawsens = Gaw * (air.T - Tw);
  Qawlat = wc * (air.hv - wallsat.hl);
// Connector equations
  air.T = heatPort.T;
  pa1 = air.p;
  pa2 = air.p;
  xaout1 = air.X;
  xaout2 = air.X;
  haout1 = air.h;
  haout2 = air.h;
end AirVolumeWithWall_Condensing;
```

EEB.Components.BaseComponents.Air.Volumes.AirVolume_wawvQdPort

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Volume	V	10	volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
Temperature	Tstart	273.15 + 20	Initial moist air temperature [K]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
HeatPort	heatPort	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
model AirVolume_wawvQdPort
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Volume V = 10 "volume";
  parameter Pressure Pstart = 101325 "Initial moist air pressure";
  parameter Temperature Tstart = 273.15 + 20 "Initial moist air temperature";
  parameter MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  EEB.Media.Substances.MoistAir air(is4cap=true);
  Mass Ma(stateSelect=StateSelect.avoid) "Total dry air mass";
  Mass Mv(stateSelect=StateSelect.avoid) "Total vapour mass";
  Energy Ea(stateSelect=StateSelect.avoid) "Energy of the moist air";
  Pressure p(start=Pstart);
  Temperature T(start=Tstart);
  MassFraction X(start=Xstart);
  // wawvQd port variables (for mixing with other volumes)
  MassFlowRate wamix, wvmix;
  HeatFlowRate Qdiff;
  Interfaces.Thermal.HeatPort heatPort;
//public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart vapour;
  EEB.Interfaces.Thermal.HeatPort diffuse;
equation
  air.p = p;
  air.T = T;
  air.X = X;
  // Masses and energy
  Ma + Mv = V * air.d;
  Mv = Ma * air.X;
  Ea = Ma * (air.h - air.p / air.d);
  // Balances
  der(Ma) = wa1 + wa2 + wamix;
  der(Mv) = wa1 * actualStream(air_flange1.xa)
            + wa2 * actualStream(air_flange2.xa) + wvmix;
  der(Ea) = wa1 * actualStream(air_flange1.ha)
            + wa2 * actualStream(air_flange2.ha) + heatPort.Q_flow
            + wamix * actualStream(dryair.ha)
            + wvmix * actualStream(vapour.hv) + Qdiff;

  // Connector equations
  air.T = heatPort.T;
  pa1 = air.p;
  pa2 = air.p;
  xaout1 = air.X;
  xaout2 = air.X;
  haout1 = air.h;
  haout2 = air.h;
  dryair.pa = air.p;
  dryair.wa = wamix;
  dryair.ha = air.ha;
  vapour.pv = air.pv;
  vapour.wv = wvmix;
  vapour.hv = air.hv;
  diffuse.T = air.T;
  diffuse.Q_flow = Qdiff;
end AirVolume_wawvQdPort;
```

EEB.Components.BaseComponents.Air.Volumes.AirVolume_only_wawvQdport

Parameters

Type	Name	Default	Description
Volume	V	0.001	volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
Temperature	Tstart	273.15 + 20	Initial moist air temperature [K]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]

Connectors

Type	Name	Description
HeatPort	heatPort	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

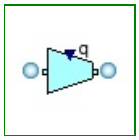
```
model AirVolume_only_wawvQdport
  EEB.Media.Substances.MoistAir air;
  parameter Volume V = 0.001 "volume";
  parameter Pressure Pstart = 101325 "Initial moist air pressure";
  parameter Temperature Tstart = 273.15 + 20 "Initial moist air temperature";
  parameter MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  Mass Ma "Total dry air mass";
```

```
  Mass Mv "Total vapour mass";
  Energy Ea "Energy of the moist air";
  Pressure p(start = Pstart) "Pressure of the air";
  // wawvQd port variables (for mixing with other volumes)
  MassFlowRate wamix, wvmix;
  HeatFlowRate Qdiff;
  Interfaces.Thermal.HeatPort heatPort;
 // Interfaces.Air.MoistAirFlange_wawvQd air_flange_wawvQd annotation(Placement(transformation(extent = {{-10, -90}, {10, -70}}), iconTransformation(extent = {{-20, -100}, {20, -58}})));
public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart
                                                  dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart
                                                  vapour;
  EEB.Interfaces.Thermal.HeatPort
                                diffuse;
initial equation
  air.T = Tstart;
  air.X = Xstart;
  air.p = Pstart;
equation
  air.p = p;
  Ma + Mv = V * air.d;
  Mv = Ma * air.X;
  Ea = Ma * (air.h - air.p / air.d);
  der(Ea) = heatPort.Q_flow + Qdiff;
  der(Ma) = wamix;
  der(Mv) = wvmix;
  air.T = heatPort.T;
  // wawvQd port
//   air_flange_wawvQd.dryair.pa = air.p;
//   air_flange_wawvQd.dryair.wa = wamix;
//   air_flange_wawvQd.dryair.ha = air.ha;
//   air_flange_wawvQd.vapour.pv = air.pv;
//   air_flange_wawvQd.vapour.wv = wvmix;
//   air_flange_wawvQd.vapour.hv = air.hv;
//   air_flange_wawvQd.diffuse.T = air.T;
//   air_flange_wawvQd.diffuse.Q_flow = Qdiff;
  dryair.pa = air.p;
  dryair.wa = wamix;
  dryair.ha = air.ha;
  vapour.pv = air.pv;
  vapour.wv = wvmix;
  vapour.hv = air.hv;
  diffuse.T = air.T;
  diffuse.Q_flow = Qdiff;
end AirVolume_only_wawvQdport;
```

EEB.Components.BaseComponents.Air.Movers

Package Content

Name	Description
AirPrescribedFlowRate_Volume	



EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input RealInput	iq	

Modelica definition

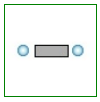
```
model AirPrescribedFlowRate_Volume
  extends Interfaces.Air.PartialTwoPort_waxa;
  Media.Substances.MoistAir air;
  Modelica.Blocks.Interfaces.RealInput iq;
equation
  // mass balance
  wa1 + wa2 = 0;
  wa1 = iq * air.d;
  // pressure drop
  air.p = 0.5 * (pa1 + pa2);
  // air humidity ratio boundary conditions
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  air.X = xaout1;
  // enthalpies boundary conditions
  haout1 = inStream(air_flange2.ha);
  haout2 = inStream(air_flange1.ha);
  air.h = haout1;
end AirPrescribedFlowRate_Volume;
```

EEB.Components.BaseComponents.Air.Pdrops

Package Content

Name	Description
AirPdrop_Lin	
AirPdrop_Lin_Gcmd01	
AirPdrop_Lin_NomPoint	
AirPdrop_Lin_NomPoint_Gcmd01	
AirPdrop_Lin_NomPoint_mix	
AirPdrop_Lin_NomPoint_mix_Gcmd01	

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin



° °

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Real	kf	5	Friction coefficient
Length	dz	1	Length of the press. drop [m]

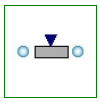
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	

Modelica definition

```
model AirPdrop_Lin
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Real kf = 5 "Friction coefficient";
  parameter SI.Length dz = 1 "Length of the press. drop";
protected
  parameter Real G = 1 / kf / dz "Equivalent conductance";
equation
  // No mass storage
  wa1 + wa2 = 0;
  // Linear pressure drop
  (pa1 - pa2) * G = wa1;
  // other equations
  haout1 = inStream(air_flange2.ha);
  haout2 = inStream(air_flange1.ha);
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
end AirPdrop_Lin;
```

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_Gcmd01



° °

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Real	kf	5	Friction coefficient
Length	dz	1	Length of the press. drop [m]

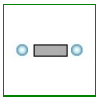
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input RealInput	cmd01	

Modelica definition

```
model AirPdrop_Lin_Gcmd01
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Real kf = 5 "Friction coefficient";
  parameter SI.Length dz = 1 "Length of the press. drop";
  Modelica.Blocks.Interfaces.RealInput cmd01;
protected
  parameter Real G = 1 / kf / dz "Equivalent conductance";
equation
  // No mass storage
  wa1 + wa2 = 0;
  // Linear pressure drop
  (pa1 - pa2) * G * cmd01 = wa1;
  // other equations
  haout1 = inStream(air_flange2.ha);
  haout2 = inStream(air_flange1.ha);
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
end AirPdrop_Lin_Gcmd01;
```

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint



Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Pressure	dpnom	1000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal mass flowrate [kg/s]

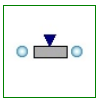
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	

Modelica definition

```
model AirPdrop_Lin_NomPoint
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Pressure dpnom = 1000 "Nominal pressure drop";
  parameter MassFlowRate wnom = 0.01 "Nominal mass flowrate";
protected
  parameter Real G = wnom / dpnom "Equivalent conductance";
equation
  // No mass storage
  wa1 + wa2 = 0;
  // Linear pressure drop
  (pa1 - pa2) * G = wa1;
  // other equations
  haout1 = inStream(air_flange2.ha);
  haout2 = inStream(air_flange1.ha);
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
end AirPdrop_Lin_NomPoint;
```

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_Gcmd01



Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Pressure	dpnom	1000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal mass flowrate [kg/s]

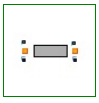
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input RealInput	cmd01	

Modelica definition

```
model AirPdrop_Lin_NomPoint_Gcmd01
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Pressure dpnom = 1000 "Nominal pressure drop";
  parameter MassFlowRate wnom = 0.01 "Nominal mass flowrate";
  Modelica.Blocks.Interfaces.RealInput cmd01;
protected
  parameter Real G = wnom / dpnom "Equivalent conductance";
equation
  // No mass storage
  wa1 + wa2 = 0;
  // Linear pressure drop
  (pa1 - pa2) * G * cmd01 = wa1;
  // other equations
  haout1 = inStream(air_flange2.ha);
  haout2 = inStream(air_flange1.ha);
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
end AirPdrop_Lin_NomPoint_Gcmd01;
```

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix



Parameters

Type	Name	Default	Description
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

Type	Name	Description
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	
MoistAirFlange_wawvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
model AirPdrop_Lin_NomPoint_mix
  parameter Pressure dpnom = 10000 "Nominal pressure drop";
  parameter MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  Pressure pa1, pa2, pv1, pv2;
  // pa's are _total_ pressures
  MassFlowRate wa1, wa2, wv1, wv2;
  Temperature Ta1, Ta2;
  HeatFlowRate Qd;
  ThermalConductance Gd;
  //Interfaces.Air.MoistAirFlange_wawvQd side1 annotation(Placement(transformation(extent = {{-90, -10}, {-70, 10}}), iconTransformation(extent = {{-100, -20}, {-60, 20}})));
  //Interfaces.Air.MoistAirFlange_wawvQd side2 annotation(Placement(transformation(extent = {{70, -10}, {90, 10}}), iconTransformation(extent = {{60, -20}, {100, 20}})));
protected
  parameter Real Ga = wnom / dpnom "Equivalent conductance";
  parameter Real Gv = GvOverGa * Ga "Equivalent conductance";
  parameter Real a = (Gdw0 - Gdwnom) / (Gdwnom * wnom ^ 2);
public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart vapour;
  EEB.Interfaces.Thermal.HeatPort diffuse;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart Bdryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart Bvapour;
  EEB.Interfaces.Thermal.HeatPort Bdiffuse;
equation
  //pa1 = side1.dryair.pa - side1.vapour.pv;
  //pa2 = side2.dryair.pa - side2.vapour.pv;
  //pv1 = side1.vapour.pv;
  //pv2 = side2.vapour.pv;
  //wa1 = side1.dryair.wa;
  //wa2 = side2.dryair.wa;
  //wv1 = side1.vapour.wv;
  //wv2 = side2.vapour.wv;
  //Ta1 = side1.diffuse.T;
  //Ta2 = side2.diffuse.T;
  pa1 = dryair.pa - vapour.pv;
  pa2 = Bdryair.pa - Bvapour.pv;
  pv1 = vapour.pv;
  pv2 = Bvapour.pv;
  wa1 = dryair.wa;
  wa2 = Bdryair.wa;
  wv1 = vapour.wv;
  wv2 = Bvapour.wv;
  Ta1 = diffuse.T;
  Ta2 = Bdiffuse.T;
  wa1 + wa2 = 0;
  wv1 + wv2 = 0;
  wa1 = Ga * (pa1 - pa2);
  wv1 = Gv * (pv1 - pv2);
  //side1.diffuse.Q_flow + side2.diffuse.Q_flow = 0;
  diffuse.Q_flow + Bdiffuse.Q_flow = 0;
  //Qd = side1.diffuse.Q_flow;
  Qd = diffuse.Q_flow;
  Gd = Gdw0 / (1 + a * (wa1 + wv1) ^ 2);
  Qd = Gd * (Ta1 - Ta2);
  //side1.dryair.ha = inStream(side2.dryair.ha);
  //side2.dryair.ha = inStream(side1.dryair.ha);
  //side1.vapour.hv = inStream(side2.vapour.hv);
  //side2.vapour.hv = inStream(side1.vapour.hv);
  dryair.ha = inStream(Bdryair.ha);
  Bdryair.ha = inStream(dryair.ha);
  vapour.hv = inStream(Bvapour.hv);
  Bvapour.hv = inStream(vapour.hv);
end AirPdrop_Lin_NomPoint_mix;
```

EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01

Parameters

Type	Name	Default	Description
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

Type	Name	Description
input RealInput	cmd01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	
MoistAirFlange_wawvQd_wvPart	Bvapour	

HeatPort	Bdiffuse	

Modelica definition

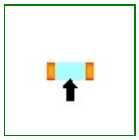
```
model AirPdrop_Lin_NomPoint_mix_Gcmd01
  parameter Pressure dpnom = 10000 "Nominal pressure drop";
  parameter MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  Pressure pa1, pa2, pv1, pv2;
  // pa's are _total_ pressures
  MassFlowRate wa1, wa2, wv1, wv2;
  Temperature Ta1, Ta2;
  HeatFlowRate Qd;
  ThermalConductance Gd;
  //Interfaces.Air.MoistAirFlange_wavvQd side1 annotation(Placement(transformation(extent = {{-90, -10}, {-70, 10}}), iconTransformation(extent = {{-100, -20}, {-60, 20}})));
  //Interfaces.Air.MoistAirFlange_wavvQd side2 annotation(Placement(transformation(extent = {{70, -10}, {90, 10}}), iconTransformation(extent = {{60, -20}, {100, 20}})));
  Modelica.Blocks.Interfaces.RealInput cmd01;
protected
  parameter Real Ga = wnom / dpnom "Equivalent conductance";
  parameter Real Gv = GvOverGa * Ga "Equivalent conductance";
  parameter Real a = (Gdw0 - Gdwnom) / (Gdwnom * wnom ^ 2);
public
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_waPart
                                                  dryair;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_wvPart
                                                  vapour;
  EEB.Interfaces.Thermal.HeatPort
                                diffuse;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_waPart
                                                  Bdryair;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_wvPart
                                                  Bvapour;
  EEB.Interfaces.Thermal.HeatPort
                                Bdiffuse;
equation
//   pa1 = side1.dryair.pa - side1.vapour.pv;
//   pa2 = side2.dryair.pa - side2.vapour.pv;
//   pv1 = side1.vapour.pv;
//   pv2 = side2.vapour.pv;
//   wa1 = side1.dryair.wa;
//   wa2 = side2.dryair.wa;
//   wv1 = side1.vapour.wv;
//   wv2 = side2.vapour.wv;
//   Ta1 = side1.diffuse.T;
//   Ta2 = side2.diffuse.T;
  pa1 = dryair.pa - vapour.pv;
  pa2 = Bdryair.pa - Bvapour.pv;
  pv1 = vapour.pv;
  pv2 = Bvapour.pv;
  wa1 = dryair.wa;
  wa2 = Bdryair.wa;
  wv1 = vapour.wv;
  wv2 = Bvapour.wv;
  Ta1 = diffuse.T;
  Ta2 = Bdiffuse.T;
  wa1 + wa2 = 0;
  wv1 + wv2 = 0;
  wa1 = cmd01 * Ga * (pa1 - pa2);
  wv1 = cmd01 * Gv * (pv1 - pv2);
//   side1.diffuse.Q_flow + side2.diffuse.Q_flow = 0;
//   Qd = side1.diffuse.Q_flow;
  diffuse.Q_flow + Bdiffuse.Q_flow = 0;
  Qd = diffuse.Q_flow;
  Gd = Gdw0 / (1 + a * (wa1 + wv1) ^ 2);
  Qd = cmd01 * Gd * (Ta1 - Ta2);
//   side1.dryair.ha = inStream(side2.dryair.ha);
//   side2.dryair.ha = inStream(side1.dryair.ha);
//   side1.vapour.hv = inStream(side2.vapour.hv);
//   side2.vapour.hv = inStream(side1.vapour.hv);
  dryair.ha = inStream(Bdryair.ha);
  Bdryair.ha = inStream(dryair.ha);
  vapour.hv = inStream(Bvapour.hv);
  Bvapour.hv = inStream(vapour.hv);
end AirPdrop_Lin_NomPoint_mix_Gcmd01;
```

EEB.Components.BaseComponents.Air.Renovation

Package Content

Name	Description
AirRenovation	

EEB.Components.BaseComponents.Air.Renovation.AirRenovation



Parameters

Type	Name	Default	Description
Volume	V	1	Room volume [m3]
Real	k	0.6/3600	recycle air parameter
MassFlowRate	wr	V*k	Air flow rate [kg/s]
SpecificHeatCapacity	cp_a	1005	cp air [J/(kg.K)]

Connectors

Type	Name	Description
HeatPort	heatPort1	
HeatPort	heatPort2	

Modelica definition

```
model AirRenovation
  //Model of air recycle inside a room; It is advisable to recycle 0.6 volume/hour
  parameter Volume V = 1 "Room volume";
  parameter Real k = 0.6 / 3600 "recycle air parameter";
  parameter MassFlowRate wr = V * k "Air flow rate";
  parameter SpecificHeatCapacity cp_a = 1005 "cp air";
  HeatFlowRate Q "Heat imposed by air florw rate";
  Interfaces.Thermal.HeatPort heatPort1;
  Interfaces.Thermal.HeatPort heatPort2;
equation
  Q = wr * cp_a * (heatPort1.T - heatPort2.T);
  heatPort1.Q_flow = Q;
  heatPort1.Q_flow + heatPort2.Q_flow = 0;
end AirRenovation;
```

EEB.Components.BaseComponents.Water

Package with models of components with water

Information

This package contains models of components with water.

Package Content

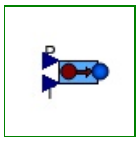
Name	Description
Volumes	
Tanks	
Pipes	
Pumps	
Valves	
Pressurisers	
Sources	
Sinks	

EEB.Components.BaseComponents.Water.Sources

Package Content

Name	Description
WaterSource_PT	
WaterSource_PT_fixed	
WaterSource_WT	
WaterSource_WT_fixed	

EEB.Components.BaseComponents.Water.Sources.WaterSource_PT

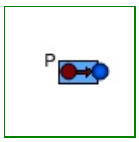


Connectors

Type	Name	Description
input RealInput	Pin	
input RealInput	Tin	
WaterFlange	water_flange	

Modelica definition

```
model WaterSource_PT
  Media.Substances.SubcooledWater water;
  Modelica.Blocks.Interfaces.RealInput Pin;
  Modelica.Blocks.Interfaces.RealInput Tin;
  Interfaces.Water.WaterFlange water_flange;
equation
  water.p = water_flange.p;
  water.p = Pin;
  water.T = Tin;
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSource_PT;
```



EEB.Components.BaseComponents.Water.Sources.WaterSource_PT_fixed

Parameters

Type	Name	Default	Description
Pressure	P0	101325	[Pa]
Temperature	T0	273.15 + 25	[K]

Connectors

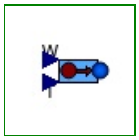
Type	Name	Description
WaterFlange	water_flange	

Modelica definition

```
model WaterSource_PT_fixed
  Media.Substances.SubcooledWater water;
  parameter Pressure P0 = 101325;
  parameter Temperature T0 = 273.15 + 25;
  Interfaces.Water.WaterFlange water_flange;
equation
```

```
  water.p = water_flange.p;
  water.p = P0;
  water.T = T0;
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSource_PT_fixed;
```

EEB.Components.BaseComponents.Water.Sources.WaterSource_WT

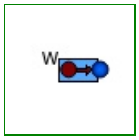


Connectors

Type	Name	Description
input RealInput	Win	
input RealInput	Tin	
WaterFlange	water_flange	

Modelica definition

```
model WaterSource_WT
  Media.Substances.SubcooledWater water;
  Modelica.Blocks.Interfaces.RealInput Win;
  Modelica.Blocks.Interfaces.RealInput Tin;
  Interfaces.Water.WaterFlange water_flange;
equation
  water_flange.w = -Win;
  water.T = Tin;
  water.p = water_flange.p;
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSource_WT;
```



EEB.Components.BaseComponents.Water.Sources.WaterSource_WT_fixed

Parameters

Type	Name	Default	Description
Temperature	T0	273.15 + 25	[K]
MassFlowRate	W0		[kg/s]

Connectors

Type	Name	Description
WaterFlange	water_flange	

Modelica definition

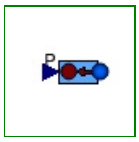
```
model WaterSource_WT_fixed
  Media.Substances.SubcooledWater water;
  parameter Temperature T0 = 273.15 + 25;
  parameter MassFlowRate W0;
  Interfaces.Water.WaterFlange water_flange;
equation
  water_flange.w = -W0;
  water.T = T0;
  water.p = water_flange.p;
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSource_WT_fixed;
```

EEB.Components.BaseComponents.Water.Sinks

Package Content

Name	Description
WaterSink_P	
WaterSink_P_fixed	
WaterSink_W	
WaterSink_W_fixed	

EEB.Components.BaseComponents.Water.Sinks.WaterSink_P



Pin •

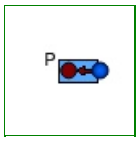
Connectors

Type	Name	Description
input RealInput	Pin	
WaterFlange	water_flange	

Modelica definition

```
model WaterSink_P
  Media.Substances.SubcooledWater water;
  Modelica.Blocks.Interfaces.RealInput Pin;
  Interfaces.Water.WaterFlange water_flange;
equation
  water.p = water_flange.p;
  water.p = Pin;
  water_flange.h = inStream(water_flange.h);
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSink_P;
```

EEB.Components.BaseComponents.Water.Sinks.WaterSink_P_fixed



•

Parameters

Type	Name	Default	Description
Pressure	Pin	101325	[Pa]

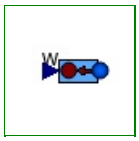
Connectors

Type	Name	Description
WaterFlange	water_flange	

Modelica definition

```
model WaterSink_P_fixed
  Media.Substances.SubcooledWater water;
  parameter Pressure Pin = 101325;
  Interfaces.Water.WaterFlange water_flange;
equation
  water.p = water_flange.p;
  water.p = Pin;
  water_flange.h = inStream(water_flange.h);
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSink_P_fixed;
```

EEB.Components.BaseComponents.Water.Sinks.WaterSink_W



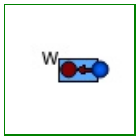
Connectors

Type	Name	Description
input RealInput	Win	
WaterFlange	water_flange	

Modelica definition

```
model WaterSink_W
  Media.Substances.SubcooledWater water;
  Modelica.Blocks.Interfaces.RealInput Win;
  Interfaces.Water.WaterFlange water_flange;
equation
  water.p = water_flange.p;
  water_flange.w = Win;
  water_flange.h = inStream(water_flange.h);
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSink_W;
```

EEB.Components.BaseComponents.Water.Sinks.WaterSink_W_fixed



Parameters

Type	Name	Default	Description
MassFlowRate	Win	0.001	[kg/s]

Connectors

Type	Name	Description
WaterFlange	water_flange	

Modelica definition

```
model WaterSink_W_fixed
  Media.Substances.SubcooledWater water;
  parameter MassFlowRate Win = 0.001;
  Interfaces.Water.WaterFlange water_flange;
equation
  water.p = water_flange.p;
  water_flange.w = Win;
  water_flange.h = inStream(water_flange.h);
  // enthalpy boundary condition
  water.h = water_flange.h;
end WaterSink_W_fixed;
```

EEB.Components.BaseComponents.Water.Pressurisers

Package Content

Name	Description
IdealWaterPressuriser	
IdealWaterPressuriser_pfixed	

EEB.Components.BaseComponents.Water.Pressurisers.IdealWaterPressuriser

Information

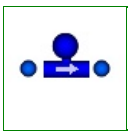
Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
input RealInput	p	

Modelica definition

```
model IdealWaterPressuriser
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  Pressure p0;
  Modelica.Blocks.Interfaces.RealInput p;
equation
  p0 = p;
  // impose pressure on both connectors
  p1 = p0;
  p2 = p0;
  water.p = 0.5 * (p1 + p2);
  // mean fluid pressure
  // energy conservation
  hout1 = inStream(water_flange2.h);
  hout2 = inStream(water_flange1.h);
  // enthalpy boundary condition
  hout1 = water.h;
end IdealWaterPressuriser;
```



EEB.Components.BaseComponents.Water.Pressurisers.IdealWaterPressuriser_pfixed

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Pressure	Pref	101325	reference pressure [Pa]

Connectors

Type	Name	Description

WaterFlange	water_flange2	
WaterFlange	water_flange1	

Modelica definition

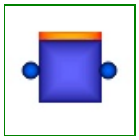
```
model IdealWaterPressuriser_pfixed
  extends Interfaces.Water.PartialTwoPort_water;
  parameter Pressure Pref = 101325 "reference pressure";
  Pressure p0;
equation
  p0 = Pref;
  // impose pressure on both connectors
  p1 = p0;
  p2 = p0;
  // mean fluid pressure
  // energy conservation
  hout1 = inStream(water_flange2.h);
  hout2 = inStream(water_flange1.h);
  // enthalpy boundary condition
end IdealWaterPressuriser_pfixed;
```

EEB.Components.BaseComponents.Water.Volumes

Package Content

Name	Description
WaterVolume	Tank with wall

EEB.Components.BaseComponents.Water.Volumes.WaterVolume



Tank with wall

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Volume	V	1	volume [m3]
Temperature	Tstart	273.15 + 25	initial temp [K]

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
HeatPort	fluid	

Modelica definition

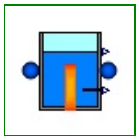
```
model WaterVolume "Tank with wall"
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter Volume V = 1 "volume";
  parameter Temperature Tstart = 273.15 + 25 "initial temp";
  Temperature Tf(start=Tstart,stateSelect=StateSelect.prefer) "fluid temperature";
  Energy E;
  Interfaces.Thermal.HeatPort fluid;
equation
  // Total mass balance
  w1 + w2 = 0;
  // No pressure drop
  p1 = p2;
  // fluid conditions
  water.p = p1;
  water.T = Tf;
  // Energy balance for the fluid
  E = V * water.d * water.h;
  der(E) = w1 * actualStream(water_flange1.h)
           + w2 * actualStream(water_flange2.h) + fluid.Q_flow;
  fluid.T = Tf;
  // Enthalpies boundary condition
  hout1 = water.h;
  hout2 = water.h;
end WaterVolume;
```

EEB.Components.BaseComponents.Water.Tanks

Package Content

Name	Description
Tank_adiabaticWall	Tank with wall
Tank_exchangingWall	Tank with wall

EEB.Components.BaseComponents.Water.Tanks.Tank_adiabaticWall



Tank with wall

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Volume	V	1	max contained water volume [m3]
Volume	Vstart	0.001	initial water volume [m3]
Area	S	1	exchanging surface (inner and outer) [m2]
Area	Sb	0.2	base area for level/pressure [m2]
Temperature	Tstart	273.15 + 25	initial temp (fluid and metal) [K]
Pressure	pin	0	mean pressure at the flanges(used in case of a boiler) [Pa]

Connectors

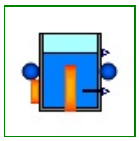
Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
HeatPort	fluid	
output RealOutput	oM	Mass of water
output RealOutput	oT	Temperature of water

Modelica definition

```
model Tank_adiabaticWall "Tank with wall"
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter Volume V = 1 "max contained water volume";
  parameter Volume Vstart = 0.001 "initial water volume";
  parameter Area S = 1 "exchanging surface (inner and outer)";
  parameter Area Sb = 0.2 "base area for level/pressure";
  parameter Temperature Tstart = 273.15 + 25 "initial temp (fluid and metal)";
  parameter Pressure pin = 0 "mean pressure at the flanges(used in case of a boiler)";
  Temperature Tf(start=Tstart,stateSelect=StateSelect.prefer) "fluid temperature";
  Mass Mw(start = Vstart * 1000) "Mass of water";
  Pressure p "preassure of water at input, inside and output";
  Energy E;
  Interfaces.Thermal.HeatPort fluid;
  Modelica.Blocks.Interfaces.RealOutput oM "Mass of water";
  Modelica.Blocks.Interfaces.RealOutput oT "Temperature of water";
equation
  // Total mass balance
  der(Mw) = w1 + w2;
  // No pressure drop
  p = Mw / Sb * Modelica.Constants.g_n + pin;
  p1 = p;
  p1 = p2;
  water.p =p;
  // Energy balances for the fluid
  der(E) = w1 * actualStream(water_flange1.h) + w2 * actualStream(water_flange2.h) + fluid.Q_flow;
  E = Mw * water.h;
```

```
  // Temperature
  water.T = Tf;
  fluid.T = Tf;
  //Output
  oM = Mw;
  oT = Tf;
  // Enthalpies boundary condition
  hout1 = water.h;
  hout2 = water.h;
end Tank_adiabaticWall;
```

EEB.Components.BaseComponents.Water.Tanks.Tank_exchangingWall



Tank with wall

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Volume	V	1	max contained water volume [m3]
Volume	Vstart	0.001	initial water volume [m3]
Area	S	1	exchanging surface (inner and outer) [m2]
Area	Sb	0.2	base area for level/pressure BEFORE Sb=0.01 [m2]
Mass	Mm	10	metal mass [kg]
SpecificHeatCapacity	cm	500	metal specific heat [J/(kg.K)]
CoefficientOfHeatTransfer	gwm	5	heat transfer coeff (fluid/metal) [W/(m2.K)]
Temperature	Tstart	273.15 + 25	initial temp (fluid and metal) [K]
Pressure	pin	0	mean pressure at the flanges(used in case of a boiler) [Pa]

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
HeatPort	fluid	
HeatPort	wall	
output RealOutput	oM	Mass of water
output RealOutput	oT	Temperature of water

Modelica definition

```
model Tank_exchangingWall "Tank with wall"
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter Volume V = 1 "max contained water volume";
  parameter Volume Vstart = 0.001 "initial water volume";
  parameter Area S = 1 "exchanging surface (inner and outer)";
  parameter Area Sb = 0.2 "base area for level/pressure BEFORE Sb=0.01";
  parameter Mass Mm = 10 "metal mass";
  parameter SpecificHeatCapacity cm = 500 "metal specific heat";
  parameter CoefficientOfHeatTransfer gwm = 5 "heat transfer coeff (fluid/metal)";
  parameter Temperature Tstart = 273.15 + 25 "initial temp (fluid and metal)";
  parameter Pressure pin = 0 "mean pressure at the flanges(used in case of a boiler)";
  HeatFlowRate Qwm "water to metal heat rate";
  Temperature Tf(start=Tstart,stateSelect=StateSelect.prefer) "fluid temperature";
  Temperature Tm(start=Tstart,stateSelect=StateSelect.prefer) "wall temperature";
  Mass Mw(start = Vstart * 1000) "Mass of water";
  Pressure p "preassure of water at input, inside and output";
  Energy E;
  Interfaces.Thermal.HeatPort fluid;
  Interfaces.Thermal.HeatPort wall;
  Modelica.Blocks.Interfaces.RealOutput oM "Mass of water";
  Modelica.Blocks.Interfaces.RealOutput oT "Temperature of water";
```

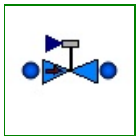
```
equation
  // Total mass balance
  der(Mw) = w1 + w2;
  // No pressure drop
  p = Mw / Sb * Modelica.Constants.g_n + pin;
  p1 = p;
  p1 = p2;
  water.p = 0.5 * (p1 + p2);
  // Energy balances for the fluid
  der(E) = w1 * actualStream(water_flange1.h) + w2 * actualStream(water_flange2.h) + Qwm + fluid.Q_flow;
  E = Mw * water.h;
  Qwm = gwm * S * (Tm - Tf);
  // Energy balances for metal
  Mm * cm * der(Tm) = (-Qwm) + wall.Q_flow;
  // Temperature
  water.T = Tf;
  fluid.T = Tf;
  wall.T = Tm;
  //Output
  oM = Mw;
  oT = Tf;
  // Enthalpies boundary condition
  hout1 = water.h;
  hout2 = water.h;
end Tank_exchangingWall;
```

EEB.Components.BaseComponents.Water.Valves

Package Content

Name	Description
WaterValve_LinChar	
WaterValve3Ways_LinChar	

EEB.Components.BaseComponents.Water.Valves.WaterValve_LinChar



Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Real	cvmax	0.01	kg/s/sqrt(Pa)
Time	T	1	positioner TC [s]
Real	xstart	0	initial position

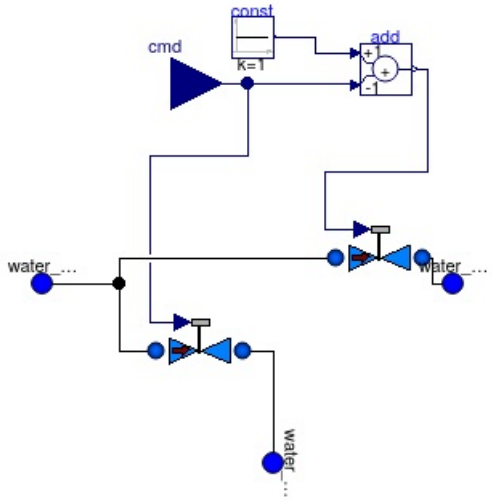
Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
input RealInput	cmd	

Modelica definition

```
model WaterValve_LinChar
  import EEB.Functions.*;
  extends Interfaces.Water.PartialTwoPort_water;
  parameter Real cvmax = 0.01 "kg/s/sqrt(Pa)";
  parameter Time T = 1 "positioner TC";
  parameter Real xstart = 0 "initial position";
  Real x(start = xstart, fixed = true);
  Modelica.Blocks.Interfaces.RealInput cmd;
equation
  // dynamic aperture
  x + T * der(x) = cmd;
  // no mass storage inside the valve
  w1 + w2 = 0;
  // relationship between pressure drop and mass flow rate
  w1 = if p1 >= p2 then cvmax * x * sqrtReg(p1 - p2,1e5) else -cvmax * x * sqrtReg(p2 - p1,1e5);
  // energy conservation
  hout1 = inStream(water_flange2.h);
  hout2 = inStream(water_flange1.h);
end WaterValve_LinChar;
```

EEB.Components.BaseComponents.Water.Valves.WaterValve3Ways_LinChar



Parameters

Type	Name	Default	Description
Real	cvmax	0.01	kg/s/sqrt(Pa)
Time	Tc	1	positioner TC [s]
Real	xstart	0	initial position

Connectors

Type	Name	Description
input RealInput	cmd	0=i->1,1=i->2
WaterFlange	water_flange1	
WaterFlange	water_flange2	
WaterFlange	water_flange3	

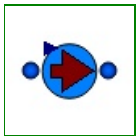
Modelica definition

```
model WaterValve3Ways_LinChar
  parameter Real cvmax = 0.01 "kg/s/sqrt(Pa)";
  parameter Time Tc = 1 "positioner TC";
  parameter Real xstart = 0 "initial position";
  WaterValve_LinChar valveLinChar1(cvmax = cvmax, xstart = 1 - xstart, T = Tc);
  Modelica.Blocks.Interfaces.RealInput cmd "0=i->1,1=i->2";
  Modelica.Blocks.Math.Add add(k2 = -1);
  Modelica.Blocks.Sources.Constant const(k = 1);
  WaterValve_LinChar valveLinChar2(cvmax = cvmax, xstart = xstart, T = Tc);
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
  Interfaces.Water.WaterFlange water_flange3;
equation
  connect(const.y, add.u1);
  connect(cmd, add.u2);
  connect(add.y, valveLinChar1.cmd);
  connect(valveLinChar2.cmd, add.u2);
  connect(valveLinChar2.water_flange2, water_flange3);
  connect(valveLinChar1.water_flange2, water_flange2);
  connect(valveLinChar1.water_flange1, water_flange1);
  connect(valveLinChar2.water_flange1, water_flange1);
end WaterValve3Ways_LinChar;
```

EEB.Components.BaseComponents.Water.Pumps

Package Content

Name	Description
WaterPump_Volumetric	
WaterPump_Centrifugal	



EEB.Components.BaseComponents.Water.Pumps.WaterPump_Volumetric

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
VolumeFlowRate	qmax	0.0005	volume flowrate at cmd=1 [m3/s]
VolumeFlowRate	qstart	0	[m3/s]
Time	T	0.5	pump TC [s]

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
input RealInput	cmd	

Modelica definition

```
model WaterPump_Volumetric
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter VolumeFlowRate qmax = 0.0005 "volume flowrate at cmd=1";
  parameter VolumeFlowRate qstart = 0;
  parameter Time T = 0.5 "pump TC";
  VolumeFlowRate q(start = qstart);
  Modelica.Blocks.Interfaces.RealInput cmd;
equation
  // no mass storage inside the pump
  w1 + w2 = 0;
  // pump dynamics
  q + T * der(q) = cmd * qmax;
  // variable mass flow rate
  w1 = water.d * q;
  water.p = 0.5 * (p1 + p2);
  // mean fluid pressure
  // energy conservation
  hout1 = inStream(water_flange2.h);
  hout2 = inStream(water_flange1.h);
  // enthalpy boundary condition
  hout2 = water.h;
end WaterPump_Volumetric;
```

EEB.Components.BaseComponents.Water.Pumps.WaterPump_Centrifugal

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Pressure	DP0	5000	DP at w=0, cmd=1 [Pa]
Real	DP1	200	pressure decr. ratio, cmd=1 Pa/(kg/s)^2
Time	T	0.5	pump TC [s]
Real	cmdstart	0	

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
input RealInput	cmd	

Modelica definition

```
model WaterPump_Centrifugal
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter Pressure DP0 = 5000 "DP at w=0, cmd=1";
  parameter Real DP1 = 200 "pressure decr. ratio, cmd=1 Pa/(kg/s)^2";
  parameter Time T = 0.5 "pump TC";
  parameter Real cmdstart = 0;
  Real x(start = cmdstart);
  Modelica.Blocks.Interfaces.RealInput cmd;
equation
  // no mass storage inside the pump
  w1 + w2 = 0;
  // command dynamics
  x + T * der(x) = cmd;
  // pressure drop
  p2 - p1 = DP0 * x - DP1 * x * w1*noEvent(abs(w1));
  water.p = 0.5 * (p1 + p2);
  // mean fluid pressure
  // energy conservation
  hout1 = inStream(water_flange2.h);
  hout2 = inStream(water_flange1.h);
  // enthalpy boundary condition
  hout1 = water.h;
end WaterPump_Centrifugal;
```

EEB.Components.BaseComponents.Water.Pipes

Package Content

Name	Description
BaseClasses	
WaterPipeExchanging_Nvols	

EEB.Components.BaseComponents.Water.Pipes.WaterPipeExchanging_Nvols

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Integer	n	2	number of lumps
Length	Ltube	30	tube length [m]
Length	Dtube	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
Temperature	Tstart	273.15 + 25	initial fluid temp [K]

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
HeatPortVec	heatPort	

Modelica definition

```
model WaterPipeExchanging_Nvols
  extends Interfaces.Water.PartialTwoPort_water;
  parameter Integer n = 2 "number of lumps";
  parameter Length Ltube = 30 "tube length";
  parameter Length Dtube = 0.05 "tube inner diameter";
  parameter Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter Temperature Tstart = 273.15 + 25 "initial fluid temp";
  EEB.Components.BaseComponents.Water.Pipes.BaseClasses.WaterPipeExchangingElement
                                                                    pipe[n](each L = Ltube / n, each D = Dtube, each Dz = Dz / n, each Cf = Cftube, each Tstart = Tstart)
  EEB.Utilities.HPvector_VectOfHPs hps(n = n);
  Temperature T[n];
  Interfaces.Thermal.HeatPortVec heatPort(n = n);
equation
  connect(water_flange1, pipe[1].water_flange1);
  connect(pipe[n].water_flange2, water_flange2);
  connect(pipe[1].heatPort, hps.vecOfHPs[1]);
  T[1] = pipe[1].Ti;
  for i in 2:n loop
    connect(pipe[i - 1].water_flange2, pipe[i].water_flange1);
    connect(pipe[i].heatPort, hps.vecOfHPs[i]);
    T[i] = pipe[i].Ti;
  end for;
  connect(hps.vecHP, heatPort);
end WaterPipeExchanging_Nvols;
```

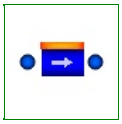
EEB.Components.BaseComponents.Water.Pipes.BaseClasses

Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
WaterPipeExchangingElement	



EEB.Components.BaseComponents.Water.Pipes.BaseClasses.WaterPipeExchangingElement

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Length	L	10	Pipe length [m]
Length	D	0.05	Pipe Diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cf	1e-6	friction coefficient
Temperature	Tstart	273.15 + 25	initial fluid temperature [K]

Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
HeatPort	heatPort	

Modelica definition

```
model WaterPipeExchangingElement
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater water;
  parameter Length L = 10 "Pipe length";
  parameter Length D = 0.05 "Pipe Diameter";
  parameter Length Dz = 0 "height diff (out-in)";
  parameter Real Cf = 1e-6 "friction coefficient";
  parameter Temperature Tstart = 273.15 + 25 "initial fluid temperature";
  Energy Ew(stateSelect = StateSelect.avoid);
  Temperature Ti(start = Tstart, stateSelect = StateSelect.always) "fluid temperature";
  Velocity u "fluid speed";
  Interfaces.Thermal.HeatPort heatPort;
protected
  final parameter Area S = pi * (D / 2) ^ 2;
  final parameter Volume V = S * L;

equation
  // no mass storage inside the component
  w1 + w2 = 0;
  // pressure drop
  p1 - p2 = 64 * L * Cf / (water.d * pi ^ 2 * D ^ 5) * w1 * noEvent(abs(w1)) + water.d * g_n * Dz;
  water.p = 0.5 * (p1 + p2);
  // mean fluid pressure
  //energy conservation
  der(Ew) = w1 * actualStream(water_flange1.h) + w2 * actualStream(water_flange2.h) + heatPort.Q_flow;
  Ew = water.d * V * water.h;
  // Energy = mass * specific enthalpy
  //temperature
  water.T = Ti;
  heatPort.T = Ti;
  // heat port temperature
  // enthalpies boundary condition
  hout1 = water.h;
  hout2 = water.h;
  u = w1/S/water.d;
end WaterPipeExchangingElement;
```

EEB.Components.BaseComponents.Thermal

Package with heat transfer elements

Information

This package contains models for heat transfer elements.

Package Content

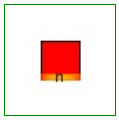
Name	Description
Capacities	
HeatTransfer	
Sources	
Sinks	

EEB.Components.BaseComponents.Thermal.Sources

Package Content

Name	Description
FixedHeatFlowVecDivided	Q_flow divided on each heatPort of the vector
FixedHeatFlowVec	
SolarRadiation_TransparentSurf	
SolarRadiation_OpaqueSurf	

EEB.Components.BaseComponents.Thermal.Sources.FixedHeatFlowVecDivided



Q_flow divided on each heatPort of the vector

Parameters

Type	Name	Default	Description
HeatFlowRate	Q_flow		Fixed heat flow rate at port [W]
Integer	n		number of heatPort

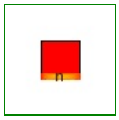
Connectors

Type	Name	Description
HeatPortVec	heatPortN	

Modelica definition

```
model FixedHeatFlowVecDivided "Q_flow divided on each heatPort of the vector"
  parameter HeatFlowRate Q_flow "Fixed heat flow rate at port";
  parameter Integer n "number of heatPort";
  Interfaces.Thermal.HeatPortVec heatPortN(n = n);
equation
  for i in 1:n loop
    heatPortN.Q_flow[i] = -Q_flow / n;
  end for;
end FixedHeatFlowVecDivided;
```

EEB.Components.BaseComponents.Thermal.Sources.FixedHeatFlowVec



Parameters

Type	Name	Default	Description
HeatFlowRate	Q_flow[n]		[W]
Integer	n		number of heatPort

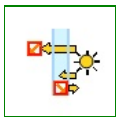
Connectors

Type	Name	Description
HeatPortVec	heatPortN	

Modelica definition

```
model FixedHeatFlowVec
  parameter HeatFlowRate Q_flow[n];
  parameter Integer n "number of heatPort";
  Interfaces.Thermal.HeatPortVec heatPortN(n = n);
equation
  for i in 1:n loop
    heatPortN.Q_flow[i] = -Q_flow[i];
  end for;
end FixedHeatFlowVec;
```

EEB.Components.BaseComponents.Thermal.Sources.SolarRadiation_TransparentSurf



Parameters

Type	Name	Default	Description
Length	L	5	Length of the glass [m]

Length	H	3	Height of the glass [m]
Area	S	L*H	Surface of the glass [m2]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	inclination	90	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.01	Absorbed coefficient
Real	trasCoef	0.9	Trasmitted coefficient

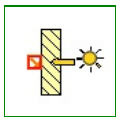
Connectors

Type	Name	Description
HeatPortRad	Trasmitted	
HeatPortRad	Absorbed	

Modelica definition

```
model SolarRadiation_TransparentSurf
  parameter Length L = 5 "Length of the glass";
  parameter Length H = 3 "Height of the glass";
  parameter Area S = L * H "Surface of the glass";
  parameter Real orientation(min = 0, max = 360) = 0
    "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real inclination(min = 0, max = 180) = 90
    "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.01 "Absorbed coefficient";
  parameter Real trasCoef = 0.9 "Trasmitted coefficient";
  // match default comp name in AmbientSettings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  HeatFlowRate Q_rad;
  Real alpha "Orietation coefficient";
  Real beta "Inclination coefficient";
  HeatFlux solarRad_eff;
  Power Qreflected;
  Real az;
  Real ze;
  Interfaces.Thermal.HeatPortRad Trasmitted;
  Interfaces.Thermal.HeatPortRad Absorbed;
equation
  az = ambient_settings.azimuth;
  ze = ambient_settings.zenith;
// FIXME check
// ORIGINAL
// alpha = max(0,Modelica.Math.cos((ambient_settings.azimuth-orientation)/180*Modelica.Constants.pi));
// beta = max(0,Modelica.Math.cos((ambient_settings.zenith-(inclination-90))/180*Modelica.Constants.pi));
  alpha = max(0, Modelica.Math.cos((az - (orientation - 90)) / 180 * Modelica.Constants.pi));
  beta = max(0, Modelica.Math.cos((ze - inclination) / 180 * Modelica.Constants.pi));
  if ze <= eps then
    solarRad_eff = 0;
  elseif not (inclination < 180 or inclination > 180) then
    solarRad_eff = sin(ze / 180 * Modelica.Constants.pi);
  else
    solarRad_eff = alpha * beta;
  end if;
// solarRad_eff = ambient_settings.solarRad*(if inclination==180 then sin(ambient_settings.zenith/180*Modelica.Constants.pi) else alpha*beta);
  Q_rad = solarRad_eff * S * ambient_settings.solarRad;
  Absorbed.Q_flow = -absCoef * Q_rad;
  Trasmitted.Q_flow = -trasCoef * Q_rad;
  Absorbed.Q_flow + Trasmitted.Q_flow + Qreflected+ Q_rad = 0;
end SolarRadiation_TransparentSurf;
```

EEB.Components.BaseComponents.Thermal.Sources.SolarRadiation_OpaqueSurf



Parameters

Type	Name	Default	Description
Length	L	5	Length of the body [m]
Length	H	3	Height of the body [m]
Area	S	L*H	Surface of the body [m2]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient

Connectors

Type	Name	Description
HeatPortRad	Absorbed	

Modelica definition

```
model SolarRadiation_OpaqueSurf
  parameter Length L = 5 "Length of the body";
  parameter Length H = 3 "Height of the body";
```

```
  parameter Area S = L * H "Surface of the body";
  parameter Real orientation(min = 0, max = 360) = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real inclination(min = 0, max = 180) = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  // match default comp name in Ambientambient_settings class
  outer BoundaryConditions.AmbientConditions ambient_settings;
  HeatFlowRate Q_rad;
  Real alpha "Orietation coefficient";
  Real beta "Inclination coefficient";
  HeatFlux solarRad_eff;
  Real az;
  Real ze;
  Interfaces.Thermal.HeatPortRad Absorbed;
equation
  az = ambient_settings.azimuth;
  ze = ambient_settings.zenith;
  // FIXME check
  // ORIGINAL
  // alpha = max(0,Modelica.Math.cos((ambient_settings.azimuth-orientation)/180*Modelica.Constants.pi));
  // beta = max(0,Modelica.Math.cos((ambient_settings.zenith-(inclination-90))/180*Modelica.Constants.pi));
  alpha = max(0, Modelica.Math.cos((az - (orientation - 90)) / 180 * Modelica.Constants.pi));
  beta = max(0, Modelica.Math.cos((ze - inclination) / 180 * Modelica.Constants.pi));
  if ze <= eps then
    solarRad_eff = 0;
  elseif inclination >= 180 then
    solarRad_eff = sin(ze / 180 * Modelica.Constants.pi);
  else
    solarRad_eff = alpha * beta;
  end if;
  // solarRad_eff = ambient_settings.solarRad*(if inclination==180 then sin(ambient_settings.zenith/180*Modelica.Constants.pi) else alpha*beta);
  Q_rad = solarRad_eff * S * ambient_settings.solarRad;
  Absorbed.Q_flow = -absCoef * Q_rad;
end SolarRadiation_OpaqueSurf;
```

EEB.Components.BaseComponents.Thermal.Sinks

Package Content

Name	Description
PrescribedHeatLossesSink	

EEB.Components.BaseComponents.Thermal.Sinks.PrescribedHeatLossesSink

Connectors

Type	Name	Description
HeatPort	port	

Modelica definition

```
model PrescribedHeatLossesSink
  Interfaces.Thermal.HeatPort port;
equation
  port.T = 0;
  // Do-not-care value
end PrescribedHeatLossesSink;
```

EEB.Components.BaseComponents.Thermal.Capacities

Package Content

Name	Description
MassT	Mass of a generic element
ThermalCap	Mass of a generic element

EEB.Components.BaseComponents.Thermal.Capacities.MassT

Mass of a generic element

Parameters

Type	Name	Default	Description
Mass	M	10	[kg]
SpecificHeatCapacity	cp	500	[J/(kg.K)]
Temperature	Tstart	273.15 + 20	[K]

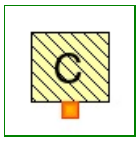
Connectors

Type	Name	Description
HeatPort	surf	

Modelica definition

```
model MassT "Mass of a generic element"
  parameter Mass M = 10;
  parameter SpecificHeatCapacity cp = 500;
  parameter Temperature Tstart = 273.15 + 20;
  Temperature T(start = Tstart);
  EEB.Interfaces.Thermal.HeatPort surf;
equation
  cp * M * der(T) = surf.Q_flow;
  surf.T = T;
end MassT;
```

EEB.Components.BaseComponents.Thermal.Capacities.ThermalCap



Mass of a generic element

Parameters

Type	Name	Default	Description
HeatCapacity	C	500	[J/K]
Temperature	Tstart	273.15 + 20	[K]

Connectors

Type	Name	Description
HeatPort	surf	

Modelica definition

```
model ThermalCap "Mass of a generic element"
  parameter HeatCapacity C = 500;
  parameter Temperature Tstart = 273.15 + 20;
  Temperature T(start = Tstart);
  EEB.Interfaces.Thermal.HeatPort surf;
equation
  C * der(T) = surf.Q_flow;
  surf.T = T;
end ThermalCap;
```

EEB.Components.BaseComponents.Thermal.HeatTransfer

Package Content

Name	Description
Conduction_SS	
Convection_SS	Scalar to scalar convection, fixed heat transfer coefficient
Convection_SS_gammaIn	Scalar to scalar convection, externally provided heat transfer coefficient
Convection_SV	Scalar to vector convection, fixed heat transfer coefficient
Convection_SV_gammaIn	Scalar to vector convection, externally provided heat transfer coefficient
Convection_VV	Vector to vector convection, fixed heat transfer coefficient
Convection_VV_gammaIn	Vector to vector convection, externally provided heat transfer coefficient
Convection_Wall2air_BI74	
Convection_Wall2air_BRIS	
Convection_Wall2Ext_Clarke	
PrescribedHeatFlowThrough	

EEB.Components.BaseComponents.Thermal.HeatTransfer.Conduction_SS

Parameters

Type	Name	Default	Description
Area	A	1	total surf, both sides [m2]
Length	d	0.2	thickness [m]
ThermalConductivity	lambda	5	heat transfer coefficient [W/(m.K)]

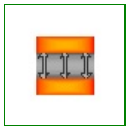
Connectors

Type	Name	Description
HeatPort	ss1	
HeatPort	ss2	

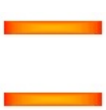
Modelica definition

```
model Conduction_SS
  parameter Area A = 1 "total surf, both sides";
  parameter Length d = 0.2 "thickness";
  parameter ThermalConductivity lambda = 5 "heat transfer coefficient";
  Interfaces.Thermal.HeatPort ss1;
  Interfaces.Thermal.HeatPort ss2;
equation
  ss1.Q_flow + ss2.Q_flow = 0;
  ss1.Q_flow = lambda * A * (ss1.T - ss2.T) / d;
end Conduction_SS;
```

EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS



Scalar to scalar convection, fixed heat transfer coefficient



Parameters

Type	Name	Default	Description
Area	S	1	Area, both sides [m2]
CoefficientOfHeatTransfer	gamma	5	Heat transfer coefficient [W/(m2.K)]

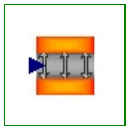
Connectors

Type	Name	Description
HeatPort	ss1	Scalar side 1

HeatPort	ss2	Scalar side 2

Modelica definition

```
model Convection_SS "Scalar to scalar convection, fixed heat transfer coefficient"
  parameter Area S = 1 "Area, both sides";
  parameter CoefficientOfHeatTransfer gamma = 5 "Heat transfer coefficient";
  Interfaces.Thermal.HeatPort ss1 "Scalar side 1";
  Interfaces.Thermal.HeatPort ss2 "Scalar side 2";
equation
  ss1.Q_flow + ss2.Q_flow = 0;
  ss1.Q_flow = gamma * S * (ss1.T - ss2.T);
end Convection_SS;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS_gammaIn

Scalar to scalar convection, externally provided heat transfer coefficient

Parameters

Type	Name	Default	Description
Area	S	1	Area, both sides [m2]

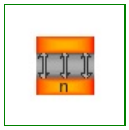
Connectors

Type	Name	Description
input RealInput	gamma	
HeatPort	ss1	Scalar side 1
HeatPort	ss2	Scalar side 2

Modelica definition

```
model Convection_SS_gammaIn "Scalar to scalar convection, externally provided heat transfer coefficient"
  parameter Area S = 1 "Area, both sides";
  Modelica.Blocks.Interfaces.RealInput gamma;
  Interfaces.Thermal.HeatPort ss1 "Scalar side 1";
  Interfaces.Thermal.HeatPort ss2 "Scalar side 2";
equation
  ss1.Q_flow + ss2.Q_flow = 0;
  ss1.Q_flow = gamma * S * (ss1.T - ss2.T);
end Convection_SS_gammaIn;
```

EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SV



Scalar to vector convection, fixed heat transfer coefficient

Parameters

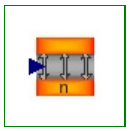
Type	Name	Default	Description
Integer	n	2	Lumps, vector side
Area	S	1	Area, both sides [m2]
CoefficientOfHeatTransfer	gamma	5	Heat transfer coefficient [W/(m2.K)]

Connectors

Type	Name	Description
HeatPort	ss	Scalar side
HeatPortVec	vs	Vector side

Modelica definition

```
model Convection_SV "Scalar to vector convection, fixed heat transfer coefficient"
  parameter Integer n = 2 "Lumps, vector side";
  parameter Area S = 1 "Area, both sides";
  parameter CoefficientOfHeatTransfer gamma = 5 "Heat transfer coefficient";
  Interfaces.Thermal.HeatPort ss "Scalar side";
  Interfaces.Thermal.HeatPortVec vs(n = n) "Vector side";
equation
  ss.Q_flow + sum(vs.Q_flow) = 0;
  for i in 1:n loop
    vs.Q_flow[i] = gamma * S / n * (vs.T[i] - ss.T);
  end for;
end Convection_SV;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SV_gammaIn

Scalar to vector convection, externally provided heat transfer coefficient

Parameters

Type	Name	Default	Description
Integer	n	2	Lumps, vector side
Area	S	1	Area, both sides [m2]

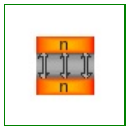
Connectors

Type	Name	Description
HeatPort	ss	Scalar side
HeatPortVec	vs	Vector side
input RealInput	gamma	

Modelica definition

```
model Convection_SV_gammaIn "Scalar to vector convection, externally provided heat transfer coefficient"
  parameter Integer n = 2 "Lumps, vector side";
  parameter Area S = 1 "Area, both sides";
  Interfaces.Thermal.HeatPort ss "Scalar side";
  Interfaces.Thermal.HeatPortVec vs(n = n) "Vector side";
  Modelica.Blocks.Interfaces.RealInput gamma;
equation
  ss.Q_flow + sum(vs.Q_flow) = 0;
  for i in 1:n loop
    vs.Q_flow[i] = gamma * S / n * (vs.T[i] - ss.T);
  end for;
end Convection_SV_gammaIn;
```

EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_VV



Vector to vector convection, fixed heat transfer coefficient

Parameters

Type	Name	Default	Description
Integer	n	2	Lumps, both sides
Area	S	1	Area, both sides [m2]
Boolean	counterFlow	false	Swap vectors for counterflow
CoefficientOfHeatTransfer	gamma	5	Heat transfer coefficient [W/(m2.K)]

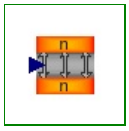
Connectors

Type	Name	Description
HeatPortVec	vs1	side vector

HeatPortVec	vs2	side vector

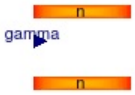
Modelica definition

```
model Convection_VV "Vector to vector convection, fixed heat transfer coefficient"
  parameter Integer n = 2 "Lumps, both sides";
  parameter Area S = 1 "Area, both sides";
  parameter Boolean counterFlow = false "Swap vectors for counterflow";
  parameter CoefficientOfHeatTransfer gamma = 5 "Heat transfer coefficient";
  Interfaces.Thermal.HeatPortVec vs1(n = n) "side vector";
  Interfaces.Thermal.HeatPortVec vs2(n = n) "side vector";
equation
  for i in 1:n loop
    if counterFlow then
      vs1.Q_flow[i] + vs2.Q_flow[n + 1 - i] = 0;
      vs1.Q_flow[i] = gamma * S / n * (vs1.T[i] - vs2.T[n + 1 - i]);
    else
      vs1.Q_flow[i] + vs2.Q_flow[i] = 0;
      vs1.Q_flow[i] = gamma + S / n * (vs1.T[i] - vs2.T[i]);
    end if;
  end for;
end Convection_VV;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_VV_gammaIn

Vector to vector convection, externally provided heat transfer coefficient



Parameters

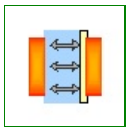
Type	Name	Default	Description
Integer	n	2	Lumps, both sides
Area	S	1	Area, both sides [m2]
Boolean	counterFlow	false	Swap vectors for counterflow

Connectors

Type	Name	Description
HeatPortVec	vs1	side vector
HeatPortVec	vs2	side vector
input RealInput	gamma	

Modelica definition

```
model Convection_VV_gammaIn "Vector to vector convection, externally provided heat transfer coefficient"
  parameter Integer n = 2 "Lumps, both sides";
  parameter Area S = 1 "Area, both sides";
  parameter Boolean counterFlow = false "Swap vectors for counterflow";
  Interfaces.Thermal.HeatPortVec vs1(n = n) "side vector";
  Interfaces.Thermal.HeatPortVec vs2(n = n) "side vector";
  Modelica.Blocks.Interfaces.RealInput gamma;
equation
  for i in 1:n loop
    if counterFlow then
      vs1.Q_flow[i] + vs2.Q_flow[n + 1 - i] = 0;
      vs1.Q_flow[i] = gamma * S / n * (vs1.T[i] - vs2.T[n + 1 - i]);
    else
      vs1.Q_flow[i] + vs2.Q_flow[i] = 0;
      vs1.Q_flow[i] = gamma + S / n * (vs1.T[i] - vs2.T[i]);
    end if;
  end for;
end Convection_VV_gammaIn;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74

Parameters

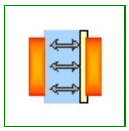
Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Boolean	vertical	true	true for vertical, false for horizontal

Connectors

Type	Name	Description
HeatPort	wall	
HeatPort	air	

Modelica definition

```
model Convection_Wall2air_BI74
  // Brown & Isfalt, 1974
  parameter Length L = 5 "wall surface length";
  parameter Length H = 3 "wall surface height";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  Interfaces.Thermal.HeatPort wall;
  Interfaces.Thermal.HeatPort air;
protected
  Modelica.SIunits.Conversions.NonSIunits.Temperature_degC DT;
  CoefficientOfHeatTransfer g;
equation
  DT = air.T - wall.T;
  if vertical then
    g = smooth(0, noEvent(if DT <= 0.0 then 0.55 else 0.55 + 1.4 * DT ^ 2));
  else
    g = smooth(0, noEvent(if DT <= 0.0 then 0.55 else 0.55 + 1.6 * DT ^ 2));
  end if;
  wall.Q_flow + air.Q_flow = 0;
  air.Q_flow = g * L * H * DT;
end Convection_Wall2air_BI74;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Real	a	3	
Real	c	0.1	
Real	n	0.4	

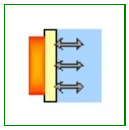
Connectors

Type	Name	Description
HeatPort	wall	
HeatPort	air	

Modelica definition

```
model Convection_Wall2air_BRIS
  // BRIS, Brown 1990
  parameter Length L = 5 "wall surface length";
  parameter Length H = 3 "wall surface height";
  parameter Real a = 3;
  parameter Real c = 0.1;
  parameter Real n = 0.4;
```

```
  Interfaces.Thermal.HeatPort wall;
  Interfaces.Thermal.HeatPort air;
protected
  Modelica.SIunits.Conversions.NonSIunits.Temperature_degC DT;
  CoefficientOfHeatTransfer g;
equation
  DT = air.T - wall.T;
  g = a + c * smooth(0, noEvent(abs(DT))) ^ n;
  wall.Q_flow + air.Q_flow = 0;
  air.Q_flow = g * L * H * DT;
end Convection_Wall2air_BRIS;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke

Parameters

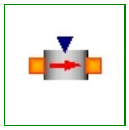
Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Boolean	fixedCoeff	false	switch the coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise

Connectors

Type	Name	Description
HeatPort	wall	

Modelica definition

```
model Convection_Wall2Ext_Clarke
  // Clarke, 1985
  parameter Length L = 5 "wall surface length";
  parameter Length H = 3 "wall surface height";
  parameter Boolean fixedCoeff = false "switch the coefficient";
  parameter CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation(min = 0, max = 360) = 0
    "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  outer BoundaryConditions.AmbientConditions ambient_settings;
  Interfaces.Thermal.HeatPort wall;
  HeatFlowRate Q_ext "Heat flow rate exchanged with the ambient";
  CoefficientOfHeatTransfer g;
  Boolean windward "T windward, F leeward";
  Real u;
equation
  windward = noEvent(abs(ambient_settings.wdir - orientation)) < 90;
  if windward and (ambient_settings.wv < 0 or ambient_settings.wv>0) then
    u = smooth(0, noEvent(if ambient_settings.wv < 2 then 0.5 else 0.25 * ambient_settings.wv));
  else
    u = 0.3 + 0.05 * ambient_settings.wv;
  end if;
  if u < 4.88 then
    g = 5.6780000000 * (1.09 + 0.23 * (u / 0.3048) ^ 1.00);
  else
    g = 0.5486590003 * (0.00 + 0.53 * (u / 0.3048) ^ 0.78);
  end if;
  wall.Q_flow + Q_ext = 0;
  if fixedCoeff then
    Q_ext = h0 * L * H * (ambient_settings.Tamb - wall.T);
  else
    Q_ext = g * L * H * (ambient_settings.Tamb - wall.T);
  end if;
end Convection_Wall2Ext_Clarke;
```



EEB.Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough

Connectors

Type	Name	Description
HeatPort	heatSrc	
HeatPort	heatSnk	
input RealInput	pwr	

Modelica definition

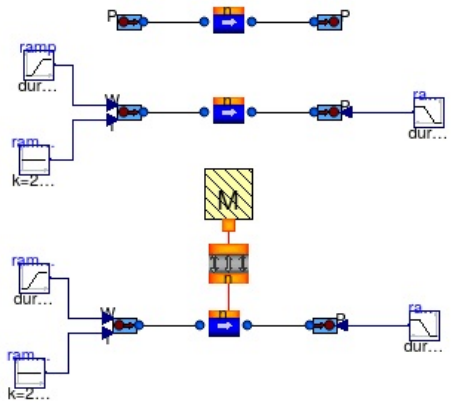
```
model PrescribedHeatFlowThrough
  Interfaces.Thermal.HeatPort heatSrc;
  Interfaces.Thermal.HeatPort heatSnk;
  Modelica.Blocks.Interfaces.RealInput pwr;
equation
  heatSrc.Q_flow + heatSnk.Q_flow = 0;
  heatSrc.Q_flow = pwr;
end PrescribedHeatFlowThrough;
```

EEB.Components.BaseComponents.Test

Package Content

Name	Description
test1_Pipe	
test2_Pipe	
test3_VolumetricPump	
test4_CentrifugalPump	
test5_Tank	
test6_AirVolume	
test7_AirVolumeWithWall_Condensing	
test8_HeatPump	
test9_AirRenovation	
test_Cell_VIchar	
test_Cell_series	
test_Cell	
testModule1	
Test_ControlledHeater	
Test_ControlledCooler	
Test_ControlledHandler	
TestPhasorSwitch	
TestRadiation	
TestDCloadI0	
TestDCloadACgen	
TestNaturalMix	
TestNaturalMix_Gcmd	
Test_LoadPcosphi_Pin	

EEB.Components.BaseComponents.Test.test1_Pipe

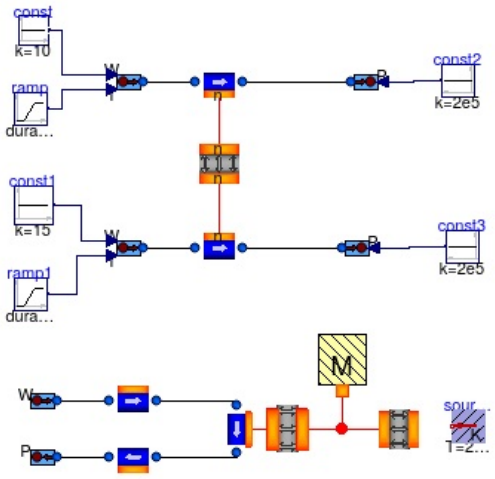


Modelica definition

```
model test1_Pipe
  Water.Sources.WaterSource_PT_fixed source_PT_fixed2(P0(displayUnit = "bar") = 100000, T0 = 273.15 + 25);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed2(Pin = 100000);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = 5, Ltube = 10, Dz = 1);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN1(n = 5, Ltube = 10);
  Water.Sources.WaterSource_WT source_WT;
  Water.Sinks.WaterSink_P sink_P;
  Modelica.Blocks.Sources.Ramp ramp(duration = 10);
  Modelica.Blocks.Sources.Ramp ramp1(height = 101325, duration = 10, startTime = 10);
  Modelica.Blocks.Sources.Constant ramp2(k = 273.15 + 25);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN2(n = 5, Ltube = 10);
  Water.Sources.WaterSource_WT source_WT1;
  Water.Sinks.WaterSink_P sink_P1;
  Modelica.Blocks.Sources.Ramp ramp3(duration = 10);
  Modelica.Blocks.Sources.Ramp ramp4(height = 101325, duration = 10, startTime = 10);
  Modelica.Blocks.Sources.Constant ramp5(k = 273.15 + 50);
  Thermal.Capacities.MassT massT(Tstart = 273.15 + 5);
  Thermal.HeatTransfer.Convection_SV convVec2Sca1(n = 5, S = 1, gamma = 40);
equation
  connect(ramp.y, source_WT.Win);
  connect(ramp1.y, sink_P.Pin);
```

```
  connect(ramp2.y, source_WT.Tin);
  connect(ramp3.y, source_WT1.Win);
  connect(ramp4.y, sink_P1.Pin);
  connect(ramp5.y, source_WT1.Tin);
  connect(massT.surf, convVec2Sca1.ss);
  connect(convVec2Sca1.vs, pipeExchangingN2.heatPort);
  connect(source_PT_fixed2.water_flange, pipeExchangingN.water_flange1);
  connect(pipeExchangingN.water_flange2, sink_P_fixed2.water_flange);
  connect(source_WT.water_flange, pipeExchangingN1.water_flange1);
  connect(pipeExchangingN1.water_flange2, sink_P.water_flange);
  connect(source_WT1.water_flange, pipeExchangingN2.water_flange1);
  connect(pipeExchangingN2.water_flange2, sink_P1.water_flange);
end test1_Pipe;
```

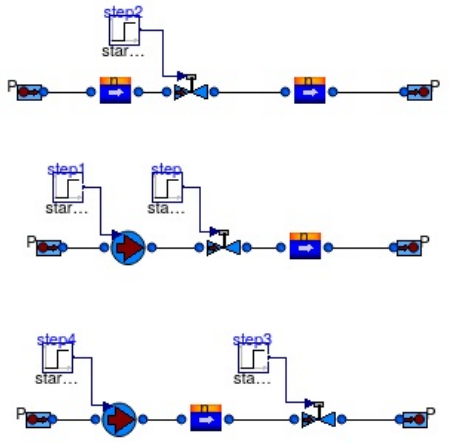
EEB.Components.BaseComponents.Test.test2_Pipe



Modelica definition

```
model test2_Pipe
  Water.Sources.WaterSource_WT source_WT;
  Water.Sources.WaterSource_WT source_WT1;
  Water.Sinks.WaterSink_P sink_P;
  Modelica.Blocks.Sources.Ramp ramp(height = 5, offset = 273.15 + 25, startTime = 100, duration = 0);
  Modelica.Blocks.Sources.Ramp ramp1(height = 0, duration = 2, offset = 273.15 + 50, startTime = 100);
  Modelica.Blocks.Sources.Constant const(k = 10);
  Modelica.Blocks.Sources.Constant const1(k = 15);
  Modelica.Blocks.Sources.Constant const2(k = 2e5);
  Water.Sinks.WaterSink_P sink_P1;
  Modelica.Blocks.Sources.Constant const3(k = 2e5);
  Thermal.HeatTransfer.Convection_VV convVec2Vec(counterFlow = false, n = 4, S = 1, gamma = 40);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = 4, Dtube = 0.1);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN1(n = 4);
  Thermal.HeatTransfer.Convection_SS convVec2Sca1(S = 1, gamma = 40);
  Water.Sources.WaterSource_WT_fixed source_PT_fixed1(T0 = 273.15 + 50, W0 = 0.1);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed1(Pin = 0);
  Thermal.Capacities.MassT massT(Tstart = 273.15 + 20);
  Thermal.HeatTransfer.Convection_SS convSca2Sca;
  Water.Pipes.BaseClasses.WaterPipeExchangingElement pipeExchanging;
  Water.Pipes.BaseClasses.WaterPipeExchangingElement pipeExchangingN3(Tstart = 273.15 + 50);
  Water.Pipes.BaseClasses.WaterPipeExchangingElement pipeExchangingN4(Tstart = 273.15 + 50);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature source_T_fixed1(T = 273.15 + 5);
equation
  connect(const.y, source_WT.Win);
  connect(sink_P1.Pin, const3.y);
  connect(sink_P.Pin, const2.y);
  connect(ramp.y, source_WT.Tin);
  connect(const1.y, source_WT1.Win);
  connect(ramp1.y, source_WT1.Tin);
  connect(convVec2Sca1.ss1, convSca2Sca.ss1);
  connect(convVec2Sca1.ss1, massT.surf);
  connect(pipeExchanging.heatPort, convVec2Sca1.ss2);
  connect(convVec2Vec.vs2, pipeExchangingN1.heatPort);
  connect(convVec2Vec.vs1, pipeExchangingN.heatPort);
  connect(source_WT.water_flange, pipeExchangingN.water_flange1);
  connect(pipeExchangingN.water_flange2, sink_P.water_flange);
  connect(source_WT1.water_flange, pipeExchangingN1.water_flange1);
  connect(pipeExchangingN1.water_flange2, sink_P1.water_flange);
  connect(source_PT_fixed1.water_flange, pipeExchangingN3.water_flange1);
  connect(pipeExchangingN3.water_flange2, pipeExchanging.water_flange1);
  connect(pipeExchangingN4.water_flange1, pipeExchanging.water_flange2);
  connect(sink_P_fixed1.water_flange, pipeExchangingN4.water_flange2);
end test2_Pipe;
```

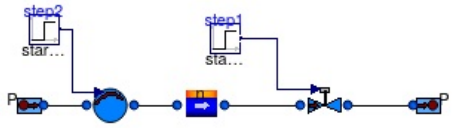
EEB.Components.BaseComponents.Test.test3_VolumetricPump



Modelica definition

```
model test3_VolumetricPump
  Modelica.Blocks.Sources.Step step(height = 0.4, offset = 0.2, startTime = 200);
  Water.Valves.WaterValve_LinChar valveLinChar(cvmax = 1e-5);
  Water.Pumps.WaterPump_Volumetric volumetricPump;
  Modelica.Blocks.Sources.Step step1(height = 0.5, offset = 0.5, startTime = 100);
  Water.Valves.WaterValve_LinChar valveLinChar1(cvmax = 1e-5);
  Modelica.Blocks.Sources.Step step2(height = 0.4, offset = 0.2, startTime = 50);
  Water.Sources.WaterSource_PT_fixed source_PT_fixed2(T0 = 273.15 + 25);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed2(Pin = 101325);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN1;
  Water.Sources.WaterSource_PT_fixed source_PT_fixed3(T0 = 273.15 + 25);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed3(Pin = 101325);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN2;
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN3(Dz = -1);
  Modelica.Blocks.Sources.Step step3(height = 0.4, offset = 0.2, startTime = 200);
  Water.Valves.WaterValve_LinChar valveLinChar2(cvmax = 1e-5);
  Water.Pumps.WaterPump_Volumetric volumetricPump1;
  Modelica.Blocks.Sources.Step step4(height = 0.5, offset = 0.5, startTime = 100);
  Water.Sources.WaterSource_PT_fixed source_PT_fixed1(T0 = 273.15 + 25);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed1(Pin = 0);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN4;
equation
  connect(step2.y, valveLinChar1.cmd);
  connect(step.y, valveLinChar.cmd);
  connect(step1.y, volumetricPump.cmd);
  connect(step3.y, valveLinChar2.cmd);
  connect(step4.y, volumetricPump1.cmd);
  connect(source_PT_fixed2.water_flange, pipeExchangingN3.water_flange1);
  connect(pipeExchangingN3.water_flange2, valveLinChar1.water_flange1);
  connect(valveLinChar1.water_flange2, pipeExchangingN1.water_flange1);
  connect(pipeExchangingN1.water_flange2, sink_P_fixed2.water_flange);
  connect(source_PT_fixed3.water_flange, volumetricPump.water_flange1);
  connect(volumetricPump.water_flange2, valveLinChar.water_flange1);
  connect(valveLinChar.water_flange2, pipeExchangingN2.water_flange1);
  connect(pipeExchangingN2.water_flange2, sink_P_fixed3.water_flange);
  connect(source_PT_fixed1.water_flange, volumetricPump1.water_flange1);
  connect(volumetricPump1.water_flange2, pipeExchangingN4.water_flange1);
  connect(pipeExchangingN4.water_flange2, valveLinChar2.water_flange1);
  connect(valveLinChar2.water_flange2, sink_P_fixed1.water_flange);
end test3_VolumetricPump;
```

EEB.Components.BaseComponents.Test.test4_CentrifugalPump

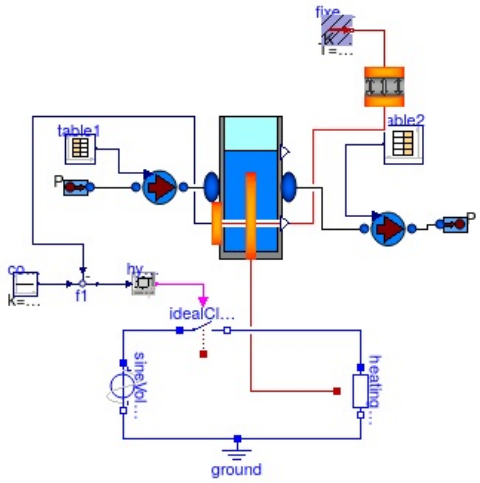


Modelica definition

```
model test4_CentrifugalPump
  Water.Valves.WaterValve_LinChar valveLinChar(cvmax = 1e-3);
  Water.Sources.WaterSource_PT_fixed source_PT_fixed;
  Water.Sinks.WaterSink_P_fixed sink_P_fixed(Pin = 101325);
  Modelica.Blocks.Sources.Step step1( height = 0.2,offset = 0, startTime = 10);
  Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = 1, Dtube = 500, Dtube = 0.01);
  Modelica.Blocks.Sources.Step step2( height = 0.5, offset = 0.5,startTime = 1);
  Water.Pumps.WaterPump_Centrifugal centrifugalPump(DP1 = 5000, DP0 = 200000);
```

```
equation
  connect(step1.y, valveLinChar.cmd);
  connect(step2.y, centrifugalPump.cmd);
  connect(source_PT_fixed.water_flange, centrifugalPump.water_flange1);
  connect(centrifugalPump.water_flange2, pipeExchangingN.water_flange1);
  connect(pipeExchangingN.water_flange2, valveLinChar.water_flange1);
  connect(valveLinChar.water_flange2, sink_P_fixed.water_flange);
end test4_CentrifugalPump;
```

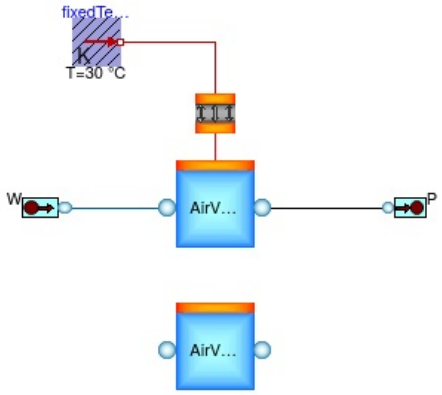
EEB.Components.BaseComponents.Test.test5_Tank



Modelica definition

```
model test5_Tank
  Water.Pumps.WaterPump_Volumetric volumetricPump(qmax = 0.0001);
  Water.Sinks.WaterSink_P_fixed sink_P_fixed(Pin = 101325);
  Thermal.HeatTransfer.Convection_SS convSca2Sca(S = 1, gamma = 45);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 273.15 + 20);
  Water.Sources.WaterSource_PT_fixed source_PT_fixed1(T0 = 273.15 + 20);
  Water.Tanks.Tank_exchangingWall tank(S = 0.5, Sb = 0.01, Mm = 2, cm = 300, gwm = 10, Tstart = 273.15 + 20, V = 1);
  Water.Pumps.WaterPump_Volumetric volumetricPump1(qmax = 0.0001);
  Modelica.Electrical.Analog.Basic.HeatingResistor heatingResistor(R_ref = 220 ^ 2 / 3000);
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Blocks.Sources.CombiTimeTable table1(table = [0, 1; 60, 1; 60, 0]);
  Modelica.Blocks.Sources.CombiTimeTable table2(startTime = 2000, table = [0, 0; 100, 0; 100, 1; 150, 1; 150, 0]);
  Modelica.Electrical.Analog.Ideal.IdealClosingSwitch idealClosingSwitch;
  Modelica.Electrical.Analog.Sources.SineVoltage sineVoltage(V = 220, freqHz = 50);
  Modelica.Blocks.Sources.Constant const(k = 273.15 + 25);
  Modelica.Blocks.Math.Feedback f1;
  Modelica.Blocks.Logical.Hysteresis hysteresis(uLow = -1, uHigh = 1);
equation
  connect(table1.y[1], volumetricPump.cmd);
  connect(table2.y[1], volumetricPump1.cmd);
  connect(heatingResistor.heatPort, tank.fluid);
  connect(heatingResistor.p, idealClosingSwitch.n);
  connect(idealClosingSwitch.p, sineVoltage.p);
  connect(ground.p, sineVoltage.n);
  connect(heatingResistor.n, ground.p);
  connect(const.y, f1.u1);
  connect(f1.y, hysteresis.u);
  connect(hysteresis.y, idealClosingSwitch.control);
  connect(tank.oT, f1.u2);
  connect(tank.wall, convSca2Sca.ss2);
  connect(fixedTemperature.port, convSca2Sca.ss1);
  connect(source_PT_fixed1.water_flange, volumetricPump.water_flange1);
  connect(volumetricPump.water_flange2, tank.water_flange1);
  connect(tank.water_flange2, volumetricPump1.water_flange1);
  connect(volumetricPump1.water_flange2, sink_P_fixed.water_flange);
end test5_Tank;
```

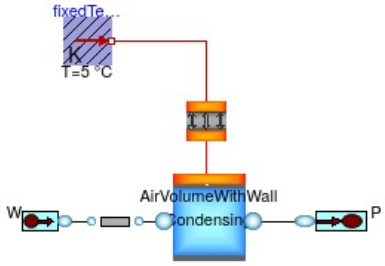
EEB.Components.BaseComponents.Test.test6_AirVolume



Modelica definition

```
model test6_AirVolume
  EEB.Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(X0 = 8e-3, w0 = 1.21606, T0 = 273.15 + 25);
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed sink_W_fixed;
  EEB.Components.BaseComponents.Air.Volumes.AirVolume airVolume(Tstart = 273.15 + 20, V = 10);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca(gamma = 10);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 303.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume airVolume1(Tstart = 273.15 + 20, V = 10);
equation
  connect(airVolume.air_flange2, sink_W_fixed.air_flange);
  connect(convSca2Sca.ss2, airVolume.heatPort);
  connect(airSource_fixed_wTX.air_flange, airVolume.air_flange1);
  connect(fixedTemperature.port, convSca2Sca.ss1);
end test6_AirVolume;
```

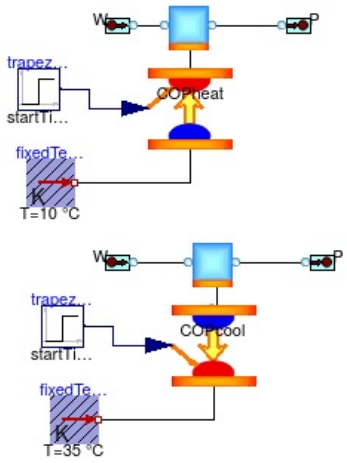
EEB.Components.BaseComponents.Test.test7_AirVolumeWithWall_Condensing



Modelica definition

```
model test7_AirVolumeWithWall_Condensing
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX( T0 = 273.15 + 20, X0 = 8e-3,w0 = 1.21606);
  Air.Sinks.AirSink_P_fixed sink_W_fixed;
  EEB.Components.BaseComponents.Air.Volumes.AirVolumeWithWall_Condensing airVolumeWithWall_Condensing(Xstart = 0.001, V = 10);
  Air.Pdrops.AirPdrop_Lin pdrop;
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca(S = 10, gamma = 100);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 278.15);
equation
  connect(airVolumeWithWall_Condensing.air_flange2, sink_W_fixed.air_flange);
  connect(airSource_fixed_wTX.air_flange, pdrop.air_flange1);
  connect(pdrop.air_flange2, airVolumeWithWall_Condensing.air_flange1);
  connect(airVolumeWithWall_Condensing.heatPort, convSca2Sca.ss2);
  connect(fixedTemperature.port, convSca2Sca.ss1);
end test7_AirVolumeWithWall_Condensing;
```

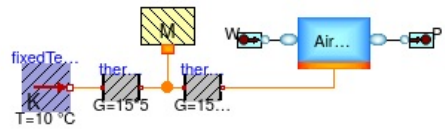
EEB.Components.BaseComponents.Test.test8_HeatPump

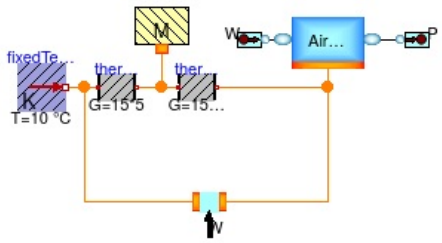


Modelica definition

```
model test8_HeatPump
  HVAC.HeatPumps.HeatPump_ConstantCOPheat heatPump_ConstantCOPheat(Wmax = 1000, constCOPheat = 3);
  Modelica.Blocks.Sources.Step trapezoid(height = -0.01, offset = 0.01, startTime = 3600);
  Air.Volumes.AirVolume airVolume(V = 5 * 20);
  Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 283.15);
  HVAC.HeatPumps.HeatPump_ConstantCOPcool heatPump_ConstantCOPheat1(Wmax = 1500, constCOPcool = 3);
  Modelica.Blocks.Sources.Step trapezoid1(height = -0.01, offset = 0.01, startTime = 3600);
  Air.Volumes.AirVolume airVolume1(V = 5 * 20);
  Air.Sinks.AirSink_P_fixed airSink_P_fixed1;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 308.15);
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(w0 = 0);
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX1(w0 = 0);
equation
  connect(trapezoid.y, heatPump_ConstantCOPheat.cmd01);
  connect(airSink_P_fixed.air_flange, airVolume.air_flange2);
  connect(airVolume.heatPort, heatPump_ConstantCOPheat.hotPort);
  connect(trapezoid1.y, heatPump_ConstantCOPheat1.cmd01);
  connect(airSink_P_fixed1.air_flange, airVolume1.air_flange2);
  connect(airVolume1.heatPort, heatPump_ConstantCOPheat1.coldPort);
  connect(heatPump_ConstantCOPheat1.hotPort, fixedTemperature1.port);
  connect(heatPump_ConstantCOPheat.coldPort, fixedTemperature.port);
  connect(airVolume.air_flange1, airSource_fixed_wTX.air_flange);
  connect(airVolume1.air_flange1, airSource_fixed_wTX1.air_flange);
end test8_HeatPump;
```

EEB.Components.BaseComponents.Test.test9_AirRenovation



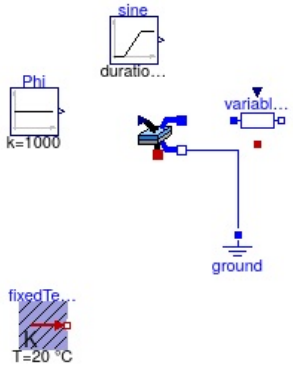


Modelica definition

```
model test9_AirRenovation
  Air.Renovation.AirRenovation airRenovation;
  Air.Volumes.AirVolume airVolume1(Tstart = 273.15 + 25, V = 5 * 5 * 3);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 283.15);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 15 * 5);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 15 * 3.5);
  Thermal.Capacities.MassT massT(M = 100, cp = 920);
  Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Air.Volumes.AirVolume airVolume(Tstart = 273.15 + 25, V = 5 * 5 * 3);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 283.15);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor2(G = 15 * 5);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor3(G = 15 * 3.5);
  Thermal.Capacities.MassT massT1(M = 100, cp = 920);
```

```
  Air.Sinks.AirSink_P_fixed airSink_P_fixed1;
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(w0 = 0);
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX1(w0 = 0);
equation
  connect(fixedTemperature.port, thermalConductor.port_a);
  connect(thermalConductor.port_b, thermalConductor1.port_a);
  connect(massT.surf, thermalConductor1.port_a);
  connect(airRenovation.heatPort2, airVolume1.heatPort);
  connect(thermalConductor1.port_b, airVolume1.heatPort);
  connect(airVolume1.air_flange2, airSink_P_fixed.air_flange);
  connect(airRenovation.heatPort1, thermalConductor.port_a);
  connect(fixedTemperature1.port, thermalConductor2.port_a);
  connect(thermalConductor2.port_b, thermalConductor3.port_a);
  connect(massT1.surf, thermalConductor3.port_a);
  connect(thermalConductor3.port_b, airVolume.heatPort);
  connect(airVolume.air_flange2, airSink_P_fixed1.air_flange);
  connect(airVolume.air_flange1, airSource_fixed_wTX.air_flange);
  connect(airVolume1.air_flange1, airSource_fixed_wTX1.air_flange);
end test9_AirRenovation;
```

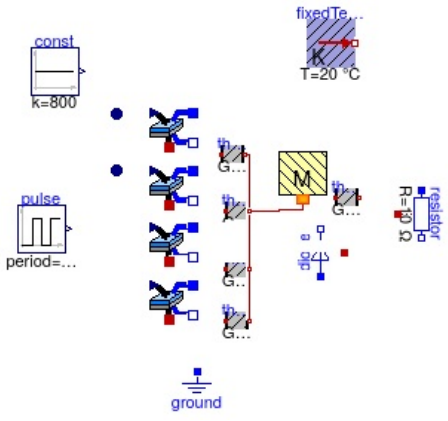
EEB.Components.BaseComponents.Test.test_Cell_VIchar



Modelica definition

```
model test_Cell_VIchar
  Electrical.PV.Cell cell;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 293.15);
  Modelica.Blocks.Sources.Constant Phi(k = 1000);
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Electrical.Analog.Basic.VariableResistor variableResistor;
  Modelica.Blocks.Sources.Ramp sine(height = 1e4, duration = 100, offset = 0);
equation
  connect(cell.n, ground.p);
  connect(Phi.y, cell.Phi);
  connect(fixedTemperature.port, cell.surf);
  connect(cell.p, variableResistor.p);
  connect(variableResistor.n, ground.p);
  connect(sine.y, variableResistor.R);
end test_Cell_VIchar;
```

EEB.Components.BaseComponents.Test.test_Cell_series

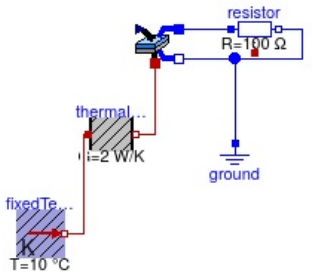


Modelica definition

```
model test_Cell_series
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Blocks.Sources.Constant const(k = 800);
  Electrical.PV.Cell cell(Rp = 1e6, Rs = 2e-5, i10 = 2, Kphi = -0.01, irs = 1e-12, dif = 5);
```

```
  Electrical.PV.Cell cell1(Rp = 1e6, Rs = 2e-5, il0 = 2, Kphi = -0.01, irs = 1e-12, dif = 5);
  Electrical.PV.Cell cell2(Rp = 1e6, Rs = 2e-5, il0 = 2, Kphi = -0.01, irs = 1e-12, dif = 5);
  Electrical.PV.Cell cell3(Rp = 1e6, Rs = 2e-5, il0 = 2, Kphi = -0.01, irs = 1e-12, dif = 5);
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 10);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 2);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T(displayUnit = "degC") = 293.15);
  Thermal.Capacities.MassT heatCap(Tstart = 293.15);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 2);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor2(G = 2);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor3(G = 2);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor4(G = 2);
  Modelica.Blocks.Sources.Pulse pulse(amplitude = 1000, period = 10);
  Modelica.Electrical.Analog.Semiconductors.HeatingDiode diode(useHeatPort = true);
equation
  connect(const.y, cell3.Phi);
  connect(const.y, cell2.Phi);
  connect(const.y, cell.Phi);
  connect(cell.n, ground.p);
  connect(cell.p, cell1.n);
  connect(cell1.p, cell2.n);
  connect(cell2.p, cell3.n);
  connect(cell3.p, resistor.p);
  connect(resistor.n, ground.p);
  connect(cell3.surf, thermalConductor.port_a);
  connect(cell1.surf, thermalConductor2.port_a);
  connect(cell.surf, thermalConductor3.port_a);
  connect(cell2.surf, thermalConductor1.port_a);
  connect(thermalConductor4.port_b, fixedTemperature.port);
  connect(pulse.y, cell1.Phi);
  connect(cell1.p, diode.n);
  connect(cell1.n, diode.p);
  connect(cell1.surf, diode.heatPort);
  connect(thermalConductor.port_b, thermalConductor1.port_b);
  connect(thermalConductor1.port_b, thermalConductor2.port_b);
  connect(thermalConductor2.port_b, thermalConductor3.port_b);
  connect(thermalConductor1.port_b, heatCap.surf);
end test_Cell_series;
```

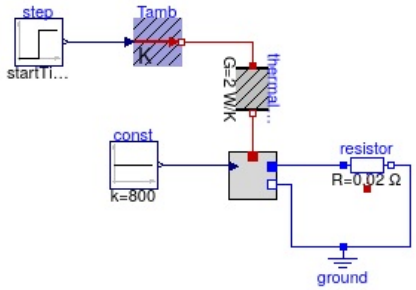
EEB.Components.BaseComponents.Test.test_Cell



Modelica definition

```
model test_Cell
  Electrical.PV.Cell cell;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 2);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 283.15);
  Modelica.Blocks.Sources.Sine sine(freqHz = 1 / 20, offset = 10, amplitude = 9);
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 100);
  Modelica.Electrical.Analog.Basic.Ground ground;
equation
  connect(cell.surf, thermalConductor.port_b);
  connect(thermalConductor.port_a, fixedTemperature.port);
  connect(cell.p, resistor.p);
  connect(resistor.n, cell.n);
  connect(cell.n, ground.p);
  connect(sine.y, cell.Phi);
end test_Cell;
```

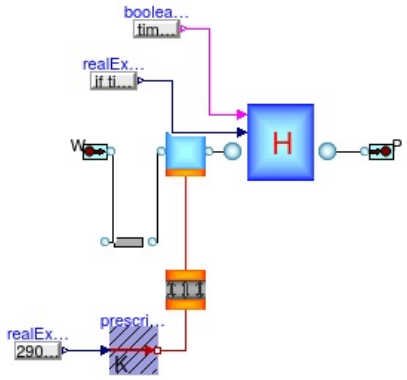
EEB.Components.BaseComponents.Test.testModule1



Modelica definition

```
model testModule1
  Electrical.PV.Module module1_1(Vt_d = 0.7);
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 0.02);
  Modelica.Blocks.Sources.Constant const(k = 800);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Tamb;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 2);
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Blocks.Sources.Step step(height = -10, offset = 300, startTime = 20000);
equation
  connect(thermalConductor.port_b, module1_1.heatPort);
  connect(Tamb.port, thermalConductor.port_a);
  connect(step.y, Tamb.T);
  connect(const.y, module1_1.Phi);
  connect(module1_1.p, resistor.p);
  connect(resistor.n, ground.p);
  connect(module1_1.n, ground.p);
end testModule1;
```

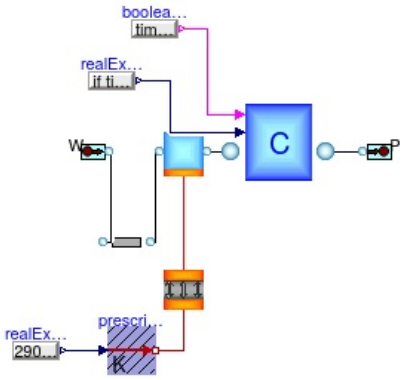
EEB.Components.BaseComponents.Test.Test_ControlledHeater



Modelica definition

```
model Test_ControlledHeater
  HVAC.AirHandling.ControlledHeater_SensQbal controlledHeater;
  Air.Sources.AirSource_wTX_fixed airSource_wTX_fixed;
  Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Modelica.Blocks.Sources.RealExpression realExpression(y = if time < 100 or time > 150 then 300 else 280);
  Modelica.Blocks.Sources.BooleanExpression booleanExpression(y = time < 30 or time > 60);
  Air.Volumes.AirVolume airVolume;
  Air.Pdrops.AirPdrop_Lin airPdrop_Lin;
  Thermal.HeatTransfer.Convection_SS convection_SS(gamma = 100);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature prescribedTemperature;
  Modelica.Blocks.Sources.RealExpression realExpression1(y = 290 + 5 * sin(time / 5));
equation
  connect(controlledHeater.air_flange2, airSink_P_fixed.air_flange);
  connect(booleanExpression.y, controlledHeater.ON);
  connect(realExpression.y, controlledHeater.Tsp);
  connect(airVolume.air_flange2, controlledHeater.air_flange1);
  connect(airSource_wTX_fixed.air_flange, airPdrop_Lin.air_flange1);
  connect(airPdrop_Lin.air_flange2, airVolume.air_flange1);
  connect(airVolume.heatPort, convection_SS.ss1);
  connect(prescribedTemperature.port, convection_SS.ss2);
  connect(realExpression1.y, prescribedTemperature.T);
end Test_ControlledHeater;
```

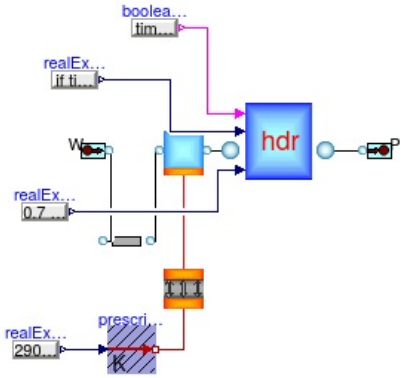
EEB.Components.BaseComponents.Test.Test_ControlledCooler



Modelica definition

```
model Test_ControlledCooler
  HVAC.AirHandling.ControlledCooler_SensQbal controlledCooler;
  Air.Sources.AirSource_wTX_fixed airSource_wTX_fixed;
  Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Modelica.Blocks.Sources.RealExpression realExpression(y = if time < 100 or time > 150 then 280 else 300);
  Modelica.Blocks.Sources.BooleanExpression booleanExpression(y = time < 30 or time > 60);
  Air.Volumes.AirVolume airVolume;
  Air.Pdrops.AirPdrop_Lin airPdrop_Lin;
  Thermal.HeatTransfer.Convection_SS convection_SS(gamma = 100);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature prescribedTemperature;
  Modelica.Blocks.Sources.RealExpression realExpression1(y = 290 + 5 * sin(time / 5));
equation
  connect(controlledCooler.air_flange2, airSink_P_fixed.air_flange);
  connect(booleanExpression.y, controlledCooler.ON);
  connect(realExpression.y, controlledCooler.Tsp);
  connect(airVolume.air_flange2, controlledCooler.air_flange1);
  connect(airSource_wTX_fixed.air_flange, airPdrop_Lin.air_flange1);
  connect(airPdrop_Lin.air_flange2, airVolume.air_flange1);
  connect(airVolume.heatPort, convection_SS.ss1);
  connect(prescribedTemperature.port, convection_SS.ss2);
  connect(realExpression1.y, prescribedTemperature.T);
end Test_ControlledCooler;
```

EEB.Components.BaseComponents.Test.Test_ControlledHandler

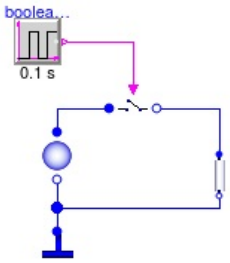


Modelica definition

```
model Test_ControlledHandler
  HVAC.AirHandling.ControlledHandler_Tphi_AlgQbal controlledHandler;
  Air.Sources.AirSource_wTX_fixed airSource_wTX_fixed;
  Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Modelica.Blocks.Sources.RealExpression realExpression(y = if time < 100 or time > 150 then 280 else 300);
  Modelica.Blocks.Sources.BooleanExpression booleanExpression(y = time < 30 or time > 60);
  Air.Volumes.AirVolume airVolume;
  Air.Pdrops.AirPdrop_Lin airPdrop_Lin;
  Thermal.HeatTransfer.Convection_SS convection_SS(gamma = 100);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature prescribedTemperature;
  Modelica.Blocks.Sources.RealExpression realExpression1(y = 290 + 5 * sin(time / 5));
  Modelica.Blocks.Sources.RealExpression realExpression2(y = 0.7 + 0.2 * sin(time / 20));
equation
  connect(controlledHandler.air_flange2, airSink_P_fixed.air_flange);
  connect(booleanExpression.y, controlledHandler.ON);
  connect(realExpression.y, controlledHandler.Tsp);
  connect(airVolume.air_flange2, controlledHandler.air_flange1);
  connect(airSource_wTX_fixed.air_flange, airPdrop_Lin.air_flange1);
  connect(airPdrop_Lin.air_flange2, airVolume.air_flange1);
  connect(airVolume.heatPort, convection_SS.ss1);
  connect(prescribedTemperature.port, convection_SS.ss2);
  connect(realExpression1.y, prescribedTemperature.T);
```

```
  connect(realExpression2.y, controlledHandler.phisp);
end Test_ControlledHandler;
```

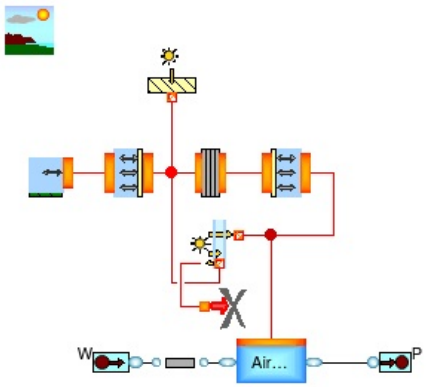
EEB.Components.BaseComponents.Test.TestPhasorSwitch



Modelica definition

```
model TestPhasorSwitch
  Electrical.Phasors.Ground ground;
  Electrical.Phasors.Vgen_Sine_Fixed vgen_Sine_Fixed;
  Electrical.Phasors.Switch switch;
  Electrical.Phasors.Load_VPcosphi_nom load_VPcosphi_nom;
  Modelica.Blocks.Sources.BooleanPulse booleanPulse(period = 0.1);
equation
  connect(vgen_Sine_Fixed.p, switch.p);
  connect(switch.n, load_VPcosphi_nom.p);
  connect(load_VPcosphi_nom.n, ground.p);
  connect(ground.p, vgen_Sine_Fixed.n);
  connect(booleanPulse.y, switch.close);
end TestPhasorSwitch;
```

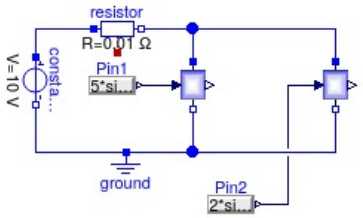
EEB.Components.BaseComponents.Test.TestRadiation



Modelica definition

```
model TestRadiation
  inner BoundaryConditions.AmbientConditions ambient_settings(Tgnd(displayUnit = "K"), Tsky(displayUnit = "degC") = 283.15);
  Thermal.Sources.SolarRadiation_OpaqueSurf solarRadiation_OpaqueSurf;
  Envelope.SolidMultilayer_Homogeneous Wall;
  Thermal.Sources.SolarRadiation_TransparentSurf Glass;
  Ambient.AmbientAirTemp ambientAirTemp;
  Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(X0 = 8e-3, T0 = 273.15 + 20, w0 = 1);
  Air.Sinks.AirSink_P_fixed sink_W_fixed;
  Air.Pdrops.AirPdrop_Lin pdrop;
  Air.Volumes.AirVolume airVolume1(Tstart = 273.15 + 20, V = 10);
  Thermal.HeatTransfer.Convection_Wall2air_BRIS convection_Wall2air_Internal_BRIS;
  Thermal.HeatTransfer.Convection_Wall2air_BI74 convection_Wall2air_Internal_BRIS1;
  Thermal.Sinks.PrescribedHeatLossesSink prescribedHeatLossesSink;
equation
  connect(solarRadiation_OpaqueSurf.Absorbed, Wall.side1);
  connect(airSource_fixed_wTX.air_flange, pdrop.air_flange1);
  connect(pdrop.air_flange2, airVolume1.air_flange1);
  connect(airVolume1.air_flange2, sink_W_fixed.air_flange);
  connect(Glass.Trasmitted, airVolume1.heatPort);
  connect(convection_Wall2air_Internal_BRIS.wall, Wall.side1);
  connect(ambientAirTemp.port, convection_Wall2air_Internal_BRIS.air);
  connect(Glass.Absorbed, Wall.side1);
  connect(Wall.side2, convection_Wall2air_Internal_BRIS1.wall);
  connect(convection_Wall2air_Internal_BRIS1.air, airVolume1.heatPort);
  connect(Glass.Reflected, prescribedHeatLossesSink.port);
end TestRadiation;
```

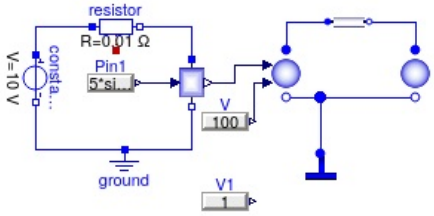
EEB.Components.BaseComponents.Test.TestDCloadI0



Modelica definition

```
model TestDCloadI0
  Electrical.DC.Load_Pin_I0 Load1;
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 0.01);
  Modelica.Electrical.Analog.Sources.ConstantVoltage constantVoltage(V = 10);
  Electrical.DC.Load_Pin_I0 Load2;
  Modelica.Blocks.Sources.RealExpression Pin1(y = 5 * sin(10 * time));
  Modelica.Blocks.Sources.RealExpression Pin2(y = 2 * sin(15 * time));
equation
  connect(constantVoltage.p, resistor.p);
  connect(constantVoltage.n, ground.p);
  connect(resistor.n, Load1.p);
  connect(Load1.n, ground.p);
  connect(Pin1.y, Load1.iPabs);
  connect(Pin2.y, Load2.iPabs);
  connect(Load2.p, Load1.p);
  connect(Load2.n, ground.p);
end TestDCloadI0;
```

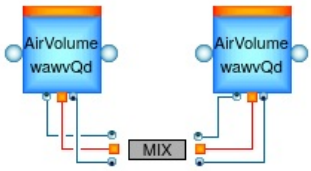
EEB.Components.BaseComponents.Test.TestDCloadACgen



Modelica definition

```
model TestDCloadACgen
  Electrical.DC.Load_Pin_I0 Load1;
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 0.01);
  Modelica.Electrical.Analog.Sources.ConstantVoltage constantVoltage(V = 10);
  Modelica.Blocks.Sources.RealExpression Pin1(y = 5 * sin(10 * time));
  Electrical.Phasors.Vgen_Sine_VP vgen_Sine_VP;
  Electrical.Phasors.Ground ground1;
  Electrical.Phasors.Vgen_Sine_Fixed vgen_Sine_Fixed;
  Modelica.Blocks.Sources.RealExpression V(y = 100);
  Modelica.Blocks.Sources.RealExpression V1(y = 1);
  Electrical.Phasors.Load_VPcosphi_nom load_VPcosphi_nom(Vnom = 1, Pnom = 1);
equation
  connect(constantVoltage.p, resistor.p);
  connect(constantVoltage.n, ground.p);
  connect(resistor.n, Load1.p);
  connect(Load1.n, ground.p);
  connect(Pin1.y, Load1.iPabs);
  connect(vgen_Sine_VP.n, ground1.p);
  connect(V.y, vgen_Sine_VP.V);
  connect(vgen_Sine_VP.p, load_VPcosphi_nom.p);
  connect(vgen_Sine_Fixed.n, ground1.p);
  connect(Load1.Pabs, vgen_Sine_VP.P);
  connect(load_VPcosphi_nom.n, vgen_Sine_Fixed.p);
end TestDCloadACgen;
```

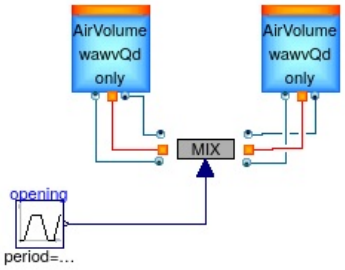
EEB.Components.BaseComponents.Test.TestNaturalMix



Modelica definition

```
model TestNaturalMix
  EEB.Components.BaseComponents.Air.Volumes.AirVolume_wawvQdPort V1(V = 10);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume_wawvQdPort V2(Xstart = 0.002, V = 30);
  EEB.Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix mix;
equation
  connect(V1.dryair, mix.dryair);
  connect(V1.diffuse, mix.diffuse);
  connect(V1.vapour, mix.vapour);
  connect(mix.Bdiffuse, V2.diffuse);
  connect(V2.dryair, mix.Bdryair);
  connect(V2.vapour, mix.Bvapour);
end TestNaturalMix;
```

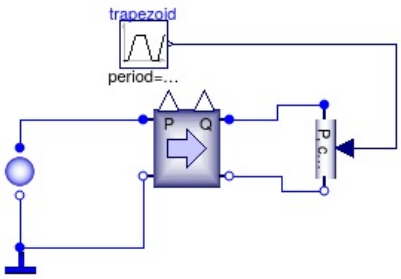
EEB.Components.BaseComponents.Test.TestNaturalMix_Gcmd



Modelica definition

```
model TestNaturalMix_Gcmd
  Air.Volumes.AirVolume_only_wawvQdport V1(V = 10);
  Air.Volumes.AirVolume_only_wawvQdport V2(Xstart = 0.002, V = 30);
  Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix;
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, width = 180, falling = 20, period = 400, startTime = 100);
equation
  connect(opening.y, mix.cmd01);
  connect(V1.diffuse, mix.diffuse);
  connect(V1.vapour, mix.vapour);
  connect(V1.dryair, mix.dryair);
  connect(mix.Bvapour, V2.vapour);
  connect(V2.dryair, mix.Bdryair);
  connect(mix.Bdiffuse, V2.diffuse);
end TestNaturalMix_Gcmd;
```

EEB.Components.BaseComponents.Test.Test_LoadPcosphi_Pin



Modelica definition

```
model Test_LoadPcosphi_Pin
  Electrical.Phasors.Load_Pcosphi_nom_Pin load_Pcosphi_nom_Pin;
  Electrical.Phasors.Ground ground;
  Electrical.Phasors.PQmeter pQmeter;
  Electrical.Phasors.Vgen_Sine_Fixed vgen_Sine_Fixed;
  Modelica.Blocks.Sources.Trapezoid trapezoid(amplitude = 100, rising = 5, width = 5, falling = 5, period = 20);
equation
  connect(pQmeter.p2, load_Pcosphi_nom_Pin.p);
  connect(pQmeter.n2, load_Pcosphi_nom_Pin.n);
  connect(pQmeter.n1, ground.p);
  connect(ground.p, vgen_Sine_Fixed.n);
```

```
  connect(vgen_Sine_Fixed.p, pQmeter.p1);
  connect(trapezoid.y, load_Pcosphi_nom_Pin.P);
end Test_LoadPcosphi_Pin;
```

EEB.Components.BaseComponents.HVAC

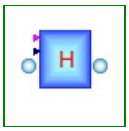
Package Content

Name	Description
AirHandling	
HeatPumps	

EEB.Components.BaseComponents.HVAC.AirHandling

Package Content

Name	Description
ControlledHeater_AlgQbal	
ControlledHeater_SensQbal	
ControlledCooler_AlgQbal	
ControlledCooler_SensQbal	
ControlledHandler_Tphi_AlgQbal	



EEB.Components.BaseComponents.HVAC.AirHandling.ControlledHeater_AlgQbal

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

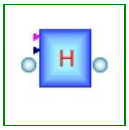
Type	Name	Default	Description
Temperature	Tstart	273.15 + 20	initial T [K]
Time	TC	4	T control TC [s]
Real	eta	0.8	efficiency

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input BooleanInput	ON	
input RealInput	Tsp	

Modelica definition

```
model ControlledHeater_AlgQbal
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  parameter Time TC = 4 "T control TC";
  parameter Real eta = 0.8 "efficiency";
  EEB.Media.Substances.MoistAir airin;
  EEB.Media.Substances.MoistAir air;
  Temperature T(start = Tstart) "controlled T";
  Power Qbal, Qcons_est;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput Tsp;
equation
  assert(air_flange1.wa >= 0, "No flow reversal here");
  pa1 = pa2;
  wa1 + wa2 = 0;
  airin.p = air_flange1.pa;
  airin.h = inStream(air_flange1.ha);
  airin.X = inStream(air_flange1.xa);
  T + TC * der(T) = if ON and Tsp > airin.T then Tsp else airin.T;
  air.p = airin.p;
  air.X = airin.X;
  air.T = T;
  haout1 = air.h;
  haout2 = air.h;
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  // Algebraic balance for Qbal, sign convention s.t. + is heating
  Qbal = air_flange1.wa * (air.h - airin.h);
  Qcons_est = max(Qbal, 0) / eta;
end ControlledHeater_AlgQbal;
```



EEB.Components.BaseComponents.HVAC.AirHandling.ControlledHeater_SensQbal

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

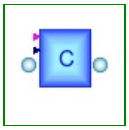
Type	Name	Default	Description
Volume	V	1	[m3]
Temperature	Tstart	273.15 + 20	initial T [K]
Time	TC	4	T control TC [s]
Real	eta	0.8	efficiency

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input BooleanInput	ON	
input RealInput	Tsp	

Modelica definition

```
model ControlledHeater_SensQbal
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Volume V = 1;
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  parameter Time TC = 4 "T control TC";
  parameter Real eta = 0.8 "efficiency";
  EEB.Media.Substances.MoistAir airin;
  EEB.Media.Substances.MoistAir air;
  Temperature T(start = Tstart) "controlled T";
  Power Qbal, Qcons_est;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput Tsp;
equation
  assert(air_flange1.wa >= 0, "No flow reversal here");
  pa1 = pa2;
  wa1 + wa2 = 0;
  airin.p = air_flange1.pa;
  airin.h = inStream(air_flange1.ha);
  airin.X = inStream(air_flange1.xa);
  T + TC * der(T) = if ON and Tsp > airin.T then Tsp else airin.T;
  air.p = airin.p;
  air.X = airin.X;
  air.T = T;
  haout1 = air.h;
  haout2 = air.h;
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  // Sens bal for Qbal, sign convention s.t. + is heating
  air.cp * air.d * V * der(T) = air_flange1.wa * (air.h - airin.h) - Qbal;
  Qcons_est = max(Qbal, 0) / eta;
end ControlledHeater_SensQbal;
```



EEB.Components.BaseComponents.HVAC.AirHandling.ControlledCooler_AlgQbal

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

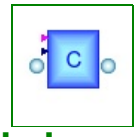
Type	Name	Default	Description
Temperature	Tstart	273.15 + 20	initial T [K]
Time	TC	4	T control TC [s]
Real	eta	0.8	efficiency

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input BooleanInput	ON	
input RealInput	Tsp	

Modelica definition

```
model ControlledCooler_AlgQbal
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  parameter Time TC = 4 "T control TC";
  parameter Real eta = 0.8 "efficiency";
  EEB.Media.Substances.MoistAir airin;
  EEB.Media.Substances.MoistAir air;
  Temperature T(start = Tstart) "controlled T";
  Power Qbal, Qcons_est;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput Tsp;
equation
  assert(air_flange1.wa >= 0, "No flow reversal here");
  pa1 = pa2;
  wa1 + wa2 = 0;
  airin.p = air_flange1.pa;
  airin.h = inStream(air_flange1.ha);
  airin.X = inStream(air_flange1.xa);
  T + TC * der(T) = if ON and Tsp < airin.T then Tsp else airin.T;
  air.p = airin.p;
  air.X = airin.X;
  air.T = T;
  haout1 = air.h;
  haout2 = air.h;
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  // Algebraic balance for Qbal, sign convention s.t. + is cooling
  Qbal = -air_flange1.wa * (air.h - airin.h);
  Qcons_est = max(Qbal, 0) / eta;
end ControlledCooler_AlgQbal;
```



EEB.Components.BaseComponents.HVAC.AirHandling.ControlledCooler_SensQbal

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Volume	V	1	[m3]
Temperature	Tstart	273.15 + 20	initial T [K]
Time	TC	4	T control TC [s]
Real	eta	0.8	efficiency

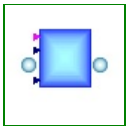
Connectors

Type	Name	Description

MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input BooleanInput	ON	
input RealInput	Tsp	

Modelica definition

```
model ControlledCooler_SensQbal
  extends Interfaces.Air.PartialTwoPort_waxa;
  parameter Volume V = 1;
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  parameter Time TC = 4 "T control TC";
  parameter Real eta = 0.8 "efficiency";
  EEB.Media.Substances.MoistAir airin;
  EEB.Media.Substances.MoistAir air;
  Temperature T(start = Tstart) "controlled T";
  Power Qbal, Qcons_est;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput Tsp;
equation
  assert(air_flange1.wa >= 0, "No flow reversal here");
  pa1 = pa2;
  wa1 + wa2 = 0;
  airin.p = air_flange1.pa;
  airin.h = inStream(air_flange1.ha);
  airin.X = inStream(air_flange1.xa);
  T + TC * der(T) = if ON and Tsp < airin.T then Tsp else airin.T;
  air.p = airin.p;
  air.X = airin.X;
  air.T = T;
  haout1 = air.h;
  haout2 = air.h;
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  // Sens bal for Qbal, sign convention s.t. + is cooling
  air.cp * air.d * V * der(T) = air_flange1.wa * (air.h - airin.h) + Qbal;
  Qcons_est = max(Qbal, 0) / eta;
end ControlledCooler_SensQbal;
```



EEB.Components.BaseComponents.HVAC.AirHandling.ControlledHandler_Tphi_AlgQbal

Information

Extends from Interfaces.Air.PartialTwoPort_waxa (Partial two port waxa).

Parameters

Type	Name	Default	Description
Temperature	Tstart	273.15 + 20	initial T [K]
Real	phistart	0.1	initial T
Time	TC	4	T control TC [s]
Real	eta	0.8	efficiency

Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
input BooleanInput	ON	
input RealInput	Tsp	
input RealInput	phisp	

Modelica definition


```
model ControlledHandler_Tphi_AlgQbal
  extends Interfaces.Air.PartialTwoPort_waxa;
```

```
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  parameter Real phistart = 0.1 "initial T";
  parameter Time TC = 4 "T control TC";
  parameter Real eta = 0.8 "efficiency";
  EEB.Media.Substances.MoistAir airin;
  EEB.Media.Substances.MoistAir air;
  Temperature T(start = Tstart) "controlled T";
  Real phi(start = phistart);
  Power Qbal, Qcons_est;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput Tsp;
  Modelica.Blocks.Interfaces.RealInput phisp;
equation
  assert(air_flange1.wa >= 0, "No flow reversal here");
  pa1 = pa2;
  wa1 + wa2 = 0;
  airin.p = air_flange1.pa;
  airin.h = inStream(air_flange1.ha);
  airin.X = inStream(air_flange1.xa);
  T + TC * der(T) = if ON then Tsp else airin.T;
  phi + TC * der(phi) = if ON then phisp else airin.phi;
  air.p = airin.p;
  air.phi = phi;
  air.T = T;
  haout1 = air.h;
  haout2 = air.h;
  xaout1 = inStream(air_flange2.xa);
  xaout2 = inStream(air_flange1.xa);
  // Algebraic balance for Qbal, sign convention s.t. + is heating
  Qbal = air_flange1.wa * (air.h - airin.h);
  Qcons_est = abs(Qbal) / eta;
end ControlledHandler_Tphi_AlgQbal;
```

EEB.Components.BaseComponents.HVAC.HeatPumps

Package Content

Name	Description
BaseClasses	
HeatPump_ConstantCOPheat	
HeatPump_ConstantCOPcool	
HeatPump_CarnotCOPheat_eta	
HeatPump_CarnotCOPcool_eta	
HeatPump_LinearCOPheat	
HeatPump_LinearCOPcool	

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPheat

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	constCOPheat	4	conatant heating COP

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_ConstantCOPheat
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real constCOPheat = 4 "conatant heating COP";
equation
  W = Wmax * cmd01;
  COPheat = constCOPheat;
end HeatPump_ConstantCOPheat;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPcool

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	constCOPcool	1	conatant cooling COP

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_ConstantCOPcool
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real constCOPcool = 1 "conatant cooling COP";
equation
  W = Wmax * cmd01;
  COPcool = constCOPcool;
end HeatPump_ConstantCOPcool;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_CarnotCOPheat_eta

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	eta	0.5	efficiency wrt Carnot COP
Real	maxCOPheat	2	maximum heating COP

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_CarnotCOPheat_eta
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real eta = 0.5 "efficiency wrt Carnot COP";
  parameter Real maxCOPheat = 2 "maximum heating COP";
  SI.Temperature DTmin;
equation
  eta * Th = maxCOPheat * DTmin;
  W = Wmax * cmd01;
  if Th - Tc > DTmin then
    COPheat = eta * Th / (Th - Tc);
  else
    COPheat = maxCOPheat;
  end if;
end HeatPump_CarnotCOPheat_eta;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_CarnotCOPcool_eta

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	eta	0.5	efficiency wrt Carnot COP
Real	maxCOPcool	1	maximum cooling COP

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_CarnotCOPcool_eta
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real eta = 0.5 "efficiency wrt Carnot COP";
  parameter Real maxCOPcool = 1 "maximum cooling COP";
  SI.Temperature DTmin;
equation
  eta * Tc = maxCOPcool * DTmin;
  W = Wmax * cmd01;
  if Th - Tc > DTmin then
    COPcool = eta * Tc / (Th - Tc);
  else
    COPcool = maxCOPcool;
  end if;
end HeatPump_CarnotCOPcool_eta;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_LinearCOPheat

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	maxCOPheat	2	maximum heating COP (DT=0)
Real	DT0	20	

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port

HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_LinearCOPheat
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real maxCOPheat = 2 "maximum heating COP (DT=0)";
  parameter Real DT0 = 20;
equation
  W = Wmax * cmd01;
  COPheat = maxCOPheat * (1 - (Th - Tc) / DT0);
end HeatPump_LinearCOPheat;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.HeatPump_LinearCOPcool

Information

Extends from BaseClasses.BaseHeatPump.

Parameters

Type	Name	Default	Description
HeatFlowRate	Wmax	100	W at cmd01=1 [W]
Real	maxCOPcool	2	maximum cooling COP (DT=0)
Real	DT0	20	

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
model HeatPump_LinearCOPcool
  extends BaseClasses.BaseHeatPump;
  parameter HeatFlowRate Wmax = 100 "W at cmd01=1";
  parameter Real maxCOPcool = 2 "maximum cooling COP (DT=0)";
  parameter Real DT0 = 20;
equation
  W = Wmax * cmd01;
  COPcool = maxCOPcool * (1 - (Th - Tc) / DT0);
end HeatPump_LinearCOPcool;
```

EEB.Components.BaseComponents.HVAC.HeatPumps.BaseClasses

Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
BaseHeatPump	

EEB.Components.BaseComponents.HVAC.HeatPumps.BaseClasses.BaseHeatPump

Connectors

Type	Name	Description
HeatPort	hotPort	hot side thermal port
HeatPort	coldPort	cold side thermal port
input RealInput	cmd01	command, 0-1 range

Modelica definition

```
partial model BaseHeatPump
  Temperature Tc, Th;
  HeatFlowRate Qc, Qh, W;
  Real COPcool, COPheat;
  EEB.Interfaces.Thermal.HeatPort hotPort "hot side thermal port";
  EEB.Interfaces.Thermal.HeatPort coldPort "cold side thermal port";
  Modelica.Blocks.Interfaces.RealInput cmd01 "command, 0-1 range";
equation
  Tc = coldPort.T;
  Th = hotPort.T;
  Qc = coldPort.Q_flow;
  Qh = -hotPort.Q_flow;
  Qh = Qc + W;
  Qc = COPcool * W;
  Qh = COPheat * W;
end BaseHeatPump;
```

EEB.Components.BaseComponents.Electrical

Package Content

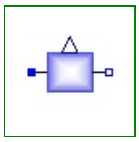
Name	Description
DC	
Phasors	
PV	

EEB.Components.BaseComponents.Electrical.DC

Package Content

Name	Description
Load_Pfixed_I0	
Load_Pin_I0	
Line	
Pmeter	

EEB.Components.BaseComponents.Electrical.DC.Load_Pfixed_I0



Information

Extends from Modelica.Electrical.Analog.Interfaces.OnePort (Component with two electrical pins p and n and current i from p to n).

Parameters

Type	Name	Default	Description
Power	Pnom	10	Nominal absorbed power [W]

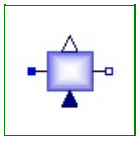
Connectors

Type	Name	Description
PositivePin	p	Positive electrical pin
NegativePin	n	Negative electrical pin
output RealOutput	Pabs	

Modelica definition

```
model Load_Pfixed_I0
  parameter Power Pnom(final min = 0) = 10 "Nominal absorbed power";
  extends Modelica.Electrical.Analog.Interfaces.OnePort;
  Modelica.Blocks.Interfaces.RealOutput Pabs;
equation
  if Pnom > 0 then
    v * i = Pnom;
    Pabs = Pnom;
  else
    i = 0;
    Pabs = 0;
  end if;
end Load_Pfixed_I0;
```

EEB.Components.BaseComponents.Electrical.DC.Load_Pin_I0



Information

Extends from Modelica.Electrical.Analog.Interfaces.OnePort (Component with two electrical pins p and n and current i from p to n).

Connectors

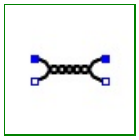
Type	Name	Description
PositivePin	p	Positive electrical pin
NegativePin	n	Negative electrical pin

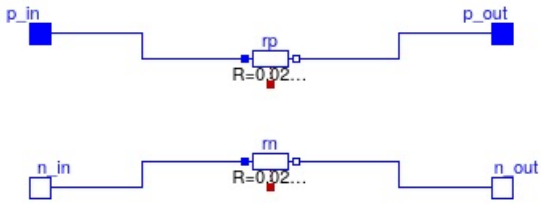
input RealInput	iPabs	
output RealOutput	Pabs	

Modelica definition

```
model Load_Pin_I0
  extends Modelica.Electrical.Analog.Interfaces.OnePort;
  Modelica.Blocks.Interfaces.RealInput iPabs;
  Modelica.Blocks.Interfaces.RealOutput Pabs;
equation
  if iPabs > 0 then
    v * i = iPabs;
    Pabs = iPabs;
  else
    i = 0;
    Pabs = 0;
  end if;
end Load_Pin_I0;
```

EEB.Components.BaseComponents.Electrical.DC.Line





Parameters

Type	Name	Default	Description
Length	Lw	10	Wire length [m]

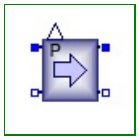
Connectors

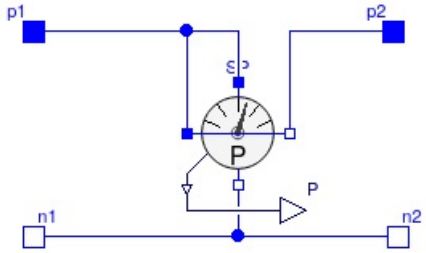
Type	Name	Description
PositivePin	p_in	input positive pin
NegativePin	n_in	input negative pin
PositivePin	p_out	output positive pin
NegativePin	n_out	output negative pin

Modelica definition

```
model Line
  parameter Length Lw = 10 "Wire length";
  Modelica.Electrical.Analog.Interfaces.PositivePin p_in "input positive pin";
  Modelica.Electrical.Analog.Interfaces.NegativePin n_in "input negative pin";
  Modelica.Electrical.Analog.Interfaces.PositivePin p_out "output positive pin";
  Modelica.Electrical.Analog.Interfaces.NegativePin n_out "output negative pin";
  Modelica.Electrical.Analog.Basic.Resistor rp(R = 0.02 * Lw);
  Modelica.Electrical.Analog.Basic.Resistor rn(R = 0.02 * Lw);
equation
  connect(p_in, rp.p);
  connect(rp.n, p_out);
  connect(n_in, rn.p);
  connect(rn.n, n_out);
end Line;
```

EEB.Components.BaseComponents.Electrical.DC.Pmeter





Connectors

Type	Name	Description
output RealOutput	P	
PositivePin	p1	
PositivePin	p2	
NegativePin	n1	
NegativePin	n2	

Modelica definition

```
model Pmeter
  Modelica.Blocks.Interfaces.RealOutput P;
  Modelica.Electrical.Analog.Interfaces.PositivePin p1;
  Modelica.Electrical.Analog.Interfaces.PositivePin p2;
  Modelica.Electrical.Analog.Interfaces.NegativePin n1;
  Modelica.Electrical.Analog.Interfaces.NegativePin n2;
  Modelica.Electrical.Analog.Sensors.PowerSensor SP;
equation
  connect(SP.pv, p1);
  connect(SP.nv, n1);
  connect(n2, n1);
  connect(SP.pc, p1);
  connect(SP.nc, p2);
  connect(SP.power, P);
end Pmeter;
```

EEB.Components.BaseComponents.Electrical.Phasors

Package Content

Name	Description
Ground	
Vgen_Sine_Fixed	
Vgen_Sine_Vphi	
Vgen_Sine_VP	
Resistor	
Capacitor	
Inductor	
R_plus_jX	
Load_VPcosphi_nom	
Load_Pcosphi_nom_Pfixed	
Load_Pcosphi_nom_Pin	
TransmissionLine	
PQmeter	
Switch	

EEB.Components.BaseComponents.Electrical.Phasors.Ground

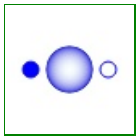
Connectors

Type	Name	Description
PositivePhasorPin	p	

Modelica definition

```
model Ground
  Interfaces.Electrical.PositivePhasorPin p;
equation
  p.vre = 0;
  p.vim = 0;
end Ground;
```

EEB.Components.BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

Type	Name	Default	Description
Voltage	V	100	[V]
Angle	phi	0	[rad]

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Vgen_Sine_Fixed
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Voltage V = 100;
  parameter Angle phi = 0;
equation
  Vre = V * cos(phi);
  Vim = V * sin(phi);
end Vgen_Sine_Fixed;
```

EEB.Components.BaseComponents.Electrical.Phasors.Vgen_Sine_Vphi

Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	
input RealInput	V	
input RealInput	phi	

Modelica definition

```
model Vgen_Sine_Vphi
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  Modelica.Blocks.Interfaces.RealInput V;
  Modelica.Blocks.Interfaces.RealInput phi;
equation
  Vre = V * cos(phi);
  Vim = V * sin(phi);
end Vgen_Sine_Vphi;
```

EEB.Components.BaseComponents.Electrical.Phasors.Vgen_Sine_VP

Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

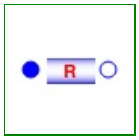
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	
input RealInput	V	
input RealInput	P	

Modelica definition

```
model Vgen_Sine_VP
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  ReactivePower Q;
  Modelica.Blocks.Interfaces.RealInput V;
  Modelica.Blocks.Interfaces.RealInput P;
equation
  P = Ire * Vre + Iim * Vim;
  Q = Ire * Vim - Iim * Vre;
  V ^ 2 = Vre ^ 2 + Vim * 2;
end Vgen_Sine_VP;
```

EEB.Components.BaseComponents.Electrical.Phasors.Resistor



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

Type	Name	Default	Description
Resistance	R	1	[Ohm]

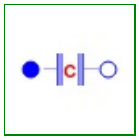
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Resistor
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Resistance R = 1;
equation
  Vre = R * Ire;
  Vim = R * Iim;
end Resistor;
```

EEB.Components.BaseComponents.Electrical.Phasors.Capacitor



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

Type	Name	Default	Description
Capacitance	C	0.01	[F]
Frequency	fo	50	[Hz]

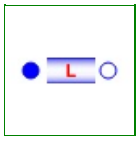
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Capacitor
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Capacitance C = 0.01;
  parameter Frequency fo = 50;
equation
  Vre = Iim / (2 * pi * fo * C);
  Vim = -Ire / (2 * pi * fo * C);
end Capacitor;
```

EEB.Components.BaseComponents.Electrical.Phasors.Inductor



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

Type	Name	Default	Description
Inductance	L	0.01	[H]
Frequency	fo	50	[Hz]

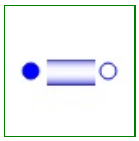
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Inductor
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Inductance L = 0.01;
  parameter Frequency fo = 50;
equation
  Vre = -2 * pi * fo * L * Iim;
  Vim = 2 * pi * fo * L * Ire;
end Inductor;
```

EEB.Components.BaseComponents.Electrical.Phasors.R_plus_jX



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

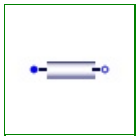
Type	Name	Default	Description
Resistance	R	1	[Ohm]
Reactance	X	1	[Ohm]

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model R_plus_jX
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Resistance R = 1;
  parameter Reactance X = 1;
equation
  Vre = R * Ire - X * Iim;
  Vim = R * Iim + X * Ire;
end R_plus_jX;
```



EEB.Components.BaseComponents.Electrical.Phasors.Load_VPcosphi_nom

Parameters

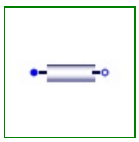
Type	Name	Default	Description
Voltage	Vnom	100	nominal V [V]
Frequency	fnom	50	nominal f [Hz]
Power	Pnom	100	nominal active P at Vnom,fnom [W]
Real	cphinom	0.9	nominal power factor

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Load_VPcosphi_nom
  parameter Voltage Vnom = 100 "nominal V";
  parameter Frequency fnom = 50 "nominal f";
  parameter Power Pnom = 100 "nominal active P at Vnom,fnom";
  parameter Real cphinom = 0.9 "nominal power factor";
  Voltage Vre, Vim;
  Current Ire, Iim;
  Resistor res(R = R);
  Inductor ind(L = L, fo = fnom);
  Interfaces.Electrical.PositivePhasorPin p;
  Interfaces.Electrical.NegativePhasorPin n;
protected
  parameter Resistance R = Vnom ^ 2 * cphinom ^ 2 / Pnom;
  parameter Inductance L = Vnom ^ 2 * cphinom / Pnom / (2 * pi * fnom) * sqrt(1 - cphinom ^ 2);
equation
  Vre = p.vre - n.vre;
  Vim = p.vim - n.vim;
  Ire = p.ire;
  Iim = p.iim;
  connect(res.n, ind.p);
  connect(p, res.p);
  connect(ind.n, n);
end Load_VPcosphi_nom;
```



EEB.Components.BaseComponents.Electrical.Phasors.Load_Pcosphi_nom_Pfixed

Parameters

Type	Name	Default	Description
Power	Pnom	10	Nominal power [W]
Real	cphinom	0.9	nominal power factor

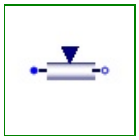
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	

Modelica definition

```
model Load_Pcosphi_nom_Pfixed
  parameter Power Pnom = 10 "Nominal power";
  parameter Real cphinom = 0.9 "nominal power factor";
```

```
  Voltage Vre, Vim;
  Current Ire, Iim;
  Interfaces.Electrical.PositivePhasorPin p;
  Interfaces.Electrical.NegativePhasorPin n;
equation
  p.ire + n.ire = 0;
  p.iim + n.iim = 0;
  Vre = p.vre - n.vre;
  Vim = p.vim - n.vim;
  Ire = p.ire;
  Iim = p.iim;
  Vre * Ire + Vim * Iim = Pnom;
  atan2(Iim, Ire) - atan2(Vim, Vre) = acos(cphinom);
end Load_Pcosphi_nom_Pfixed;
```



EEB.Components.BaseComponents.Electrical.Phasors.Load_Pcosphi_nom_Pin

Parameters

Type	Name	Default	Description
Real	cphinom	0.9	nominal power factor

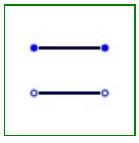
Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	
input RealInput	P	

Modelica definition

```
model Load_Pcosphi_nom_Pin
  parameter Real cphinom = 0.9 "nominal power factor";
  Voltage Vre, Vim;
  Current Ire, Iim;
  Interfaces.Electrical.PositivePhasorPin p;
  Interfaces.Electrical.NegativePhasorPin n;
  Modelica.Blocks.Interfaces.RealInput P;
equation
  p.ire + n.ire = 0;
  p.iim + n.iim = 0;
  Vre = p.vre - n.vre;
  Vim = p.vim - n.vim;
  Ire = p.ire;
  Iim = p.iim;
  Vre * Ire + Vim * Iim = P;
  atan2(Iim, Ire) - atan2(Vim, Vre) = acos(cphinom);
end Load_Pcosphi_nom_Pin;
```

EEB.Components.BaseComponents.Electrical.Phasors.TransmissionLine



Parameters

Type	Name	Default	Description
Length	l	1000	line length [m]
Length	r	0.002	wire radius [m]
Length	d	0.1	conductors distance [m]
Frequency	fo	50	nominal f [Hz]

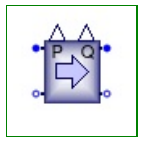
Connectors

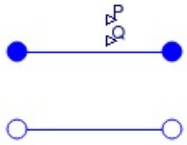
Type	Name	Description
PositivePhasorPin	p1	
PositivePhasorPin	p2	
NegativePhasorPin	n1	
NegativePhasorPin	n2	

Modelica definition

```
model TransmissionLine
  Interfaces.Electrical.PositivePhasorPin p1;
  Interfaces.Electrical.PositivePhasorPin p2;
  Interfaces.Electrical.NegativePhasorPin n1;
  Interfaces.Electrical.NegativePhasorPin n2;
  Resistor res1(R = R);
  Inductor ind1(L = L / 2, fo = fo);
  Resistor res2(R = R);
  Inductor ind2(L = L / 2, fo = fo);
  parameter Length l = 1000 "line length";
  parameter Length r = 0.002 "wire radius";
  parameter Length d = 0.1 "conductors distance";
  parameter Modelica.SIunits.Frequency fo = 50 "nominal f";
protected
  constant Resistivity ro = 1.68e-8 "copper";
  parameter Resistance R = ro * l / (pi * r ^ 2 / 4);
  parameter Inductance L = 4e-7 * log(d / (0.7788 * r));
equation
  connect(p1, res1.p);
  connect(res1.n, ind1.p);
  connect(ind1.n, p2);
  connect(n1, ind2.n);
  connect(ind2.p, res2.n);
  connect(res2.p, n2);
end TransmissionLine;
```

EEB.Components.BaseComponents.Electrical.Phasors.PQmeter





Connectors

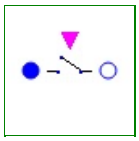
Type	Name	Description
PositivePhasorPin	p1	
PositivePhasorPin	p2	
NegativePhasorPin	n1	
NegativePhasorPin	n2	
output RealOutput	P	
output RealOutput	Q	

Modelica definition

```
model PQmeter
  Interfaces.Electrical.PositivePhasorPin p1;
  Interfaces.Electrical.PositivePhasorPin p2;
  Interfaces.Electrical.NegativePhasorPin n1;
```

```
  Interfaces.Electrical.NegativePhasorPin n2;
  Modelica.Blocks.Interfaces.RealOutput P;
  Modelica.Blocks.Interfaces.RealOutput Q;
protected
  Voltage Vre, Vim;
  Current Ire, Iim;
equation
  Vre = p1.vre - n1.vre;
  Vim = p1.vim - n1.vim;
  Ire = p1.ire;
  Iim = p1.iim;
  P = Vre * Ire + Vim * Iim;
  Q = Vim * Ire - Vre * Iim;
  connect(p1, p2);
  connect(n1, n2);
end PQmeter;
```

EEB.Components.BaseComponents.Electrical.Phasors.Switch



Information

Extends from EEB.Interfaces.Electrical.PhasorTwoPin (Phasor two pins).

Parameters

Type	Name	Default	Description
Time	TC	0.001	Time constant [s]
Resistance	Ron	1e-6	[Ohm]
Conductance	Goff	1e-6	[S]

Connectors

Type	Name	Description
PositivePhasorPin	p	
NegativePhasorPin	n	
input BooleanInput	close	

Modelica definition

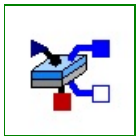
```
model Switch
  extends EEB.Interfaces.Electrical.PhasorTwoPin;
  parameter Time TC = 0.001 "Time constant";
  parameter Resistance Ron = 1e-6;
  parameter Conductance Goff = 1e-6;
  Conductance G;
  Modelica.Blocks.Interfaces.BooleanInput close;
initial equation
  G = if close then 1 / Ron else Goff;
equation
  G + TC * der(G) = if close then 1 / Ron else Goff;
  Ire = G * Vre;
  Iim = G * Vim;
end Switch;
```

EEB.Components.BaseComponents.Electrical.PV

Package Content

Name	Description
Cell	
Module	

EEB.Components.BaseComponents.Electrical.PV.Cell



Parameters

Type	Name	Default	Description
Current	irs	1e-6	Diode reverse seturation current [A]
Real	dif	5	Diode ideality factor
Resistance	Rp	1e3	Parallel (shunt) resistance [Ohm]
Resistance	Rs	0.1	Series resistance [Ohm]
Current	il0	0.2	Nominal light (photo) current [A]
HeatFlux	Phi0	1000	Nominal heat flux [W/m2]
Temperature	T0	293.15	Nominal temperature [K]
Real	Kphi	-0.01	Light (photo) current temperature corfficient
HeatCapacity	C	5	Total heat capacity [J/K]
Temperature	Tstart	293.15	Initial temperature [K]

Connectors

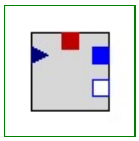
Type	Name	Description
PositivePin	p	
NegativePin	n	
HeatPort_a	surf	
input RealInput	Phi	

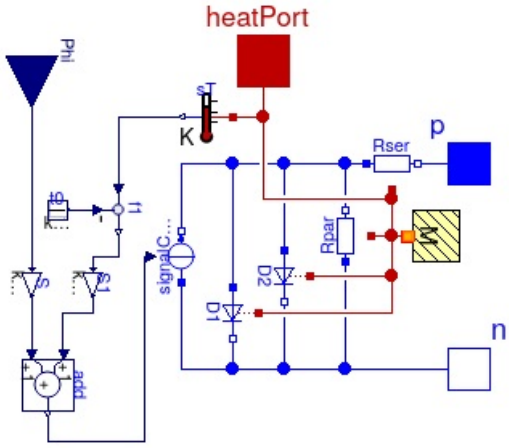
Modelica definition

```
model Cell
  constant ElectricCharge q = 1.6021892e-19 "Elementary (electron) charge";
  constant Real k = 1.380648813e-23 "Boltzmann contant (J/K)";
  parameter Current irs = 1e-6 "Diode reverse seturation current";
  parameter Real dif = 5 "Diode ideality factor";
  parameter Resistance Rp = 1e3 "Parallel (shunt) resistance";
  parameter Resistance Rs = 0.1 "Series resistance";
  parameter Current il0 = 0.2 "Nominal light (photo) current";
  parameter HeatFlux Phi0 = 1000 "Nominal heat flux";
  parameter Temperature T0 = 293.15 "Nominal temperature";
  parameter Real Kphi = -0.01 "Light (photo) current temperature corfficient";
  parameter HeatCapacity C = 5 "Total heat capacity";
  parameter Temperature Tstart = 293.15 "Initial temperature";
  Voltage vj "Junction voltage";
  Current il "Light (photo) current";
  Current id "Diode current";
  Current ip "Parallel (shunt) current";
  Temperature T(start = Tstart);
  Voltage v;
  Current i;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
  Modelica.Thermal.HeatTransfer.Interfaces.HeatPort_a surf;
  Modelica.Blocks.Interfaces.RealInput Phi;
equation
  v = p.v - n.v;
  0 = p.i + n.i;
```

```
  i = -p.i;
  T = surf.T;
  il = il0 * Phi / Phi0 * (1 + Kphi * (T - T0));
  id = irs * (exp(q * vj / dif / k / T) - 1);
  vj = Rp * ip;
  v = vj - Rs * i;
  i = il - id - ip;
  C * der(T) = surf.Q_flow + vj * id + vj * ip + (vj - v) * i;
end Cell;
```

EEB.Components.BaseComponents.Electrical.PV.Module





Parameters

Type	Name	Default	Description
Resistance	Ron_d	1.E-5	Diodes Ron [Ohm]
Conductance	Goff_d	1.E-5	Diodes Goff [S]
Voltage	Vt_d	0.7	Diodes threshold V [V]
Resistance	Rp	200	Module parallel R [Ohm]
Resistance	Rs	0.02	Module series R [Ohm]
Current	Isc	8.33	Short circuit current [A]
Real	alpha	0.02	Current/temperature gain
HeatFlux	Irr0	1000	Reference radiative flux [W/m2]
Temperature	T0	300	Reference temperature [K]
Mass	M	19	Module mass [kg]
SpecificHeatCapacity	cp	300	Module cp [J/(kg.K)]

Connectors

Type	Name	Description
PositivePin	p	
NegativePin	n	
HeatPort_a	heatPort	
input RealInput	Phi	

Modelica definition

```
model Module
  parameter Resistance Ron_d = 1.E-5 "Diodes Ron";
  parameter Conductance Goff_d = 1.E-5 "Diodes Goff";
  parameter Voltage Vt_d = 0.7 "Diodes threshold V";
  parameter Resistance Rp = 200 "Module parallel R";
  parameter Resistance Rs = 0.02 "Module series R";
  parameter Current Isc = 8.33 "Short circuit current";
  parameter Real alpha = 0.02 "Current/temperature gain";
  parameter HeatFlux Irr0 = 1000 "Reference radiative flux";
  parameter Temperature T0 = 300 "Reference temperature";
  parameter Mass M = 19 "Module mass";
  parameter SpecificHeatCapacity cp = 300 "Module cp";
  Modelica.Electrical.Analog.Ideal.IdealDiode D1(Ron = Ron_d, Goff = Goff_d, Vknee = Vt_d, useHeatPort = true);
  Modelica.Electrical.Analog.Ideal.IdealDiode D2(Ron = Ron_d, Goff = Goff_d, Vknee = Vt_d, useHeatPort = true);
  Modelica.Electrical.Analog.Sources.SignalCurrent signalCurrent;
```

```
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
  Modelica.Electrical.Analog.Basic.HeatingResistor Rpar(R_ref = Rp);
  Modelica.Electrical.Analog.Basic.HeatingResistor Rser(R_ref = Rs);
  Thermal.Capacities.MassT heatCap(Tstart = T0, M = M, cp = cp);
  Modelica.Thermal.HeatTransfer.Interfaces.HeatPort_a heatPort;
  Modelica.Blocks.Interfaces.RealInput Phi;
  Modelica.Blocks.Math.Gain S(k = Isc / Irr0);
  Modelica.Blocks.Math.Gain S1(k = alpha);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sT;
  Modelica.Blocks.Math.Feedback f1;
  Modelica.Blocks.Sources.Constant t0(k = T0);
  Modelica.Blocks.Math.Add add;
equation
  connect(Rser.n, p);
  connect(Rser.p, Rpar.n);
  connect(Rser.p, D2.p);
  connect(Rser.p, D1.p);
  connect(Rser.p, signalCurrent.n);
  connect(n, Rpar.p);
  connect(n, D2.n);
  connect(n, D1.n);
  connect(n, signalCurrent.p);
  connect(heatPort, Rser.heatPort);
  connect(Phi, S.u);
  connect(sT.port, heatPort);
  connect(sT.T, f1.u1);
  connect(t0.y, f1.u2);
  connect(f1.y, S1.u);
  connect(S.y, add.u2);
  connect(add.y, signalCurrent.i);
  connect(S1.y, add.u1);
  connect(Rser.heatPort, heatCap.surf);
  connect(Rpar.heatPort, heatCap.surf);
  connect(D2.heatPort, heatCap.surf);
  connect(D1.heatPort, heatCap.surf);
end Module;
```

EEB.Components.AggregateComponents

Package Content

Name	Description
Electrical	
Heating	
Envelope	
Test	

EEB.Components.AggregateComponents.Envelope

Package Content

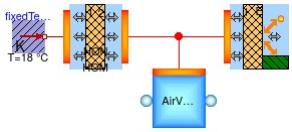
Name	Description
Walls	
Openings	
Test	

EEB.Components.AggregateComponents.Envelope.Test

Package Content

Name	Description
Test_walls	
Test_windows	
Test_windows_openings_int	
Test_windows_openings_ext	
Test_doors_openings_int	
Test_doors_openings_ext	

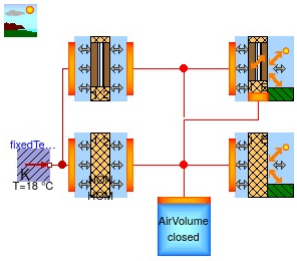
EEB.Components.AggregateComponents.Envelope.Test.Test_walls



Modelica definition

```
model Test_walls
  Walls.ExtenallWall_NoOpenings_Homogeneous WE(lambda = 0.2);
  inner BoundaryConditions.AmbientConditions ambient_settings( Ta_Yex = 20, Ta_avg = 288.15,acv = EEB.Types.AmbCondVariabilty.ACV_constant);
  BaseComponents.Air.Volumes.AirVolume Room(V = 20);
  Walls.InternalWall_NoOpenings_NonHomogeneous WI;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 291.15);
equation
  connect(WE.airInt, Room.heatPort);
  connect(Room.heatPort, WI.airSide2);
  connect(fixedTemperature.port, WI.airSide1);
end Test_walls;
```

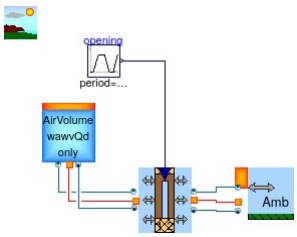
EEB.Components.AggregateComponents.Envelope.Test.Test_windows



Modelica definition

```
model Test_windows
  Walls.ExtenallWall_NoOpenings_Homogeneous WE(lambda = 0.2);
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15, Ta_Yex = 20);
  BaseComponents.Air.Volumes.AirVolume_closed Room(V = 20);
  Walls.InternalWall_NoOpenings_NonHomogeneous WI;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 291.15);
  Openings.InternalWindow_Closed_DoubleGlass internalWindow_SingleGlass(material_glass = EEB.Media.Materials.Glasses.Glass(), material_gas = EEB.Media.Materials.EnvelopeGases.Argon());
  Openings.ExtenallWindow_Closed_DoubleGlass extenallWindow_DoubleGlass(L = 2, H = 2, s = 0.02, material_glass = EEB.Media.Materials.Glasses.Glass(), material_gas = EEB.Media.Materials.Enve
equation
  connect(WE.airInt, Room.heatPort);
  connect(Room.heatPort, WI.airSide2);
  connect(fixedTemperature.port, WI.airSide1);
  connect(fixedTemperature.port, internalWindow_SingleGlass.airSide1);
  connect(WI.airSide2, internalWindow_SingleGlass.airSide2);
  connect(extenallWindow_DoubleGlass.airInt, internalWindow_SingleGlass.airSide2);
  connect(extenallWindow_DoubleGlass.absToWall, Room.heatPort);
end Test_windows;
```

EEB.Components.AggregateComponents.Envelope.Test.Test_windows_openings_int

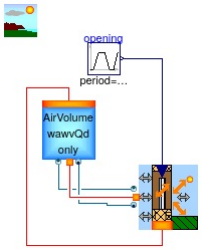


Modelica definition

```
model Test_windows_openings_int
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15, Ta_Yex = 20);
  BaseComponents.Air.Volumes.AirVolume_only_wavvQdport V1(V = 20);
  Openings.InternalWindow_Opening_SingleGlass Window(material = EEB.Media.Materials.Glasses.Glass());
  BaseComponents.Ambient.AmbientAirTempWithOpenings Amb;
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, startTime = 100, width = 600, period = 3600, amplitude = 1);
equation
  connect(opening.y, Window.opening01);
  connect(Window.Bdryair, Amb.dryair);
  connect(Amb.diffuse, Window.Bdiffuse);
  connect(Window.Bvapour, Amb.vapour);
  connect(Window.dryair, V1.dryair);
  connect(V1.diffuse, Window.diffuse);
  connect(Window.vapour, V1.vapour);
```

```
end Test_windows_openings_int;
```

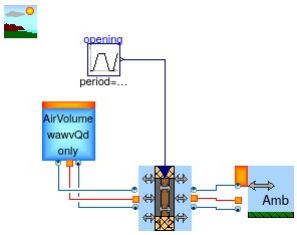
EEB.Components.AggregateComponents.Envelope.Test.Test_windows_openings_ext



Modelica definition

```
model Test_windows_openings_ext
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15, Ta_Yex = 20);
  BaseComponents.Air.Volumes.AirVolume_only_wawvQdport V1(V = 20);
  Openings.ExtenallWindow_Opening_DoubleGlass Window(material_glass = EEB.Media.Materials.Glasses.Glass(), material_gas = EEB.Media.Materials.EnvelopeGases.Air());
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, startTime = 100, width = 600, period = 3600, amplitude = 1);
equation
  connect(opening.y, Window.opening01);
  connect(V1.heatPort, Window.absToWall);
  connect(V1.dryair, Window.dryair);
  connect(V1.vapour, Window.vapour);
  connect(Window.diffuse, V1.diffuse);
end Test_windows_openings_ext;
```

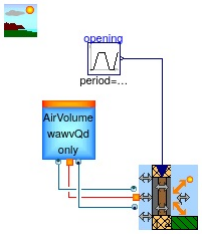
EEB.Components.AggregateComponents.Envelope.Test.Test_doors_openings_int



Modelica definition

```
model Test_doors_openings_int
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15, Ta_Yex = 20);
  BaseComponents.Air.Volumes.AirVolume_only_wawvQdport V1(V = 20);
  Openings.InternalDoor_Opening Door;
  BaseComponents.Ambient.AmbientAirTempWithOpenings Amb;
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, startTime = 100, width = 600, period = 3600, amplitude = 1);
equation
  connect(opening.y, Door.opening01);
  connect(V1.vapour, Door.vapour);
  connect(Door.diffuse, V1.diffuse);
  connect(V1.dryair, Door.dryair);
  connect(Door.Bdryair, Amb.dryair);
  connect(Door.Bdiffuse, Amb.diffuse);
  connect(Door.Bvapour, Amb.vapour);
end Test_doors_openings_int;
```

EEB.Components.AggregateComponents.Envelope.Test.Test_doors_openings_ext



Modelica definition

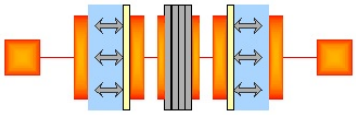
```
model Test_doors_openings_ext
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15, Ta_Yex = 20);
  BaseComponents.Air.Volumes.AirVolume_only_wawvQdport V1(V = 20);
  Openings.ExternallDoor_Opening Door;
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, startTime = 100, width = 600, period = 3600, amplitude = 1);
equation
  connect(opening.y, Door.opening01);
  connect(V1.diffuse, Door.diffuse);
  connect(Door.dryair, V1.dryair);
  connect(V1.vapour, Door.vapour);
end Test_doors_openings_ext;
```

EEB.Components.AggregateComponents.Envelope.Walls

Package Content

Name	Description
BaseClasses	
InternalWall_NoOpenings_Homogeneous	
InternalWall_NoOpenings_NonHomogeneous	
ExtenallWall_NoOpenings_Homogeneous	
ExtenallWall_NoOpenings_NonHomogeneous	

EEB.Components.AggregateComponents.Envelope.Walls.InternalWall_NoOpenings_Homogeneous



Information

Extends from BaseClasses.BaseInternalWall_NoOpenings.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

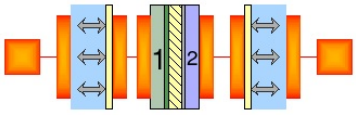
Connectors

Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
model InternalWall_NoOpenings_Homogeneous
  extends BaseClasses.BaseInternalWall_NoOpenings;
  BaseComponents.Envelope.SolidMultilayer_Homogeneous MultiLayerWall(A = L * H, s = s, ro = ro, cp = cp, lambda = lambda, n = n, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 2400 "wall density";
  parameter SI.SpecificHeatCapacity cp = 880 "wall cp";
  parameter SI.ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
equation
  connect(ConvSideA.wall, MultiLayerWall.side1);
  connect(MultiLayerWall.side2, ConvSideB.wall);
  connect(airSide1, ConvSideA.air);
  connect(ConvSideB.air, airSide2);
end InternalWall_NoOpenings_Homogeneous;
```

EEB.Components.AggregateComponents.Envelope.Walls.InternalWall_NoOpenings_NonHomogeneous



Information

Extends from BaseClasses.BaseInternalWall_NoOpenings.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s[:]	{0.05,0.4,0.05}	layer thicknesses [m]
Density	ro[:]	{1600,2400,1600}	layer densities [kg/m3]
SpecificHeatCapacity	cp[:]	{400,880,400}	layer cps [J/(kg.K)]
ThermalConductivity	lambda[:]	{0.2,1.91,0.2}	layer thermal conds [W/(m.K)]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

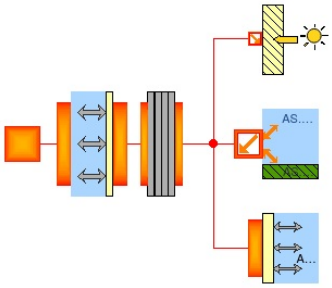
Connectors

Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
model InternalWall_NoOpenings_NonHomogeneous
  extends BaseClasses.BaseInternalWall_NoOpenings;
  BaseComponents.Envelope.SolidMultilayer_NonHomogeneous MultiLayerWall(A = L * H, s = s, ro = ro, cp = cp, lambda = lambda, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter Length s[:] = {0.05, 0.4, 0.05} "layer thicknesses";
  parameter Density ro[:] = {1600, 2400, 1600} "layer densities";
  parameter SpecificHeatCapacity cp[:] = {400, 880, 400} "layer cps";
  parameter ThermalConductivity lambda[:] = {0.2, 1.91, 0.2} "layer thermal conds";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  //parameter Integer n=size(s,1) "number of layers";
equation
  connect(ConvSideA.wall, MultiLayerWall.side1);
  connect(MultiLayerWall.side2, ConvSideB.wall);
  connect(airSide1, ConvSideA.air);
  connect(ConvSideB.air, airSide2);
end InternalWall_NoOpenings_NonHomogeneous;
```

EEB.Components.AggregateComponents.Envelope.Walls.ExtenalWall_NoOpenings_Homogeneous



Information

Extends from BaseClasses.BaseExternalWall_NoOpenings.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient

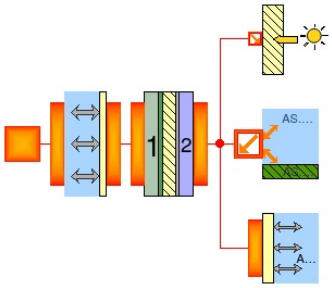
Connectors

Type	Name	Description
HeatPort	airInt	

Modelica definition

```
model ExtenalWall_NoOpenings_Homogeneous
  extends BaseClasses.BaseExternalWall_NoOpenings;
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 2400 "wall density";
  parameter SI.SpecificHeatCapacity cp = 880 "wall cp";
  parameter SI.ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  BaseComponents.Envelope.SolidMultilayer_Homogeneous MultiLayerWall(A = L * H, s = s, ro = ro, cp = cp, lambda = lambda, n = n, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_OpaqueSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
equation
  connect(ConvInt.wall, MultiLayerWall.side1);
  connect(airInt, ConvInt.air);
  connect(MultiLayerWall.side2, RadToSkyGround.wall);
  connect(MultiLayerWall.side2, solarRadiation_OpaqueSurf.Absorbed);
  connect(MultiLayerWall.side2, ConvToExt.wall);
end ExtenalWall_NoOpenings_Homogeneous;
```

EEB.Components.AggregateComponents.Envelope.Walls.ExtenalWall_NoOpenings_NonHomogeneous



Information

Extends from BaseClasses.BaseExternalWall_NoOpenings.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s[:]	{0.05,0.4,0.05}	layer thicknesses [m]
Density	ro[:]	{1600,2400,1600}	layer densities [kg/m3]
SpecificHeatCapacity	cp[:]	{400,880,400}	layer cps [J/(kg.K)]
ThermalConductivity	lambda[:]	{0.2,1.91,0.2}	layer thermal conds [W/(m.K)]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient

Connectors

Type	Name	Description
HeatPort	airInt	

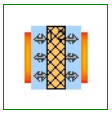
Modelica definition

```
model ExtenalWall_NoOpenings_NonHomogeneous
  extends BaseClasses.BaseExternalWall_NoOpenings;
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter Length s[:] = {0.05, 0.4, 0.05} "layer thicknesses";
  parameter Density ro[:] = {1600, 2400, 1600} "layer densities";
  parameter SpecificHeatCapacity cp[:] = {400, 880, 400} "layer cps";
  parameter ThermalConductivity lambda[:] = {0.2, 1.91, 0.2} "layer thermal conds";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  BaseComponents.Envelope.SolidMultilayer_NonHomogeneous MultiLayerWall(A = L * H, s = s, ro = ro, cp = cp, lambda = lambda, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_OpaqueSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
equation
  connect(ConvInt.wall, MultiLayerWall.side1);
  connect(airInt, ConvInt.air);
  connect(MultiLayerWall.side2, RadToSkyGround.wall);
  connect(MultiLayerWall.side2, solarRadiation_OpaqueSurf.Absorbed);
  connect(MultiLayerWall.side2, ConvToExt.wall);
end ExtenalWall_NoOpenings_NonHomogeneous;
```

EEB.Components.AggregateComponents.Envelope.Walls.BaseClasses

Package Content

Name	Description
BaseInternalWall_NoOpenings	
BaseExternalWall_NoOpenings	



EEB.Components.AggregateComponents.Envelope.Walls.BaseClasses.BaseInternalWall_NoOpenings

Connectors

Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
partial model BaseInternalWall_NoOpenings
  Interfaces.Thermal.HeatPort airSide1;
  Interfaces.Thermal.HeatPort airSide2;
end BaseInternalWall_NoOpenings;
```

EEB.Components.AggregateComponents.Envelope.Walls.BaseClasses.BaseExternalWall_NoOpenings

Connectors

Type	Name	Description
HeatPort	airInt	

Modelica definition

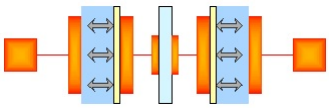
```
partial model BaseExternalWall_NoOpenings
  Interfaces.Thermal.HeatPort airInt;
end BaseExternalWall_NoOpenings;
```

EEB.Components.AggregateComponents.Envelope.Openings

Package Content

Name	Description
BaseClasses	
InternalWindow_Closed_SingleGlass	
InternalWindow_Closed_DoubleGlass	
ExtenallWindow_Closed_SingleGlass	
ExtenallWindow_Closed_DoubleGlass	
InternalWindow_Opening_SingleGlass	
InternalWindow_Opening_DoubleGlass	
ExtenallWindow_Opening_SingleGlass	
ExtenallWindow_Opening_DoubleGlass	
InternalDoor_Opening	
ExternallDoor_Opening	

EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_SingleGlass



Information

Extends from BaseClasses.BaseInternalWindow_Closed.

Parameters

Type	Name	Default	Description
Length	L	5	glass surface length [m]
Length	H	3	glass surface height [m]
Length	s	0.4	glass thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]
BaseGlass	material		Type of glass

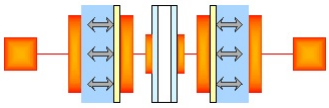
Connectors

Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
model InternalWindow_Closed_SingleGlass
  extends BaseClasses.BaseInternalWindow_Closed;
  BaseComponents.Envelope.GlassLayer Glass(L = L, H = H, d = s, Tstart = Tstart, material = material);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "glass surface length";
  parameter SI.Length H = 3 "glass surface height";
  parameter SI.Length s = 0.4 "glass thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
equation
  connect(ConvSideA.wall, Glass.side1);
  connect(Glass.side2, ConvSideB.wall);
  connect(airSide1, ConvSideA.air);
  connect(ConvSideB.air, airSide2);
end InternalWindow_Closed_SingleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_DoubleGlass



Information

Extends from BaseClasses.BaseInternalWindow_Closed.

Parameters

Type	Name	Default	Description
Length	L	5	glass surface length [m]
Length	H	3	glass surface height [m]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]
Length	d_glass	0.02	Width of the single glass [m]
Length	d_int	0.05	Distance between two glasses [m]
BaseGlass	material_glass		Type of glass
BaseEnvelopeGas	material_gas		Type of internal gas between the glasses
CoefficientOfHeatTransfer	gamma_g	5	Heat transfer coefficient between the glasses and internal gas [W/(m2.K)]

Connectors

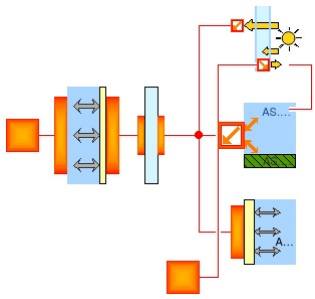
Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
model InternalWindow_Closed_DoubleGlass
  extends BaseClasses.BaseInternalWindow_Closed;
  BaseComponents.Envelope.DoubleGlass DoubleGlass(L = L, H = H, Tstart = Tstart, d_glass = d_glass, d_int = d_int, material_glass = material_glass, material_gas = material_gas, gamma =
```

```
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "glass surface length";
  parameter SI.Length H = 3 "glass surface height";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  parameter SI.Length d_glass = 0.02 "Width of the single glass";
  parameter SI.Length d_int = 0.05 "Distance between two glasses";
  parameter Media.Materials.Glasses.BaseGlass material_glass "Type of glass";
  parameter Media.Materials.EnvelopeGases.BaseEnvelopeGas material_gas "Type of internal gas between the glasses";
  parameter SI.CoefficientOfHeatTransfer gamma_g = 5 "Heat transfer coefficient between the glasses and internal gas";
equation
  connect(ConvSideA.wall, DoubleGlass.side1);
  connect(DoubleGlass.side2, ConvSideB.wall);
  connect(airSide1, ConvSideA.air);
  connect(ConvSideB.air, airSide2);
end InternalWindow_Closed_DoubleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.ExtenalWindow_Closed_SingleGlass



Information

Extends from BaseClasses.BaseExternalWindow_Closed.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient
BaseGlass	material		Type of glass

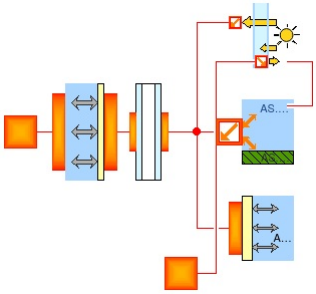
Connectors

Type	Name	Description
HeatPort	airInt	
HeatPort	absToWall	

Modelica definition

```
model ExtenalWindow_Closed_SingleGlass
  extends BaseClasses.BaseExternalWindow_Closed;
  BaseComponents.Envelope.GlassLayer Glass(Tstart = Tstart, L = L, H = H, d = s, material = material);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 2400 "wall density";
  parameter SI.SpecificHeatCapacity cp = 880 "wall cp";
  parameter SI.ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_TransparentSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
equation
  connect(ConvInt.wall, Glass.side1);
  connect(airInt, ConvInt.air);
  connect(solarRadiation_OpaqueSurf.Trasmitted, Glass.side2);
  connect(RadToSkyGround.wall, Glass.side2);
  connect(ConvToExt.wall, Glass.side2);
  connect(solarRadiation_OpaqueSurf.Absorbed, absToWall);
  connect(solarRadiation_OpaqueSurf.Reflected, RadToSkyGround.SkyTemp);
end ExtenalWindow_Closed_SingleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.ExtenalWindow_Closed_DoubleGlass



Information

Extends from BaseClasses.BaseExternalWindow_Closed.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient
Length	d_glass	0.02	Width of the single glass [m]
Length	d_int	0.05	Distance between two glasses [m]
BaseGlass	material_glass		Type of glass
BaseEnvelopeGas	material_gas		Type of internal gas between the glasses
CoefficientOfHeatTransfer	gamma_g	5	Heat transfer coefficient between the glasses and internal gas [W/(m2.K)]

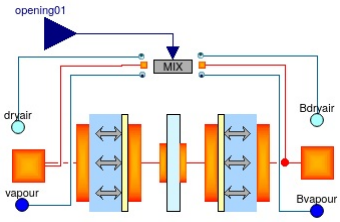
Connectors

Type	Name	Description
HeatPort	airInt	
HeatPort	absToWall	

Modelica definition

```
model ExtenallWindow_Closed_DoubleGlass
  extends BaseClasses.BaseExternalWindow_Closed;
  BaseComponents.Envelope.DoubleGlass Glass(Tstart = Tstart, L = L, H = H, material_glass = material_glass, material_gas = material_gas);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 2400 "wall density";
  parameter SI.SpecificHeatCapacity cp = 880 "wall cp";
  parameter SI.ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_TransparentSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  parameter SI.Length d_glass = 0.02 "Width of the single glass";
  parameter SI.Length d_int = 0.05 "Distance between two glasses";
  parameter Media.Materials.Glasses.BaseGlass material_glass "Type of glass";
  parameter Media.Materials.EnvelopeGases.BaseEnvelopeGas material_gas "Type of internal gas between the glasses";
  parameter SI.CoefficientOfHeatTransfer gamma_g = 5 "Heat transfer coefficient between the glasses and internal gas";
equation
  connect(ConvInt.wall, Glass.side1);
  connect(airInt, ConvInt.air);
  connect(solarRadiation_OpaqueSurf.Trasmitted, Glass.side2);
  connect(RadToSkyGround.wall, Glass.side2);
  connect(ConvToExt.wall, Glass.side2);
  connect(solarRadiation_OpaqueSurf.Absorbed, absToWall);
  connect(solarRadiation_OpaqueSurf.Reflected, RadToSkyGround.SkyTemp);
end ExtenallWindow_Closed_DoubleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Opening_SingleGlass



Information

Extends from BaseClasses.BaseInternalWindow_Opening.

Parameters

Type	Name	Default	Description
Length	L	5	glass surface length [m]
Length	H	3	glass surface height [m]
Length	s	0.4	glass thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]
BaseGlass	material		Type of glass
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

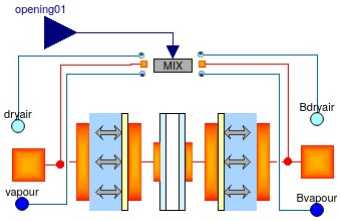
Connectors

Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	
MoistAirFlange_wawvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
model InternalWindow_Opening_SingleGlass
  extends BaseClasses.BaseInternalWindow_Opening;
  BaseComponents.Envelope.GlassLayer Glass(L = L, H = H, d = s, Tstart = Tstart, material = material);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "glass surface length";
  parameter SI.Length H = 3 "glass surface height";
  parameter SI.Length s = 0.4 "glass thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
equation
  connect(ConvSideA.wall, Glass.side1);
  connect(Glass.side2, ConvSideB.wall);
  connect(opening01, mix.cmd01);
  connect(ConvSideB.air, Bdiffuse);
  connect(ConvSideA.air, diffuse);
  connect(mix.diffuse, diffuse);
  connect(mix.Bdiffuse, Bdiffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
  connect(mix.Bdryair, Bdryair);
  connect(mix.Bvapour, Bvapour);
end InternalWindow_Opening_SingleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Opening_DoubleGlass



Information

Extends from BaseClasses.BaseInternalWindow_Opening.

Parameters

Type	Name	Default	Description
Length	L	5	glass surface length [m]
Length	H	3	glass surface height [m]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	Initial temperature, all layers [K]
BaseGlass	material		Type of glass
Length	d_glass	0.02	Width of the single glass [m]
Length	d_int	0.05	Distance between two glasses [m]
BaseGlass	material_glass		Type of glass
BaseEnvelopeGas	material_gas		Type of internal gas between the glasses
CoefficientOfHeatTransfer	gamma_g	5	Heat transfer coefficient between the glasses and internal gas [W/(m2.K)]
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

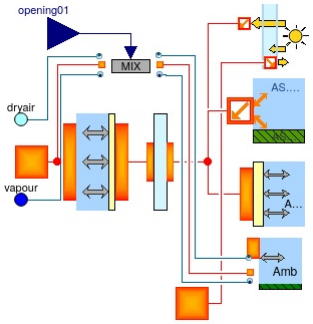
Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	

MoistAirFlange_wawvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
model InternalWindow_Opening_DoubleGlass
  extends BaseClasses.BaseInternalWindow_Opening;
  BaseComponents.Envelope.DoubleGlass Glass(L = L, H = H, Tstart = Tstart, d_glass = d_glass, d_int = d_int, gamma = gamma_g, material_glass = material_glass, material_gas = material_g
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideA(L = L, H = H, vertical = vertical);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvSideB(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "glass surface length";
  parameter SI.Length H = 3 "glass surface height";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Temperature Tstart = 273.15 + 25 "Initial temperature, all layers";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
  parameter SI.Length d_glass = 0.02 "Width of the single glass";
  parameter SI.Length d_int = 0.05 "Distance between two glasses";
  parameter Media.Materials.Glasses.BaseGlass material_glass "Type of glass";
  parameter Media.Materials.EnvelopeGases.BaseEnvelopeGas material_gas "Type of internal gas between the glasses";
  parameter SI.CoefficientOfHeatTransfer gamma_g = 5 "Heat transfer coefficient between the glasses and internal gas";
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
equation
  connect(ConvSideA.wall, Glass.side1);
  connect(Glass.side2, ConvSideB.wall);
  connect(opening01, mix.cmd01);
  connect(mix.Bdiffuse, Bdiffuse);
  connect(ConvSideB.air, Bdiffuse);
  connect(mix.diffuse, diffuse);
  connect(ConvSideA.air, diffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
  connect(mix.Bvapour, Bvapour);
  connect(mix.Bdryair, Bdryair);
end InternalWindow_Opening_DoubleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_SingleGlass



Information

Extends from BaseClasses.BaseExternalWindow_Opening.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.01	Absorption coefficient
Real	trasCoef	0.9	Trasmitted coefficient
BaseGlass	material		Type of glass
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

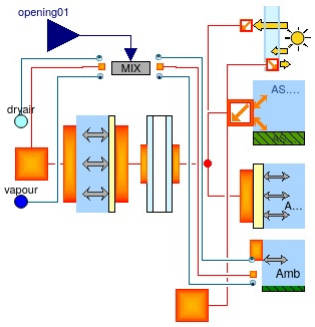
Type	Name	Description
HeatPort	absToWall	
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
model ExtenallWindow_Opening_SingleGlass
  extends BaseClasses.BaseExternalWindow_Opening;
  BaseComponents.Envelope.GlassLayer Glass(Tstart = Tstart, L = L, H = H, d = s, material = material);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  //   parameter SI.Density ro=2400 "wall density";
  //   parameter SI.SpecificHeatCapacity cp=880 "wall cp";
  //   parameter SI.ThermalConductivity lambda=1.91 "wall thermal cond";
  //   parameter Integer n=4 "number of layers";
```

```
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_TransparentSurf solarRadiation_TransparentSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination, tras
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.01 "Absorption coefficient";
  parameter Real trasCoef = 0.9 "Trasmitted coefficient";
  parameter Media.Materials.Glasses.BaseGlass material "Type of glass";
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  EEB.Components.BaseComponents.Ambient.AmbientAirTempWithOpenings ambientAirTempWithOpenings;
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
equation
  connect(ConvInt.wall, Glass.side1);
  connect(solarRadiation_TransparentSurf.Trasmitted, Glass.side2);
  connect(ConvToExt.wall, Glass.side2);
  connect(solarRadiation_TransparentSurf.Absorbed, absToWall);
  connect(RadToSkyGround.wall, Glass.side2);
  connect(opening01, mix.cmd01);
  connect(ConvInt.air, diffuse);
  connect(mix.diffuse, diffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
  connect(mix.Bvapour, ambientAirTempWithOpenings.vapour);
  connect(ambientAirTempWithOpenings.diffuse, mix.Bdiffuse);
  connect(mix.Bdryair, ambientAirTempWithOpenings.dryair);
end ExtenallWindow_Opening_SingleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_DoubleGlass



Information

Extends from BaseClasses.BaseExternalWindow_Opening.

Parameters

Type	Name	Default	Description
Length	L	5	wall surface length [m]
Length	H	3	wall surface height [m]
Length	s	0.4	wall thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	2400	wall density [kg/m3]
SpecificHeatCapacity	cp	880	wall cp [J/(kg.K)]
ThermalConductivity	lambda	1.91	wall thermal cond [W/(m.K)]
Integer	n	4	number of layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient
Length	d_glass	0.02	Width of the single glass [m]
Length	d_int	0.05	Distance between two glasses [m]
BaseGlass	material_glass		Type of glass
BaseEnvelopeGas	material_gas		Type of internal gas between the glasses
CoefficientOfHeatTransfer	gamma_g	5	Heat transfer coefficient between the glasses and internal gas [W/(m2.K)]
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

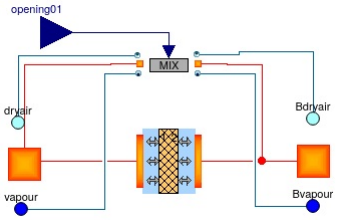
Type	Name	Description
HeatPort	absToWall	
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
model ExtenallWindow_Opening_DoubleGlass
  extends BaseClasses.BaseExternalWindow_Opening;
  BaseComponents.Envelope.DoubleGlass Glass(Tstart = Tstart, L = L, H = H, d_glass = d_glass, d_int = d_int, gamma = gamma_g, material_glass = material_glass, material_gas = material_g
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BI74 ConvInt(L = L, H = H, vertical = vertical);
  parameter SI.Length L = 5 "wall surface length";
  parameter SI.Length H = 3 "wall surface height";
  parameter SI.Length s = 0.4 "wall thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
```

```
  parameter SI.Density ro = 2400 "wall density";
  parameter SI.SpecificHeatCapacity cp = 880 "wall cp";
  parameter SI.ThermalConductivity lambda = 1.91 "wall thermal cond";
  parameter Integer n = 4 "number of layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.Sources.SolarRadiation_TransparentSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  parameter SI.Length d_glass = 0.02 "Width of the single glass";
  parameter SI.Length d_int = 0.05 "Distance between two glasses";
  parameter Media.Materials.Glasses.BaseGlass material_glass "Type of glass";
  parameter Media.Materials.EnvelopeGases.BaseEnvelopeGas material_gas "Type of internal gas between the glasses";
  parameter SI.CoefficientOfHeatTransfer gamma_g = 5 "Heat transfer coefficient between the glasses and internal gas";
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  BaseComponents.Ambient.AmbientAirTempWithOpenings ambientAirTempWithOpenings;
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
equation
  connect(ConvInt.wall, Glass.side1);
  connect(solarRadiation_OpaqueSurf.Trasmitted, Glass.side2);
  connect(ConvToExt.wall, Glass.side2);
  connect(solarRadiation_OpaqueSurf.Absorbed, absToWall);
  connect(RadToSkyGround.wall, Glass.side2);
  connect(opening01, mix.cmd01);
  connect(mix.Bdryair, ambientAirTempWithOpenings.dryair);
  connect(mix.Bdiffuse, ambientAirTempWithOpenings.diffuse);
  connect(ambientAirTempWithOpenings.vapour, mix.Bvapour);
  connect(ConvInt.air, diffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
  connect(mix.diffuse, diffuse);
end Extenal1Window_Opening_DoubleGlass;
```

EEB.Components.AggregateComponents.Envelope.Openings.InternalDoor_Opening



Information

Extends from BaseClasses.BaseInternalDoor_Opening.

Parameters

Type	Name	Default	Description
Length	L	1	door width [m]
Length	H	2	door height [m]
Length	s	0.04	door thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	800	door material density [kg/m3]
SpecificHeatCapacity	cp	480	door material cp [J/(kg.K)]
ThermalConductivity	lambda	0.8	door material thermal cond [W/(m.K)]
Integer	n	4	number of door layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

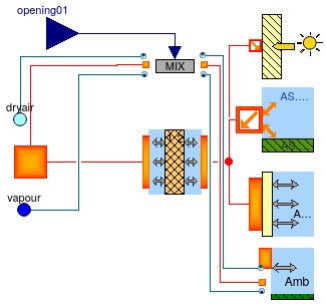
Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	
MoistAirFlange_wawvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
model InternalDoor_Opening
  extends BaseClasses.BaseInternalDoor_Opening;
  Walls.InternalWall_NoOpenings_Homogeneous internalWall_NoOpenings_Homogeneous(L = L, H = H, s = s, vertical = vertical, ro = ro, cp = cp, lambda = lambda, n = n, Tstart = Tstart);
  parameter SI.Length L = 1 "door width";
  parameter SI.Length H = 2 "door height";
  parameter SI.Length s = 0.04 "door thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 800 "door material density";
  parameter SI.SpecificHeatCapacity cp = 480 "door material cp";
  parameter SI.ThermalConductivity lambda = 0.8 "door material thermal cond";
  parameter Integer n = 4 "number of door layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
equation
  connect(opening01, mix.cmd01);
  connect(internalWall_NoOpenings_Homogeneous.airSide1, diffuse);
```

```
  connect(internalWall_NoOpenings_Homogeneous.airSide2, Bdiffuse);
  connect(mix.Bdryair, Bdryair);
  connect(mix.Bdiffuse, Bdiffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
  connect(mix.diffuse, diffuse);
  connect(mix.Bvapour, Bvapour);
end InternalDoor_Opening;
```

EEB.Components.AggregateComponents.Envelope.Openings.ExternalDoor_Opening



Information

Extends from BaseClasses.BaseExternalDoor_Opening.

Parameters

Type	Name	Default	Description
Length	L	1	door width [m]
Length	H	2	door height [m]
Length	s	0.04	door thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	800	door material density [kg/m3]
SpecificHeatCapacity	cp	480	door material cp [J/(kg.K)]
ThermalConductivity	lambda	0.8	door material thermal cond [W/(m.K)]
Integer	n	4	number of door layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]
Boolean	fixedCoeff	false	fixed heat transfer coefficient
CoefficientOfHeatTransfer	h0	25	fixed convective heat transfer [W/(m2.K)]
Real	orientation	0	Orientation of the exiting normal direction relative to North: 0? North, clockwise
Real	es	0.9	Surface emissivity
Real	eg	0.9	Ground emissivity
Real	inclination	0	Inclination of the surface: 90? vertical, 180? horizontal
Real	absCoef	0.9	Absorption coefficient
Pressure	dpnom	10000	Nominal pressure drop [Pa]
MassFlowRate	wnom	0.01	Nominal total mass flowrate [kg/s]
Real	GvOverGa	0.1	Vapour/dry air conductance ratio
ThermalConductance	Gdw0	200	Diffusive thermal cond at w=0 [W/K]
ThermalConductance	Gdwnom	1	Diffusive thermal cond at w=wnom [W/K]

Connectors

Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
model ExternalDoor_Opening
  extends BaseClasses.BaseExternalDoor_Opening;
  Walls.InternalWall_NoOpenings_Homogeneous internalWall_NoOpenings_Homogeneous(L = L, H = H, s = s, vertical = vertical, ro = ro, cp = cp, lambda = lambda, n = n, Tstart = Tstart);
  parameter SI.Length L = 1 "door width";
  parameter SI.Length H = 2 "door height";
  parameter SI.Length s = 0.04 "door thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 800 "door material density";
  parameter SI.SpecificHeatCapacity cp = 480 "door material cp";
  parameter SI.ThermalConductivity lambda = 0.8 "door material thermal cond";
  parameter Integer n = 4 "number of door layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
  parameter Boolean fixedCoeff = false "fixed heat transfer coefficient";
  parameter SI.CoefficientOfHeatTransfer h0 = 25 "fixed convective heat transfer";
  parameter Real orientation = 0 "Orientation of the exiting normal direction relative to North: 0? North, clockwise";
  parameter Real es = 0.9 "Surface emissivity";
  parameter Real eg = 0.9 "Ground emissivity";
  parameter Real inclination = 0 "Inclination of the surface: 90? vertical, 180? horizontal";
  parameter Real absCoef = 0.9 "Absorption coefficient";
  parameter SI.Pressure dpnom = 10000 "Nominal pressure drop";
  parameter SI.MassFlowRate wnom = 0.01 "Nominal total mass flowrate";
  parameter Real GvOverGa = 0.1 "Vapour/dry air conductance ratio";
  parameter SI.ThermalConductance Gdw0 = 200 "Diffusive thermal cond at w=0";
  parameter SI.ThermalConductance Gdwnom = 1 "Diffusive thermal cond at w=wnom";
  BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint_mix_Gcmd01 mix(dpnom = dpnom, wnom = wnom, GvOverGa = GvOverGa, Gdw0 = Gdw0, Gdwnom = Gdwnom);
  BaseComponents.Thermal.Sources.SolarRadiation_OpaqueSurf solarRadiation_OpaqueSurf(L = L, H = H, absCoef = absCoef, orientation = orientation, inclination = inclination);
  BaseComponents.Ambient.Radiaton_SkyGround RadToSkyGround(L = L, H = H, inclination = inclination, es = es, eg = eg);
  BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke ConvToExt(L = L, H = H, fixedCoeff = fixedCoeff, h0 = h0, orientation = orientation);
  BaseComponents.Ambient.AmbientAirTempWithOpenings ambientAirTempWithOpenings;
equation
  connect(opening01, mix.cmd01);
  connect(solarRadiation_OpaqueSurf.Absorbed, internalWall_NoOpenings_Homogeneous.airSide2);
  connect(RadToSkyGround.wall, internalWall_NoOpenings_Homogeneous.airSide2);
  connect(internalWall_NoOpenings_Homogeneous.airSide2, ConvToExt.wall);
  connect(mix.Bdiffuse, ambientAirTempWithOpenings.diffuse);
  connect(mix.Bdryair, ambientAirTempWithOpenings.dryair);
  connect(mix.Bvapour, ambientAirTempWithOpenings.vapour);
  connect(internalWall_NoOpenings_Homogeneous.airSide1, diffuse);
  connect(mix.diffuse, diffuse);
  connect(mix.dryair, dryair);
  connect(mix.vapour, vapour);
end ExternalDoor_Opening;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses

Package Content

Name	Description
BaseInternalWindow_Closed	
BaseExternalWindow_Closed	
BaseInternalWindow_Opening	
BaseExternalWindow_Opening	
BaseInternalDoor_Opening	
BaseExternalDoor_Opening	

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseInternalWindow_Closed

Connectors

Type	Name	Description
HeatPort	airSide1	
HeatPort	airSide2	

Modelica definition

```
partial model BaseInternalWindow_Closed
  Interfaces.Thermal.HeatPort airSide1;
  Interfaces.Thermal.HeatPort airSide2;
end BaseInternalWindow_Closed;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseExternalWindow_Closed

Connectors

Type	Name	Description
HeatPort	airInt	
HeatPort	absToWall	

Modelica definition

```
partial model BaseExternalWindow_Closed
  Interfaces.Thermal.HeatPort airInt;
  Interfaces.Thermal.HeatPort absToWall;
end BaseExternalWindow_Closed;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseInternalWindow_Opening

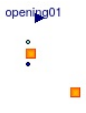
Connectors

Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wawvQd_waPart	Bdryair	
MoistAirFlange_wawvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
partial model BaseInternalWindow_Opening
  //Interfaces.Air.MoistAirFlange_wawvQd airSide1 annotation(Placement(transformation(extent = {{-100, -10}, {-80, 10}}), iconTransformation(extent = {{-110, -20}, {-70, 20}})));
  Modelica.Blocks.Interfaces.RealInput opening01;
  //Interfaces.Air.MoistAirFlange_wawvQd airSide2 annotation(Placement(transformation(extent = {{80, -10}, {100, 10}}), iconTransformation(extent = {{70, -20}, {110, 20}})));
public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart
                                              dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart
                                              vapour;
  EEB.Interfaces.Thermal.HeatPort
                              diffuse;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart
                                              Bdryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart
                                              Bvapour;
  EEB.Interfaces.Thermal.HeatPort
                              Bdiffuse;
end BaseInternalWindow_Opening;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseExternalWindow_Opening



Connectors

Type	Name	Description
HeatPort	absToWall	
input RealInput	opening01	
MoistAirFlange_wavvQd_waPart	dryair	
MoistAirFlange_wavvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

```
partial model BaseExternalWindow_Opening
  //Interfaces.Air.MoistAirFlange_wavvQd airInt annotation(Placement(transformation(extent = {{-100, -10}, {-80, 10}}), iconTransformation(extent = {{-120, -20}, {-80, 20}})));
  Interfaces.Thermal.HeatPort absToWall;
  Modelica.Blocks.Interfaces.RealInput opening01;
public
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_waPart
                                              dryair;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_wvPart
                                              vapour;
  EEB.Interfaces.Thermal.HeatPort
                                diffuse;
end BaseExternalWindow_Opening;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseInternalDoor_Opening

Parameters

Type	Name	Default	Description
Length	L	1	door width [m]
Length	H	2	door height [m]
Length	s	0.04	door thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal
Density	ro	800	door material density [kg/m3]
SpecificHeatCapacity	cp	480	door material cp [J/(kg.K)]
ThermalConductivity	lambda	0.8	door material thermal cond [W/(m.K)]
Integer	n	4	number of door layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

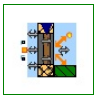
Connectors

Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wavvQd_waPart	dryair	
MoistAirFlange_wavvQd_wvPart	vapour	
HeatPort	diffuse	
MoistAirFlange_wavvQd_waPart	Bdryair	
MoistAirFlange_wavvQd_wvPart	Bvapour	
HeatPort	Bdiffuse	

Modelica definition

```
partial model BaseInternalDoor_Opening
  //Interfaces.Air.MoistAirFlange_wavvQd airSide1 annotation(Placement(transformation(extent = {{-100, -10}, {-80, 10}}), iconTransformation(extent = {{-110, -20}, {-70, 20}})));
  Modelica.Blocks.Interfaces.RealInput opening01;
  //Interfaces.Air.MoistAirFlange_wavvQd airSide2 annotation(Placement(transformation(extent = {{80, -10}, {100, 10}}), iconTransformation(extent = {{70, -20}, {110, 20}})));
  parameter SI.Length L = 1 "door width";
  parameter SI.Length H = 2 "door height";
  parameter SI.Length s = 0.04 "door thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 800 "door material density";
  parameter SI.SpecificHeatCapacity cp = 480 "door material cp";
  parameter SI.ThermalConductivity lambda = 0.8 "door material thermal cond";
  parameter Integer n = 4 "number of door layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
public
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_waPart
                                              dryair;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_wvPart
                                              vapour;
  EEB.Interfaces.Thermal.HeatPort
                                diffuse;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_waPart
                                              Bdryair;
  EEB.Interfaces.Air.MoistAirFlange_wavvQd_wvPart
                                              Bvapour;
  EEB.Interfaces.Thermal.HeatPort
                                Bdiffuse;
end BaseInternalDoor_Opening;
```

EEB.Components.AggregateComponents.Envelope.Openings.BaseClasses.BaseExternalDoor_Opening



Parameters

Type	Name	Default	Description
Length	L	1	door width [m]
Length	H	2	door height [m]
Length	s	0.04	door thickness [m]
Boolean	vertical	true	true for vertical, false for horizontal

Density	ro	800	door material density [kg/m3]
SpecificHeatCapacity	cp	480	door material cp [J/(kg.K)]
ThermalConductivity	lambda	0.8	door material thermal cond [W/(m.K)]
Integer	n	4	number of door layers
Temperature	Tstart	273.15 + 25	initial T, all layers [K]

Connectors

Type	Name	Description
input RealInput	opening01	
MoistAirFlange_wawvQd_waPart	dryair	
MoistAirFlange_wawvQd_wvPart	vapour	
HeatPort	diffuse	

Modelica definition

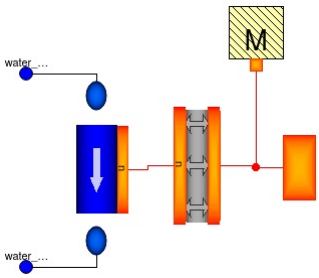
```
partial model BaseExternalDoor_Opening
  //Interfaces.Air.MoistAirFlange_wawvQd airInt annotation(Placement(transformation(extent = {{-100, -10}, {-80, 10}}), iconTransformation(extent = {{-120, -20}, {-80, 20}})));
  Modelica.Blocks.Interfaces.RealInput opening01;
  parameter SI.Length L = 1 "door width";
  parameter SI.Length H = 2 "door height";
  parameter SI.Length s = 0.04 "door thickness";
  parameter Boolean vertical = true "true for vertical, false for horizontal";
  parameter SI.Density ro = 800 "door material density";
  parameter SI.SpecificHeatCapacity cp = 480 "door material cp";
  parameter SI.ThermalConductivity lambda = 0.8 "door material thermal cond";
  parameter Integer n = 4 "number of door layers";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial T, all layers";
public
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_waPart
                                              dryair;
  EEB.Interfaces.Air.MoistAirFlange_wawvQd_wvPart
                                              vapour;
  EEB.Interfaces.Thermal.HeatPort
                              diffuse;
end BaseExternalDoor_Opening;
```

EEB.Components.AggregateComponents.Heating

Package Content

Name	Description
RoomHeater	
RoomHeaterWithValve	
WaterCoilWithWall	
FanCoil	
IdealControlledFluidHeater	
ElectricBoiler	
FuelBoiler	
AirCaptureCoil	
AirCaptureCoilWithMaterialConductance	

EEB.Components.AggregateComponents.Heating.RoomHeater



Parameters

Type	Name	Default	Description
Integer	nLumps	2	number of lumps
Length	Ltube	30	tube length [m]
Length	Dtube	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
Mass	Mmetal	10	metal mass [kg]
SpecificHeatCapacity	cpMetal	500	metal specific heat [J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal and fluid temp [K]
Area	STube	pi*Dtube*Ltube	total surf, both sides [m2]
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]

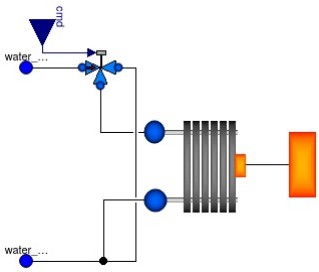
Connectors

Type	Name	Description
HeatPort	heatPort	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

```
model RoomHeater
  parameter Integer nLumps = 2 "number of lumps";
  parameter SI.Length Ltube = 30 "tube length";
  parameter SI.Length Dtube = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.Mass Mmetal = 10 "metal mass";
  parameter SI.SpecificHeatCapacity cpMetal = 500 "metal specific heat";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal and fluid temp";
  parameter SI.Area STube = pi * Dtube * Ltube "total surf, both sides";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  Interfaces.Thermal.HeatPort heatPort;
  BaseComponents.Thermal.Capacities.MassT massT(Tstart = Tstart, M = Mmetal, cp = cpMetal);
  BaseComponents.Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = nLumps, Ltube = Ltube, Dtube = Dtube, Dz = Dz, Cftube = Cftube, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_SV convVec2Sca1(n = nLumps, S = STube, gamma = gammaTubeMetal);
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(pipeExchangingN.heatPort, convVec2Sca1.vs);
  connect(convVec2Sca1.ss, heatPort);
  connect(massT.surf, heatPort);
  connect(water_flange1, pipeExchangingN.water_flange1);
  connect(pipeExchangingN.water_flange2, water_flange2);
end RoomHeater;
```

EEB.Components.AggregateComponents.Heating.RoomHeaterWithValve



Parameters

Type	Name	Default	Description
Integer	nLumps	2	number of lumps
Length	Ltube	30	tube length [m]
Length	Dtube	0.05	tube inner diameter [m]

Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
Mass	Mmetal	10	[kg]
SpecificHeatCapacity	cpMetal	500	[J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal and fluid temp [K]
Real	cvmaxValve	0.01	kg/s/sqrt(Pa)
Time	TpValve	1	positioner TC [s]
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]
Area	STube	pi*Dtube*Ltube	total surf, both sides [m2]

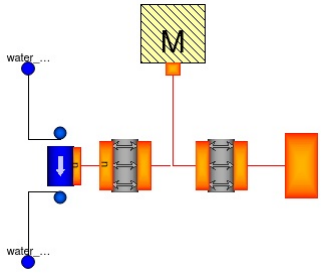
Connectors

Type	Name	Description
HeatPort	heatPort	
input RealInput	cmd	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

```
model RoomHeaterWithValve
  parameter Integer nLumps = 2 "number of lumps";
  parameter SI.Length Ltube = 30 "tube length";
  parameter SI.Length Dtube = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.Mass Mmetal = 10;
  parameter SI.SpecificHeatCapacity cpMetal = 500;
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal and fluid temp";
  parameter Real cvmaxValve = 0.01 "kg/s/sqrt(Pa)";
  parameter SI.Time TpValve = 1 "positioner TC";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  parameter SI.Area STube = pi * Dtube * Ltube "total surf, both sides";
  Interfaces.Thermal.HeatPort heatPort;
  Modelica.Blocks.Interfaces.RealInput cmd;
  RoomHeater roomHeater(nLumps = nLumps, Ltube = Ltube, Dtube = Dtube, Dz = Dz, Cftube = Cftube, Mmetal = Mmetal, cpMetal = cpMetal, Tstart = Tstart, STube = STube, gammaTubeMetal = g
  BaseComponents.Water.Valves.WaterValve3Ways_LinChar valve3WaysLinChar;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(roomHeater.heatPort, heatPort);
  connect(cmd, valve3WaysLinChar.cmd);
  connect(valve3WaysLinChar.water_flange1, water_flange1);
  connect(valve3WaysLinChar.water_flange3, roomHeater.water_flange1);
  connect(valve3WaysLinChar.water_flange2, water_flange2);
  connect(roomHeater.water_flange2, water_flange2);
end RoomHeaterWithValve;
```

EEB.Components.AggregateComponents.Heating.WaterCoilWithWall



Parameters

Type	Name	Default	Description
Integer	nLumps	2	number of lumps
Length	Ltube	30	tube length [m]
Length	Dtube	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]
Mass	Mmetal	10	metal mass [kg]
SpecificHeatCapacity	cpMetal	500	metal specific heat [J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal and fluid temp [K]
CoefficientOfHeatTransfer	gammaMetalExternal	5	heat transfer coefficient [W/(m2.K)]
Area	STube	pi*Dtube*Ltube	total surf tube [m2]

Connectors

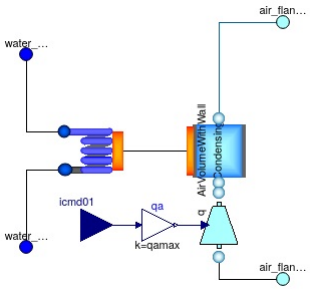
Type	Name	Description
HeatPort	heatPort	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

```
model WaterCoilWithWall
  parameter Integer nLumps = 2 "number of lumps";
  parameter SI.Length Ltube = 30 "tube length";
  parameter SI.Length Dtube = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  parameter SI.Mass Mmetal = 10 "metal mass";
  parameter SI.SpecificHeatCapacity cpMetal = 500 "metal specific heat";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal and fluid temp";
  parameter SI.CoefficientOfHeatTransfer gammaMetalExternal = 5 "heat transfer coefficient";
  parameter SI.Area STube = pi * Dtube * Ltube "total surf tube";
  BaseComponents.Thermal.HeatTransfer.Convection_SV convVec2Sca(n = nLumps, gamma = gammaTubeMetal, S = STube);
  Interfaces.Thermal.HeatPort heatPort;
  BaseComponents.Thermal.Capacities.MassT massT(Tstart = Tstart, M = Mmetal, cp = cpMetal);
  BaseComponents.Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = nLumps, Ltube = Ltube, Dtube = Dtube, Dz = Dz, Cftube = Cftube, Tstart = Tstart);
  BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca(S = STube, gamma = gammaMetalExternal);
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(convVec2Sca.ss, convSca2Sca.ss1);
```

```
  connect(convSca2Sca.ss2, heatPort);
  connect(massT.surf, convSca2Sca.ss1);
  connect(pipeExchangingN.heatPort, convVec2Sca.vs);
  connect(pipeExchangingN.water_flange1, water_flange1);
  connect(pipeExchangingN.water_flange2, water_flange2);
end WaterCoilWithWall;
```

EEB.Components.AggregateComponents.Heating.FanCoil



Parameters

Type	Name	Default	Description
Integer	ncoil	1	number of lumps
Length	Lcoil	30	tube length [m]
Length	Dcoil	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]
Mass	Mmetal	10	metal mass [kg]
SpecificHeatCapacity	cpMetal	500	metal specific heat [J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal, fluid temp and moist air [K]
CoefficientOfHeatTransfer	gammaMetalExternal	5	heat transfer coefficient [W/(m2.K)]
Area	STube	pi*waterCoilWithWall.Dtube*w...	total surf tube [m2]
Volume	V	0.001	Moist air volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]
ThermalConductance	Gaw	100	air-wall thermal conductance [W/K]
Real	qamax	0.02	max air volumetric flowrate (m3/s)

Connectors

Type	Name	Description
input RealInput	icmd01	
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

```
model FanCoil
  parameter Integer ncoil = 1 "number of lumps";
  parameter SI.Length Lcoil = 30 "tube length";
  parameter SI.Length Dcoil = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  parameter SI.Mass Mmetal = 10 "metal mass";
  parameter SI.SpecificHeatCapacity cpMetal = 500 "metal specific heat";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal, fluid temp and moist air";
  parameter SI.CoefficientOfHeatTransfer gammaMetalExternal = 5 "heat transfer coefficient";
  parameter SI.Area STube = pi * waterCoilWithWall.Dtube * waterCoilWithWall.Ltube "total surf tube";
  parameter SI.Volume V = 0.001 "Moist air volume";
  parameter SI.Pressure Pstart = 101325 "Initial moist air pressure";
  parameter SI.MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  parameter SI.ThermalConductance Gaw = 100 "air-wall thermal conductance";
  parameter Real qamax = 0.02 "max air volumetric flowrate (m3/s)";
  Heating.WaterCoilWithWall waterCoilWithWall(nLumps = ncoil, Ltube = Lcoil, Dtube = Dcoil, Dz = Dz, Cftube = Cftube, gammaTubeMetal = gammaTubeMetal, Mmetal = Mmetal, cpMetal = cpMeta
  BaseComponents.Air.Volumes.AirVolumeWithWall_Condensing airVolumeWithWall_Condensing(V = V, Pstart = Pstart, Tstart = Tstart, Xstart = Xstart, Cw = cpMetal * Mmetal, Gaw = Gaw);
  BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume airPrescribedFlowRate_Volume;
  Modelica.Blocks.Interfaces.RealInput icmd01;
  Modelica.Blocks.Math.Gain qa(k = qamax);
  Interfaces.Air.MoistAirFlange_waxa air_flange1;
  Interfaces.Air.MoistAirFlange_waxa air_flange2;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(waterCoilWithWall.heatPort, airVolumeWithWall_Condensing.heatPort);
  connect(airVolumeWithWall_Condensing.air_flange1, airPrescribedFlowRate_Volume.air_flange2);
  connect(airPrescribedFlowRate_Volume.air_flange1, air_flange1);
  connect(air_flange2, airVolumeWithWall_Condensing.air_flange2);
  connect(icmd01, qa.u);
  connect(qa.y, airPrescribedFlowRate_Volume.iq);
  connect(waterCoilWithWall.water_flange1, water_flange1);
  connect(waterCoilWithWall.water_flange2, water_flange2);
end FanCoil;
```

EEB.Components.AggregateComponents.Heating.IdealControlledFluidHeater

Information

Extends from Interfaces.Water.PartialTwoPort_water (Partial component with two ports).

Parameters

Type	Name	Default	Description
Time	Tcl	5	CL temp ctrl TC [s]
Time	Thc	20	intrinsic cooling TC [s]
Temperature	Tstart	273.15 + 25	initial fluid temp [K]

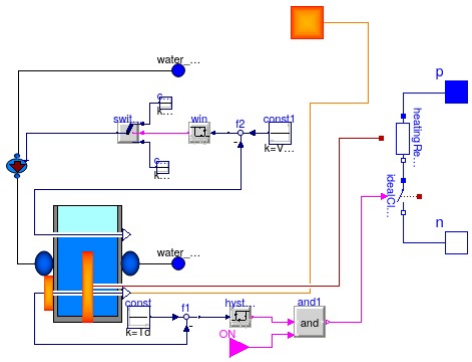
Connectors

Type	Name	Description
WaterFlange	water_flange2	
WaterFlange	water_flange1	
input BooleanInput	ON	
input RealInput	To	
output RealOutput	Pc	
output RealOutput	Ec	
output RealOutput	oTi	inlet T meas
output RealOutput	oTo	outlet T meas

Modelica definition

```
model IdealControlledFluidHeater
  extends Interfaces.Water.PartialTwoPort_water;
  Media.Substances.SubcooledWater waterIN, water;
  parameter SI.Time Tcl = 5 "CL temp ctrl TC";
  parameter SI.Time Thc = 20 "intrinsic cooling TC";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial fluid temp";
  SI.Temperature Tfi "inlet fluid temp";
  SI.Temperature Tfo(start = Tstart) "outlet fluid temp";
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Interfaces.RealInput To;
  Modelica.Blocks.Interfaces.RealOutput Pc;
  Modelica.Blocks.Interfaces.RealOutput Ec;
  Modelica.Blocks.Interfaces.RealOutput oTi "inlet T meas";
  Modelica.Blocks.Interfaces.RealOutput oTo "outlet T meas";
equation
// no pressure drop ("water" is water out)
  p1 = p2;
  waterIN.p = p1;
  water.p = p1;
// no mass storage
  w1 + w2 = 0;
// enthalpy boundary conditions
  hout1 = water.h;
  hout2 = water.h;
  waterIN.h = actualStream(water_flange1.h);
// definition of the input and output temperatures
  waterIN.T = Tfi;
  water.T = Tfo;
  if ON then
    Tfo + Tcl * der(Tfo) = To;
  else
    Tfo + Thc * der(Tfo) = Tfi;
  end if;
// Power consumption
  Pc = noEvent(max(w1 * (actualStream(water_flange2.h) - actualStream(water_flange1.h)), 0.0));
  Pc = der(Ec);
// Temp meas
  oTi = Tfi;
  oTo = Tfo;
end IdealControlledFluidHeater;
```

EEB.Components.AggregateComponents.Heating.ElectricBoiler



Parameters

Type	Name	Default	Description
Power	Pnom	500	Nominal power consumption [W]
Voltage	Vnom	220	Nominal voltage [V]
Real	Td	273.15 + 50	Desired temperature
Volume	Vboiler	0.1	max contained water volume [m3]
Volume	Vstart	0.001	initial water volume [m3]

Connectors

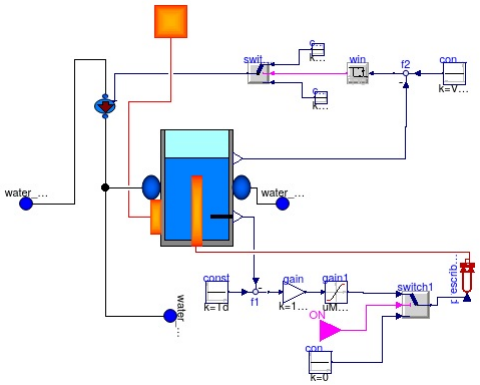
Type	Name	Description
HeatPort	e	
input BooleanInput	ON	
WaterFlange	water_flange1	
WaterFlange	water_flange2	
PositivePin	p	
NegativePin	n	

Modelica definition

```
model ElectricBoiler
  parameter SI.Power Pnom = 500 "Nominal power consumption";
  parameter SI.Voltage Vnom = 220 "Nominal voltage";
  parameter Real Td = 273.15 + 50 "Desired temperature";
  parameter SI.Volume Vboiler = 0.1 "max contained water volume";
  parameter SI.Volume Vstart = 0.001 "initial water volume";
  Real Ec "Energy consuption in kWh";
  Interfaces.Thermal.HeatPort e;
  Modelica.Blocks.Sources.Constant const(k = Td);
  Modelica.Blocks.Math.Feedback f1;
  Modelica.Blocks.Logical.Hysteresis hysteresis(uLow = -1, uHigh = 1);
  Modelica.Blocks.Sources.Constant const1(k = Vboiler * 1000);
  Modelica.Blocks.Math.Feedback f2;
  Modelica.Blocks.Logical.Hysteresis win(uLow = 0, uHigh = 1);
  Modelica.Electrical.Analog.Basic.HeatingResistor heatingResistor(R_ref = Vnom ^ 2 / Pnom, useHeatPort = true);
```

```
  Modelica.Electrical.Analog.Ideal.IdealClosingSwitch idealClosingSwitch;
  Components.BaseComponents.Water.Tanks.Tank_exchangingWall tank(V = Vboiler, Mm = 20, Vstart = Vstart, pin = 101325);
  Components.BaseComponents.Water.Pumps.WaterPump_Volumetric volumetricPump(qmax(displayUnit = "l/min") = 0.0003);
  Modelica.Blocks.Sources.Constant const2(k = 1);
  Modelica.Blocks.Logical.Switch switch2;
  Modelica.Blocks.Sources.Constant const3(k = 0);
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Blocks.Logical.And and1;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
  der(Ec) = (p.v - n.v) * p.i / (3600 * 1000);
  connect(const.y, f1.u1);
  connect(f1.y, hysteresis.u);
  connect(f2.y, win.u);
  connect(f2.u1, const1.y);
  connect(win.y, switch2.u2);
  connect(switch2.y, volumetricPump.cmd);
  connect(tank.oM, f2.u2);
  connect(const2.y, switch2.u1);
  connect(const3.y, switch2.u3);
  connect(tank.wall, e);
  connect(heatingResistor.heatPort, tank.fluid);
  connect(tank.oT, f1.u2);
  connect(heatingResistor.n, idealClosingSwitch.p);
  connect(hysteresis.y, and1.u1);
  connect(ON, and1.u2);
  connect(and1.y, idealClosingSwitch.control);
  connect(volumetricPump.water_flange1, water_flange1);
  connect(tank.water_flange1, volumetricPump.water_flange2);
  connect(tank.water_flange2, water_flange2);
  connect(heatingResistor.p, p);
  connect(idealClosingSwitch.n, n);
end ElectricBoiler;
```

EEB.Components.AggregateComponents.Heating.FuelBoiler



Parameters

Type	Name	Default	Description
Real	Td	273.15 + 50	Desired temperature
Volume	Vboiler	0.1	max contained water volume [m3]
Volume	Vstart	0.001	initial water volume [m3]
Power	Qmax	20000	Max thermal power of heater [W]

Connectors

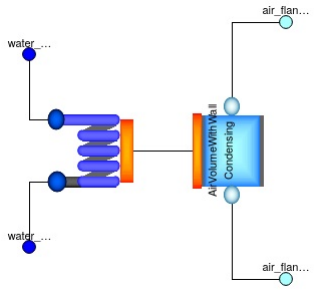
Type	Name	Description
HeatPort	e	
input BooleanInput	ON	
WaterFlange	water_flange1	
WaterFlange	water_flange2	
WaterFlange	water_flange3	

Modelica definition

```
model FuelBoiler
  parameter Real Td = 273.15 + 50 "Desired temperature";
  parameter SI.Volume Vboiler = 0.1 "max contained water volume";
  parameter SI.Volume Vstart = 0.001 "initial water volume";
  parameter SI.Power Qmax = 20000 "Max thermal power of heater";
  Interfaces.Thermal.HeatPort e;
  Modelica.Blocks.Sources.Constant const(k = Td);
  Modelica.Blocks.Math.Feedback f1;
  Modelica.Blocks.Sources.Constant const1(k = Vboiler * 1000);
  Modelica.Blocks.Math.Feedback f2;
  Modelica.Blocks.Logical.Hysteresis win(uLow = 0, uHigh = 1);
  Components.BaseComponents.Water.Tanks.Tank_exchangingWall tank(V = Vboiler, Mm = 20, Vstart = Vstart, pin = 101325);
  Components.BaseComponents.Water.Pumps.WaterPump_Volumetric volumetricPump(qmax(displayUnit = "l/min") = 0.0003);
  Modelica.Blocks.Sources.Constant const2(k = 1);
  Modelica.Blocks.Logical.Switch switch2;
  Modelica.Blocks.Sources.Constant const3(k = 0);
  Modelica.Blocks.Math.Gain gain(k = 10000);
  Modelica.Blocks.Nonlinear.Limiter gain1(uMax = Qmax, uMin = 0);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow prescribedHeatFlow;
  Modelica.Blocks.Logical.Switch switch1;
  Modelica.Blocks.Sources.Constant const4(k = 0);
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
  Interfaces.Water.WaterFlange water_flange3;
equation
  connect(const.y, f1.u1);
  connect(f2.y, win.u);
  connect(f2.u1, const1.y);
  connect(win.y, switch2.u2);
  connect(const2.y, switch2.u1);
  connect(const3.y, switch2.u3);
  connect(gain.y, gain1.u);
  connect(f1.y, gain.u);
  connect(e, e);
  connect(tank.oM, f2.u2);
  connect(tank.oT, f1.u2);
  connect(switch2.y, volumetricPump.cmd);
  connect(switch1.y, prescribedHeatFlow.Q_flow);
```

```
  connect(gain1.y, switch1.u1);
  connect(prescribedHeatFlow.port, tank.fluid);
  connect(const4.y, switch1.u3);
  connect(ON, switch1.u2);
  connect(e, tank.wall1);
  connect(volumetricPump.water_flange1, water_flange1);
  connect(tank.water_flange2, water_flange2);
  connect(water_flange3, volumetricPump.water_flange2);
  connect(tank.water_flange1, volumetricPump.water_flange2);
end FuelBoiler;
```

EEB.Components.AggregateComponents.Heating.AirCaptureCoil



Parameters

Type	Name	Default	Description
Integer	ncoil	1	number of lumps
Length	Lcoil	30	tube length [m]
Length	Dcoil	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]
Mass	Mmetal	10	metal mass [kg]
SpecificHeatCapacity	cpMetal	500	metal specific heat [J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal, fluid temp and moist air [K]
CoefficientOfHeatTransfer	gammaMetalExternal	5	heat transfer coefficient [W/(m2.K)]
Area	STube	pi*waterCoilWithWall.Dtube*w...	total surf tube [m2]
Volume	V	0.001	Moist air volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]
ThermalConductance	Gaw	100	air-wall thermal conductance [W/K]

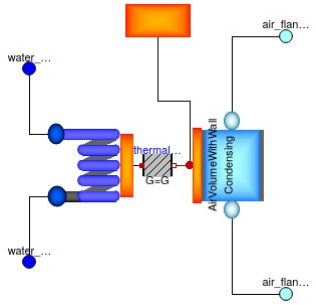
Connectors

Type	Name	Description
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

```
model AirCaptureCoil
  parameter Integer ncoil = 1 "number of lumps";
  parameter SI.Length Lcoil = 30 "tube length";
  parameter SI.Length Dcoil = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  parameter SI.Mass Mmetal = 10 "metal mass";
  parameter SI.SpecificHeatCapacity cpMetal = 500 "metal specific heat";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal, fluid temp and moist air";
  parameter SI.CoefficientOfHeatTransfer gammaMetalExternal = 5 "heat transfer coefficient";
  parameter SI.Area STube = pi * waterCoilWithWall.Dtube * waterCoilWithWall.Ltube "total surf tube";
  parameter SI.Volume V = 0.001 "Moist air volume";
  parameter SI.Pressure Pstart = 101325 "Initial moist air pressure";
  parameter SI.MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  parameter SI.ThermalConductance Gaw = 100 "air-wall thermal conductance";
  Heating.WaterCoilWithWall waterCoilWithWall(nLumps = ncoil, Ltube = Lcoil, Dtube = Dcoil, Dz = Dz, Cftube = Cftube, gammaTubeMetal = gammaTubeMetal, Mmetal = Mmetal, cpMetal = cpMeta
  BaseComponents.Air.Volumes.AirVolumeWithWall_Condensing airVolumeWithWall_Condensing(V = V, Pstart = Pstart, Tstart = Tstart, Xstart = Xstart, Cw = cpMetal * Mmetal, Gaw = Gaw);
  Interfaces.Air.MoistAirFlange_waxa air_flange1;
  Interfaces.Air.MoistAirFlange_waxa air_flange2;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(airVolumeWithWall_Condensing.air_flange2, air_flange2);
  connect(airVolumeWithWall_Condensing.air_flange1, air_flange1);
  connect(waterCoilWithWall.heatPort, airVolumeWithWall_Condensing.heatPort);
  connect(waterCoilWithWall.water_flange1, water_flange1);
  connect(waterCoilWithWall.water_flange2, water_flange2);
end AirCaptureCoil;
```

EEB.Components.AggregateComponents.Heating.AirCaptureCoilWithMaterialConductance



Parameters

Type	Name	Default	Description
Integer	ncoil	1	number of lumps
Length	Lcoil	30	tube length [m]
Length	Dcoil	0.05	tube inner diameter [m]
Length	Dz	0	height diff (out-in) [m]
Real	Cftube	1e-6	tube friction coefficient
CoefficientOfHeatTransfer	gammaTubeMetal	5	heat transfer coefficient [W/(m2.K)]
Mass	Mmetal	10	metal mass [kg]
SpecificHeatCapacity	cpMetal	500	metal specific heat [J/(kg.K)]
Temperature	Tstart	273.15 + 25	initial metal, fluid temp and moist air [K]
CoefficientOfHeatTransfer	gammaMetalExternal	5	heat transfer coefficient [W/(m2.K)]
Area	STube	pi*waterCoilWithWall.Dtube*w...	total surf tube [m2]
Volume	V	0.001	Moist air volume [m3]
Pressure	Pstart	101325	Initial moist air pressure [Pa]
MassFraction	Xstart	0.001	Initial absolute umidity [kg_H20/kg_DA] [1]
ThermalConductance	G	10	Constant thermal conductance of material [W/K]
ThermalConductance	Gaw	100	air-wall thermal conductance [W/K]

Connectors

Type	Name	Description
HeatPort	heatPort	
MoistAirFlange_waxa	air_flange1	
MoistAirFlange_waxa	air_flange2	
WaterFlange	water_flange1	
WaterFlange	water_flange2	

Modelica definition

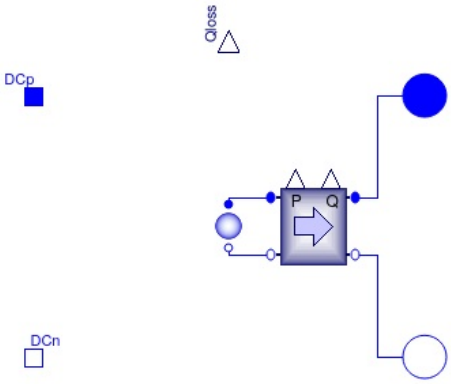
```
model AirCaptureCoilWithMaterialConductance
  parameter Integer ncoil = 1 "number of lumps";
  parameter SI.Length Lcoil = 30 "tube length";
  parameter SI.Length Dcoil = 0.05 "tube inner diameter";
  parameter SI.Length Dz = 0 "height diff (out-in)";
  parameter Real Cftube = 1e-6 "tube friction coefficient";
  parameter SI.CoefficientOfHeatTransfer gammaTubeMetal = 5 "heat transfer coefficient";
  parameter SI.Mass Mmetal = 10 "metal mass";
  parameter SI.SpecificHeatCapacity cpMetal = 500 "metal specific heat";
  parameter SI.Temperature Tstart = 273.15 + 25 "initial metal, fluid temp and moist air";
  parameter SI.CoefficientOfHeatTransfer gammaMetalExternal = 5 "heat transfer coefficient";
  parameter SI.Area STube = pi * waterCoilWithWall.Dtube * waterCoilWithWall.Ltube "total surf tube";
  parameter SI.Volume V = 0.001 "Moist air volume";
  parameter SI.Pressure Pstart = 101325 "Initial moist air pressure";
  parameter SI.MassFraction Xstart = 0.001 "Initial absolute umidity [kg_H20/kg_DA]";
  parameter SI.ThermalConductance G = 10 "Constant thermal conductance of material";
  parameter SI.ThermalConductance Gaw = 100 "air-wall thermal conductance";
  Heating.WaterCoilWithWall waterCoilWithWall(nLumps = ncoil, Ltube = Lcoil, Dtube = Dcoil, Dz = Dz, Cftube = Cftube, gammaTubeMetal = gammaTubeMetal, Mmetal = Mmetal, cpMetal = cpMeta
  BaseComponents.Air.Volumes.AirVolumeWithWall_Condensing airVolumeWithWall_Condensing(V = V, Pstart = Pstart, Tstart = Tstart, Xstart = Xstart, Cw = cpMetal * Mmetal, Gaw = Gaw);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = G);
  Interfaces.Thermal.HeatPort heatPort;
  Interfaces.Air.MoistAirFlange_waxa air_flange1;
  Interfaces.Air.MoistAirFlange_waxa air_flange2;
  Interfaces.Water.WaterFlange water_flange1;
  Interfaces.Water.WaterFlange water_flange2;
equation
  connect(airVolumeWithWall_Condensing.air_flange2, air_flange2);
  connect(airVolumeWithWall_Condensing.air_flange1, air_flange1);
  connect(waterCoilWithWall.heatPort, thermalConductor.port_a);
  connect(thermalConductor.port_b, airVolumeWithWall_Condensing.heatPort);
  connect(heatPort, airVolumeWithWall_Condensing.heatPort);
  connect(waterCoilWithWall.water_flange1, water_flange1);
  connect(waterCoilWithWall.water_flange2, water_flange2);
end AirCaptureCoilWithMaterialConductance;
```

EEB.Components.AggregateComponents.Electrical

Package Content

Name	Description
BaseClasses	
Inverter_Phi0_ConstantEfficiency	
Inverter_Pcontrol_ConstantEfficiency	
Inverter_PcontrolOrPhi0_ConstantEfficiency	
Inverter_IdealPcontrol_ConstantEfficiency	
DCsupply_constantEfficiency	
GenericOnOffDCload	
FixedACsupply_socket_withGround	

EEB.Components.AggregateComponents.Electrical.Inverter_Phi0_ConstantEfficiency



Information

Extends from Components.AggregateComponents.Electrical.BaseClasses.BaseInverter.

Parameters

Type	Name	Default	Description
Voltage	Vac	100	AC voltage [V]
Real	eta	0.95	Efficiency

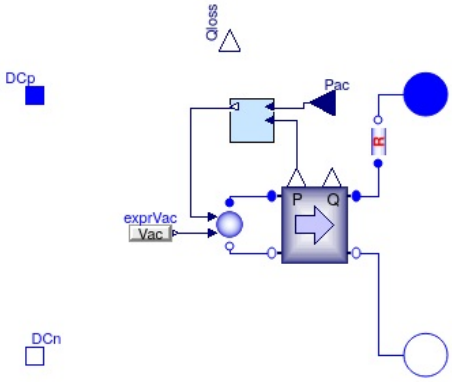
Connectors

Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	

Modelica definition

```
model Inverter_Phi0_ConstantEfficiency
  extends Components.AggregateComponents.Electrical.BaseClasses.BaseInverter;
  parameter SI.Voltage Vac = 100 "AC voltage";
  parameter Real eta = 0.95 "Efficiency";
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed GenAC(V = Vac);
  BaseComponents.Electrical.Phasors.PQmeter PQ;
equation
  DCp.i + DCn.i = 0;
  DCp.i * (DCp.v - DCn.v) = PQ.P / eta;
  Qloss = PQ.P * (1 / eta - 1);
  connect(GenAC.p, PQ.p1);
  connect(GenAC.n, PQ.n1);
  connect(PQ.p2, ACp);
  connect(PQ.n2, ACn);
end Inverter_Phi0_ConstantEfficiency;
```

EEB.Components.AggregateComponents.Electrical.Inverter_Pcontrol_ConstantEfficiency



Information

Extends from Components.AggregateComponents.Electrical.BaseClasses.BaseInverter.

Parameters

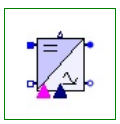
Type	Name	Default	Description
Voltage	Vac	100	AC voltage [V]
Real	eta	0.95	Efficiency
Resistance	Rout	1	Output R [Ohm]
Real	K	0.00001	C gain
Time	Ti	0.01	C integral time [s]

Connectors

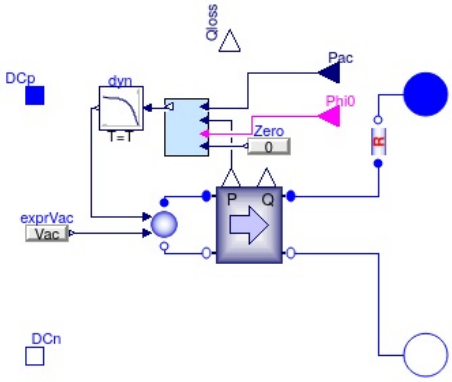
Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	
input RealInput	Pac	

Modelica definition

```
model Inverter_Pcontrol_ConstantEfficiency
  extends Components.AggregateComponents.Electrical.BaseClasses.BaseInverter;
  parameter SI.Voltage Vac = 100 "AC voltage";
  parameter Real eta = 0.95 "Efficiency";
  BaseComponents.Electrical.Phasors.Vgen_Sine_Vphi GenAC;
  BaseComponents.Electrical.Phasors.PQmeter PQ;
  Modelica.Blocks.Interfaces.RealInput Pac;
  Modelica.Blocks.Sources.RealExpression exprVac(y = Vac);
  Controllers.Blocks.Analogue.AWPI_1dof PI_P(CSmax = pi / 4, CSmin = -pi / 4, K = K, Ti = Ti);
  BaseComponents.Electrical.Phasors.Resistor Ro(R = Rout);
  parameter SI.Resistance Rout = 1 "Output R";
  parameter Real K = 0.00001 "C gain";
  parameter SI.Time Ti = 0.01 "C integral time";
equation
  DCp.i + DCn.i = 0;
  DCp.i * (DCp.v - DCn.v) = PQ.P / eta;
  Qloss = PQ.P * (1 / eta - 1);
  connect(GenAC.p, PQ.p1);
  connect(GenAC.n, PQ.n1);
  connect(PQ.n2, ACn);
  connect(exprVac.y, GenAC.V);
  connect(PI_P.SP, Pac);
  connect(PI_P.PV, PQ.P);
  connect(PI_P.CS, GenAC.phi);
  connect(PQ.p2, Ro.p);
  connect(Ro.n, ACp);
end Inverter_Pcontrol_ConstantEfficiency;
```



EEB.Components.AggregateComponents.Electrical.Inverter_PcontrolOrPhi0_ConstantEfficiency



Information

Extends from Components.AggregateComponents.Electrical.BaseClasses.BaseInverter.

Parameters

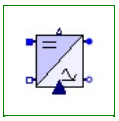
Type	Name	Default	Description
Voltage	Vac	100	AC voltage [V]
Real	eta	0.95	Efficiency
Resistance	Rout	1	Output R [Ohm]
Time	T	0.01	Time Constant [s]

Connectors

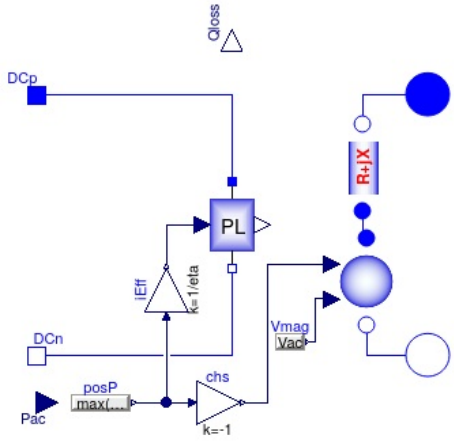
Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	
input RealInput	Pac	
input BooleanInput	Phi0	

Modelica definition

```
model Inverter_PcontrolOrPhi0_ConstantEfficiency
  extends Components.AggregateComponents.Electrical.BaseClasses.BaseInverter;
  parameter SI.Voltage Vac = 100 "AC voltage";
  parameter Real eta = 0.95 "Efficiency";
  BaseComponents.Electrical.Phasors.Vgen_Sine_Vphi GenAC;
  BaseComponents.Electrical.Phasors.PQmeter PQ;
  Modelica.Blocks.Interfaces.RealInput Pac;
  Modelica.Blocks.Sources.RealExpression exprVac(y = Vac);
  Controllers.Blocks.Analogue.AWPI_1dof_trk PI_P(CSmax = pi / 4, CSmin = -pi / 4, K = 0.00001, Ti = 0.01);
  BaseComponents.Electrical.Phasors.Resistor Ro(R = Rout);
  parameter SI.Resistance Rout = 1 "Output R";
  Modelica.Blocks.Sources.RealExpression Zero(y = 0);
  Modelica.Blocks.Interfaces.BooleanInput Phi0;
  Modelica.Blocks.Continuous.FirstOrder dyn(k = 1, T = T);
  parameter SI.Time T = 0.01 "Time Constant";
equation
  DCp.i + DCn.i = 0;
  DCp.i * (DCp.v - DCn.v) = PQ.P / eta;
  Qloss = PQ.P * (1 / eta - 1);
  connect(GenAC.p, PQ.p1);
  connect(GenAC.n, PQ.n1);
  connect(PQ.n2, ACn);
  connect(exprVac.y, GenAC.V);
  connect(PI_P.SP, Pac);
  connect(PI_P.PV, PQ.P);
  connect(PQ.p2, Ro.p);
  connect(Ro.n, ACp);
  connect(PI_P.TR, Zero.y);
  connect(PI_P.TS, Phi0);
  connect(PI_P.CS, dyn.u);
  connect(dyn.y, GenAC.phi);
end Inverter_PcontrolOrPhi0_ConstantEfficiency;
```



EEB.Components.AggregateComponents.Electrical.Inverter_IdealPcontrol_ConstantEfficiency



Information

Extends from Components.AggregateComponents.Electrical.BaseClasses.BaseInverter.

Parameters

Type	Name	Default	Description
Voltage	Vac	100	AC voltage [V]
Real	eta	0.95	Efficiency
Resistance	Rout	1	Output R [Ohm]
Reactance	Xout	0.1	Output X [Ohm]

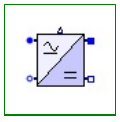
Connectors

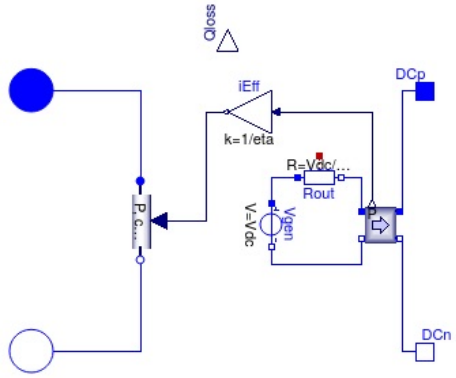
Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	
input RealInput	Pac	

Modelica definition

```
model Inverter_IdealPcontrol_ConstantEfficiency
  extends Components.AggregateComponents.Electrical.BaseClasses.BaseInverter;
  parameter SI.Voltage Vac = 100 "AC voltage";
  parameter Real eta = 0.95 "Efficiency";
  parameter SI.Resistance Rout = 1 "Output R";
  parameter Reactance Xout = 0.1 "Output X";
  Modelica.Blocks.Interfaces.RealInput Pac;
  BaseComponents.Electrical.DC.Load_Pin_I0 Load1;
  BaseComponents.Electrical.Phasors.Vgen_Sine_VP vgen_Sine_VP;
  BaseComponents.Electrical.Phasors.R_plus_jX Zout(R = Rout, X = Xout);
  Modelica.Blocks.Sources.RealExpression posP(y = max(Pac, 0));
  Modelica.Blocks.Sources.RealExpression Vmag(y = Vac);
  Modelica.Blocks.Math.Gain iEff(k = 1 / eta);
  Modelica.Blocks.Math.Gain chs(k = -1);
equation
  Qloss = iEff.y - iEff.u;
  connect(DCn, Load1.n);
  connect(DCp, Load1.p);
  connect(Vmag.y, vgen_Sine_VP.V);
  connect(Zout.n, ACp);
  connect(Zout.p, vgen_Sine_VP.p);
  connect(vgen_Sine_VP.n, ACn);
  connect(iEff.y, Load1.iPabs);
  connect(posP.y, iEff.u);
  connect(posP.y, chs.u);
  connect(chs.y, vgen_Sine_VP.P);
end Inverter_IdealPcontrol_ConstantEfficiency;
```

EEB.Components.AggregateComponents.Electrical.DCsupply_constantEfficiency





Information

Extends from BaseClasses.BaseDCsupply.

Parameters

Type	Name	Default	Description
Voltage	Vdc	10	DC voltage [V]
Current	Isc	10	SC current [A]
Real	cphinom	0.9	nominal power factor
Real	eta	0.85	efficiency

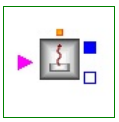
Connectors

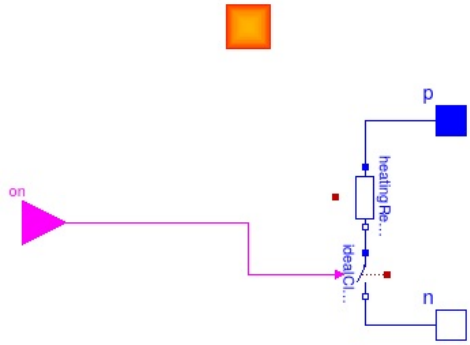
Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	

Modelica definition

```
model DCsupply_constantEfficiency
  extends BaseClasses.BaseDCsupply;
  Modelica.Electrical.Analog.Sources.ConstantVoltage Vgen(V = Vdc);
  parameter SI.Voltage Vdc = 10 "DC voltage";
  parameter Current Isc = 10 "SC current";
  Modelica.Electrical.Analog.Basic.Resistor Rout(R = Vdc / Isc);
  BaseComponents.Electrical.DC.Pmeter PM;
  BaseComponents.Electrical.Phasors.Load_Pcosphi_nom_Pin ACload(cphinom = cphinom);
  parameter Real cphinom = 0.9 "nominal power factor";
  parameter Real eta = 0.85 "efficiency";
  Modelica.Blocks.Math.Gain iEff(k = 1 / eta);
equation
  Qloss = iEff.y - iEff.u;
  connect(PM.n2, DCn);
  connect(PM.p2, DCp);
  connect(Vgen.p, Rout.p);
  connect(Rout.n, PM.p1);
  connect(Vgen.n, PM.n1);
  connect(ACp, ACload.p);
  connect(ACn, ACload.n);
  connect(iEff.y, ACload.P);
  connect(iEff.u, PM.P);
end DCsupply_constantEfficiency;
```

EEB.Components.AggregateComponents.Electrical.GenericOnOffDCload





Parameters

Type	Name	Default	Description

Power	Pnom	500	Nominal power consumption [W]
Real	D	20	% Power dissipation
Voltage	Vnom	220	Nominal voltage [V]

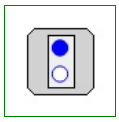
Connectors

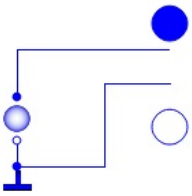
Type	Name	Description
HeatPort	e	
input BooleanInput	on	
PositivePin	p	
NegativePin	n	

Modelica definition

```
model GenericOnOffDCload
  parameter SI.Power Pnom = 500 "Nominal power consumption";
  parameter Real D(min = 0, max = 100) = 20 "% Power dissipation";
  parameter SI.Voltage Vnom = 220 "Nominal voltage";
  Real Ec "Energy consumption";
  Interfaces.Thermal.HeatPort e;
  Modelica.Blocks.Interfaces.BooleanInput on;
  Modelica.Electrical.Analog.Basic.HeatingResistor heatingResistor(R_ref = Vnom ^ 2 / Pnom, useHeatPort = false);
  Modelica.Electrical.Analog.Ideal.IdealClosingSwitch idealClosingSwitch;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
  e.Q_flow = -heatingResistor.LossPower * D / 100;
  der(Ec) = (p.v - n.v) * p.i / (1000 * 3600);
  connect(on, idealClosingSwitch.control);
  connect(heatingResistor.n, idealClosingSwitch.p);
  connect(heatingResistor.p, p);
  connect(idealClosingSwitch.n, n);
end GenericOnOffDCload;
```

EEB.Components.AggregateComponents.Electrical.FixedACsupply_socket_withGround





Parameters

Type	Name	Default	Description
Voltage	V	220	Voltage [V]

Connectors

Type	Name	Description
PosNegPhasorPins	socket	

Modelica definition

```
model FixedACsupply_socket_withGround
  Interfaces.Electrical.PosNegPhasorPins socket;
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed Vgen(V = V, phi = 0);
  BaseComponents.Electrical.Phasors.Ground gnd;
  parameter SI.Voltage V = 220 "Voltage";
equation
  connect(gnd.p, Vgen.n);
  connect(socket.n, gnd.p);
  connect(Vgen.p, socket.p);
end FixedACsupply_socket_withGround;
```

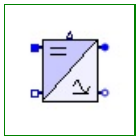
EEB.Components.AggregateComponents.Electrical.BaseClasses

Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
BaseInverter	
BaseDCsupply	



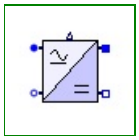
EEB.Components.AggregateComponents.Electrical.BaseClasses.BaseInverter

Connectors

Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	

Modelica definition

```
partial model BaseInverter
  Interfaces.Electrical.PositivePhasorPin ACp;
  Interfaces.Electrical.NegativePhasorPin ACn;
  Modelica.Electrical.Analog.Interfaces.PositivePin DCp;
  Modelica.Electrical.Analog.Interfaces.NegativePin DCn;
  Modelica.Blocks.Interfaces.RealOutput Qloss;
end BaseInverter;
```



EEB.Components.AggregateComponents.Electrical.BaseClasses.BaseDCsupply

Connectors

Type	Name	Description
PositivePhasorPin	ACp	
NegativePhasorPin	ACn	
PositivePin	DCp	
NegativePin	DCn	
output RealOutput	Qloss	

Modelica definition

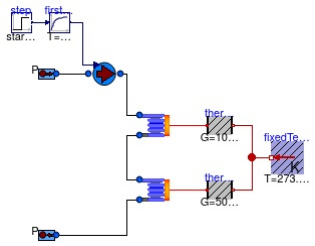
```
partial model BaseDCsupply
  Interfaces.Electrical.PositivePhasorPin ACp;
  Interfaces.Electrical.NegativePhasorPin ACn;
  Modelica.Electrical.Analog.Interfaces.PositivePin DCp;
  Modelica.Electrical.Analog.Interfaces.NegativePin DCn;
  Modelica.Blocks.Interfaces.RealOutput Qloss;
end BaseDCsupply;
```

EEB.Components.AggregateComponents.Test

Package Content

Name	Description
test1_WaterCoil	
test2_AirCaptureCoil	
test3_Heater	
Test_Inverter_Phi0andPcontrol_Nswitch	
Test_Inverter_Phi0andPcontrol_ConstantEfficiency	
Test_Inverter_IdealPcontrol_ConstantEfficiency	
test1_IdealControlledFluidHeater	
test_DCsupply	
test_room_cooling	

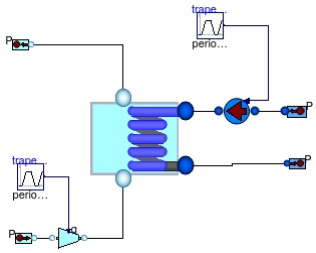
EEB.Components.AggregateComponents.Test.test1_WaterCoil



Modelica definition

```
model test1_WaterCoil
  BaseComponents.Water.Sources.WaterSource_PT_fixed source_PT_fixed(T0 = 273.15 + 50);
  BaseComponents.Water.Sinks.WaterSink_P_fixed sink_P_fixed;
  Heating.WaterCoilWithWall waterCoilWithWall1(Ltube = 0.5, Dtube = 0.005, gammaTubeMetal = 100, cpMetal = 300, Tstart = 273.15 + 20, Mmetal = 0.06, STube = 1, gammaMetalExternal = 100)
  BaseComponents.Water.Pumps.WaterPump_Volumetric volumetricPump(qmax = 0.001);
  Modelica.Blocks.Sources.Step step;
  Modelica.Blocks.Continuous.FirstOrder firstOrder(T = 0.1);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 10);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 50);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 273.15 + 20);
  Heating.WaterCoilWithWall waterCoilWithWall2(Ltube = 0.5, Dtube = 0.005, gammaTubeMetal = 100, cpMetal = 300, Tstart = 273.15 + 20, Mmetal = 0.06, STube = 1, gammaMetalExternal = 10(
equation
  connect(firstOrder.y, volumetricPump.cmd);
  connect(step.y, firstOrder.u);
  connect(thermalConductor.port_b, fixedTemperature.port);
  connect(thermalConductor1.port_b, fixedTemperature.port);
  connect(waterCoilWithWall2.heatPort, thermalConductor1.port_a);
  connect(waterCoilWithWall.heatPort, thermalConductor.port_a);
  connect(volumetricPump.water_flange1, source_PT_fixed.water_flange);
  connect(waterCoilWithWall.water_flange1, volumetricPump.water_flange2);
  connect(waterCoilWithWall.water_flange2, waterCoilWithWall2.water_flange1);
  connect(waterCoilWithWall2.water_flange2, sink_P_fixed.water_flange);
end test1_WaterCoil;
```

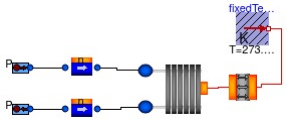
EEB.Components.AggregateComponents.Test.test2_AirCaptureCoil



Modelica definition

```
model test2_AirCaptureCoil
  Heating.AirCaptureCoil airCaptureCoil(Lcoil = 0.5, Dcoil = 0.005, gammaTubeMetal = 100, cpMetal = 300, Tstart = 273.15 + 20, STube = 1, V = 0.002, Mmetal = 0.06, gammaMetalExternal =
  BaseComponents.Water.Pumps.WaterPump_Volumetric volumetricPump(qmax(displayUnit = "l/min") = 3.3333333333333e-05);
  BaseComponents.Water.Sinks.WaterSink_P_fixed sink_P_fixed;
  BaseComponents.Water.Sources.WaterSource_PT_fixed source_PT_fixed(T0 = 273.15 + 20);
  BaseComponents.Air.Sources.AirSource_pTX_fixed airSource_fixed_pTX(                  X0 = 0.5, T0=
        273.15 + 85);
  BaseComponents.Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume airPrescribedFlowRate_Volume;
  Modelica.Blocks.Sources.Trapezoid trapezoid(amplitude = 0.1, rising = 20, width = 100, falling = 20, period = 200, nperiod = 1, startTime = 20);
  Modelica.Blocks.Sources.Trapezoid trapezoid1(amplitude = 0.5, rising = 50, width = 200, falling = 50, period = 300, nperiod = 1, startTime = 10);
equation
  connect(airSink_P_fixed.air_flange, airCaptureCoil.air_flange2);
  connect(airSource_fixed_pTX.air_flange, airPrescribedFlowRate_Volume.air_flange1);
  connect(airPrescribedFlowRate_Volume.air_flange2, airCaptureCoil.air_flange1);
  connect(trapezoid1.y, volumetricPump.cmd);
  connect(airPrescribedFlowRate_Volume.iq, trapezoid.y);
  connect(source_PT_fixed.water_flange, volumetricPump.water_flange1);
  connect(volumetricPump.water_flange2, airCaptureCoil.water_flange1);
  connect(sink_P_fixed.water_flange, airCaptureCoil.water_flange2);
end test2_AirCaptureCoil;
```

EEB.Components.AggregateComponents.Test.test3_Heater

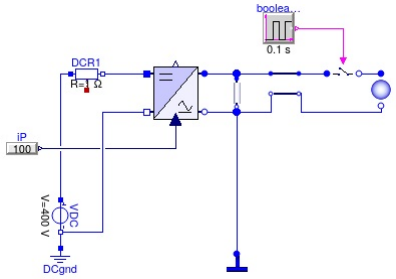


Modelica definition

```
model test3_Heater
```

```
  Heating.RoomHeater roomHeater(nLumps = 5, Ltube = 10, Dtube = 0.01, Mmetal = 10, gammaTubeMetal = 40);
  BaseComponents.Water.Sources.WaterSource_PT_fixed source_PT_fixed(T0 = 273.15 + 50);
  BaseComponents.Water.Sinks.WaterSink_P_fixed sink_P_fixed;
  BaseComponents.Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN(n = 1, Ltube = 10, Dtube = 0.01, Tstart = 323.15);
  BaseComponents.Water.Pipes.WaterPipeExchanging_Nvols pipeExchangingN1(n = 1, Ltube = 10, Dtube = 0.01, Tstart = 323.15);
  BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca;
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 273.15 + 5);
equation
  connect(roomHeater.heatPort, convSca2Sca.ss2);
  connect(convSca2Sca.ss1, fixedTemperature.port);
  connect(source_PT_fixed.water_flange, pipeExchangingN.water_flange1);
  connect(pipeExchangingN.water_flange2, roomHeater.water_flange1);
  connect(sink_P_fixed.water_flange, pipeExchangingN1.water_flange1);
  connect(pipeExchangingN1.water_flange2, roomHeater.water_flange2);
end test3_Heater;
```

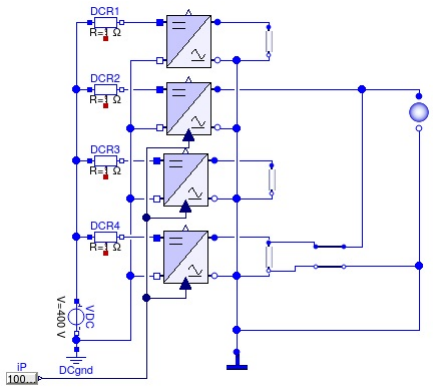
EEB.Components.AggregateComponents.Test.Test_Inverter_Phi0andPcontrol_Nswitch



Modelica definition

```
model Test_Inverter_Phi0andPcontrol_Nswitch
  Electrical.Inverter_Pcontrol_ConstantEfficiency INV1(eta = 0.9);
  Modelica.Electrical.Analog.Basic.Ground DCgnd;
  Modelica.Electrical.Analog.Basic.Resistor DCR1(R = 1);
  Modelica.Electrical.Analog.Sources.ConstantVoltage VDC(V = 400);
  BaseComponents.Electrical.Phasors.Load_VPcosphi_nom ACload1;
  BaseComponents.Electrical.Phasors.Ground ACgnd;
  Modelica.Blocks.Sources.RealExpression iP(y = 100);
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed Network1;
  BaseComponents.Electrical.Phasors.TransmissionLine transmissionLine(l = 10);
  Modelica.Blocks.Sources.BooleanPulse booleanPulse(period = 0.1);
  BaseComponents.Electrical.Phasors.Switch switch;
equation
  connect(DCgnd.p, VDC.n);
  connect(VDC.p, DCR1.p);
  connect(DCR1.n, INV1.DCp);
  connect(INV1.ACn, ACload1.n);
  connect(iP.y, INV1.Pac);
  connect(INV1.ACp, ACload1.p);
  connect(ACload1.p, transmissionLine.p1);
  connect(ACload1.n, ACgnd.p);
  connect(ACload1.n, transmissionLine.n1);
  connect(transmissionLine.n2, Network1.n);
  connect(INV1.DCn, VDC.n);
  connect(transmissionLine.p2, switch.p);
  connect(switch.n, Network1.p);
  connect(booleanPulse.y, switch.close);
end Test_Inverter_Phi0andPcontrol_Nswitch;
```

EEB.Components.AggregateComponents.Test.Test_Inverter_Phi0andPcontrol_ConstantEfficiency

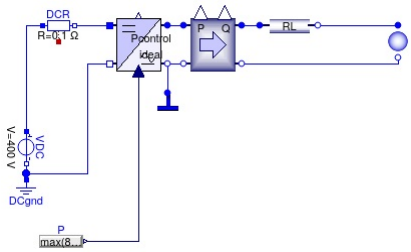


Modelica definition

```
model Test_Inverter_Phi0andPcontrol_ConstantEfficiency
  Electrical.Inverter_Phi0_ConstantEfficiency INV1(eta = 0.9);
  Modelica.Electrical.Analog.Basic.Ground DCgnd;
  Modelica.Electrical.Analog.Basic.Resistor DCR1(R = 1);
  Modelica.Electrical.Analog.Sources.ConstantVoltage VDC(V = 400);
  BaseComponents.Electrical.Phasors.Load_VPcosphi_nom ACload1;
  BaseComponents.Electrical.Phasors.Ground ACgnd;
  Modelica.Electrical.Analog.Basic.Resistor DCR2(R = 1);
  Electrical.Inverter_Pcontrol_ConstantEfficiency INV2(eta = 0.9);
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed Network;
  Modelica.Blocks.Sources.RealExpression iP(y = 100 + 20 * sin(2 * time) * exp(-time / 4));
  Modelica.Electrical.Analog.Basic.Resistor DCR3(R = 1);
  Electrical.Inverter_Pcontrol_ConstantEfficiency INV3(eta = 0.9);
  BaseComponents.Electrical.Phasors.Load_VPcosphi_nom ACload3;
  Modelica.Electrical.Analog.Basic.Resistor DCR4(R = 1);
  Electrical.Inverter_Pcontrol_ConstantEfficiency INV4(eta = 0.9);
  BaseComponents.Electrical.Phasors.Load_VPcosphi_nom ACload4;
  BaseComponents.Electrical.Phasors.TransmissionLine transmissionLine(l = 10);
equation
  connect(DCgnd.p, VDC.n);
  connect(VDC.p, DCR1.p);
  connect(DCR1.n, INV1.DCp);
  connect(INV1.ACp, ACload1.p);
  connect(INV1.ACn, ACload1.n);
  connect(ACgnd.p, ACload1.n);
  connect(DCR2.p, DCR1.p);
  connect(DCR2.n, INV2.DCp);
  connect(INV2.ACp, Network.p);
  connect(INV2.ACn, ACload1.n);
```

```
  connect(DCR3.p, DCR1.p);
  connect(iP.y, INV2.Pac);
  connect(DCR3.n, INV3.DCp);
  connect(INV3.Pac, INV2.Pac);
  connect(INV3.ACn, ACload1.n);
  connect(INV3.ACp, ACload3.p);
  connect(ACload3.n, ACload1.n);
  connect(INV1.DCn, VDC.n);
  connect(INV2.DCn, VDC.n);
  connect(INV3.DCn, VDC.n);
  connect(DCR4.p, DCR1.p);
  connect(INV4.DCp, DCR4.n);
  connect(INV4.DCn, VDC.n);
  connect(INV4.Pac, INV2.Pac);
  connect(INV4.ACn, ACload1.n);
  connect(ACload4.n, ACload1.n);
  connect(INV4.ACp, ACload4.p);
  connect(Network.n, ACload1.n);
  connect(transmissionLine.p2, Network.p);
  connect(transmissionLine.n2, ACload1.n);
  connect(transmissionLine.p1, ACload4.p);
  connect(ACload4.n, transmissionLine.n1);
end Test_Inverter_Phi0andPcontrol_ConstantEfficiency;
```

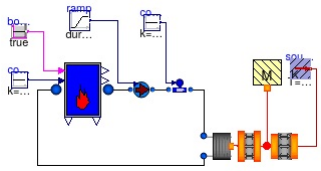
EEB.Components.AggregateComponents.Test.Test_Inverter_IdealPcontrol_ConstantEfficiency



Modelica definition

```
model Test_Inverter_IdealPcontrol_ConstantEfficiency
  Modelica.Electrical.Analog.Basic.Ground DCgnd;
  Modelica.Electrical.Analog.Basic.Resistor DCR(R = 0.1);
  Modelica.Electrical.Analog.Sources.ConstantVoltage VDC(V = 400);
  BaseComponents.Electrical.Phasors.Ground ACgnd;
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed Network;
  BaseComponents.Electrical.Phasors.Load_VPcosphi_nom Line(Vnom = 1, Pnom = 1);
  Electrical.Inverter_IdealPcontrol_ConstantEfficiency inverter_IdealPcontrol_ConstantEfficiency(Xout = 0, Rout = 0.02);
  Modelica.Blocks.Sources.RealExpression P(y = max(80 * sin(time), 0));
  BaseComponents.Electrical.Phasors.PQmeter PQmeter;
equation
  connect(DCgnd.p, VDC.n);
  connect(VDC.p, DCR.p);
  connect(DCR.n, inverter_IdealPcontrol_ConstantEfficiency.DCp);
  connect(inverter_IdealPcontrol_ConstantEfficiency.DCn, VDC.n);
  connect(P.y, inverter_IdealPcontrol_ConstantEfficiency.Pac);
  connect(inverter_IdealPcontrol_ConstantEfficiency.ACp, PQmeter.p1);
  connect(inverter_IdealPcontrol_ConstantEfficiency.ACn, PQmeter.n1);
  connect(ACgnd.p, inverter_IdealPcontrol_ConstantEfficiency.ACn);
  connect(PQmeter.p2, Line.p);
  connect(Line.n, Network.p);
  connect(Network.n, PQmeter.n2);
end Test_Inverter_IdealPcontrol_ConstantEfficiency;
```

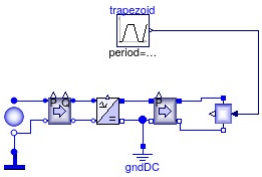
EEB.Components.AggregateComponents.Test.test1_IdealControlledFluidHeater



Modelica definition

```
model test1_IdealControlledFluidHeater
  Components.AggregateComponents.Heating.IdealControlledFluidHeater idealControlledFluidHeater(Tc1 = 240, Tstart = 273.15 + 20);
  Modelica.Blocks.Sources.BooleanConstant booleanConstant;
  Modelica.Blocks.Sources.Constant const1(k = 101325);
  Modelica.Blocks.Sources.Constant const2(k = 273.15 + 60);
  Components.BaseComponents.Water.Pressurisers.IdealWaterPressuriser idealPressuriser;
  Components.BaseComponents.Water.Pumps.WaterPump_Volumetric volumetricPump(qmax = 0.0002);
  Components.BaseComponents.Thermal.Capacities.MassT massT(cp = 1000, Tstart = 273.15 + 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca(S = 1.5, gamma = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(S = 10);
  Components.AggregateComponents.Heating.RoomHeater roomHeater(nLumps = 5, Ltube = 2.5, Dtube = 0.1, Cftube = 0.001, gammaTubeMetal = 45, Mmetal = 50, Tstart = 273.15 + 20);
  Modelica.Blocks.Sources.Ramp ramp(duration = 100);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature source_T_fixed1(T = 273.15 + 5);
equation
  connect(const1.y, idealPressuriser.p);
  connect(booleanConstant.y, idealControlledFluidHeater.ON);
  connect(const2.y, idealControlledFluidHeater.To);
  connect(ramp.y, volumetricPump.cmd);
  connect(convSca2Sca.ss2, convSca2Sca1.ss1);
  connect(convSca2Sca.ss2, massT.surf);
  connect(source_T_fixed1.port, convSca2Sca1.ss2);
  connect(roomHeater.heatPort, convSca2Sca.ss1);
  connect(idealControlledFluidHeater.water_flange2, volumetricPump.water_flange1);
  connect(volumetricPump.water_flange2, idealPressuriser.water_flange1);
  connect(idealPressuriser.water_flange2, roomHeater.water_flange1);
  connect(roomHeater.water_flange2, idealControlledFluidHeater.water_flange1);
end test1_IdealControlledFluidHeater;
```

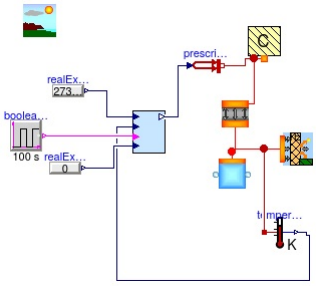
EEB.Components.AggregateComponents.Test.test_DCsupply



Modelica definition

```
model test_DCsupply
  Electrical.DCsupply_constantEfficiency Psupply(eta = 1 / 1.2);
  BaseComponents.Electrical.DC.Load_Pin_IO LoadDC;
  BaseComponents.Electrical.DC.Pmeter PM;
  BaseComponents.Electrical.Phasors.PQmeter PQM;
  BaseComponents.Electrical.Phasors.Vgen_Sine_Fixed Vac;
  BaseComponents.Electrical.Phasors.Ground gndAC;
  Modelica.Electrical.Analog.Basic.Ground gndDC;
  Modelica.Blocks.Sources.Trapezoid trapezoid(rising = 5, width = 5, falling = 5, period = 20, amplitude = 20);
equation
  connect(gndAC.p, Vac.n);
  connect(Vac.p, PQM.p1);
  connect(Vac.n, PQM.n1);
  connect(PQM.p2, Psupply.ACp);
  connect(PQM.n2, Psupply.ACn);
  connect(Psupply.DCp, PM.p1);
  connect(Psupply.DCn, PM.n1);
  connect(gndDC.p, PM.n1);
  connect(PM.p2, LoadDC.p);
  connect(PM.n2, LoadDC.n);
  connect(trapezoid.y, LoadDC.iPabs);
end test_DCsupply;
```

EEB.Components.AggregateComponents.Test.test_room_cooling



Modelica definition

```
model test_room_cooling
  Envelope.Walls.ExtenallWall_NoOpenings_Homogeneous extenallWall_NoOpenings_Homogeneous(s = 0.3, n = 10, inclination = 90, L = 10, orientation = 180, lambda = 0.8);
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_constant, Ta_avg = 273.15 + 35, Phimax = 1000, Xa_avg = 0.002, azi0 = 180);
  BaseComponents.Air.Volumes.AirVolume airVolume_closed(Xstart = 0.006, V = 80 * 3);
  Controllers.Blocks.Analogue.AWPI_1dof_trk aWPI_1dof( CSmax = 0, CSmin = -2000, K = 100,Ti = 200);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow prescribedHeatFlow;
  Modelica.Blocks.Sources.RealExpression realExpression(y = 273.15 + 20);
  Modelica.Blocks.Sources.RealExpression realExpression1(y = 0);
  Modelica.Blocks.Sources.BooleanPulse booleanPulse(width = 0.5, period = 100);
  BaseComponents.Thermal.Capacities.ThermalCap thermalCap(C = 5000);
  BaseComponents.Thermal.HeatTransfer.Convection_SS convection_SS(S = 1, gamma = 100);
equation
  connect(airVolume_closed.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(temperatureSensor.port, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(temperatureSensor.T, aWPI_1dof.PV);
  connect(aWPI_1dof.CS, prescribedHeatFlow.Q_flow);
  connect(realExpression.y, aWPI_1dof.SP);
  connect(realExpression1.y, aWPI_1dof.TR);
  connect(booleanPulse.y, aWPI_1dof.TS);
  connect(prescribedHeatFlow.port, thermalCap.surf);
  connect(convection_SS.ss1, thermalCap.surf);
  connect(convection_SS.ss2, extenallWall_NoOpenings_Homogeneous.airInt);
end test_room_cooling;
```

EEB.Appliances

Package Content

Name	Description
BaseClasses	
Office	
Accessories	
Test	

EEB.Appliances.BaseClasses

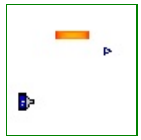
Information

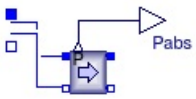
Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
BaseAppliance_DC	
BaseAppliance_AC	

EEB.Appliances.BaseClasses.BaseAppliance_DC





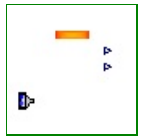
Connectors

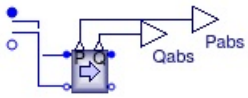
Type	Name	Description
PosNegPins	plug	
HeatPort	heatPort	
output RealOutput	Pabs	Instantaneous power as output signal

Modelica definition

```
partial model BaseAppliance_DC
  Interfaces.Electrical.PosNegPins plug;
  Interfaces.Thermal.HeatPort heatPort;
  Modelica.Blocks.Interfaces.RealOutput Pabs "Instantaneous power as output signal";
  Components.BaseComponents.Electrical.DC.Pmeter PM;
equation
  connect(plug.p, PM.p1);
  connect(plug.n, PM.n1);
  connect(PM.P, Pabs);
end BaseAppliance_DC;
```

EEB.Appliances.BaseClasses.BaseAppliance_AC





Connectors

Type	Name	Description
PosNegPhasorPins	plug	
HeatPort	heatPort	
output RealOutput	Pabs	Instantaneous power as output signal
output RealOutput	Qabs	Instantaneous power as output signal

Modelica definition

```
partial model BaseAppliance_AC
  Interfaces.Electrical.PosNegPhasorPins plug;
  Interfaces.Thermal.HeatPort heatPort;
  Modelica.Blocks.Interfaces.RealOutput Pabs "Instantaneous power as output signal";
  Modelica.Blocks.Interfaces.RealOutput Qabs "Instantaneous power as output signal";
  Components.BaseComponents.Electrical.Phasors.PQmeter PQM;
equation
  connect(PQM.p1, plug.p);
  connect(PQM.n1, plug.n);
  connect(Pabs, PQM.P);
  connect(PQM.Q, Qabs);
```

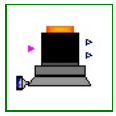
```
end BaseAppliance_AC;
```

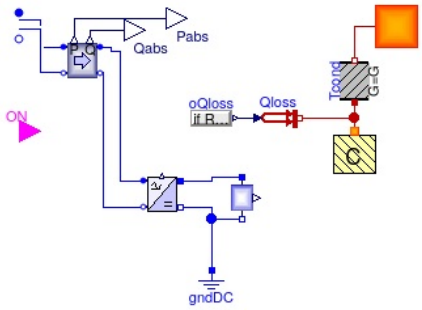
EEB.Appliances.Office

Package Content

Name	Description
DesktopComputer	

EEB.Appliances.Office.DesktopComputer





Information

Extends from BaseClasses.BaseAppliance_AC.

Parameters

Type	Name	Default	Description
Power	Pnom	20	Nominal absorbed power (avg) [W]
Voltage	Vdc	10	DC voltage [V]
Real	cphinom	0.9	nominal power factor
Real	eta	0.85	DC supply efficiency
Boolean	ReleaseQ	false	release heat to port
HeatCapacity	C	100	equivalent heat capacity [J/K]
ThermalConductance	G	2	equivalent thermal conductance [W/K]
Temperature	Tstart	273.15 + 20	initial T [K]

Connectors

Type	Name	Description
PosNegPhasorPins	plug	
HeatPort	heatPort	
output RealOutput	Pabs	Instantaneous power as output signal
output RealOutput	Qabs	Instantaneous power as output signal
input BooleanInput	ON	

Modelica definition

```
model DesktopComputer
  extends BaseClasses.BaseAppliance_AC;
  Components.AggregateComponents.Electrical.DCsupply_constantEfficiency Psupply(Vdc = Vdc, Isc = 20 * Pnom / Vdc, cphinom = cphinom, eta = eta);
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Components.BaseComponents.Electrical.DC.Load_Pfixed_I0 Load(Pnom = Pnom);
  parameter SI.Power Pnom = 20 "Nominal absorbed power (avg)";
  parameter SI.Voltage Vdc = 10 "DC voltage";
  parameter Real cphinom = 0.9 "nominal power factor";
  parameter Real eta = 0.85 "DC supply efficiency";
  parameter Boolean ReleaseQ = false "release heat to port";
  parameter HeatCapacity C = 100 "equivalent heat capacity";
  parameter ThermalConductance G = 2 "equivalent thermal conductance";
  parameter Temperature Tstart = 273.15 + 20 "initial T";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow Qloss;
  Modelica.Electrical.Analog.Basic.Ground gndDC;
  Modelica.Blocks.Sources.RealExpression oQloss(y = if ReleaseQ then Pabs else 0);
  Components.BaseComponents.Thermal.Capacities.ThermalCap Hcap(Tstart = Tstart);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor Tcond(G = G);
equation
  connect(PQM.n2, Psupply.ACn);
  connect(Psupply.DCp, Load.p);
  connect(Psupply.DCn, Load.n);
  connect(PQM.p2, Psupply.ACp);
  connect(gndDC.p, Load.n);
  connect(oQloss.y, Qloss.Q_flow);
  connect(Qloss.port, Hcap.surf);
  connect(Tcond.port_a, Hcap.surf);
  connect(Tcond.port_b, heatPort);
end DesktopComputer;
```

EEB.Appliances.Accessories

Information

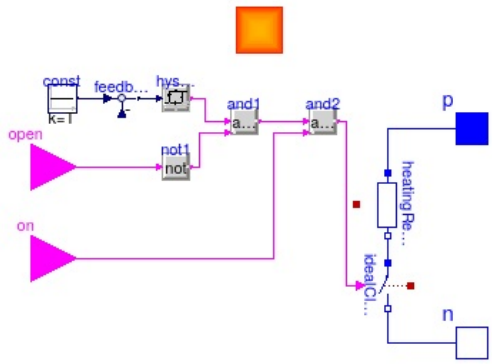
Extends from Modelica.Icons.Package (Icon for standard packages).

Package Content

Name	Description
BaseClasses	
MicrowaveOven	
SimpleRefrigerator	
CompactRefrigerator_FSC_2C	Compact refrigerator with variable speed compressor (PI), 2 independent circuits
CompactRefrigerator_VSC_2C	Compact refrigerator with variable speed compressor (PI), 2 independent circuits
CompactRefrigerator_FSC_1Cdamper	Compact refrigerator with variable speed compressor (PI), 1 circuit + damper

---

EEB.Appliances.Accessories.MicrowaveOven



Parameters

Type	Name	Default	Description
Real	T	200	?C Cooking temperature
Power	Pnom	1600	Nominal power consumption [W]
Voltage	Vnom	220	Nominal voltage [V]
Volume	Vo	0.2	Oven internal volume [m3]

Connectors

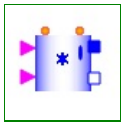
Type	Name	Description
input BooleanInput	open	
input BooleanInput	on	
HeatPort	e	
PositivePin	p	
NegativePin	n	

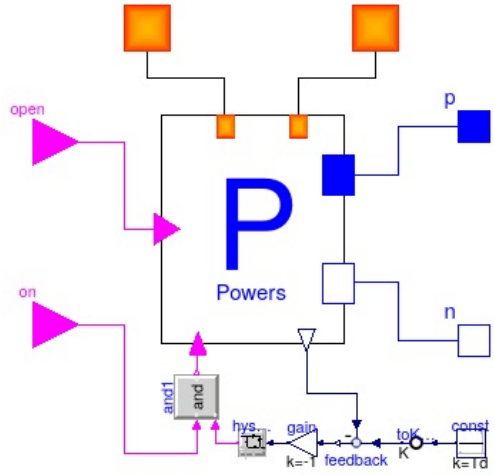
Modelica definition

```
model MicrowaveOven
  parameter Real T(min = 100, max = 250) = 200 "?C Cooking temperature";
  parameter SI.Power Pnom = 1600 "Nominal power consumption";
  parameter SI.Voltage Vnom = 220 "Nominal voltage";
  parameter SI.Volume Vo = 0.2 "Oven internal volume";
  SI.ThermalConductance G;
  SI.Temp_K Ta;
  SI.Temp_K Tae;
  Real Eel "Electrical consumption";
  Modelica.Blocks.Interfaces.BooleanInput open;
  Modelica.Blocks.Interfaces.BooleanInput on;
  Modelica.Blocks.Math.Feedback feedback;
  Modelica.Blocks.Logical.Hysteresis hysteresis(uLow = -10, uHigh = 10);
  Modelica.Blocks.Logical.And and1;
  Modelica.Blocks.Logical.And and2;
  Modelica.Blocks.Logical.Not not1;
  Modelica.Blocks.Sources.Constant const(k = T);
  Interfaces.Thermal.HeatPort e;
  Modelica.Electrical.Analog.Basic.HeatingResistor heatingResistor(R_ref = Vnom ^ 2 / Pnom, useHeatPort = false);
  Modelica.Electrical.Analog.Ideal.IdealClosingSwitch idealClosingSwitch;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
```

```
  G = if open then 1000 else 0.5;
  feedback.u2 = Ta - 273.15;
  1008 * Vo * der(Ta) = heatingResistor.LossPower + e.Q_flow;
  e.Q_flow = 10 * (e.T - Tae);
  e.Q_flow = G * (Tae - Ta);
  der(Eel) = (p.v - n.v) * p.i / 1000 / 3600;
  connect(const.y, feedback.u1);
  connect(feedback.y, hysteresis.u);
  connect(hysteresis.y, and1.u1);
  connect(not1.y, and1.u2);
  connect(open, not1.u);
  connect(and1.y, and2.u1);
  connect(and2.u2, on);
  connect(heatingResistor.n, idealClosingSwitch.p);
  connect(and2.y, idealClosingSwitch.control);
  connect(heatingResistor.p, p);
  connect(idealClosingSwitch.n, n);
end MicrowaveOven;
```

EEB.Appliances.Accessories.SimpleRefrigerator





Parameters

Type	Name	Default	Description
Power	Pnom	90	Compressor power [W]
Voltage	Vnom	220	Nominal voltage [V]
Volume	Vi	0.3	Refrigerator internal capacity [m3]
Real	Td	4	?C Desired refrigerator temperature

Connectors

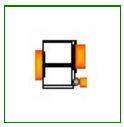
Type	Name	Description
HeatPort	ef	
HeatPort	er	
input BooleanInput	open	
input BooleanInput	on	
PositivePin	p	
NegativePin	n	

Modelica definition

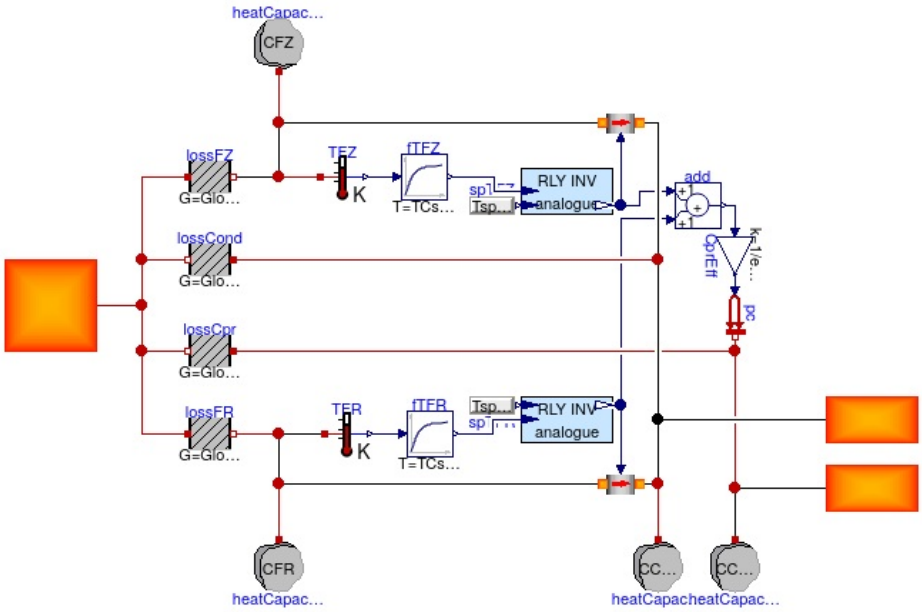
```
model SimpleRefrigerator
  parameter SI.Power Pnom = 90 "Compressor power";
  parameter SI.Voltage Vnom = 220 "Nominal voltage";
  parameter SI.Volume Vi = 0.3 "Refrigerator internal capacity";
  parameter Real Td(min = 2, max = 6) = 4 "?C Desired refrigerator temperature";
  Real Eel "Electrical consumption";
  BaseClasses.Refrigerator_powers Powers;
  Interfaces.Thermal.HeatPort ef;
  Interfaces.Thermal.HeatPort er;
  Modelica.Blocks.Interfaces.BooleanInput open;
  Modelica.Blocks.Interfaces.BooleanInput on;
  Modelica.Blocks.Logical.And and1;
  Modelica.Blocks.Math.Feedback feedback;
  Modelica.Blocks.Math.Gain gain(k = -1);
  Modelica.Blocks.Logical.Hysteresis hysteresis(uLow = -1, uHigh = 1);
  Modelica.Blocks.Sources.Constant const(k = Td);
  Modelica.Thermal.HeatTransfer.Celsius.ToKelvin toKelvin;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
```

```
  der(Eel) = (p.v - n.v) * p.i / 1000 / 3600;
  connect(ef, Powers.ef);
  connect(er, Powers.er);
  connect(open, Powers.open);
  connect(on, and1.u1);
  connect(and1.y, Powers.ON);
  connect(and1.u2, hysteresis.y);
  connect(gain.y, hysteresis.u);
  connect(gain.u, feedback.y);
  connect(Powers.T, feedback.u2);
  connect(feedback.u1, toKelvin.Kelvin);
  connect(toKelvin.Celsius, const.y);
  connect(Powers.p, p);
  connect(Powers.n, n);
end SimpleRefrigerator;
```

EEB.Appliances.Accessories.CompactRefrigerator_FSC_2C



Compact refrigerator with variable speed compressor (PI), 2 independent circuits



Parameters

Type	Name	Default	Description
Temperature	Tambnom	273.15 + 20	nominal ambient temperature [K]
Temperature	Tambmax	273.15 + 30	max rated ambient temperature [K]
Temperature	Tcondnom	273.15 + 40	nominal condenser temperature [K]
Temperature	Tcprnom	273.15 + 35	nominal condenser temperature [K]
Temperature	TspFR	273.15 + 4	fridge temperature SP [K]
Temperature	TspFZ	273.15 - 18	freezer temperature SP [K]
HeatFlowRate	QFRnom	60	nominal fridge cooling power [W]
HeatFlowRate	QFZnom	40	nominal freezer cooling power [W]
HeatCapacity	CFR	10000	fridge heat capacity [J/K]
HeatCapacity	CFZ	8000	freezer heat capacity [J/K]
HeatCapacity	CCond	100	condenser heat capacity [J/K]
Real	etaC	0.8	Compressor efficiency
Temperature	hist2FR	1	fridge hist/2 [K]
Temperature	hist2FZ	1	freezer hist/2 [K]
Time	TCsensTFZ	10	TFZ sensor TC [s]
Time	TCsensTFR	10	TFR sensor TC [s]

Connectors

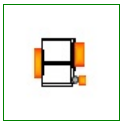
Type	Name	Description
HeatPort	portAmbient	
HeatPort	portHeatRecCompressor	
HeatPort	portHeatRecCondenser	

Modelica definition

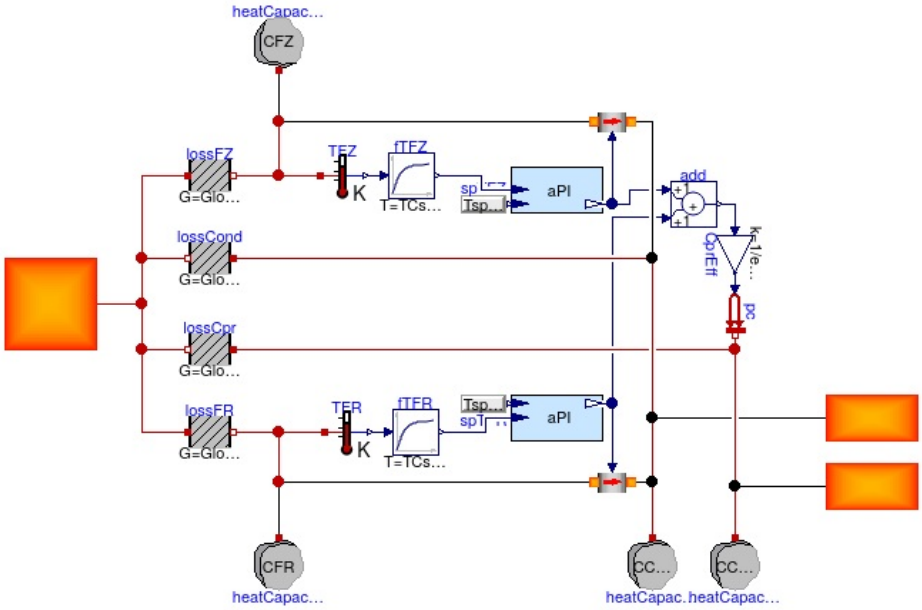
```
model CompactRefrigerator_FSC_2C "Compact refrigerator with variable speed compressor (PI), 2 independent circuits"
  parameter SI.Temperature Tambnom = 273.15 + 20 "nominal ambient temperature";
  parameter SI.Temperature Tambmax = 273.15 + 30 "max rated ambient temperature";
```

```
  parameter SI.Temperature Tcondnom = 273.15 + 40 "nominal condenser temperature";
  parameter SI.Temperature Tcprnom = 273.15 + 35 "nominal condenser temperature";
  parameter SI.Temperature TspFR = 273.15 + 4 "fridge temperature SP";
  parameter SI.Temperature TspFZ = 273.15 - 18 "freezer temperature SP";
  parameter SI.HeatFlowRate QFRnom = 60 "nominal fridge cooling power";
  parameter SI.HeatFlowRate QFZnom = 40 "nominal freezer cooling power";
  parameter SI.HeatCapacity CFR = 10000 "fridge heat capacity";
  parameter SI.HeatCapacity CFZ = 8000 "freezer heat capacity";
  parameter SI.HeatCapacity CCond = 100 "condenser heat capacity";
  parameter Real etaC = 0.8 "Compressor efficiency";
  parameter SI.Temperature hist2FR = 1 "fridge hist/2";
  parameter SI.Temperature hist2FZ = 1 "freezer hist/2";
  parameter SI.Time TCsensTFZ = 10 "TFZ sensor TC";
  parameter SI.Time TCsensTFR = 10 "TFR sensor TC";
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFZ;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFR;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFR(G = GlossFR);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFZ(G = GlossFZ);
  Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough QoFZ;
  Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough QoFR;
  Controllers.Blocks.Analogue.RLY_inv RFZ(CSlo = 0, CShist2 = hist2FZ, CShi = QmaxFZ);
  Controllers.Blocks.Analogue.RLY_inv RFR(CShi = QmaxFR, CSlo = 0, CShist2 = hist2FR);
  Modelica.Blocks.Sources.RealExpression spTFZ(y = TspFZ);
  Modelica.Blocks.Sources.RealExpression spTFR(y = TspFR);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCond(G = GlossCond) "condenser to ambient conductance";
  Interfaces.Thermal.HeatPort portAmbient;
  Modelica.Blocks.Math.Add add;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCpr(G = GlossCpr) "compressor to ambient conductance";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pc;
  Modelica.Blocks.Math.Gain CprEff(k = 1 / etaC - 1);
  Interfaces.Thermal.HeatPort portHeatRecCompressor;
  Interfaces.Thermal.HeatPort portHeatRecCondenser;
  Modelica.Blocks.Continuous.FirstOrder fTFZ(T = TCsensTFZ, initType = Modelica.Blocks.Types.Init.InitialOutput, y_start = TspFZ);
  Modelica.Blocks.Continuous.FirstOrder fTFR(initType = Modelica.Blocks.Types.Init.InitialOutput, T = TCsensTFR, y_start = TspFR);
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor1(C = CFR, T(start = TspFR, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor2(T(start = Tcondnom, fixed = true), C = CCond);
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor3(C = CCond, T(start = Tcondnom, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor(C = CFZ, T(start = TspFZ, fixed = true));
protected
  parameter SI.ThermalConductance GlossFR = QFRnom / (Tambnom - TspFR);
  parameter SI.ThermalConductance GlossFZ = QFZnom / (Tambnom - TspFZ);
  parameter SI.ThermalConductance GlossCond = (QFRnom + QFZnom) / (Tcondnom - Tambnom);
  parameter SI.ThermalConductance GlossCpr = (QFRnom + QFZnom) / etaC / (Tcprnom - Tambnom);
  parameter SI.HeatFlowRate QmaxFR = GlossFR * (Tambmax - TspFR);
  parameter SI.HeatFlowRate QmaxFZ = GlossFZ * (Tambmax - TspFZ);
equation
  connect(RFZ.CS, QoFZ.pwr);
  connect(RFR.CS, QoFR.pwr);
  connect(spTFZ.y, RFZ.SP);
  connect(spTFR.y, RFR.SP);
  connect(lossFZ.port_a, portAmbient);
  connect(lossFR.port_a, portAmbient);
  connect(RFZ.CS, add.u1);
  connect(RFR.CS, add.u2);
  connect(lossCond.port_b, portAmbient);
  connect(lossCpr.port_b, portAmbient);
  connect(add.y, CprEff.u);
  connect(CprEff.y, pc.Q_flow);
  connect(TFZ.T, fTFZ.u);
  connect(fTFZ.y, RFZ.PV);
  connect(TFR.T, fTFR.u);
  connect(fTFR.y, RFR.PV);
  connect(heatCapacitor.port, lossFZ.port_b);
  connect(TFZ.port, lossFZ.port_b);
  connect(QoFZ.heatSrc, lossFZ.port_b);
  connect(heatCapacitor2.port, QoFZ.heatSnk);
  connect(lossCond.port_a, QoFZ.heatSnk);
  connect(pc.port, heatCapacitor3.port);
  connect(lossCpr.port_a, heatCapacitor3.port);
  connect(lossFR.port_b, TFR.port);
  connect(heatCapacitor1.port, TFR.port);
  connect(QoFR.heatSrc, TFR.port);
  connect(QoFR.heatSnk, QoFZ.heatSnk);
  connect(portHeatRecCondenser, QoFZ.heatSnk);
  connect(portHeatRecCompressor, heatCapacitor3.port);
end CompactRefrigerator_FSC_2C;
```

EEB.Appliances.Accessories.CompactRefrigerator_VSC_2C



Compact refrigerator with variable speed compressor (PI), 2 independent circuits



Parameters

Type	Name	Default	Description
Temperature	Tambnom	273.15 + 20	nominal ambient temperature [K]
Temperature	Tambmax	273.15 + 30	max rated ambient temperature [K]
Temperature	Tcondnom	273.15 + 40	nominal condenser temperature [K]
Temperature	Tcprnom	273.15 + 35	nominal condenser temperature [K]
Temperature	TspFR	273.15 + 4	fridge temperature SP [K]
Temperature	TspFZ	273.15 - 18	freezer temperature SP [K]
HeatFlowRate	QFRnom	60	nominal fridge cooling power [W]
HeatFlowRate	QFZnom	40	nominal freezer cooling power [W]
HeatCapacity	CFR	10000	fridge heat capacity [J/K]
HeatCapacity	CFZ	8000	freezer heat capacity [J/K]
HeatCapacity	CCond	100	condenser heat capacity [J/K]
Real	etaC	0.8	Compressor efficiency
Real	KFR	1	fridge PI gain
Time	TiFR	50	fridge PI integral time [s]
Real	KFZ	1	freezer PI gain
Time	TiFZ	50	freezer PI integral time [s]
Time	TCsensTFZ	10	TFZ sensor TC [s]
Time	TCsensTFR	10	TFR sensor TC [s]

Connectors

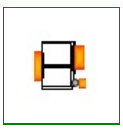
Type	Name	Description
HeatPort	portAmbient	
HeatPort	portHeatRecCompressor	
HeatPort	portHeatRecCondenser	

Modelica definition

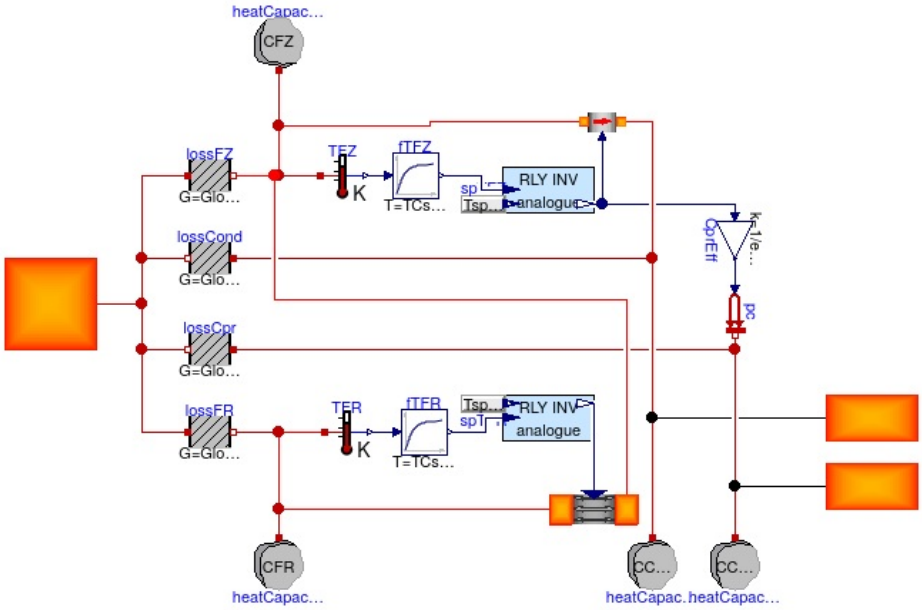
```
model CompactRefrigerator_VSC_2C "Compact refrigerator with variable speed compressor (PI), 2 independent circuits"
  parameter SI.Temperature Tambnom = 273.15 + 20 "nominal ambient temperature";
  parameter SI.Temperature Tambmax = 273.15 + 30 "max rated ambient temperature";
  parameter SI.Temperature Tcondnom = 273.15 + 40 "nominal condenser temperature";
  parameter SI.Temperature Tcprnom = 273.15 + 35 "nominal condenser temperature";
  parameter SI.Temperature TspFR = 273.15 + 4 "fridge temperature SP";
  parameter SI.Temperature TspFZ = 273.15 - 18 "freezer temperature SP";
  parameter SI.HeatFlowRate QFRnom = 60 "nominal fridge cooling power";
  parameter SI.HeatFlowRate QFZnom = 40 "nominal freezer cooling power";
  parameter SI.HeatCapacity CFR = 10000 "fridge heat capacity";
  parameter SI.HeatCapacity CFZ = 8000 "freezer heat capacity";
  parameter SI.HeatCapacity CCond = 100 "condenser heat capacity";
  parameter Real etaC = 0.8 "Compressor efficiency";
  parameter Real KFR = 1 "fridge PI gain";
  parameter SI.Time TiFR = 50 "fridge PI integral time";
  parameter Real KFZ = 1 "freezer PI gain";
  parameter SI.Time TiFZ = 50 "freezer PI integral time";
  parameter SI.Time TCsensTFZ = 10 "TFZ sensor TC";
  parameter SI.Time TCsensTFR = 10 "TFR sensor TC";
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFZ;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFR;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFR(G = GlossFR);
```

```
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFZ(G = GlossFZ);
  Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough QoFZ;
  Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough QoFR;
  Controllers.Blocks.Analogue.AWPI_1dof RFZ(Ti = TiFZ, CSmin = 0, CSmax = QmaxFZ, K = -KFZ);
  Controllers.Blocks.Analogue.AWPI_1dof RFR(Ti = TiFR, CSmax = QmaxFR, CSmin = 0, K = -KFR);
  Modelica.Blocks.Sources.RealExpression spTFZ(y = TspFZ);
  Modelica.Blocks.Sources.RealExpression spTFR(y = TspFR);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCond(G = GlossCond) "condenser to ambient conductance";
  Interfaces.Thermal.HeatPort portAmbient;
  Modelica.Blocks.Math.Add add;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCpr(G = GlossCpr) "compressor to ambient conductance";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pc;
  Modelica.Blocks.Math.Gain CprEff(k = 1 / etaC - 1);
  Interfaces.Thermal.HeatPort portHeatRecCompressor;
  Interfaces.Thermal.HeatPort portHeatRecCondenser;
  Modelica.Blocks.Continuous.FirstOrder fTFZ(T = TCsensTFZ, initType = Modelica.Blocks.Types.Init.InitialOutput, y_start = TspFZ);
  Modelica.Blocks.Continuous.FirstOrder fTFR(initType = Modelica.Blocks.Types.Init.InitialOutput, T = TCsensTFR, y_start = TspFR);
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor(C = CFZ, T(start = TspFZ, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor1(T(start = TspFR, fixed = true), C = CFR);
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor2(C = CCond, T(start = Tcondnom, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor3(C = CCond, T(start = Tcondnom, fixed = true));
protected
  parameter SI.ThermalConductance GlossFR = QFRnom / (Tambnom - TspFR);
  parameter SI.ThermalConductance GlossFZ = QFZnom / (Tambnom - TspFZ);
  parameter SI.ThermalConductance GlossCond = (QFRnom + QFZnom) / (Tcondnom - Tambnom);
  parameter SI.ThermalConductance GlossCpr = (QFRnom + QFZnom) / etaC / (Tcprnom - Tambnom);
  parameter SI.HeatFlowRate QmaxFR = GlossFR * (Tambmax - TspFR);
  parameter SI.HeatFlowRate QmaxFZ = GlossFZ * (Tambmax - TspFZ);
equation
  connect(RFZ.CS, QoFZ.pwr);
  connect(RFR.CS, QoFR.pwr);
  connect(spTFZ.y, RFZ.SP);
  connect(spTFR.y, RFR.SP);
  connect(lossFZ.port_a, portAmbient);
  connect(lossFR.port_a, portAmbient);
  connect(RFZ.CS, add.u1);
  connect(RFR.CS, add.u2);
  connect(lossCond.port_b, portAmbient);
  connect(lossCpr.port_b, portAmbient);
  connect(add.y, CprEff.u);
  connect(CprEff.y, pc.Q_flow);
  connect(fTFZ.y, RFZ.PV);
  connect(fTFR.y, RFR.PV);
  connect(TFR.T, fTFR.u);
  connect(TFZ.T, fTFZ.u);
  connect(lossFZ.port_b, heatCapacitor.port);
  connect(TFZ.port, heatCapacitor.port);
  connect(QoFZ.heatSnk, heatCapacitor2.port);
  connect(QoFZ.heatSrc, heatCapacitor.port);
  connect(lossFR.port_b, heatCapacitor1.port);
  connect(TFR.port, heatCapacitor1.port);
  connect(QoFR.heatSrc, heatCapacitor1.port);
  connect(QoFR.heatSnk, heatCapacitor2.port);
  connect(pc.port, heatCapacitor3.port);
  connect(portHeatRecCompressor, heatCapacitor3.port);
  connect(portHeatRecCondenser, heatCapacitor2.port);
  connect(lossCpr.port_a, heatCapacitor3.port);
  connect(lossCond.port_a, heatCapacitor2.port);
end CompactRefrigerator_VSC_2C;
```

EEB.Appliances.Accessories.CompactRefrigerator_FSC_1Cdamper



Compact refrigerator with variable speed compressor (PI), 1 circuit + damper



Parameters

Type	Name	Default	Description
Temperature	Tambnom	273.15 + 20	nominal ambient temperature [K]
Temperature	Tambmax	273.15 + 30	max rated ambient temperature [K]
Temperature	Tcondnom	273.15 + 40	nominal condenser temperature [K]
Temperature	Tcprnom	273.15 + 35	nominal condenser temperature [K]
Temperature	TspFR	273.15 + 4	fridge temperature SP [K]
Temperature	TspFZ	273.15 - 18	freezer temperature SP [K]
HeatFlowRate	QFRnom	60	nominal fridge cooling power [W]
HeatFlowRate	QFZnom	40	nominal freezer cooling power [W]
HeatCapacity	CFR	10000	fridge heat capacity [J/K]
HeatCapacity	CFZ	8000	freezer heat capacity [J/K]
HeatCapacity	CCond	100	condenser heat capacity [J/K]
ThermalConductance	Gdamper	100	open damper equiv conductance [W/K]
Real	etaC	0.8	Compressor efficiency
Temperature	hist2FR	0.5	fridge hist/2 [K]
Temperature	hist2FZ	0.5	freezer hist/2 [K]
Time	TCsensTFZ	10	TFZ sensor TC [s]
Time	TCsensTFR	10	TFR sensor TC [s]

Connectors

Type	Name	Description
HeatPort	portAmbient	
HeatPort	portHeatRecCompressor	
HeatPort	portHeatRecCondenser	

Modelica definition

```
model CompactRefrigerator_FSC_1Cdamper "Compact refrigerator with variable speed compressor (PI), 1 circuit + damper"
  parameter SI.Temperature Tambnom = 273.15 + 20 "nominal ambient temperature";
  parameter SI.Temperature Tambmax = 273.15 + 30 "max rated ambient temperature";
  parameter SI.Temperature Tcondnom = 273.15 + 40 "nominal condenser temperature";
  parameter SI.Temperature Tcprnom = 273.15 + 35 "nominal condenser temperature";
  parameter SI.Temperature TspFR = 273.15 + 4 "fridge temperature SP";
  parameter SI.Temperature TspFZ = 273.15 - 18 "freezer temperature SP";
  parameter SI.HeatFlowRate QFRnom = 60 "nominal fridge cooling power";
  parameter SI.HeatFlowRate QFZnom = 40 "nominal freezer cooling power";
  parameter SI.HeatCapacity CFR = 10000 "fridge heat capacity";
  parameter SI.HeatCapacity CFZ = 8000 "freezer heat capacity";
  parameter SI.HeatCapacity CCond = 100 "condenser heat capacity";
  parameter SI.ThermalConductance Gdamper = 100 "open damper equiv conductance";
  parameter Real etaC = 0.8 "Compressor efficiency";
  parameter SI.Temperature hist2FR = 0.5 "fridge hist/2";
  parameter SI.Temperature hist2FZ = 0.5 "freezer hist/2";
  parameter SI.Time TCsensTFZ = 10 "TFZ sensor TC";
  parameter SI.Time TCsensTFR = 10 "TFR sensor TC";
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFZ;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor TFR;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFR(G = GlossFR);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossFZ(G = GlossFZ);
  Components.BaseComponents.Thermal.HeatTransfer.PrescribedHeatFlowThrough QoFZ;
```

```
  Controllers.Blocks.Analogue.RLY_inv RFZ(CSlo = 0, CShist2 = hist2FZ, CShi = QmaxFZ + QmaxFR);
  Controllers.Blocks.Analogue.RLY_inv RFR(CSlo = 0, CShist2 = hist2FR, CShi = Gdamper);
  Modelica.Blocks.Sources.RealExpression spTFZ(y = TspFZ);
  Modelica.Blocks.Sources.RealExpression spTFR(y = TspFR);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCond(G = GlossCond) "condenser to ambient conductance";
  Interfaces.Thermal.HeatPort portAmbient;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor lossCpr(G = GlossCpr) "compressor to ambient conductance";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pc;
  Modelica.Blocks.Math.Gain CprEff(k = 1 / etaC - 1);
  Interfaces.Thermal.HeatPort portHeatRecCompressor;
  Interfaces.Thermal.HeatPort portHeatRecCondenser;
  Modelica.Blocks.Continuous.FirstOrder fTFZ(T = TCsensTFZ, initType = Modelica.Blocks.Types.Init.InitialOutput, y_start = TspFZ);
  Modelica.Blocks.Continuous.FirstOrder fTFR(initType = Modelica.Blocks.Types.Init.InitialOutput, T = TCsensTFR, y_start = TspFR);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS_gammaIn convSca2ScaGin;
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor(C = CFZ, T(start = TspFZ, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor1(T(start = TspFR, fixed = true), C = CFR);
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor2(C = CCond, T(start = Tcondnom, fixed = true));
  Modelica.Thermal.HeatTransfer.Components.HeatCapacitor heatCapacitor3(C = CCond, T(start = Tcondnom, fixed = true));
protected
  parameter SI.ThermalConductance GlossFR = QFRnom / (Tambnom - TspFR);
  parameter SI.ThermalConductance GlossFZ = QFZnom / (Tambnom - TspFZ);
  parameter SI.ThermalConductance GlossCond = (QFRnom + QFZnom) / (Tcondnom - Tambnom);
  parameter SI.ThermalConductance GlossCpr = (QFRnom + QFZnom) / etaC / (Tcprnom - Tambnom);
  parameter SI.HeatFlowRate QmaxFR = GlossFR * (Tambmax - TspFR);
  parameter SI.HeatFlowRate QmaxFZ = GlossFZ * (Tambmax - TspFZ);
equation
  connect(RFZ.CS, QoFZ.pwr);
  connect(spTFZ.y, RFZ.SP);
  connect(spTFR.y, RFR.SP);
  connect(lossFZ.port_a, portAmbient);
  connect(lossFR.port_a, portAmbient);
  connect(lossCond.port_b, portAmbient);
  connect(lossCpr.port_b, portAmbient);
  connect(CprEff.y, pc.Q_flow);
  connect(RFZ.CS, CprEff.u);
  connect(TFZ.T, fTFZ.u);
  connect(fTFZ.y, RFZ.PV);
  connect(TFR.T, fTFR.u);
  connect(fTFR.y, RFR.PV);
  connect(RFR.CS, convSca2ScaGin.gamma);
  connect(heatCapacitor.port, lossFZ.port_b);
  connect(TFZ.port, lossFZ.port_b);
  connect(lossCond.port_a, heatCapacitor2.port);
  connect(pc.port, heatCapacitor3.port);
  connect(portHeatRecCompressor, heatCapacitor3.port);
  connect(portHeatRecCondenser, heatCapacitor2.port);
  connect(lossCpr.port_a, heatCapacitor3.port);
  connect(lossFR.port_b, TFR.port);
  connect(heatCapacitor1.port, TFR.port);
  connect(QoFZ.heatSrc, lossFZ.port_b);
  connect(QoFZ.heatSnk, heatCapacitor2.port);
  connect(convSca2ScaGin.ss2, lossFZ.port_b);
  connect(convSca2ScaGin.ss1, TFR.port);
end CompactRefrigerator_FSC_1Cdamper;
```

EEB.Appliances.Accessories.BaseClasses

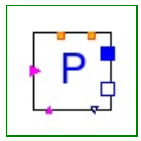
Information

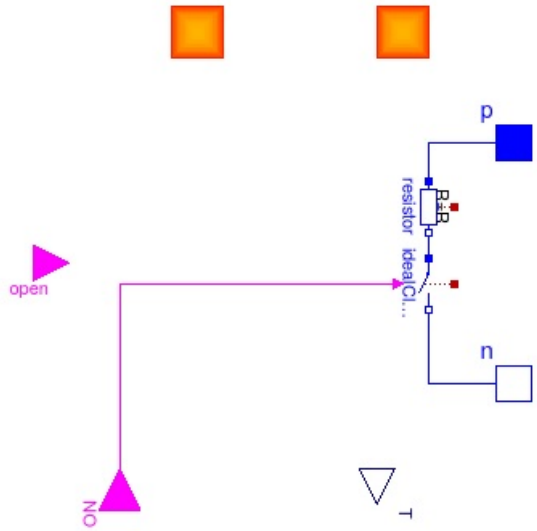
Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
Refrigerator_powers	

EEB.Appliances.Accessories.BaseClasses.Refrigerator_powers





Parameters

Type	Name	Default	Description
Power	Pnom	90	Compressor power [W]
Voltage	Vnom	220	Nominal voltage [V]
Resistance	R	Vnom^2/Pnom	Compressor resistance [Ohm]
Volume	Vi	0.3	Refrigerator internal capacity [m3]
HeatCapacity	Ci	1008*Vi	[J/K]
HeatCapacity	Cev	0.5	[J/K]
HeatCapacity	Ccond	0.5	[J/K]
ThermalConductance	Guf	5	Thermal conductance for opening of the door [W/K]
ThermalConductance	Gr	2	[W/K]
ThermalConductance	Gi_ev	0.05	[W/K]

Connectors

Type	Name	Description
input BooleanInput	ON	
input BooleanInput	open	
HeatPort	ef	
HeatPort	er	
output RealOutput	T	
PositivePin	p	
NegativePin	n	

Modelica definition

```
model Refrigerator_powers
  // electrical parameters
  parameter SI.Power Pnom = 90 "Compressor power";
  parameter SI.Voltage Vnom = 220 "Nominal voltage";
  parameter SI.Resistance R = Vnom ^ 2 / Pnom "Compressor resistance";
```

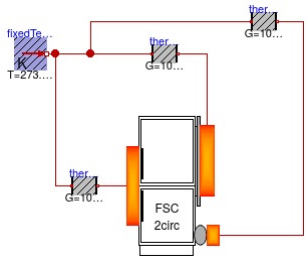
```
  // cell parameters
  parameter SI.Volume Vi = 0.3 "Refrigerator internal capacity";
  parameter SI.HeatCapacity Ci = 1008 * Vi;
  SI.Temp_K Ti;
  SI.Energy Ei(start = Ci * 277.15);
  // evaporator parameters
  parameter SI.HeatCapacity Cev = 0.5;
  SI.Temp_K Tev;
  SI.Energy Eev(start = Cev * 253.15);
  // condenser parameters
  parameter SI.HeatCapacity Ccond = 0.5;
  SI.Temp_K Tcond;
  SI.Energy Econd(start = Ccond * 318.15);
  // thermal conductance
  parameter SI.ThermalConductance Guf = 5 "Thermal conductance for opening of the door";
  parameter SI.ThermalConductance Gr = 2;
  SI.Temp_K Tuf;
  SI.ThermalConductance G;
  parameter SI.ThermalConductance Gi_ev = 0.05;
  // heat flow rate
  SI.HeatFlowRate Qi_ev;
  Modelica.Blocks.Interfaces.BooleanInput ON;
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = R);
  Modelica.Electrical.Analog.Ideal.IdealClosingSwitch idealClosingSwitch;
  Modelica.Blocks.Interfaces.BooleanInput open;
  Interfaces.Thermal.HeatPort ef;
  Interfaces.Thermal.HeatPort er;
  Modelica.Blocks.Interfaces.RealOutput T;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
  if ON then
    der(Eev) = 0;
    der(Econd) = 0;
    Tev = 253.15;
    Tcond = 318.15;
  else
    Eev = Cev * Tev;
    Econd = Ccond * Tcond;
    der(Eev) = Qi_ev;
    der(Econd) = er.Q_flow;
  end if;
  Ei = Ci * Ti;
  der(Ei) = ef.Q_flow - Qi_ev;
  Qi_ev = Gi_ev * (Ti - Tev);
  T = Ti;
  ef.Q_flow = Guf * (ef.T - Tuf);
  ef.Q_flow = G * (Tuf - Ti);
  er.Q_flow = Gr * (er.T - Tcond);
  G = if open then 1000 else 0.005;
  connect(resistor.n, idealClosingSwitch.p);
  connect(idealClosingSwitch.control, ON);
  connect(resistor.p, p);
  connect(idealClosingSwitch.n, n);
end Refrigerator_powers;
```

EEB.Appliances.Test

Package Content

Name	Description
test4_FSC_2c	
test5_FSC_Damper	
test8_ElectricBoiler	
test9_FuelBoiler	
Test_Office_01	
Test_Office_02	

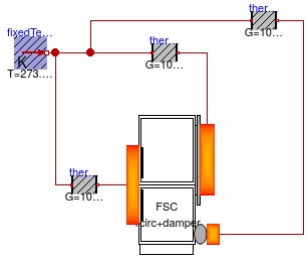
EEB.Appliances.Test.test4_FSC_2c



Modelica definition

```
model test4_FSC_2c
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor2(G = 100);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 273.15 + 20);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 100);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor3(G = 100);
  Accessories.CompactRefrigerator_FSC_2C compactRefrigerator_FSC_2C;
equation
  connect(fixedTemperature1.port, thermalConductor2.port_a);
  connect(thermalConductor2.port_b, compactRefrigerator_FSC_2C.portHeatRecCondenser);
  connect(thermalConductor1.port_a, thermalConductor2.port_a);
  connect(thermalConductor1.port_b, compactRefrigerator_FSC_2C.portHeatRecCompressor);
  connect(thermalConductor3.port_a, thermalConductor2.port_a);
  connect(thermalConductor3.port_b, compactRefrigerator_FSC_2C.portAmbient);
end test4_FSC_2c;
```

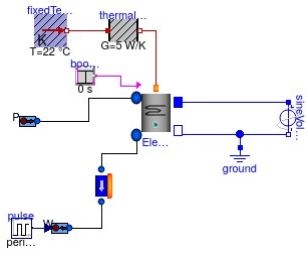
EEB.Appliances.Test.test5_FSC_Damper



Modelica definition

```
model test5_FSC_Damper
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor2(G = 100);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature1(T = 273.15 + 20);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 100);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor3(G = 100);
  Accessories.CompactRefrigerator_FSC_1Cdamper compactRefrigerator_FSC_2C;
equation
  connect(fixedTemperature1.port, thermalConductor2.port_a);
  connect(thermalConductor2.port_b, compactRefrigerator_FSC_2C.portHeatRecCondenser);
  connect(thermalConductor1.port_a, thermalConductor2.port_a);
  connect(thermalConductor1.port_b, compactRefrigerator_FSC_2C.portHeatRecCompressor);
  connect(thermalConductor3.port_a, thermalConductor2.port_a);
  connect(thermalConductor3.port_b, compactRefrigerator_FSC_2C.portAmbient);
end test5_FSC_Damper;
```

EEB.Appliances.Test.test8_ElectricBoiler

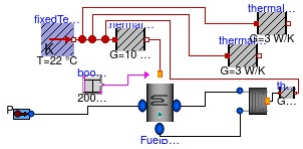


Modelica definition

```
model test8_ElectricBoiler
  Components.AggregateComponents.Heating.ElectricBoiler ElectricBoiler(Td = 273.15 + 70, Vboiler = 0.1, Pnom = 20000);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 5);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 295.15);
  Modelica.Electrical.Analog.Sources.SineVoltage sineVoltage(V = 220, freqHz = 50);
  Modelica.Electrical.Analog.Basic.Ground ground;
  Components.BaseComponents.Water.Sinks.WaterSink_W sink_P_fixed;
  Components.BaseComponents.Water.Sources.WaterSource_PT_fixed source_PT_fixed(T0 = 273.15 + 25);
  Modelica.Blocks.Sources.BooleanStep booleanStep;
  Components.BaseComponents.Water.Pipes.BaseClasses.WaterPipeExchangingElement pipeExchangingElement;
  Modelica.Blocks.Sources.Pulse pulse(startTime = 3600, period = 3600, width = 0.5, amplitude = 0.01);
equation
  connect(thermalConductor.port_b, ElectricBoiler.e);
```

```
  connect(thermalConductor.port_a, fixedTemperature.port);
  connect(ElectricBoiler.p, sineVoltage.p);
  connect(sineVoltage.n, ElectricBoiler.n);
  connect(sineVoltage.n, ground.p);
  connect(booleanStep.y, ElectricBoiler.ON);
  connect(sink_P_fixed.Win, pulse.y);
  connect(ElectricBoiler.water_flange1, source_PT_fixed.water_flange);
  connect(ElectricBoiler.water_flange2, pipeExchangingElement.water_flange1);
  connect(pipeExchangingElement.water_flange2, sink_P_fixed.water_flange);
end test8_ElectricBoiler;
```

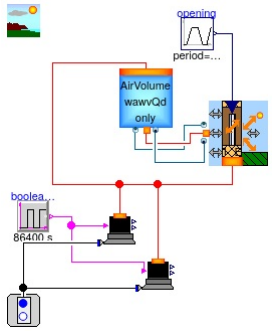
EEB.Appliances.Test.test9_FuelBoiler



Modelica definition

```
model test9_FuelBoiler
  Components.AggregateComponents.Heating.FuelBoiler FuelBoiler(Td = 273.15 + 70);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor(G = 10);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature fixedTemperature(T = 295.15);
  Components.BaseComponents.Water.Sources.WaterSource_PT_fixed source_PT_fixed(T0 = 273.15 + 15);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor1(G = 3);
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor2(G = 3);
  Modelica.Blocks.Sources.BooleanStep booleanStep(startTime = 2000);
  Components.AggregateComponents.Heating.RoomHeater roomHeater;
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor thermalConductor3(G = 3);
equation
  connect(thermalConductor.port_b, FuelBoiler.e);
  connect(thermalConductor.port_a, fixedTemperature.port);
  connect(thermalConductor1.port_a, fixedTemperature.port);
  connect(thermalConductor2.port_a, fixedTemperature.port);
  connect(booleanStep.y, FuelBoiler.ON);
  connect(thermalConductor3.port_b, fixedTemperature.port);
  connect(roomHeater.heatPort, thermalConductor3.port_a);
  connect(FuelBoiler.water_flange2, roomHeater.water_flange1);
  connect(FuelBoiler.water_flange3, roomHeater.water_flange2);
  connect(FuelBoiler.water_flange1, source_PT_fixed.water_flange);
end test9_FuelBoiler;
```

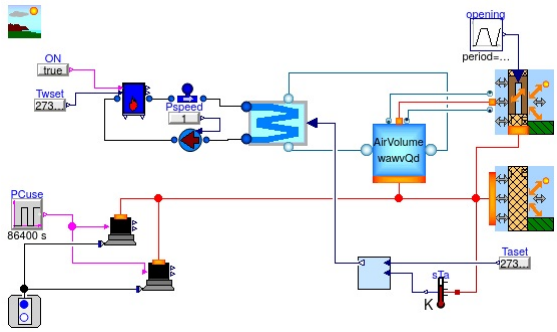
EEB.Appliances.Test.Test_Office_01



Modelica definition

```
model Test_Office_01
  Office.DesktopComputer desktopComputer(ReleaseQ = true);
  Office.DesktopComputer desktopComputer1(ReleaseQ = true);
  Components.AggregateComponents.Electrical.FixedACsupply_socket_withGround fixedACsupply_socket_withGround;
  inner BoundaryConditions.AmbientConditions ambient_settings(Ta_Yex = 20, acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15);
  Components.BaseComponents.Air.Volumes.AirVolume_only_wawvQdport V1(V = 20);
  Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_DoubleGlass
                                                                                    Window(material_glass = EEB.Media.Materials.Glasses.Glass(), material_gas = EEB.Media.Materials.Enve
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, startTime = 100, amplitude = 1, width = 120, period = 7200);
  Modelica.Blocks.Sources.BooleanPulse booleanPulse(width = 40, period = 86400, startTime = 8 * 3600);
equation
  connect(opening.y, Window.opening01);
  connect(V1.heatPort, Window.absToWall);
  connect(desktopComputer1.heatPort, Window.absToWall);
  connect(desktopComputer.heatPort, Window.absToWall);
  connect(booleanPulse.y, desktopComputer1.ON);
  connect(booleanPulse.y, desktopComputer.ON);
  connect(desktopComputer.plug, fixedACsupply_socket_withGround.socket);
  connect(desktopComputer1.plug, fixedACsupply_socket_withGround.socket);
  connect(V1.dryair, Window.dryair);
  connect(V1.diffuse, Window.diffuse);
  connect(V1.vapour, Window.vapour);
end Test_Office_01;
```

EEB.Appliances.Test.Test_Office_02



Modelica definition

```
model Test_Office_02
  Office.DesktopComputer desktopComputer(ReleaseQ = true);
  Office.DesktopComputer desktopComputer1(ReleaseQ = true);
  Components.AggregateComponents.Electrical.FixedACsupply_socket_withGround fixedACsupply_socket_withGround;
  inner BoundaryConditions.AmbientConditions ambient_settings(Ta_Yex = 20, acv = EEB.Types.AmbCondVariabilty.ACV_variable, Ta_avg = 288.15);
  Components.BaseComponents.Air.Volumes.AirVolume wawvQdPort V1(V = 5 * 4 * 3);
  Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_SingleGlass
                                                                                Window(L = 3, H = 3, s = 0.05, material = EEB.Media.Materials.Glasses.Glass(), orientation = 180,
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, amplitude = 1, width = 120, period = 12 * 3600, startTime = 8 * 3600);
  Modelica.Blocks.Sources.BooleanPulse PCuse(width = 40, period = 86400, startTime = 8 * 3600);
  Components.AggregateComponents.Envelope.Walls.ExtenallWall_NoOpenings_Homogeneous
                                                                              extenallWall_NoOpenings_Homogeneous(L = 4, H = 1, orientation = 180, lambda = 0.4);
  Components.AggregateComponents.Heating.FanCoil fanCoil(ncoil = 5, gammaTubeMetal = 100, gammaMetalExternal = 20, Lcoil = 5);
  Components.AggregateComponents.Heating.IdealControlledFluidHeater idealControlledFluidHeater;
  Components.BaseComponents.Water.Pressurisers.IdealWaterPressuriser_pfixed idealWaterPressuriser;
  Modelica.Blocks.Sources.BooleanExpression ON(y = true);
  Modelica.Blocks.Sources.RealExpression Twset(y = 273.15 + 55);
  Modelica.Blocks.Sources.RealExpression Pspeed(y = 1);
  Components.BaseComponents.Water.Pumps.WaterPump_Volumetric waterPump_Volumetric;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sTa;
  Modelica.Blocks.Sources.RealExpression Taset(y = 273.15 + 20);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_1dof(K = 1, Ti = 200);
equation
  connect(opening.y, Window.opening01);
  connect(PCuse.y, desktopComputer1.ON);
  connect(PCuse.y, desktopComputer.ON);
  connect(desktopComputer.plug, fixedACsupply_socket_withGround.socket);
  connect(desktopComputer1.plug, fixedACsupply_socket_withGround.socket);
  connect(Window.absToWall, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(V1.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(desktopComputer1.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(desktopComputer.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(fanCoil.air_flange2, V1.air_flange2);
  connect(fanCoil.air_flange1, V1.air_flange1);
  connect(ON.y, idealControlledFluidHeater.ON);
  connect(idealControlledFluidHeater.To, Twset.y);
  connect(Pspeed.y, waterPump_Volumetric.cmd);
  connect(sTa.port, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(aWPI_1dof.SP, Taset.y);
  connect(aWPI_1dof.PV, sTa.T);
  connect(fanCoil.icmd01, aWPI_1dof.CS);
  connect(idealControlledFluidHeater.water_flange2, idealWaterPressuriser.water_flange1);
  connect(idealWaterPressuriser.water_flange2, fanCoil.water_flange1);
  connect(fanCoil.water_flange2, waterPump_Volumetric.water_flange1);
  connect(waterPump_Volumetric.water_flange2, idealControlledFluidHeater.water_flange1);
  connect(V1.dryair, Window.dryair);
  connect(Window.diffuse, V1.diffuse);
  connect(Window.vapour, V1.vapour);
end Test_Office_02;
```

EEB.Controllers

Package Content

Name	Description
Templates	
Blocks	
Actuation	
AggregateBlocks	
Test	

EEB.Controllers.Templates

Package Content

Name	Description
BaseClasses	
Analogue	
Digital	

EEB.Controllers.Templates.BaseClasses

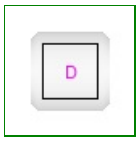
Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
DigitalControlDiagram	

EEB.Controllers.Templates.BaseClasses.DigitalControlDiagram



Modelica definition

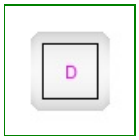
```
model DigitalControlDiagram
  inner Time Ts = settings.Ts;
  Settings.DigitalControlSettings settings;
end DigitalControlDiagram;
```

EEB.Controllers.Templates.Digital

Package Content

Name	Description
GenericDiagram	

EEB.Controllers.Templates.Digital.GenericDiagram



Information

Extends from Controllers.Templates.BaseClasses.DigitalControlDiagram.

Modelica definition

```
model GenericDiagram
  extends Controllers.Templates.BaseClasses.DigitalControlDiagram;
end GenericDiagram;
```

EEB.Controllers.Blocks

Package Content

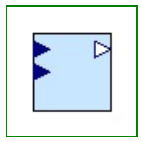
Name	Description
Analogue	
Digital	

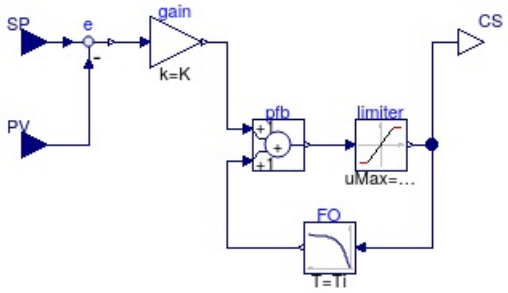
EEB.Controllers.Blocks.Analogue

Package Content

Name	Description
BaseClasses	
AWPI_1dof	
AWPI_1dof_trk	
RLY_dir	RLY, direct action
RLY_inv	RLY, inverse action

EEB.Controllers.Blocks.Analogue.AWPI_1dof





Information

Extends from BaseClasses.A_SISO_base.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

Connectors

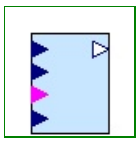
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

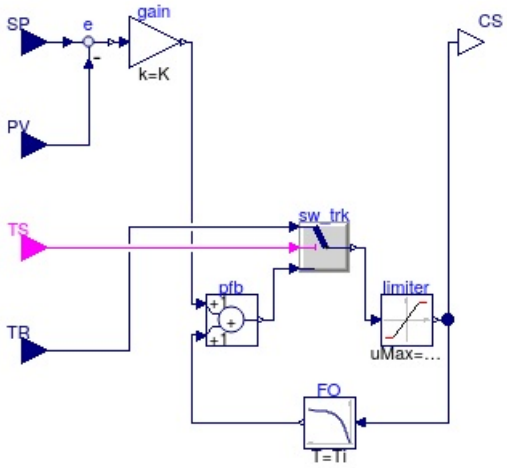
Modelica definition

```
model AWPI_1dof
  extends BaseClasses.A_SISO_base;
  parameter Real K = 1 "Gain";
  parameter Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  Modelica.Blocks.Nonlinear.Limiter limiter(uMax = CSmax, uMin = CSmin);
  Modelica.Blocks.Continuous.FirstOrder FO(k = 1, T = Ti);
  Modelica.Blocks.Math.Add pfb;
  Modelica.Blocks.Math.Gain gain(k = K);
  Modelica.Blocks.Math.Feedback e;
initial equation
  FO.y = CSstart - K * (SP - PV);
equation
  connect(limiter.y, FO.u);
  connect(FO.y, pfb.u2);
  connect(PV, e.u2);
  connect(pfb.y, limiter.u);
  connect(SP, e.u1);
```

```
  connect(limiter.y, CS);
  connect(e.y, gain.u);
  connect(gain.y, pfb.u1);
end AWPI_1dof;
```

EEB.Controllers.Blocks.Analogue.AWPI_1dof_trk





Information

Extends from BaseClasses.A_SISO_trk.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

Connectors

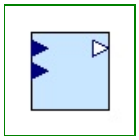
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	

Modelica definition

```
model AWPI_1dof_trk
  extends BaseClasses.A_SISO_trk;
  parameter Real K = 1 "Gain";
  parameter Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  Modelica.Blocks.Nonlinear.Limiter limiter(uMax = CSmax, uMin = CSmin);
  Modelica.Blocks.Continuous.FirstOrder FO(k = 1, T = Ti);
  Modelica.Blocks.Math.Add pfb;
  Modelica.Blocks.Math.Gain gain(k = K);
  Modelica.Blocks.Math.Feedback e;
  Modelica.Blocks.Logical.Switch sw_trk;
initial equation
  FO.y = (if TS then TR else CSstart) - K * (SP - PV);
equation
  connect(limiter.y, FO.u);
  connect(FO.y, pfb.u2);
  connect(PV, e.u2);
  connect(SP, e.u1);
  connect(limiter.y, CS);
  connect(e.y, gain.u);
```

```
  connect(gain.y, pfb.u1);
  connect(sw_trk.u1, TR);
  connect(TS, sw_trk.u2);
  connect(pfb.y, sw_trk.u3);
  connect(sw_trk.y, limiter.u);
end AWPI_1dof_trk;
```

EEB.Controllers.Blocks.Analogue.RLY_dir



RLY, direct action

Information

Extends from BaseClasses.A_SISO_base.

Parameters

Type	Name	Default	Description
Real	CShi	1	HI CS
Real	CSlo	0	LO CS
Real	CShist2	0.1	switch when |SP-PV|>CShist2
Boolean	startHI	false	start with CS=CShi

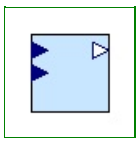
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
model RLY_dir "RLY, direct action"
  extends BaseClasses.A_SISO_base;
  parameter Real CShi = 1 "HI CS";
  parameter Real CSlo = 0 "LO CS";
  parameter Real CShist2 = 0.1 "switch when |SP-PV|>CShist2";
  parameter Boolean startHI = false "start with CS=CShi";
  discrete Real cs;
equation
  CS = cs;
algorithm
  // CS lo and SP under PV -> go hi
  when cs < CShi and PV <= SP - CShist2 then
    cs := CShi;
  end when;
  // CS hi and SP over PV -> go lo
  when cs > CSlo and PV >= SP + CShist2 then
    cs := CSlo;
  end when;
initial algorithm
  cs := if startHI then CShi else CSlo;
end RLY_dir;
```

EEB.Controllers.Blocks.Analogue.RLY_inv



RLY, inverse action

Information

Extends from BaseClasses.A_SISO_base.

Parameters

Type	Name	Default	Description
Real	CShi	1	HI CS
Real	CSlo	0	LO CS
Real	CShist2	0.1	switch when |SP-PV|>CShist2
Boolean	startHI	false	start with CS=CShi

Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
model RLY_inv "RLY, inverse action"
  extends BaseClasses.A_SISO_base;
  parameter Real CShi = 1 "HI CS";
  parameter Real CSlo = 0 "LO CS";
  parameter Real CShist2 = 0.1 "switch when |SP-PV|>CShist2";
  parameter Boolean startHI = false "start with CS=CShi";
  discrete Real cs;
equation
  CS = cs;
algorithm
  // CS lo and SP over PV -> go hi
  when cs < CShi and PV >= SP + CShist2 then
    cs := CShi;
  end when;
  // CS hi and SP under PV -> go lo
  when cs > CSlo and PV <= SP - CShist2 then
    cs := CSlo;
  end when;
initial algorithm
  cs := if startHI then CShi else CSlo;
end RLY_inv;
```

EEB.Controllers.Blocks.Analogue.BaseClasses

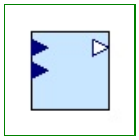
Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
A_SISO_base	
A_SISO_trk	

EEB.Controllers.Blocks.Analogue.BaseClasses.A_SISO_base



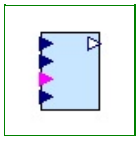
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
partial model A_SISO_base
  Modelica.Blocks.Interfaces.RealInput SP;
  Modelica.Blocks.Interfaces.RealInput PV;
  Modelica.Blocks.Interfaces.RealOutput CS;
end A_SISO_base;
```

EEB.Controllers.Blocks.Analogue.BaseClasses.A_SISO_trk



Information

Extends from A_SISO_base.

Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	

Modelica definition

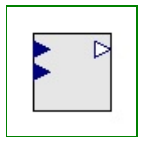
```
partial model A_SISO_trk
  extends A_SISO_base;
  Modelica.Blocks.Interfaces.BooleanInput TS;
  Modelica.Blocks.Interfaces.RealInput TR;
end A_SISO_trk;
```

EEB.Controllers.Blocks.Digital

Package Content

Name	Description
BaseClasses	
AWPI_1dof	
AWPI_1dof_trk	
AWPI_1dof_sat_lock	
AWPI_1dof_trk_sat_lock	

EEB.Controllers.Blocks.Digital.AWPI_1dof



Information

Extends from BaseClasses.D_SISO_base.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

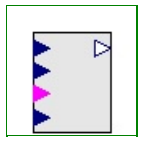
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
model AWPI_1dof
  extends BaseClasses.D_SISO_base;
  parameter Real K = 1 "Gain";
  parameter SI.Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  discrete Real xfo, xfoo, cso;
initial algorithm
  cs := CSstart;
  cso := cs;
  xfoo := cso - K * (SP - PV);
algorithm
  when sample(0, Ts) then
    xfo := (Ti * xfoo + Ts * cso) / (Ti + Ts);
    cs := max(CSmin, min(CSmax, xfo + K * (SP - PV)));
    xfoo := xfo;
    cso := cs;
  end when;
end AWPI_1dof;
```

EEB.Controllers.Blocks.Digital.AWPI_1dof_trk



Information

Extends from BaseClasses.D_SISO_trk.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

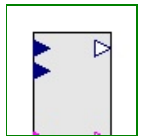
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	

Modelica definition

```
model AWPI_1dof_trk
  extends BaseClasses.D_SISO_trk;
  parameter Real K = 1 "Gain";
  parameter SI.Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  discrete Real xfo, xfoo, cso;
initial algorithm
  cs := CSstart;
  cso := cs;
  xfoo := cso - K * (SP - PV);
algorithm
  when sample(0, Ts) then
    xfo := (Ti * xfoo + Ts * cso) / (Ti + Ts);
    if TS then
      cs := TR;
    else
      cs := max(CSmin, min(CSmax, xfo + K * (SP - PV)));
    end if;
    xfoo := xfo;
    cso := cs;
  end when;
end AWPI_1dof_trk;
```

EEB.Controllers.Blocks.Digital.AWPI_1dof_sat_lock



Information

Extends from BaseClasses.D_SISO_sat_lock.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS

Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

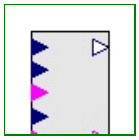
Connectors

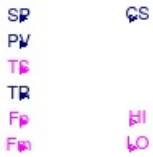
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	Fp	
input BooleanInput	Fm	
output BooleanOutput	HI	
output BooleanOutput	LO	

Modelica definition

```
model AWPI_1dof_sat_lock
  extends BaseClasses.D_SISO_sat_lock;
  parameter Real K = 1 "Gain";
  parameter SI.Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  discrete Real xfo, xfoo, cso;
initial algorithm
  cs := CSstart;
  cso := cs;
  xfoo := cso - K * (SP - PV);
algorithm
  when sample(0, Ts) then
    xfo := (Ti * xfoo + Ts * cso) / (Ti + Ts);
    cs := max(CSmin, min(CSmax, xfo + K * (SP - PV)));
    if cs > cso and Fp or cs < cso and Fm then
      cs := cso;
    end if;
    xfoo := xfo;
    cso := cs;
    HI := cs >= CSmax;
    LO := cs <= CSmin;
  end when;
end AWPI_1dof_sat_lock;
```

EEB.Controllers.Blocks.Digital.AWPI_1dof_trk_sat_lock





Information

Extends from BaseClasses.D_SISO_trk_sat_lock.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS

Connectors

Type	Name	Description
input RealInput	SP	

input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	
input BooleanInput	Fp	
input BooleanInput	Fm	
output BooleanOutput	HI	
output BooleanOutput	LO	

Modelica definition

```
model AWPI_1dof_trk_sat_lock
  extends BaseClasses.D_SISO_trk_sat_lock;
  parameter Real K = 1 "Gain";
  parameter SI.Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  discrete Real xfo, xfoo, cso;
initial algorithm
  cs := CSstart;
  cso := cs;
  xfoo := cso - K * (SP - PV);
algorithm
  when sample(0, Ts) then
    xfo := (Ti * xfoo + Ts * cso) / (Ti + Ts);
    if TS then
      cs := TR;
    else
      cs := max(CSmin, min(CSmax, xfo + K * (SP - PV)));
    end if;
    if cs > cso and Fp or cs < cso and Fm then
      cs := cso;
    end if;
    xfoo := xfo;
    cso := cs;
    HI := cs >= CSmax;
    LO := cs <= CSmin;
  end when;
end AWPI_1dof_trk_sat_lock;
```

EEB.Controllers.Blocks.Digital.BaseClasses

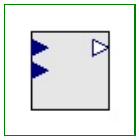
Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
D_SISO_base	
D_SISO_trk	
D_SISO_sat_lock	
D_SISO_trk_sat_lock	

EEB.Controllers.Blocks.Digital.BaseClasses.D_SISO_base



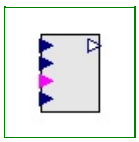
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
partial model D_SISO_base
  outer Time Ts;
  discrete Real cs;
  Modelica.Blocks.Interfaces.RealInput SP;
  Modelica.Blocks.Interfaces.RealInput PV;
  Modelica.Blocks.Interfaces.RealOutput CS;
equation
  CS = cs;
end D_SISO_base;
```

EEB.Controllers.Blocks.Digital.BaseClasses.D_SISO_trk



Information

Extends from D_SISO_base.

Connectors

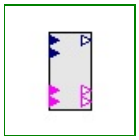
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	

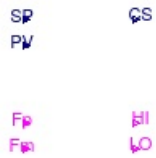
Modelica definition

```
partial model D_SISO_trk
  extends D_SISO_base;
  Modelica.Blocks.Interfaces.BooleanInput TS;
  Modelica.Blocks.Interfaces.RealInput TR;
```

```
end D_SISO_trk;
```

EEB.Controllers.Blocks.Digital.BaseClasses.D_SISO_sat_lock





Information

Extends from D_SISO_base.

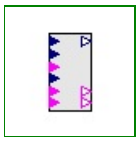
Connectors

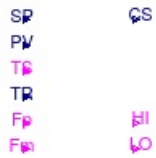
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	Fp	
input BooleanInput	Fm	
output BooleanOutput	HI	
output BooleanOutput	LO	

Modelica definition

```
partial model D_SISO_sat_lock
  extends D_SISO_base;
  Modelica.Blocks.Interfaces.BooleanInput Fp;
  Modelica.Blocks.Interfaces.BooleanInput Fm;
  Modelica.Blocks.Interfaces.BooleanOutput HI;
  Modelica.Blocks.Interfaces.BooleanOutput LO;
end D_SISO_sat_lock;
```

EEB.Controllers.Blocks.Digital.BaseClasses.D_SISO_trk_sat_lock





Information

Extends from D_SISO_base.

Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	
input BooleanInput	TS	
input RealInput	TR	
input BooleanInput	Fp	
input BooleanInput	Fm	
output BooleanOutput	HI	
output BooleanOutput	LO	

Modelica definition

```
partial model D_SISO_trk_sat_lock
```

```
  extends D_SISO_base;
  Modelica.Blocks.Interfaces.BooleanInput TS;
  Modelica.Blocks.Interfaces.RealInput TR;
  Modelica.Blocks.Interfaces.BooleanInput Fp;
  Modelica.Blocks.Interfaces.BooleanInput Fm;
  Modelica.Blocks.Interfaces.BooleanOutput HI;
  Modelica.Blocks.Interfaces.BooleanOutput LO;
end D_SISO_trk_sat_lock;
```

EEB.Controllers.AggregateBlocks

Package Content

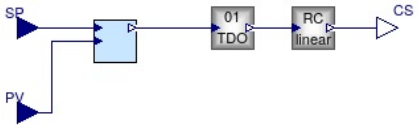
Name	Description
Analogue	

EEB.Controllers.AggregateBlocks.Analogue

Package Content

Name	Description
PI_TDO	
PI_SplitRange	
TwinPI_RangeSP	

EEB.Controllers.AggregateBlocks.Analogue.PI_TDO



Information

Extends from Controllers.Blocks.Analogue.BaseClasses.A_SISO_base.

Parameters

Type	Name	Default	Description
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSmax	1	Max CS
Real	CSmin	0	Min CS
Real	CSstart	0	Initial CS
Time	Ttdo	1	TDO timebase [s]

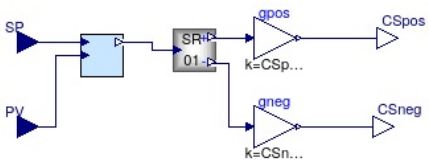
Connectors

Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CS	

Modelica definition

```
model PI_TDO
  extends Controllers.Blocks.Analogue.BaseClasses.A_SISO_base;
  Blocks.Analogue.AWPI_1dof PI(K = K, Ti = Ti, CSmax = 1, CSmin = 0, CSstart = (CSstart - CSmin) / (CSmax - CSmin));
  Actuation.TDO TDO(Ttdo = Ttdo);
  parameter Real K = 1 "Gain";
  parameter Time Ti = 1 "Integral time";
  parameter Real CSmax = 1 "Max CS";
  parameter Real CSmin = 0 "Min CS";
  parameter Real CSstart = 0 "Initial CS";
  parameter Time Ttdo = 1 "TDO timebase";
  Actuation.RangeConv_Linear RCo(CSimin = 0, CSimax = 1, CSomin = CSmin, CSomax = CSmax);
equation
  connect(PI.CS, TDO.cmd01);
  connect(PV, PI.PV);
  connect(SP, PI.SP);
  connect(TDO.TDO, RCo.CSi);
  connect(RCo.CSo, CS);
end PI_TDO;
```

EEB.Controllers.AggregateBlocks.Analogue.PI_SplitRange



Parameters

Type	Name	Default	Description
Real	DeadZoneNorm	0.1	Normalised DZ (0-1)
Real	K	1	Gain
Time	Ti	1	Integral time [s]
Real	CSposmax	1	Max CS for pos action
Real	CSnegmax	0	Min CS for neg action
Real	CSstart	0	Initial CS

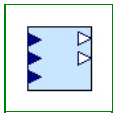
Connectors

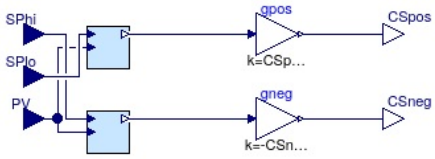
Type	Name	Description
input RealInput	SP	
input RealInput	PV	
output RealOutput	CSpos	
output RealOutput	CSneg	

Modelica definition

```
model PI_SplitRange
  Modelica.Blocks.Interfaces.RealInput SP;
  Modelica.Blocks.Interfaces.RealInput PV;
  Modelica.Blocks.Interfaces.RealOutput CSpos;
  Modelica.Blocks.Interfaces.RealOutput CSneg;
  Blocks.Analogue.AWPI_1dof PI(K = K, Ti = Ti, CSmax = 1, CSstart = (CSstart + CSnegmax) / (CSposmax + CSnegmax), CSmin = -1);
  Actuation.SplitRange_01 splitRange_01_1(DeadZone = DeadZoneNorm);
  parameter Real DeadZoneNorm = 0.1 "Normalised DZ (0-1)";
  Modelica.Blocks.Math.Gain gpos(k = CSposmax);
  Modelica.Blocks.Math.Gain gneg(k = CSnegmax);
  parameter Real K = 1 "Gain";
  parameter Time Ti = 1 "Integral time";
  parameter Real CSposmax = 1 "Max CS for pos action";
  parameter Real CSnegmax = 0 "Min CS for neg action";
  parameter Real CSstart = 0 "Initial CS";
equation
  connect(SP, PI.SP);
  connect(PV, PI.PV);
  connect(PI.CS, splitRange_01_1.CSi01);
  connect(splitRange_01_1.CSo01_pos, gpos.u);
  connect(gpos.y, CSpos);
  connect(splitRange_01_1.CSo01_neg, gneg.u);
  connect(gneg.y, CSneg);
end PI_SplitRange;
```

EEB.Controllers.AggregateBlocks.Analogue.TwinPI_RangeSP





Parameters

Type	Name	Default	Description
Real	DeadZoneNorm	0.1	Normalised DZ (0-1)
Real	Khi	1	Gain, hi
Time	Tihi	1	Integral time, hi [s]
Real	Klo	1	Gain, lo
Time	Tilo	1	Integral time, lo [s]
Real	CSposmax	1	Max CS for pos action
Real	CSnegmax	0	Min CS for neg action
Real	CSstart	0	Initial CS

Connectors

Type	Name	Description
input RealInput	SPhi	
input RealInput	PV	
output RealOutput	CSpos	
output RealOutput	CSneg	
input RealInput	SPlo	

Modelica definition

```
model TwinPI_RangeSP
  Modelica.Blocks.Interfaces.RealInput SPhi;
  Modelica.Blocks.Interfaces.RealInput PV;
  Modelica.Blocks.Interfaces.RealOutput CSpos;
  Modelica.Blocks.Interfaces.RealOutput CSneg;
  Blocks.Analogue.AWPI_1dof PIhi(CSmin = 0, CSstart = max(0, (CSstart + CSnegmax) / (CSposmax + CSnegmax)), K = Khi, Ti = Tihi, CSmax = 1);
  parameter Real DeadZoneNorm = 0.1 "Normalised DZ (0-1)";
  Modelica.Blocks.Math.Gain gpos(k = CSposmax);
  Modelica.Blocks.Math.Gain gneg(k = -CSnegmax);
  Modelica.Blocks.Interfaces.RealInput SPlo;
  Blocks.Analogue.AWPI_1dof PIlo(Ti = Tilo, CSmax = 0, CSmin = -1, CSstart = min(0, (CSstart + CSnegmax) / (CSposmax + CSnegmax)), K = Klo);
  parameter Real Khi = 1 "Gain, hi";
  parameter Time Tihi = 1 "Integral time, hi";
  parameter Real Klo = 1 "Gain, lo";
  parameter Time Tilo = 1 "Integral time, lo";
  parameter Real CSposmax = 1 "Max CS for pos action";
  parameter Real CSnegmax = 0 "Min CS for neg action";
```

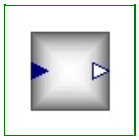
```
  parameter Real CSstart = 0 "Initial CS";
equation
  connect(gpos.y, CSpos);
  connect(gneg.y, CSneg);
  connect(PV, PIhi.PV);
  connect(PV, PIlo.PV);
  connect(PIhi.CS, gpos.u);
  connect(PIlo.CS, gneg.u);
  connect(SPhi, PIlo.SP);
  connect(SPlo, PIhi.SP);
end TwinPI_RangeSP;
```

EEB.Controllers.Actuation

Package Content

Name	Description
PWM	
TDO	
Distributor_uniform	
DaisyChain_uniform	
SplitRange_01	
RangeConv_Linear	
DaisyChain_InRanges_Out01	

EEB.Controllers.Actuation.PWM



Parameters

Type	Name	Default	Description
Frequency	fPWM	1000	PWM f [Hz]

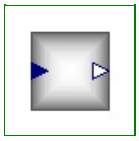
Connectors

Type	Name	Description
input RealInput	cmd01	
output RealOutput	PWM	

Modelica definition

```
model PWM
  Modelica.Blocks.Interfaces.RealInput cmd01;
  Modelica.Blocks.Interfaces.RealOutput PWM;
  parameter Frequency fPWM = 1000 "PWM f";
  discrete Real pwm;
  discrete Real timeNextEv;
equation
  PWM = pwm;
initial algorithm
  timeNextEv := 0;
algorithm
  when sample(0, 1 / fPWM) then
    if cmd01 <= 0 then
      pwm := 0;
      timeNextEv := -1;
    elseif cmd01 >= 1 then
      pwm := 1;
      timeNextEv := -1;
    else
      pwm := 1;
      timeNextEv := time + cmd01 / fPWM;
    end if;
  end when;
  when time >= timeNextEv and timeNextEv >= 0 then
    if cmd01 > 0 and cmd01 < 1 then
      pwm := 0;
    end if;
  end when;
end PWM;
```

EEB.Controllers.Actuation.TDO



Parameters

Type	Name	Default	Description
Time	Ttdo	1	TDO timebase [s]

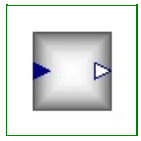
Connectors

Type	Name	Description
input RealInput	cmd01	
output RealOutput	TDO	

Modelica definition

```
model TDO
  Modelica.Blocks.Interfaces.RealInput cmd01;
  Modelica.Blocks.Interfaces.RealOutput TDO;
  parameter Time Ttdo = 1 "TDO timebase";
  discrete Real tdo;
  discrete Real timeNextEv;
equation
  TDO = tdo;
initial algorithm
  timeNextEv := 0;
  tdo := 0;
algorithm
  when sample(0, Ttdo) then
    if cmd01 <= 0 then
      tdo := 0;
      timeNextEv := -1;
    elseif cmd01 >= 1 then
      tdo := 1;
      timeNextEv := -1;
    else
      tdo := 1;
      timeNextEv := time + cmd01 * Ttdo;
    end if;
  end when;
  when time >= timeNextEv and timeNextEv >= 0 then
    if cmd01 > 0 and cmd01 < 1 then
      tdo := 0;
    end if;
  end when;
end TDO;
```

EEB.Controllers.Actuation.Distributor_uniform



Parameters

Type	Name	Default	Description
Integer	ns	2	

Connectors

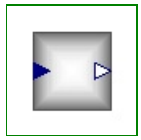
Type	Name	Description
input RealInput	CSi01	
output RealOutput	CSo01[ns]	

Modelica definition

```
model Distributor_uniform
  parameter Integer ns = 2;
  Modelica.Blocks.Interfaces.RealInput CSi01;
  Modelica.Blocks.Interfaces.RealOutput CSo01[ns];
  Real csi, cso[ns], cso01tot;
equation
  csi = CSi01;
  cso = CSo01;
  for i in 1:ns loop
    cso[i] = max(0, min(1, 1 - abs(csi * ns - i)));
  end for;
  cso01tot = sum(cso);
```

```
end Distributor_uniform;
```

EEB.Controllers.Actuation.DaisyChain_uniform



Parameters

Type	Name	Default	Description
Integer	ns	2	

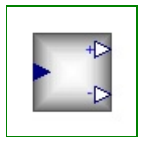
Connectors

Type	Name	Description
input RealInput	CSi01	
output RealOutput	CSo01[ns]	

Modelica definition

```
model DaisyChain_uniform
  parameter Integer ns = 2;
  Modelica.Blocks.Interfaces.RealInput CSi01;
  Modelica.Blocks.Interfaces.RealOutput CSo01[ns];
  Real csi, cso[ns], cso01tot;
equation
  csi = CSi01;
  cso = CSo01;
  for i in 1:ns loop
    cso[i] = max(0, min(1, csi - (i - 1) / ns));
  end for;
  cso01tot = sum(cso);
end DaisyChain_uniform;
```

EEB.Controllers.Actuation.SplitRange_01



Parameters

Type	Name	Default	Description
Real	DeadZone	0.1	

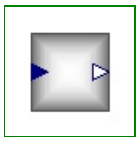
Connectors

Type	Name	Description
input RealInput	CSi01	
output RealOutput	CSo01_pos	
output RealOutput	CSo01_neg	

Modelica definition

```
model SplitRange_01
  parameter Real DeadZone(final min = 1e-6, final max = 1 - 1e-6) = 0.1;
  Modelica.Blocks.Interfaces.RealInput CSi01;
  Modelica.Blocks.Interfaces.RealOutput CSo01_pos;
  Modelica.Blocks.Interfaces.RealOutput CSo01_neg;
  // POS solve([a*dz+b=0,a*1+b=1],[a,b]);
  // NEG solve([-a*dz+b=0,-a*1+b=1],[a,b]);
protected
  parameter Real a = 1 / (DeadZone - 1);
  parameter Real b = DeadZone / (DeadZone - 1);
equation
  CSo01_pos = max(0, min(1, (-a * CSi01) + b));
  CSo01_neg = max(0, min(1, a * CSi01 + b));
end SplitRange_01;
```

EEB.Controllers.Actuation.RangeConv_Linear



CSi CSo

Parameters

Type	Name	Default	Description
Real	CSimin	0.1	
Real	CSimax	0.9	
Real	CSomin	0	
Real	CSomax	1	

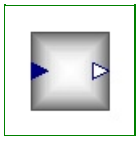
Connectors

Type	Name	Description
input RealInput	CSi	
output RealOutput	CSo	

Modelica definition

```
model RangeConv_Linear
  parameter Real CSimin = 0.1;
  parameter Real CSimax = 0.9;
  parameter Real CSomin = 0;
  parameter Real CSomax = 1;
  Modelica.Blocks.Interfaces.RealInput CSi;
  Modelica.Blocks.Interfaces.RealOutput CSo;
equation
  CSo = Functions.range_conv_linear(CSi, CSimin, CSimax, CSomin, CSomax);
end RangeConv_Linear;
```

EEB.Controllers.Actuation.DaisyChain_InRanges_Out01



CSi CSo[]

Parameters

Type	Name	Default	Description
Real	CSiRB[:]	{0.1,0.3,0.5,1,3}	Input range boundaries

Connectors

Type	Name	Description
input RealInput	CSi	
output RealOutput	CSo[n]	

Modelica definition

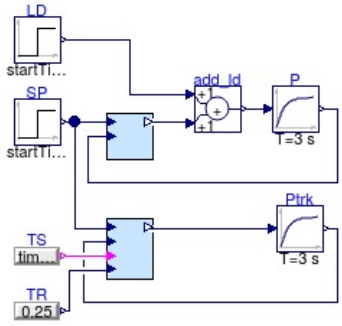
```
model DaisyChain_InRanges_Out01
  parameter Real CSiRB[:] = {0.1, 0.3, 0.5, 1, 3} "Input range boundaries";
  Modelica.Blocks.Interfaces.RealInput CSi;
  Modelica.Blocks.Interfaces.RealOutput CSo[n];
protected
  parameter Integer n = size(CSiRB, 1) - 1;
equation
  for i in 1:n loop
    CSo[i] = EEB.Functions.range_conv_linear(CSi, CSiRB[i], CSiRB[i + 1], 0, 1);
  end for;
end DaisyChain_InRanges_Out01;
```

EEB.Controllers.Test

Package Content

Name	Description
Test_AWPI_analogue	
Test_AWPI_digital	
Test_Act	
Test_AWPI_TDO_analogue	
Test_AWPI_SR_analogue	
Test_TwinPI_RangeSP	

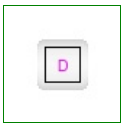
EEB.Controllers.Test.Test_AWPI_analogue

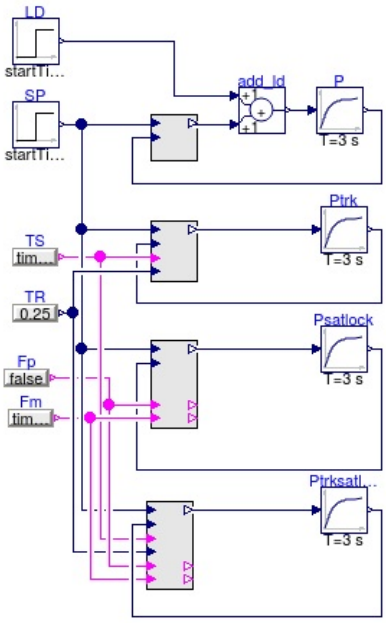


Modelica definition

```
model Test_AWPI_analogue
  Blocks.Analogue.AWPI_1dof PI(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Continuous.FirstOrder P(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Math.Add add_ld;
  Modelica.Blocks.Sources.Step SP(startTime = 1);
  Modelica.Blocks.Sources.Step LD(height = 0.5, startTime = 10);
  Blocks.Analogue.AWPI_1dof_trk PItrk(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Continuous.FirstOrder Ptrk(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Sources.BooleanExpression TS(y = time > 3 and time < 5);
  Modelica.Blocks.Sources.RealExpression TR(y = 0.25);
equation
  connect(PI.CS, add_ld.u2);
  connect(add_ld.y, P.u);
  connect(P.y, PI.PV);
  connect(SP.y, PI.SP);
  connect(LD.y, add_ld.u1);
  connect(PItrk.SP, PI.SP);
  connect(PItrk.CS, Ptrk.u);
  connect(Ptrk.y, PItrk.PV);
  connect(TS.y, PItrk.TS);
  connect(TR.y, PItrk.TR);
end Test_AWPI_analogue;
```

EEB.Controllers.Test.Test_AWPI_digital





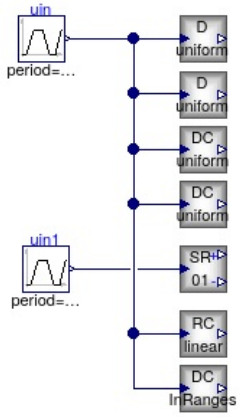
Information

Extends from Templates.BaseClasses.DigitalControlDiagram.

Modelica definition

```
model Test_AWPI_digital
  extends Templates.BaseClasses.DigitalControlDiagram;
  Blocks.Digital.AWPI_1dof PI(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Continuous.FirstOrder P(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Math.Add add_ld;
  Modelica.Blocks.Sources.Step SP(startTime = 1);
  Modelica.Blocks.Sources.Step LD(height = 0.5, startTime = 10);
  Blocks.Digital.AWPI_1dof_trk PItrk(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Continuous.FirstOrder Ptrk(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Sources.BooleanExpression TS(y = time > 3 and time < 5);
  Modelica.Blocks.Sources.RealExpression TR(y = 0.25);
  Modelica.Blocks.Continuous.FirstOrder Psatlock(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Blocks.Digital.AWPI_1dof_sat_lock PIsatlock(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Sources.BooleanExpression Fp(y = false);
  Modelica.Blocks.Sources.BooleanExpression Fm(y = time > 2.5 and time < 3.5);
  Blocks.Digital.AWPI_1dof_trk_sat_lock PItrksatlock(CSmin = 0, CSstart = 0.2, Ti = 2, K = 3, CSmax = 2);
  Modelica.Blocks.Continuous.FirstOrder Ptrksatlock(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
equation
  connect(PI.CS, add_ld.u2);
  connect(add_ld.y, P.u);
  connect(P.y, PI.PV);
  connect(SP.y, PI.SP);
  connect(LD.y, add_ld.u1);
  connect(PItrk.SP, PI.SP);
  connect(PItrk.CS, Ptrk.u);
  connect(Ptrk.y, PItrk.PV);
  connect(TS.y, PItrk.TS);
  connect(TR.y, PItrk.TR);
  connect(PIsatlock.CS, Psatlock.u);
  connect(Psatlock.y, PIsatlock.PV);
  connect(PIsatlock.SP, PI.SP);
  connect(Fp.y, PIsatlock.Fp);
  connect(Fm.y, PIsatlock.Fm);
  connect(PItrksatlock.SP, PI.SP);
  connect(PItrksatlock.TS, PItrk.TS);
  connect(PItrksatlock.TR, PItrk.TR);
  connect(PItrksatlock.Fp, PIsatlock.Fp);
  connect(PItrksatlock.Fm, PIsatlock.Fm);
  connect(PItrksatlock.CS, Ptrksatlock.u);
  connect(Ptrksatlock.y, PItrksatlock.PV);
end Test_AWPI_digital;
```

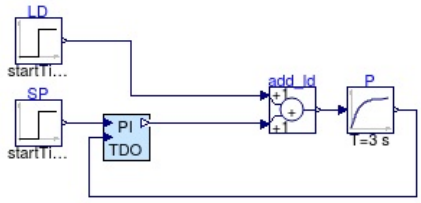
EEB.Controllers.Test.Test_Act



Modelica definition

```
model Test_Act
  Actuation.Distributor_uniform D2(ns = 2);
  Actuation.Distributor_uniform D5(ns = 5);
  Modelica.Blocks.Sources.Trapezoid uin(amplitude = 1, rising = 1, width = 1, falling = 1, period = 4);
  Actuation.DaisyChain_uniform DC2(ns = 2);
  Actuation.DaisyChain_uniform DC5(ns = 5);
  Modelica.Blocks.Sources.Trapezoid uin1(rising = 1, width = 1, falling = 1, period = 4, amplitude = 3, offset = -1.5);
  Actuation.SplitRange_01 SR;
  Actuation.RangeConv_Linear RC(CSimin = 0.4, CSimax = 0.6);
  Actuation.DaisyChain_InRanges_Out01 DC_IR(CSiRB = {0.1, 0.3, 0.4, 0.75, 0.9});
equation
  connect(uin.y, D2.CSi01);
  connect(uin.y, D5.CSi01);
  connect(DC2.CSi01, D5.CSi01);
  connect(DC5.CSi01, D5.CSi01);
  connect(uin1.y, SR.CSi01);
  connect(RC.CSi, D5.CSi01);
  connect(DC_IR.CSi, D5.CSi01);
end Test_Act;
```

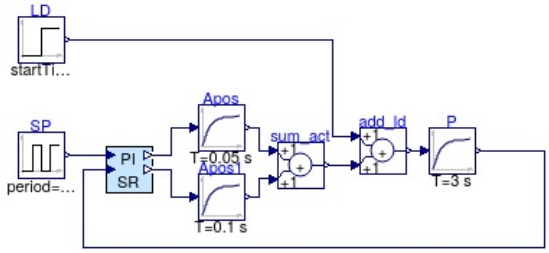
EEB.Controllers.Test.Test_AWPI_TDO_analogue



Modelica definition

```
model Test_AWPI_TDO_analogue
  AggregateBlocks.Analogue.PI_TDO PI(CSmin = 0, Ti = 2, K = 3, CSmax = 2, CSstart = 0.2, Ttdo = 0.25);
  Modelica.Blocks.Continuous.FirstOrder P(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Math.Add add_ld;
  Modelica.Blocks.Sources.Step SP(startTime = 1);
  Modelica.Blocks.Sources.Step LD(height = 0.5, startTime = 10);
equation
  connect(add_ld.y, P.u);
  connect(P.y, PI.PV);
  connect(SP.y, PI.SP);
  connect(LD.y, add_ld.u1);
  connect(PI.CS, add_ld.u2);
end Test_AWPI_TDO_analogue;
```

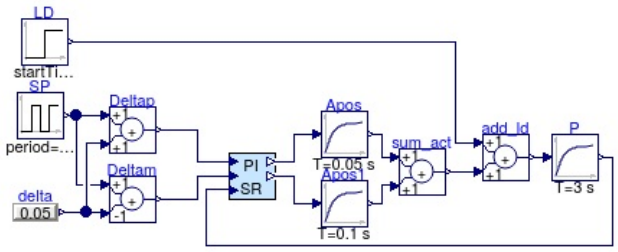
EEB.Controllers.Test.Test_AWPI_SR_analogue



Modelica definition

```
model Test_AWPI_SR_analogue
  AggregateBlocks.Analogue.PI_SplitRange PI(Ti = 2, K = 3, CSstart = 0.2, CSposmax = 100, CSnegmax = 20);
  Modelica.Blocks.Continuous.FirstOrder P(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Math.Add add_ld;
  Modelica.Blocks.Sources.Pulse SP(startTime = 1, amplitude = 1, period = 10, offset = -0.5);
  Modelica.Blocks.Sources.Step LD(height = 0.5, startTime = 10);
  Modelica.Blocks.Continuous.FirstOrder Apos(k = 1, initType = Modelica.Blocks.Types.Init.InitialOutput, T = 0.05);
  Modelica.Blocks.Math.Add sum_act;
  Modelica.Blocks.Continuous.FirstOrder Apos1(initType = Modelica.Blocks.Types.Init.InitialOutput, k = -1, T = 0.1);
equation
  connect(add_ld.y, P.u);
  connect(P.y, PI.PV);
  connect(SP.y, PI.SP);
  connect(LD.y, add_ld.u1);
  connect(PI.CSpos, Apos.u);
  connect(PI.CSneg, Apos1.u);
  connect(Apos.y, sum_act.u1);
  connect(Apos1.y, sum_act.u2);
  connect(sum_act.y, add_ld.u2);
end Test_AWPI_SR_analogue;
```

EEB.Controllers.Test.Test_TwinPI_RangeSP



Modelica definition

```
model Test_TwinPI_RangeSP
  AggregateBlocks.Analogue.TwinPI_RangeSP PI(CSstart = 0.2, CSposmax = 100, CSnegmax = 20, Khi = 3, Tihi = 2, Klo = 3, Tilo = 2);
  Modelica.Blocks.Continuous.FirstOrder P(k = 1, T = 3, initType = Modelica.Blocks.Types.Init.InitialOutput);
  Modelica.Blocks.Math.Add add_ld;
  Modelica.Blocks.Sources.Pulse SP(startTime = 1, amplitude = 1, period = 10, offset = -0.5);
  Modelica.Blocks.Sources.Step LD(height = 0.5, startTime = 10);
  Modelica.Blocks.Continuous.FirstOrder Apos(k = 1, initType = Modelica.Blocks.Types.Init.InitialOutput, T = 0.05);
  Modelica.Blocks.Math.Add sum_act;
  Modelica.Blocks.Continuous.FirstOrder Apos1(initType = Modelica.Blocks.Types.Init.InitialOutput, k = -1, T = 0.1);
  Modelica.Blocks.Math.Add Deltap;
  Modelica.Blocks.Math.Add Deltam(k2 = -1);
  Modelica.Blocks.Sources.RealExpression delta(y = 0.05);
equation
  connect(add_ld.y, P.u);
  connect(P.y, PI.PV);
  connect(LD.y, add_ld.u1);
  connect(PI.CSpos, Apos.u);
  connect(PI.CSneg, Apos1.u);
  connect(Apos.y, sum_act.u1);
  connect(Apos1.y, sum_act.u2);
  connect(sum_act.y, add_ld.u2);
  connect(SP.y, Deltap.u1);
  connect(SP.y, Deltam.u1);
  connect(delta.y, Deltam.u2);
  connect(delta.y, Deltap.u2);
  connect(Deltap.y, PI.SPhi);
  connect(Deltam.y, PI.SPlo);
end Test_TwinPI_RangeSP;
```

EEB.CaseStudies

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
DEIB	
ABC	

EEB.CaseStudies.ABC

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
TableTest	
TestAirSourcepTphi	
ReadFromTM2file	
OneRoom	
OneRoom_ExtData	
OneRoom_ExtData_noAHU	
OneRoom_ExtData_forTests	

EEB.CaseStudies.ABC.TableTest

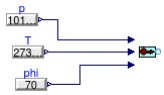
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model TableTest
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.CombiTimeTable table(tableOnFile = true, fileName = "modelica://EEB/Resources/testFile.txt", tableName = "pippo", smoothness = Modelica.Blocks.Types.Smoothness
end TableTest;
```

EEB.CaseStudies.ABC.TestAirSourcepTphi



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model TestAirSourcepTphi
  extends Icons.CaseStudyModel;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed air;
  Modelica.Blocks.Sources.RealExpression p(y = 101325);
  Modelica.Blocks.Sources.RealExpression T(y = 273.15 + 20 + 10 * sin(time / 100));
  Modelica.Blocks.Sources.RealExpression phi(y = 70);
equation
  connect(phi.y, air.iphi);
  connect(T.y, air.iT);
  connect(p.y, air.iP);
end TestAirSourcepTphi;
```

EEB.CaseStudies.ABC.ReadFromTM2file

Information

Extends from Icons.CaseStudyModel.

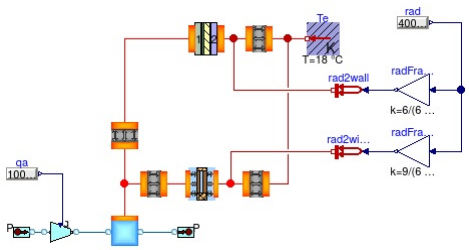
Parameters

Type	Name	Default	Description
String	fileName	"Resources/DOEData/"	File where the matrix is stored

Modelica definition

```
model ReadFromTM2file
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.CombiTimeTable table(
    tableOnFile = true,
    fileName = "/home/leva/Dropbox/Buildings_ABC/Modelica/EEB/Resources/testFile.txt",
    tableName = "pippo",
    smoothness = Modelica.Blocks.Types.Smoothness.ContinuousDerivative,
    columns = 2:3);
  parameter String fileName = "Resources/DOEData/" "File where the matrix is stored";
end ReadFromTM2file;
```

EEB.CaseStudies.ABC.OneRoom



Information

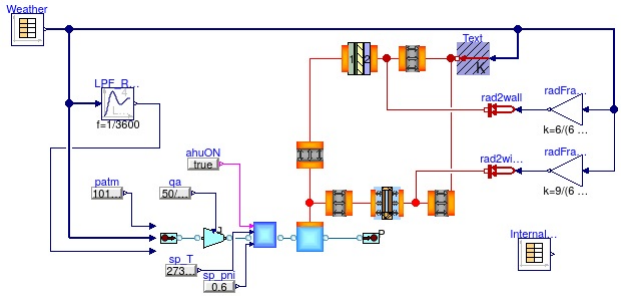
Extends from Icons.CaseStudyModel.

Modelica definition

```
model OneRoom
  extends Icons.CaseStudyModel;
  EEB.Components.BaseComponents.Air.Volumes.AirVolume Room(V = 50, Xstart = 0.0025);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSrc(X0 = 0.0025);
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression qa(y = 100 / 3600);
```

```
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2wall(S = 6, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS wall2ext(S = 6, gamma = 12);
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_NonHomogeneous wall(A = 6, Tstart = 293.15);
  Modelica.Thermal.HeatTransfer.Sources.FixedTemperature Te(T = 291.15);
  EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_SingleGlass
                                                                    window(H = 3, L = 3, Tstart = 293.15, material = EEB.Media.Materials.Glasses.Glass(), s = 0.005
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS window2ext(S = 9, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2window(S = 9, gamma = 8);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2window;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2wall;
  Modelica.Blocks.Math.Gain radFracWall(k = 6 / (6 + 9));
  Modelica.Blocks.Math.Gain radFracWindow(k = 9 / (6 + 9));
  Modelica.Blocks.Sources.RealExpression rad(y = 400 + 300 * sin(0.00001 * time));
equation
  connect(Te.port, window2ext.ss2);
  connect(window.airSide2, window2ext.ss1);
  connect(radFracWall.u, radFracWindow.u);
  connect(rad.y, radFracWall.u);
  connect(rad2window.Q_flow, radFracWindow.y);
  connect(rad2window.port, window.airSide2);
  connect(rad2wall.Q_flow, radFracWall.y);
  connect(rad2wall.port, wall.side2);
  connect(wall2ext.ss2, Te.port);
  connect(wall.side2, wall2ext.ss1);
  connect(room2wall.ss2, wall.side1);
  connect(room2window.ss2, window.airSide1);
  connect(room2window.ss1, Room.heatPort);
  connect(qa.y, fan.iq);
  connect(Room.air_flange2, airSink.air_flange);
  connect(airSrc.air_flange, fan.air_flange1);
  connect(fan.air_flange2, Room.air_flange1);
  connect(Room.heatPort, room2wall.ss1);
end OneRoom;
```

EEB.CaseStudies.ABC.OneRoom_ExtData



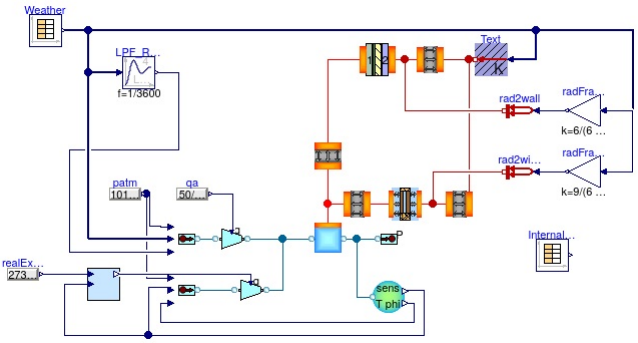
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model OneRoom_ExtData
  extends Icons.CaseStudyModel;
  Real phi100 = Room.air.phi * 100;
  EEB.Components.BaseComponents.Air.Volumes.AirVolume Room(V = 50, Xstart = 0.005);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression qa(y = 50 / 3600);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2wall(S = 6, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS wall2ext(S = 6, gamma = 12);
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_NonHomogeneous wall(A = 6, Tstart = 293.15);
  EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_SingleGlass
                                                                    window(H = 3, L = 3, Tstart = 293.15, material = EEB.Media.Materials.Glasses.Glass(), s = 0.005
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS window2ext(S = 9, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2window(S = 9, gamma = 8);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2window;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2wall;
  Modelica.Blocks.Math.Gain radFracWall(k = 6 / (6 + 9));
  Modelica.Blocks.Math.Gain radFracWindow(k = 9 / (6 + 9));
  Modelica.Blocks.Sources.CombiTimeTable Weather(columns = 2:4, fileName = "/home/artoo/Dropbox/Buildings_ABC/Modelica/EEB/Resources/TestDEIB_123_July2015_7days_weather.txt", tableName
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Text;
  Modelica.Blocks.Sources.CombiTimeTable InternalData(columns = 2:5, fileName = "/home/artoo/Dropbox/Buildings_ABC/Modelica/EEB/Resources/TestDEIB_123_July2015_7days_InternalData.txt",
  Modelica.Blocks.Sources.RealExpression patm(y = 101325);
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed airSrc;
  Modelica.Blocks.Continuous.LowpassButterworth LPF_RHext(f = 1 / 3600, initType = Modelica.Blocks.Types.Init.InitialOutput, n = 4, y_start = 50);
  EEB.Components.BaseComponents.HVAC.AirHandling.ControlledHandler_Tphi_AlgObal
                                                                    AHU(TC = 60);
  Modelica.Blocks.Sources.RealExpression sp_T(y = 273.15 + 24);
  Modelica.Blocks.Sources.RealExpression sp_phi(y = 0.6);
  Modelica.Blocks.Sources.BooleanExpression ahuON(y = true);
equation
  connect(fan.air_flange2, AHU.air_flange1);
  connect(AHU.air_flange2, Room.air_flange1);
  connect(AHU.Tsp, sp_T.y);
  connect(sp_phi.y, AHU.phisp);
  connect(ahuON.y, AHU.ON);
  connect(LPF_RHext.y, airSrc.iphi);
  connect(Weather.y[1], airSrc.iT);
  connect(patm.y, airSrc.iP);
  connect(airSrc.air_flange, fan.air_flange1);
  connect(Room.air_flange2, airSink.air_flange);
  connect(qa.y, fan.iq);
  connect(Room.heatPort, room2wall.ss1);
  connect(room2window.ss1, Room.heatPort);
  connect(room2window.ss2, window.airSide1);
  connect(window.airSide2, window2ext.ss1);
  connect(Text.port, window2ext.ss2);
  connect(rad2window.port, window.airSide2);
  connect(room2wall.ss2, wall.side1);
  connect(Weather.y[1], Text.T);
  connect(Text.port, wall2ext.ss2);
  connect(Weather.y[3], radFracWall.u);
  connect(Weather.y[2], LPF_RHext.u);
  connect(rad2window.Q_flow, radFracWindow.y);
  connect(radFracWall.u, radFracWindow.u);
  connect(rad2wall.Q_flow, radFracWall.y);
  connect(rad2wall.port, wall.side2);
  connect(wall.side2, wall2ext.ss1);
end OneRoom_ExtData;
```

EEB.CaseStudies.ABC.OneRoom_ExtData_noAHU



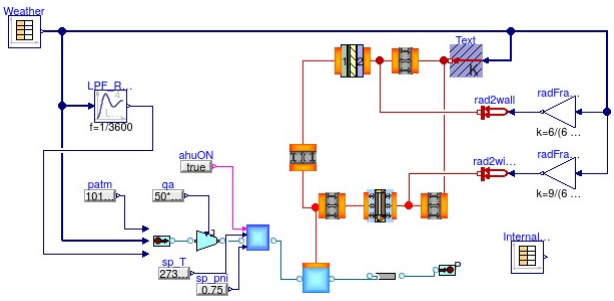
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model OneRoom_ExtData_noAHU
  extends Icons.CaseStudyModel;
  Real phi100 = Room.air.phi * 100;
  EEB.Components.BaseComponents.Air.Volumes.AirVolume Room(V = 50, Xstart = 0.0085);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression qa(y = 50 / 3600);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2wall(S = 6, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS wall2ext(S = 6, gamma = 12);
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_NonHomogeneous wall(A = 6, Tstart = 293.15);
  EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_SingleGlass
                                                                    window(H = 3, L = 3, Tstart = 293.15, material = EEB.Media.Materials.Glasses.Glass(), s = 0.005
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS window2ext(S = 9, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2window(S = 9, gamma = 8);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2window;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2wall;
  Modelica.Blocks.Math.Gain radFracWall(k = 6 / (6 + 9));
  Modelica.Blocks.Math.Gain radFracWindow(k = 9 / (6 + 9));
  Modelica.Blocks.Sources.CombiTimeTable Weather(columns = 2:4, fileName = "/home/leva/Dropbox/Buildings_ABC/Modelica/EEB/Resources/TestDEIB_123_July2015_7days_weather.txt", tableName =
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Text;
  Modelica.Blocks.Sources.CombiTimeTable InternalData(columns = 2:5, fileName = "/home/leva/Dropbox/Buildings_ABC/Modelica/EEB/Resources/TestDEIB_123_July2015_7days_InternalData.txt", t
  Modelica.Blocks.Sources.RealExpression patm(y = 101325);
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed airSrc_amb;
  Modelica.Blocks.Continuous.LowpassButterworth LPF_RHext(f = 1 / 3600, initType = Modelica.Blocks.Types.Init.InitialOutput, n = 4, y_start = 50);
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed airSrc_cond;
EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume airPrescribedFlowRate_Volume1;
EEB.Controllers.Blocks.Analogue.AWPI_1dof PI_T(CSmax = 250 / 3600);
Modelica.Blocks.Sources.RealExpression realExpression3(y = 273.15 + 22);
EEB.Interfaces.Air.sensor_Tphi sTphi;
equation
  connect(sTphi.T, PI_T.PV);
  connect(Room.air_flange2, sTphi.airSense);
  connect(airPrescribedFlowRate_Volume1.air_flange2, Room.air_flange1);
  connect(airSrc_cond.air_flange, airPrescribedFlowRate_Volume1.air_flange1);
  connect(patm.y, airSrc_cond.iP);
  connect(Weather.y[2], LPF_RHext.u);
  connect(LPF_RHext.y, airSrc_amb.iphi);
  connect(airSrc_amb.air_flange, fan.air_flange1);
  connect(patm.y, airSrc_amb.iP);
  connect(Weather.y[1], airSrc_amb.iT);
  connect(Text.port, wall2ext.ss2);
  connect(Weather.y[1], Text.T);
  connect(Text.port, window2ext.ss2);
  connect(Weather.y[3], radFracWall.u);
  connect(radFracWall.u, radFracWindow.u);
  connect(rad2window.Q_flow, radFracWindow.y);
  connect(rad2wall.Q_flow, radFracWall.y);
  connect(rad2wall.port, wall.side2);
  connect(rad2window.port, window.airSide2);
  connect(room2window.ss2, window.airSide1);
  connect(room2window.ss1, Room.heatPort);
  connect(window.airSide2, window2ext.ss1);
  connect(wall.side2, wall2ext.ss1);
  connect(room2wall.ss2, wall.side1);
  connect(Room.heatPort, room2wall.ss1);
  connect(qa.y, fan.iq);
  connect(Room.air_flange2, airSink.air_flange);
  connect(fan.air_flange2, Room.air_flange1);
  connect(sTphi.phi, airSrc_cond.iphi);
  connect(PI_T.CS, airPrescribedFlowRate_Volume1.iq);
  connect(sTphi.T, airSrc_cond.iT);
  connect(realExpression3.y, PI_T.SP);
end OneRoom_ExtData_noAHU;
```

EEB.CaseStudies.ABC.OneRoom_ExtData_forTests



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model OneRoom_ExtData_forTests
  extends Icons.CaseStudyModel;
  Real phi100 = Room.air.phi * 100;
  EEB.Components.BaseComponents.Air.Volumes.AirVolume Room(V = 50, Xstart = 0.005);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression qa(y = 50 * 2 / 3600);
```

```
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2wall(S = 6, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS wall2ext(S = 6, gamma = 12);
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_NonHomogeneous wall(A = 6, Tstart = 293.15);
  EEB.Components.AggregateComponents.Envelope.Openings.InternalWindow_Closed_SingleGlass
                                                                                window(H = 3, L = 3, Tstart = 293.15, material = EEB.Media.Materials.Glasses.Glass(), s = 0.005
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS window2ext(S = 9, gamma = 8);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_SS room2window(S = 9, gamma = 8);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2window;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow rad2wall;
  Modelica.Blocks.Math.Gain radFracWall(k = 6 / (6 + 9));
  Modelica.Blocks.Math.Gain radFracWindow(k = 9 / (6 + 9));
  Modelica.Blocks.Sources.CombiTimeTable Weather(columns = 2:4, extrapolation = Modelica.Blocks.Types.Extrapolation.HoldLastPoint, fileName = Modelica.Utilities.Files.loadResource("modeli
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Text;
  Modelica.Blocks.Sources.CombiTimeTable InternalData(columns = 2:5, fileName = Modelica.Utilities.Files.loadResource("modelica://EEB/Resources/TestDEIB_123_July2015_7days_InternalData.tx
  Modelica.Blocks.Sources.RealExpression patm(y = 101325);
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTphi_prescribed airSrc;
  Modelica.Blocks.Continuous.LowpassButterworth LPF_RHext(f = 1 / 3600, initType = Modelica.Blocks.Types.Init.InitialOutput, n = 4, y_start = 50);
  EEB.Components.BaseComponents.HVAC.AirHandling.ControlledHandler_Tphi_AlgQbal
                                                                          AHU(TC = 60);
  Modelica.Blocks.Sources.RealExpression sp_T(y = 273.15 + 24);
  Modelica.Blocks.Sources.RealExpression sp_phi(y = 0.75);
  Modelica.Blocks.Sources.BooleanExpression ahuON(y = true);
  Components.BaseComponents.Air.Pdrops.AirPdrop_Lin_NomPoint dp2;
equation
  connect(Weather.y[2], LPF_RHext.u);
  connect(Weather.y[3], radFracWall.u);
  connect(Weather.y[1], Text.T);
  connect(Weather.y[1], airSrc.iT);
  connect(fan.air_flange2, AHU.air_flange1);
  connect(AHU.air_flange2, Room.air_flange1);
  connect(AHU.Tsp, sp_T.y);
  connect(sp_phi.y, AHU.phisp);
  connect(ahuON.y, AHU.ON);
  connect(LPF_RHext.y, airSrc.iphi);
  connect(patm.y, airSrc.iP);
  connect(airSrc.air_flange, fan.air_flange1);
  connect(qa.y, fan.iq);
  connect(Room.heatPort, room2wall.ss1);
  connect(room2window.ss1, Room.heatPort);
  connect(room2window.ss2, window.airSide1);
  connect(window.airSide2, window2ext.ss1);
  connect(Text.port, window2ext.ss2);
  connect(rad2window.port, window.airSide2);
  connect(room2wall.ss2, wall.side1);
  connect(Text.port, wall2ext.ss2);
  connect(rad2window.Q_flow, radFracWindow.y);
  connect(radFracWall.u, radFracWindow.u);
  connect(rad2wall.Q_flow, radFracWall.y);
  connect(rad2wall.port, wall.side2);
  connect(wall.side2, wall2ext.ss1);
  connect(Room.air_flange2, dp2.air_flange1);
  connect(dp2.air_flange2, airSink.air_flange);
end OneRoom_ExtData_forTests;
```

EEB.CaseStudies.DEIB

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
RoomLevel	
BuildingLevel	

EEB.CaseStudies.DEIB.RoomLevel

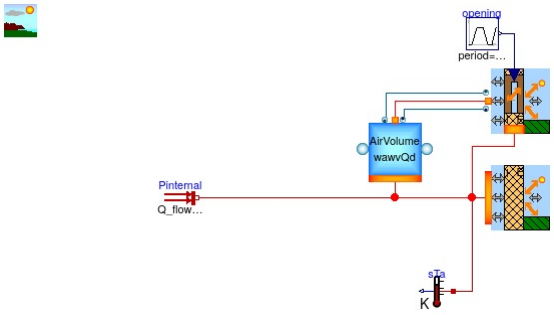
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
DEIBoffice_simple	
DEIBoffice_full	
Rooms_simplified	
Rooms_layered_wall	
Test	

EEB.CaseStudies.DEIB.RoomLevel.DEIBoffice_simple



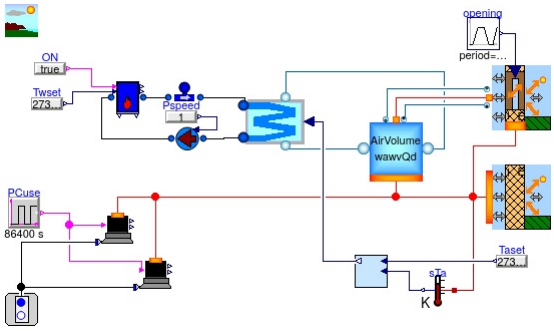
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model DEIBoffice_simple
  extends Icons.CaseStudyModel;
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_constant, Ta_avg = 295.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume wawvQdPort V1(V = 5 * 4 * 3);
  Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_SingleGlass
                                                                    Window(L = 3, H = 3, s = 0.05, material = EEB.Media.Materials.Glasses.Glass(), orientation = 180,
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, width = 120, period = 12 * 3600, startTime = 8 * 3600, amplitude = 0);
  Components.AggregateComponents.Envelope.Walls.ExtenallWall_NoOpenings_Homogeneous
                                                                    extenallWall_NoOpenings_Homogeneous(L = 4, H = 1, orientation = 180, lambda = 0.4, Tstart = 293.15
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sTa;
  Modelica.Thermal.HeatTransfer.Sources.FixedHeatFlow Pinternal(Q_flow = 10);
equation
  connect(opening.y, Window.opening01);
  connect(Window.absToWall, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(V1.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(sTa.port, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(Window.dryair, V1.dryair);
  connect(Window.diffuse, V1.diffuse);
  connect(Window.vapour, V1.vapour);
  connect(Pinternal.port, V1.heatPort);
end DEIBoffice_simple;
```

EEB.CaseStudies.DEIB.RoomLevel.DEIBoffice_full



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model DEIBoffice_full
  extends Icons.CaseStudyModel;
  Appliances.Office.DesktopComputer desktopComputer(ReleaseQ = true);
  Appliances.Office.DesktopComputer desktopComputer1(ReleaseQ = true);
  Components.AggregateComponents.Electrical.FixedACsupply_socket_withGround fixedACsupply_socket_withGround;
  inner BoundaryConditions.AmbientConditions ambient_settings(acv = EEB.Types.AmbCondVariabilty.ACV_variable);
  Components.BaseComponents.Air.Volumes.AirVolume wawvQdPort V1(V = 5 * 4 * 3);
  Components.AggregateComponents.Envelope.Openings.ExtenallWindow_Opening_SingleGlass
                                                                    Window(L = 3, H = 3, s = 0.05, material = EEB.Media.Materials.Glasses.Glass(), wnom = 0.0001, incl
  Modelica.Blocks.Sources.Trapezoid opening(rising = 20, falling = 20, width = 120, period = 12 * 3600, startTime = 8 * 3600, amplitude = 0);
  Modelica.Blocks.Sources.BooleanPulse PCuse(width = 40, period = 86400, startTime = 8 * 3600);
  Components.AggregateComponents.Envelope.Walls.ExtenallWall_NoOpenings_Homogeneous
                                                                    extenallWall_NoOpenings_Homogeneous(L = 4, H = 1, lambda = 0.4, inclination = 90, fixedCoeff = fal
  Components.AggregateComponents.Heating.FanCoil fanCoil(ncoil = 5, Lcoil = 5, gammaMetalExternal = 40, gammaTubeMetal = 200, qamax = 0.03);
  EEB.Components.AggregateComponents.Heating.IdealControlledFluidHeater idealControlledFluidHeater;
  Components.BaseComponents.Water.Pressurisers.IdealWaterPressuriser_pfixed idealWaterPressuriser;
  Modelica.Blocks.Sources.BooleanExpression ON(y = true);
  Modelica.Blocks.Sources.RealExpression Twset(y = 273.15 + 60);
  Modelica.Blocks.Sources.RealExpression Pspeed(y = 1);
  Components.BaseComponents.Water.Pumps.WaterPump_Volumetric waterPump_Volumetric(T = 0.05);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sTa;
  Modelica.Blocks.Sources.RealExpression Taset(y = 273.15 + 20);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_1dof(Ti = 200, K = 0.1, CSmax = 1);
equation
```

```
  connect(opening.y, Window.opening01);
  connect(PCuse.y, desktopComputer1.ON);
  connect(PCuse.y, desktopComputer.ON);
  connect(desktopComputer.plug, fixedACsupply_socket_withGround.socket);
  connect(desktopComputer1.plug, fixedACsupply_socket_withGround.socket);
  connect(Window.absToWall, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(V1.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(desktopComputer1.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(desktopComputer.heatPort, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(fanCoil.air_flange2, V1.air_flange2);
  connect(fanCoil.air_flange1, V1.air_flange1);
  connect(ON.y, idealControlledFluidHeater.ON);
  connect(idealControlledFluidHeater.To, Twset.y);
  connect(Pspeed.y, waterPump_Volumetric.cmd);
  connect(sTa.port, extenallWall_NoOpenings_Homogeneous.airInt);
  connect(aWPI_1dof.SP, Taset.y);
  connect(aWPI_1dof.PV, sTa.T);
  connect(fanCoil.icmd01, aWPI_1dof.CS);
  connect(idealControlledFluidHeater.water_flange2, idealWaterPressuriser.water_flange1);
  connect(idealWaterPressuriser.water_flange2, fanCoil.water_flange1);
  connect(waterPump_Volumetric.water_flange1, fanCoil.water_flange2);
  connect(Window.dryair, V1.dryair);
  connect(V1.diffuse, Window.diffuse);
  connect(Window.vapour, V1.vapour);
  connect(waterPump_Volumetric.water_flange2, idealControlledFluidHeater.water_flange1);
end DEIBoffice_full;
```

Automatically generated Wed Nov 11 09:40:02 2020.

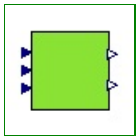
EEB.CaseStudies.DEIB.RoomLevel.Rooms_simplified

Information

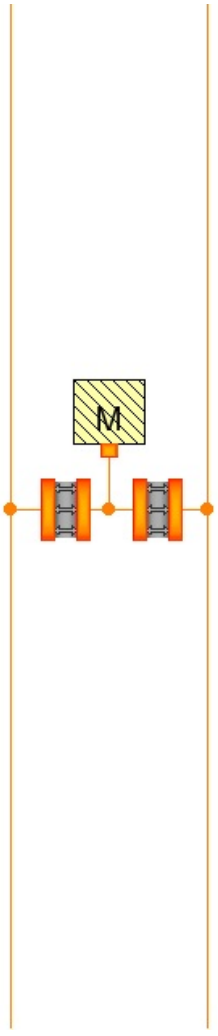
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
BaseClasses	
RoomThermalSimplified	



EEB.CaseStudies.DEIB.RoomLevel.Rooms_simplified.RoomThermalSimplified



Information

Extends from BaseClasses.BaseRoomThermalSimplified.

Parameters

Type	Name	Default	Description
Temperature	Tstart	273.15 + 15	Initial moist air and wall temperature [K]
HeatFlowRate	Wmax	1000	W heater at cmd=1 [W]
Real	COPheat	3	conatant heating COP
MassFlowRate	wr	airRenovation.V*airRenovatio...	Air flow rate [kg/s]

Connectors

Type	Name	Description
output RealOutput	Troom1	
output RealOutput	Troom2	
input RealInput	Texternal	
input RealInput	heater1	
input RealInput	heater2	

Modelica definition

```
model RoomThermalSimplified
  extends BaseClasses.BaseRoomThermalSimplified;
end RoomThermalSimplified;
```

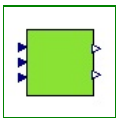
EEB.CaseStudies.DEIB.RoomLevel.Rooms_simplified.BaseClasses

Information

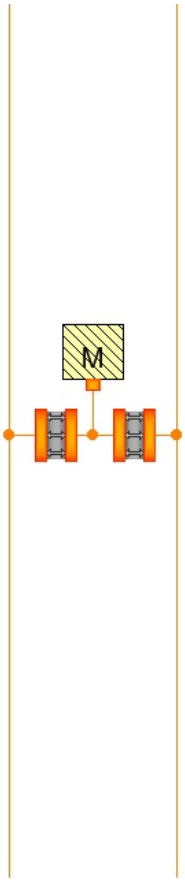
Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
BaseRoomThermalSimplified	



EEB.CaseStudies.DEIB.RoomLevel.Rooms_simplified.BaseClasses.BaseRoomThermalSimplified



Parameters

Type	Name	Default	Description
Temperature	Tstart	273.15 + 15	Initial moist air and wall temperature [K]
HeatFlowRate	Wmax	1000	W heater at cmd=1 [W]
Real	COPheat	3	conatant heating COP
MassFlowRate	wr	airRenovation.V*airRenovatio...	Air flow rate [kg/s]

Connectors

Type	Name	Description
output RealOutput	Troom1	
output RealOutput	Troom2	
input RealInput	Texternal	
input RealInput	heater1	
input RealInput	heater2	

Modelica definition

```
model BaseRoomThermalSimplified
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor Gdoor(G = 8);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature prescribedTemperature;
  Components.BaseComponents.Air.Volumes.AirVolume AirRoom1(V = 5 * 5 * 3, Xstart = 0.001, Tstart = Tstart);
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink1;
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gwe1(S = 1, gamma = 37.5) "Wall-external thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gwe2(S = 1, gamma = 37.5) "Wall-external thermal conductance";
```

```
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gaw2(S = 1, gamma = 35) "Air-Wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gi_aw2(S = 1, gamma = 28) "Wall-wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gaw1(S = 1, gamma = 35) "Air-Wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gi_aw1(S = 1, gamma = 28) "Wall-wall thermal conductance";
  Components.BaseComponents.Air.Volumes.AirVolume AirRoom2(V = 5 * 5 * 3, Xstart = 0.001, Tstart = Tstart);
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink2;
  Components.BaseComponents.Thermal.Capacities.MassT Wall1(M = 3 * (5 * 3 * 0.4) * 1500, cp = 0.22 * 1000 * 4.184, Tstart = Tstart);
  Components.BaseComponents.Thermal.Capacities.MassT InsideWall(M = 5 * 3 * 0.01 * 1500, cp = 0.22 * 1000 * 4.184, Tstart = Tstart);
  Components.BaseComponents.Thermal.Capacities.MassT Wall2(M = 3 * (5 * 3 * 0.4) * 1500, cp = 0.22 * 1000 * 4.184, Tstart = Tstart);
  Modelica.Blocks.Interfaces.RealOutput Troom1;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor4;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor1;
  Modelica.Blocks.Interfaces.RealOutput Troom2;
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin2;
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin1;
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPheat heatPump_ConstantCOPheat(Wmax = Wmax, constCOPheat = COPheat);
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPheat heatPump_ConstantCOPheat1(Wmax = Wmax, constCOPheat = COPheat);
  Modelica.Blocks.Interfaces.RealInput Texternal;
  Modelica.Blocks.Interfaces.RealInput heater1;
  Modelica.Blocks.Math.Gain gain(k = 1 / (Wmax * COPheat));
  Modelica.Blocks.Math.Gain gain1(k = 1 / (Wmax * COPheat));
  Modelica.Blocks.Interfaces.RealInput heater2;
  Components.BaseComponents.Air.Renovation.AirRenovation airRenovation(wr = wr);
  Components.BaseComponents.Air.Renovation.AirRenovation airRenovation1(wr = wr);
  parameter SI.Temperature Tstart = 273.15 + 15 "Initial moist air and wall temperature";
  parameter SI.HeatFlowRate Wmax = 1000 "W heater at cmd=1";
  parameter Real COPheat = 3 "conatant heating COP";
  parameter SI.MassFlowRate wr = airRenovation.V * airRenovation.k "Air flow rate";
  Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(w0 = 0, T0 = Tstart);
  Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX1(w0 = 0, T0 = Tstart);
equation
  connect(airSink1.air_flange, AirRoom1.air_flange2);
  connect(airSink2.air_flange, AirRoom2.air_flange2);
  connect(Wall1.surf, Gwe1.ss1);
  connect(Gaw1.ss2, Gwe1.ss1);
  connect(temperatureSensor1.T, FromKelvin2.Kelvin);
  connect(Troom2, FromKelvin2.Celsius);
  connect(temperatureSensor4.T, FromKelvin1.Kelvin);
  connect(Troom1, FromKelvin1.Celsius);
  connect(Gwe1.ss2, prescribedTemperature.port);
  connect(Gwe2.ss1, prescribedTemperature.port);
  connect(Gaw1.ss1, AirRoom1.heatPort);
  connect(Gi_aw1.ss2, AirRoom1.heatPort);
  connect(Gdoor.port_a, AirRoom1.heatPort);
  connect(Gaw2.ss2, AirRoom2.heatPort);
  connect(Gi_aw2.ss1, AirRoom2.heatPort);
  connect(Gaw2.ss1, Wall2.surf);
  connect(Gwe2.ss2, Wall2.surf);
  connect(Gi_aw1.ss1, InsideWall.surf);
  connect(Gi_aw2.ss2, InsideWall.surf);
  connect(Gdoor.port_b, AirRoom2.heatPort);
  connect(heatPump_ConstantCOPheat.hotPort, AirRoom1.heatPort);
  connect(heatPump_ConstantCOPheat1.hotPort, AirRoom2.heatPort);
  connect(temperatureSensor4.port, AirRoom1.heatPort);
  connect(temperatureSensor1.port, AirRoom2.heatPort);
  connect(prescribedTemperature.T, Texternal);
  connect(heater1, gain.u);
  connect(gain.y, heatPump_ConstantCOPheat.cmd01);
  connect(heater2, gain1.u);
  connect(gain1.y, heatPump_ConstantCOPheat1.cmd01);
  connect(heatPump_ConstantCOPheat.coldPort, prescribedTemperature.port);
  connect(heatPump_ConstantCOPheat1.coldPort, prescribedTemperature.port);
  connect(airRenovation.heatPort1, prescribedTemperature.port);
  connect(airRenovation.heatPort2, AirRoom1.heatPort);
  connect(airRenovation1.heatPort2, AirRoom2.heatPort);
  connect(airRenovation1.heatPort1, prescribedTemperature.port);
  connect(AirRoom1.air_flange1, airSource_fixed_wTX.air_flange);
  connect(airSource_fixed_wTX1.air_flange, AirRoom2.air_flange1);
end BaseRoomThermalSimplified;
```

EEB.CaseStudies.DEIB.RoomLevel.Rooms_layered_wall

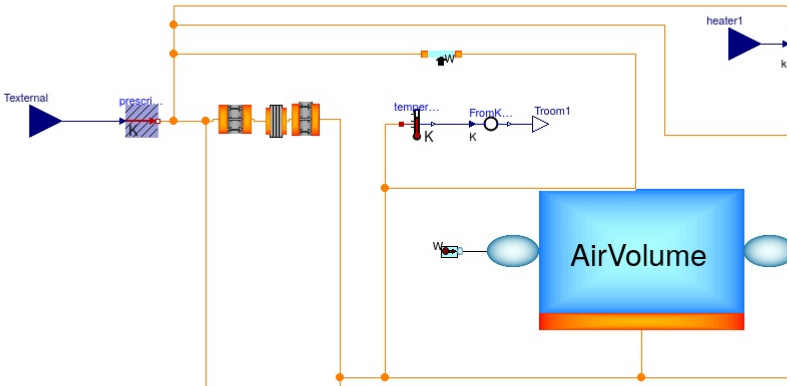
Information

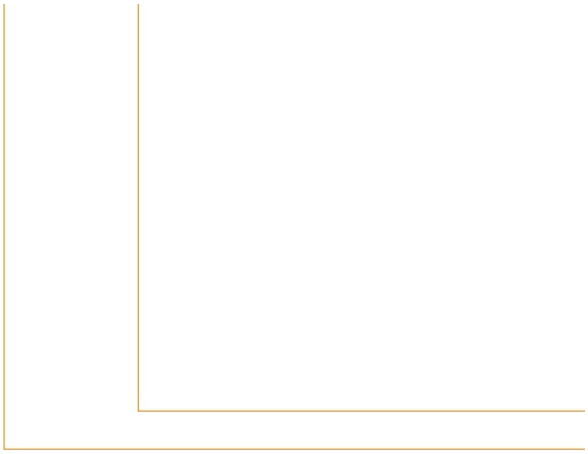
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
BaseClasses	
RoomThermal	

EEB.CaseStudies.DEIB.RoomLevel.Rooms_layered_wall.RoomThermal





Information

Extends from BaseClasses.BaseRoomThermal.

Parameters

Type	Name	Default	Description
Temperature	Tstart	273.15 + 15	Initial moist air and wall temperature [K]
HeatFlowRate	Wmax	1000	W heater at cmd=1 [W]
Real	COPheat	3	conatant heating COP
MassFlowRate	wr	airRenovation.V*airRenovatio...	Air flow rate [kg/s]

Connectors

Type	Name	Description
output RealOutput	Troom1	
output RealOutput	Troom2	
input RealInput	heater1	
input RealInput	heater2	
input RealInput	Texternal	

Modelica definition

```
model RoomThermal
  extends BaseClasses.BaseRoomThermal;
end RoomThermal;
```

EEB.CaseStudies.DEIB.RoomLevel.Rooms_layered_wall.BaseClasses

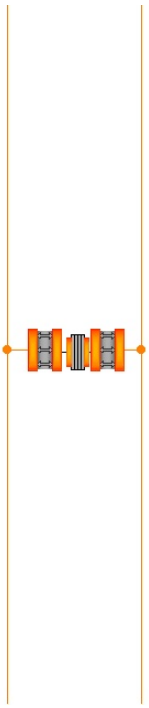
Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
BaseRoomThermal	

EEB.CaseStudies.DEIB.RoomLevel.Rooms_layered_wall.BaseClasses.BaseRoomThermal



Parameters

Type	Name	Default	Description
Temperature	Tstart	273.15 + 15	Initial moist air and wall temperature [K]
HeatFlowRate	Wmax	1000	W heater at cmd=1 [W]
Real	COPheat	3	conatant heating COP
MassFlowRate	wr	airRenovation.V*airRenovatio...	Air flow rate [kg/s]

Connectors

Type	Name	Description
output RealOutput	Troom1	
output RealOutput	Troom2	
input RealInput	heater1	
input RealInput	heater2	
input RealInput	Texternal	

Modelica definition

```
model BaseRoomThermal
  Modelica.Thermal.HeatTransfer.Components.ThermalConductor Gdoor(G = 8);
  Components.BaseComponents.Air.Volumes.AirVolume AirRoom1(V = 5 * 5 * 3, Xstart = 0.001, Tstart = Tstart);
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink1;
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gwe1(S = 3 * 5 * 3, gamma = 5) "Wall-external thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gwe2(S = 3 * 5 * 3, gamma = 5) "Wall-external thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gaw2(S = 3 * 5 * 3, gamma = 3.5) "Air-Wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gi_aw2(S = 3 * 5, gamma = 3.5) "Wall-wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gaw1(S = 3 * 5 * 3, gamma = 3.5) "Air-Wall thermal conductance";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS Gi_aw1(S = 3 * 5, gamma = 3.5) "Wall-wall thermal conductance";
  Components.BaseComponents.Air.Volumes.AirVolume AirRoom2(V = 5 * 5 * 3, Xstart = 0.001, Tstart = Tstart);
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink2;
  Modelica.Blocks.Interfaces.RealOutput Troom1;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor4;
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor1;
  Modelica.Blocks.Interfaces.RealOutput Troom2;
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin2;
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin1;
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPheat heatPump_ConstantCOPheat(Wmax = 1000, constCOPheat = COPheat);
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPheat heatPump_ConstantCOPheat1(Wmax = Wmax, constCOPheat = COPheat);
  Modelica.Blocks.Interfaces.RealInput heater1;
  Modelica.Blocks.Math.Gain gain(k = 1 / (Wmax * COPheat));
  Modelica.Blocks.Math.Gain gain1(k = 1 / (Wmax * COPheat));
  Modelica.Blocks.Interfaces.RealInput heater2;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous Wall1(ro = 1500, s = 0.4, n = 2, A = 3 * 5 * 3, cp = 0.22 * 1000 * 4.184, lambda = 0.1, Tstart = Tstart);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous InsideWall(A = 5 * 3, ro = 1500, cp = 0.22 * 1000 * 4.184, n = 2, lambda = 0.1, s = 0.1, Tstart = Tstart);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous Wall2(ro = 1500, cp = 0.22 * 1000 * 4.184, s = 0.4, n = 2, A = 3 * 5 * 3, lambda = 0.1, Tstart = Tstart);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature prescribedTemperature;
  Modelica.Blocks.Interfaces.RealInput Texternal;
  Components.BaseComponents.Air.Renovation.AirRenovation airRenovation(wr = wr);
  Components.BaseComponents.Air.Renovation.AirRenovation airRenovation1(wr = wr);
  parameter SI.Temperature Tstart = 273.15 + 15 "Initial moist air and wall temperature";
  parameter SI.HeatFlowRate Wmax = 1000 "W heater at cmd=1";
  parameter Real COPheat = 3 "conatant heating COP";
  parameter SI.MassFlowRate wr = airRenovation.V * airRenovation.k "Air flow rate";
  Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(w0 = 0, T0 = Tstart);
```

```
  Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX1(w0 = 0, T0 = Tstart);
equation
  connect(airSink1.air_flange, AirRoom1.air_flange2);
  connect(airSink2.air_flange, AirRoom2.air_flange2);
  connect(temperatureSensor1.T, FromKelvin2.Kelvin);
  connect(Troom2, FromKelvin2.Celsius);
  connect(temperatureSensor4.T, FromKelvin1.Kelvin);
  connect(Troom1, FromKelvin1.Celsius);
  connect(Gaw1.ss1, AirRoom1.heatPort);
  connect(Gi_aw1.ss2, AirRoom1.heatPort);
  connect(Gdoor.port_a, AirRoom1.heatPort);
  connect(Gaw2.ss2, AirRoom2.heatPort);
  connect(Gi_aw2.ss1, AirRoom2.heatPort);
  connect(Gdoor.port_b, AirRoom2.heatPort);
  connect(heatPump_ConstantCOPheat.hotPort, AirRoom1.heatPort);
  connect(heatPump_ConstantCOPheat1.hotPort, AirRoom2.heatPort);
  connect(temperatureSensor4.port, AirRoom1.heatPort);
  connect(temperatureSensor1.port, AirRoom2.heatPort);
  connect(heater1, gain.u);
  connect(gain.y, heatPump_ConstantCOPheat.cmd01);
  connect(heater2, gain1.u);
  connect(gain1.y, heatPump_ConstantCOPheat1.cmd01);
  connect(Gwe1.ss1, Wall1.side1);
  connect(Gaw1.ss2, Wall1.side2);
  connect(Gi_aw1.ss1, InsideWall.side1);
  connect(Gi_aw2.ss2, InsideWall.side2);
  connect(Gaw2.ss1, Wall2.side1);
  connect(Gwe2.ss2, Wall2.side2);
  connect(Gwe1.ss2, prescribedTemperature.port);
  connect(Gwe2.ss1, prescribedTemperature.port);
  connect(prescribedTemperature.T, Texternal);
  connect(heatPump_ConstantCOPheat.coldPort, prescribedTemperature.port);
  connect(heatPump_ConstantCOPheat1.coldPort, prescribedTemperature.port);
  connect(airRenovation.heatPort1, prescribedTemperature.port);
  connect(airRenovation.heatPort2, AirRoom1.heatPort);
  connect(airRenovation1.heatPort2, AirRoom2.heatPort);
  connect(airRenovation1.heatPort1, prescribedTemperature.port);
  connect(AirRoom1.air_flange1, airSource_fixed_wTX.air_flange);
  connect(airSource_fixed_wTX1.air_flange, AirRoom2.air_flange1);
end BaseRoomThermal;
```

EEB.CaseStudies.DEIB.RoomLevel.Test

Information

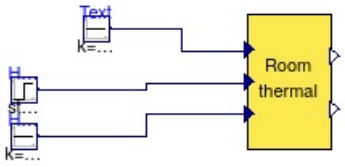
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.TestModel.

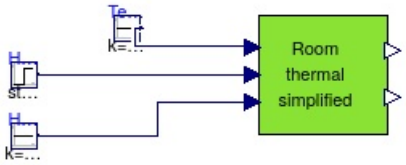
Package Content

Name	Description
test1_Rooms	Differences between wall with or without layer

EEB.CaseStudies.DEIB.RoomLevel.Test.test1_Rooms

Differences between wall with or without layer





Information

Extends from Icons.TestModel.

Modelica definition

```
model test1_Rooms "Differences between wall with or without layer"
  extends Icons.TestModel;
  EEB.CaseStudies.DEIB.RoomLevel.Rooms_simplified.RoomThermalSimplified RoomsThermalSimplified;
  Modelica.Blocks.Sources.Constant Text(k = 273.15 + 18);
  Modelica.Blocks.Sources.Constant Text1(k = 273.15 + 18);
  Modelica.Blocks.Sources.Step Heater1(startTime = 6e6, offset = 300, height = 100);
  Modelica.Blocks.Sources.Constant Heater2(k = 200);
  Modelica.Blocks.Sources.Constant Heater4(k = 200);
  Modelica.Blocks.Sources.Step Heater3(startTime = 6e6, height = 100, offset = 300);
  EEB.CaseStudies.DEIB.RoomLevel.Rooms_layered_wall.RoomThermal RoomThermalLayered;
equation
  connect(RoomsThermalSimplified.Texternal, Text1.y);
  connect(Heater3.y, RoomsThermalSimplified.heater1);
  connect(Heater4.y, RoomsThermalSimplified.heater2);
  connect(Heater2.y, RoomThermalLayered.heater2);
  connect(Heater1.y, RoomThermalLayered.heater1);
  connect(RoomThermalLayered.Texternal, Text.y);
end test1_Rooms;
```

EEB.CaseStudies.DEIB.BuildingLevel

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
ThermalManagementStudy	
ElectricEquivalentStudy	
Test	

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
PI_Rooms	
LQ_Rooms	
Rooms	
Test	

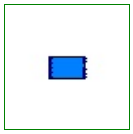
EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Rooms

Information

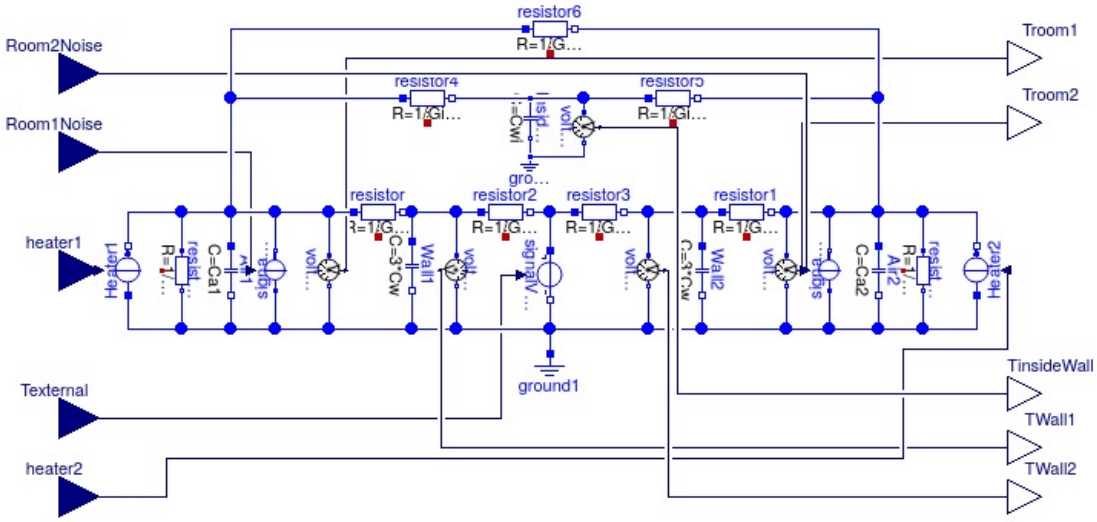
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
RoomElectrical	



EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Rooms.RoomElectrical



Parameters

Type	Name	Default	Description
Conductance	Gaw1	35	Air-Wall thermal conductance [S]
Conductance	Gaw2	35	Air-Wall thermal conductance [S]
Conductance	Gwe1	37.5	Wall-external thermal conductance [S]
Conductance	Gwe2	37.5	Wall-external thermal conductance [S]
Conductance	Gi_aw1	28	Wall-wall thermal conductance [S]
Conductance	Gi_aw2	28	Wall-wall thermal conductance [S]
Conductance	Gdoor	8	Air-Door thermal conductance [S]
Conductance	Gloss1	0.0001	Loss heater1 [S]
Conductance	Gloss2	0.0001	Loss heater2 [S]
Capacitance	Cw	3*5*0.4*1500*0.22*1000*4.184	Capacitance of Wall [F]
Capacitance	Cwi	3*5*0.01*1500*0.22*1000*4.184	Capacitance of Wall [F]
Capacitance	Ca1	5*5*3*1.225*1010	Capacitance of Air [F]
Capacitance	Ca2	5*5*3*1.225*1010	Capacitance of Air [F]

Connectors

Type	Name	Description
input RealInput	heater1	
output RealOutput	Troom1	
input RealInput	heater2	
input RealInput	Texternal	
output RealOutput	Troom2	
input RealInput	Room1Noise	
input RealInput	Room2Noise	
output RealOutput	TinsideWall	

output RealOutput	TWall2	
output RealOutput	TWall1	

Modelica definition

```
model RoomElectrical
  Modelica.Electrical.Analog.Sources.SignalCurrent Heater1;
  Modelica.Electrical.Analog.Basic.Capacitor Air1(C = Ca1);
  Modelica.Electrical.Analog.Basic.Capacitor Wall1(C = 3 * Cw);
  Modelica.Electrical.Analog.Basic.Resistor resistor(R = 1 / Gaw1);
  Modelica.Electrical.Analog.Sensors.VoltageSensor voltageSensor;
  Modelica.Electrical.Analog.Sources.SignalCurrent signalCurrent1;
  Modelica.Electrical.Analog.Basic.Ground ground1;
  Modelica.Electrical.Analog.Basic.Capacitor Wall2(C = 3 * Cw);
  Modelica.Electrical.Analog.Basic.Capacitor Air2(C = Ca2);
  Modelica.Electrical.Analog.Basic.Resistor resistor1(R = 1 / Gaw2);
  Modelica.Electrical.Analog.Basic.Resistor resistor2(R = 1 / Gwe1);
  Modelica.Electrical.Analog.Basic.Resistor resistor3(R = 1 / Gwe2);
  Modelica.Blocks.Interfaces.RealInput heater1;
  Modelica.Blocks.Interfaces.RealOutput Troom1;
  Modelica.Electrical.Analog.Sources.SignalVoltage signalVoltage;
  Modelica.Blocks.Interfaces.RealInput heater2;
  Modelica.Blocks.Interfaces.RealInput Texternal;
  Modelica.Blocks.Interfaces.RealOutput Troom2;
  Modelica.Electrical.Analog.Sensors.VoltageSensor voltageSensor1;
  Modelica.Electrical.Analog.Sources.SignalCurrent signalCurrent2;
  Modelica.Electrical.Analog.Sources.SignalCurrent Heater2;
  Modelica.Electrical.Analog.Basic.Resistor resistor4(R = 1 / Gi_aw1);
  Modelica.Electrical.Analog.Basic.Resistor resistor5(R = 1 / Gi_aw2);
  Modelica.Electrical.Analog.Basic.Capacitor Inside_Wall(C = Cwi);
  Modelica.Electrical.Analog.Basic.Ground ground2;
  parameter SI.Conductance Gaw1 = 35 "Air-Wall thermal conductance";
  parameter SI.Conductance Gaw2 = 35 "Air-Wall thermal conductance";
  parameter SI.Conductance Gwe1 = 37.5 "Wall-external thermal conductance";
  parameter SI.Conductance Gwe2 = 37.5 "Wall-external thermal conductance";
  parameter SI.Conductance Gi_aw1 = 28 "Wall-wall thermal conductance";
  parameter SI.Conductance Gi_aw2 = 28 "Wall-wall thermal conductance";
  parameter SI.Conductance Gdoor = 8 "Air-Door thermal conductance";
  parameter SI.Conductance Gloss1 = 0.0001 "Loss heater1";
  parameter SI.Conductance Gloss2 = 0.0001 "Loss heater2";
  parameter SI.Capacitance Cw = 3 * 5 * 0.4 * 1500 * 0.22 * 1000 * 4.184 "Capacitance of Wall";
  parameter SI.Capacitance Cwi = 3 * 5 * 0.01 * 1500 * 0.22 * 1000 * 4.184 "Capacitance of Wall";
  parameter SI.Capacitance Ca1 = 5 * 5 * 3 * 1.225 * 1010 "Capacitance of Air";
  parameter SI.Capacitance Ca2 = 5 * 5 * 3 * 1.225 * 1010 "Capacitance of Air";
  Modelica.Blocks.Interfaces.RealInput Room1Noise;
  Modelica.Blocks.Interfaces.RealInput Room2Noise;
  Modelica.Electrical.Analog.Basic.Resistor resistor6(R = 1 / Gdoor);
  Modelica.Electrical.Analog.Basic.Resistor resistor7(R = 1 / Gloss1);
  Modelica.Electrical.Analog.Basic.Resistor resistor8(R = 1 / Gloss2);
  Modelica.Electrical.Analog.Sensors.VoltageSensor voltageSensor2;
  Modelica.Electrical.Analog.Sensors.VoltageSensor voltageSensor3;
  Modelica.Electrical.Analog.Sensors.VoltageSensor voltageSensor4;
  Modelica.Blocks.Interfaces.RealOutput TinsideWall;
  Modelica.Blocks.Interfaces.RealOutput TWall2;
  Modelica.Blocks.Interfaces.RealOutput TWall1;
equation
  connect(Wall1.n, Air1.n);
  connect(Wall1.p, resistor.n);
  connect(Heater1.n, resistor.p);
  connect(Air1.p, resistor.p);
  connect(voltageSensor.n, Air1.n);
  connect(voltageSensor.p, resistor.p);
  connect(resistor2.n, resistor3.p);
  connect(resistor3.n, Wall2.p);
  connect(signalVoltage.n, Air1.n);
  connect(Heater1.p, Air1.n);
  connect(signalVoltage.p, resistor3.p);
  connect(Wall2.n, Air1.n);
  connect(signalCurrent1.n, resistor.p);
  connect(signalCurrent1.p, Air1.n);
  connect(resistor2.p, resistor.n);
  connect(ground1.p, Air1.n);
  connect(resistor1.p, Wall2.p);
  connect(resistor1.n, Air2.p);
  connect(Air2.n, Air1.n);
  connect(voltageSensor1.p, Air2.p);
  connect(voltageSensor1.n, Air1.n);
  connect(signalCurrent2.n, Air2.p);
  connect(signalCurrent2.p, Air1.n);
  connect(Heater2.n, Air2.p);
  connect(Heater2.p, Air1.n);
  connect(voltageSensor1.v, Troom2);
  connect(voltageSensor.v, Troom1);
```

```
  connect(heater1, Heater1.i);
  connect(Texternal, signalVoltage.v);
  connect(resistor4.p, resistor.p);
  connect(resistor4.n, Inside_Wall.p);
  connect(resistor5.p, Inside_Wall.p);
  connect(resistor5.n, Air2.p);
  connect(ground2.p, Inside_Wall.n);
  connect(Room1Noise, signalCurrent1.i);
  connect(Room2Noise, signalCurrent2.i);
  connect(resistor6.p, resistor.p);
  connect(resistor6.n, Air2.p);
  connect(resistor7.p, resistor.p);
  connect(resistor7.n, Air1.n);
  connect(resistor8.p, Air2.p);
  connect(resistor8.n, Air1.n);
  connect(voltageSensor2.p, resistor.n);
  connect(voltageSensor2.n, Air1.n);
  connect(voltageSensor3.p, Wall2.p);
  connect(voltageSensor3.n, Air1.n);
  connect(voltageSensor4.p, Inside_Wall.p);
  connect(voltageSensor4.n, ground2.p);
  connect(voltageSensor2.v, TWall1);
  connect(heater2, Heater2.i);
  connect(voltageSensor3.v, TWall2);
  connect(voltageSensor4.v, TinsideWall);
end RoomElectrical;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms

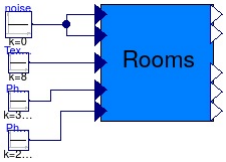
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
Desired_condition	
Step1	
Step2	
IMC	
IMC_withNoise	

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms.Desired_condition



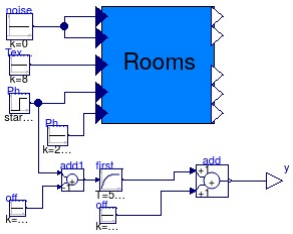
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model Desired_condition
  extends Icons.CaseStudyModel;
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Constant Pheater1(k = 300);
  Modelica.Blocks.Sources.Constant Pheater2(k = 200);
equation
  connect(noise.y, rooms.Room1Noise);
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(Pheater1.y, rooms.heater1);
  connect(Texternal.y, rooms.Texternal);
  connect(Pheater2.y, rooms.heater2);
end Desired_condition;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms.Step1



Information

Extends from Icons.CaseStudyModel.

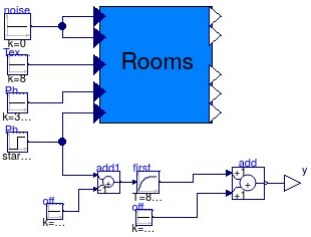
Connectors

Type	Name	Description
output RealOutput	y	

Modelica definition

```
model Step1
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Step Pheater1(height = 100, offset = 300, startTime = 6e6);
  Modelica.Blocks.Sources.Constant Pheater2(k = 200);
  Modelica.Blocks.Continuous.FirstOrder firstOrder(T = 5682, k = 0.0207);
  Modelica.Blocks.Math.Add add;
  Modelica.Blocks.Sources.Constant offset(k = 22.613);
  Modelica.Blocks.Math.Add add1(k2 = -1);
  Modelica.Blocks.Sources.Constant offset1(k = 300);
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Interfaces.RealOutput y;
equation
  connect(firstOrder.y, add.u1);
  connect(offset.y, add.u2);
  connect(offset1.y, add1.u2);
  connect(firstOrder.u, add1.y);
  connect(noise.y, rooms.Room1Noise);
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(Pheater1.y, rooms.heater1);
  connect(Pheater2.y, rooms.heater2);
  connect(add1.u1, rooms.heater1);
  connect(add.y, y);
end Step1;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms.Step2



Information

Extends from Icons.CaseStudyModel.

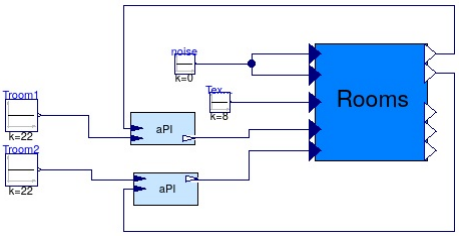
Connectors

Type	Name	Description
output RealOutput	y	

Modelica definition

```
model Step2
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Constant Pheater1(k = 300);
  Modelica.Blocks.Sources.Step Pheater2(offset = 200, height = 100, startTime = 6e6);
  Modelica.Blocks.Continuous.FirstOrder firstOrder(k = 0.0213, T = 8268);
  Modelica.Blocks.Math.Add add;
  Modelica.Blocks.Sources.Constant offset(k = 21.0024);
  Modelica.Blocks.Math.Add add1(k2 = -1);
  Modelica.Blocks.Sources.Constant offset1(k = 200);
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Interfaces.RealOutput y;
equation
  connect(offset.y, add.u2);
  connect(offset1.y, add1.u2);
  connect(firstOrder.u, add1.y);
  connect(noise.y, rooms.Room1Noise);
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(Pheater1.y, rooms.heater1);
  connect(Pheater2.y, rooms.heater2);
  connect(add1.u1, rooms.heater2);
  connect(firstOrder.y, add.u1);
  connect(add.y, y);
end Step2;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms.IMC



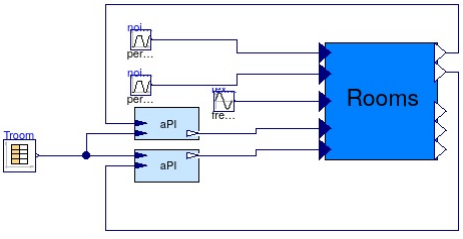
Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model IMC
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(CSmax = 1000, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue1(CSmax = 1000, Ti = 8268, K = 38816);
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.Constant Troom1(k = 22);
  Modelica.Blocks.Sources.Constant Troom2(k = 22);
equation
  connect(Texternal.y, rooms.Texternal);
  connect(noise.y, rooms.Room1Noise);
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(aWPI_analogue.CS, rooms.heater1);
  connect(aWPI_analogue1.CS, rooms.heater2);
  connect(rooms.Troom2, aWPI_analogue1.PV);
  connect(rooms.Troom1, aWPI_analogue.PV);
  connect(Troom1.y, aWPI_analogue.SP);
  connect(Troom2.y, aWPI_analogue1.SP);
end IMC;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.PI_Rooms.IMC_withNoise



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model IMC_withNoise
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.Sine Texternal(freqHz = 1 / 86400, phase = -pi / 2, offset = 8, amplitude = 3);
  Controllers.Blocks.Analogue.AWPI_1dof AWPI_analogue1(CSmax = 1000, CSmin = 0, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof AWPI_analogue2(CSmax = 1000, CSmin = 0, Ti = 8268, K = 38816);
  Modelica.Blocks.Sources.Trapezoid noise1(period = 3600 * 24, rising = 300, falling = 300, startTime = 6e5, width = 720, amplitude = 250);
  Modelica.Blocks.Sources.Trapezoid noise2(period(displayUnit = "h") = 3600 * 24, rising = 300, falling = 300, startTime = 1e6, width = 720, amplitude = 300);
  Modelica.Blocks.Sources.CombiTimeTable Troom(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 18
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
equation
  connect(Troom.y[1], AWPI_analogue1.SP);
  connect(AWPI_analogue2.SP, AWPI_analogue1.SP);
  connect(noise1.y, rooms.Room1Noise);
  connect(noise2.y, rooms.Room2Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(AWPI_analogue1.CS, rooms.heater1);
  connect(AWPI_analogue2.CS, rooms.heater2);
  connect(rooms.Troom2, AWPI_analogue2.PV);
  connect(rooms.Troom1, AWPI_analogue1.PV);
end IMC_withNoise;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.LQ_Rooms

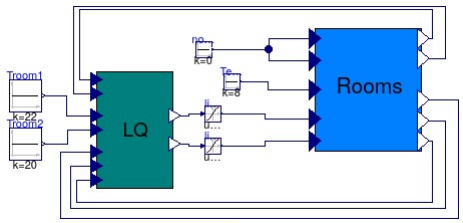
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
Base_Classes	
LQ	
LQ_withNoise	

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.LQ_Rooms.LQ



Information

Extends from Icons.CaseStudyModel.

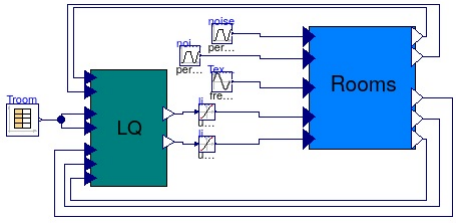
Parameters

Type	Name	Default	Description
Real	Klq[2, 5]	1e5*[9.9993, 0.0004, 0.0000,...	

Modelica definition

```
model LQ
  extends Icons.CaseStudyModel;
  parameter Real Klq[2, 5] = 1e5 * [9.9993, 0.0004, 0.0000, 0.0001, 0.0003; 0.0001, 0.0000, 0.0004, 9.9993, 0.0003];
  Base_Classes.LQ_Base lQ_Base(K = Klq);
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Nonlinear.Limiter limiter2(uMax = 1000, uMin = 0);
  Modelica.Blocks.Nonlinear.Limiter limiter1(uMax = 1000, uMin = 0);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant Troom1(k = 22);
  Modelica.Blocks.Sources.Constant Troom2(k = 20);
equation
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(rooms.TWall2, lQ_Base.Vc);
  connect(rooms.TWall1, lQ_Base.Vb);
  connect(rooms.TinsideWall, lQ_Base.Ve);
  connect(rooms.Troom2, lQ_Base.Vd);
  connect(lQ_Base.u1, limiter1.u);
  connect(limiter1.y, rooms.heater1);
  connect(limiter2.y, rooms.heater2);
  connect(lQ_Base.u2, limiter2.u);
  connect(rooms.Troom1, lQ_Base.Va);
  connect(noise.y, rooms.Room1Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(Troom1.y, lQ_Base.Vref1);
  connect(Troom2.y, lQ_Base.Vref2);
end LQ;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.LQ_Rooms.LQ_withNoise



Information

Extends from Icons.CaseStudyModel.

Parameters

Type	Name	Default	Description
Real	Klq[2, 5]	1e5*[9.9993, 0.0004, 0.0000,...	

Modelica definition

```
model LQ_withNoise
  extends Icons.CaseStudyModel;
  parameter Real Klq[2, 5] = 1e5 * [9.9993, 0.0004, 0.0000, 0.0001, 0.0003; 0.0001, 0.0000, 0.0004, 9.9993, 0.0003];
  Base_Classes.LQ_Base lQ_Base(K = Klq);
  Rooms.RoomElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Nonlinear.Limiter limiter2(uMax = 1000, uMin = 0);
  Modelica.Blocks.Nonlinear.Limiter limiter1(uMax = 1000, uMin = 0);
  Modelica.Blocks.Sources.CombiTimeTable Troom(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 18
  Modelica.Blocks.Sources.Sine Texternal(freqHz = 1 / 86400, phase = -pi / 2, offset = 8, amplitude = 3);
  Modelica.Blocks.Sources.Trapezoid noise(period = 3600 * 24, rising = 300, falling = 300, startTime = 6e5, width = 720, amplitude = 250);
  Modelica.Blocks.Sources.Trapezoid noise1(period(displayUnit = "h") = 3600 * 24, rising = 300, falling = 300, startTime = 1e6, width = 720, amplitude = 300);
equation
  connect(rooms.TWall2, lQ_Base.Vc);
  connect(rooms.TWall1, lQ_Base.Vb);
  connect(rooms.TinsideWall, lQ_Base.Ve);
  connect(rooms.Troom2, lQ_Base.Vd);
  connect(lQ_Base.u1, limiter1.u);
  connect(limiter1.y, rooms.heater1);
  connect(limiter2.y, rooms.heater2);
  connect(lQ_Base.u2, limiter2.u);
```

```
  connect(rooms.Troom1, lQ_Base.Va);
  connect(Troom.y[1], lQ_Base.Vref1);
  connect(lQ_Base.Vref2, lQ_Base.Vref1);
  connect(Texternal.y, rooms.Texternal);
  connect(noise.y, rooms.Room1Noise);
  connect(noise1.y, rooms.Room2Noise);
end LQ_withNoise;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.LQ_Rooms.Base_Classes

Information

Extends from Modelica.Icons.InternalPackage (Icon for an internal package (indicating that the package should not be directly utilized by user)).

Package Content

Name	Description
LQ_Base	

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.LQ_Rooms.Base_Classes.LQ_Base

Parameters

Type	Name	Default	Description
Real	K[2, 5]	[4.4072, 4.7416, 2.9566, 0.7...	

Connectors

Type	Name	Description
output RealOutput	u1	
output RealOutput	u2	
input RealInput	Vd	Temperature room 2
input RealInput	Va	Temperature room 1
input RealInput	Ve	Temperature Inside Wall
input RealInput	Vc	Temperature wall 2
input RealInput	Vb	Temperature wall 1
input RealInput	Vref1	Desired temperature room 1
input RealInput	Vref2	Desired temperature room 1

Modelica definition

```
model LQ_Base
  // System matrix
  parameter Real K[2, 5] = [4.4072, 4.7416, 2.9566, 0.7632, 2.0505; 0.7632, 2.9566, 4.7416, 4.4072, 2.0505];
  Real SP[2, 2];
  // Transfer function from y_setpoint to u_setpoint
  Real up[2, 1];
  // Setpoint of the control variables
  Real u[2, 1];
  // Control variables
  Real x[5, 1];
  // States
  Real yp[2, 1];
  // Setpoint of output
  Modelica.Blocks.Interfaces.RealOutput u1;
  Modelica.Blocks.Interfaces.RealOutput u2;
  Modelica.Blocks.Interfaces.RealInput Vd "Temperature room 2";
  Modelica.Blocks.Interfaces.RealInput Va "Temperature room 1";
  Modelica.Blocks.Interfaces.RealInput Ve "Temperature Inside Wall";
  Modelica.Blocks.Interfaces.RealInput Vc "Temperature wall 2";
  Modelica.Blocks.Interfaces.RealInput Vb "Temperature wall 1";
  Modelica.Blocks.Interfaces.RealInput Vref1 "Desired temperature room 1 ";
  Modelica.Blocks.Interfaces.RealInput Vref2 "Desired temperature room 1 ";
protected
  parameter Real A[5, 5] = [-0.0007651, 0.0003772, 0, 8.621e-05, 0.0003017; 1.408e-06, -2.917e-06, 0, 0, 0; 0, 0, -2.917e-06, 1.408e-06, 0; 8.621e-05, 0, 0.0003772, -0.0007651, 0.000
  parameter Real B[5, 2] = [1.078e-05, 0; 0, 0; 0, 0; 0, 0; 0, 1.078e-05; 0, 0];
  parameter Real C[2, 5] = [1, 0, 0, 0, 0; 0, 0, 0, 1, 0];
  // LQ gain
equation
  SP = -Modelica.Math.Matrices.inv(C * Modelica.Math.Matrices.inv(A - B * K) * B);
  x[1, 1] = Va;
  x[2, 1] = Vb;
  x[3, 1] = Vc;
  x[4, 1] = Vd;
  x[5, 1] = Ve;
  u[1, 1] = u1;
  u[2, 1] = u2;
  yp[1, 1] = Vref1;
  yp[2, 1] = Vref2;
  // LQ optimal control law
  up = SP * yp;
  u = (-K * x) + up;
end LQ_Base;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Test

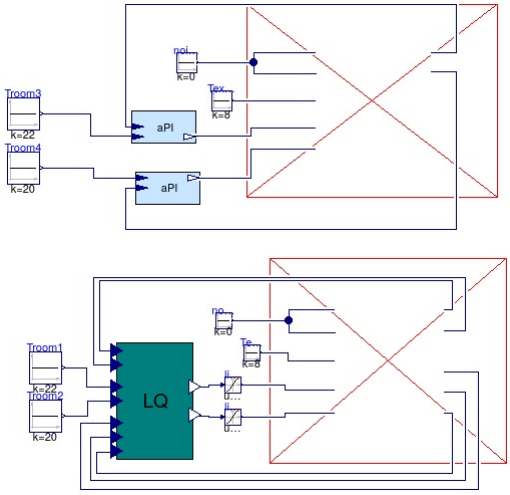
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.TestModel.

Package Content

Name	Description
test_1	
test_2	
test_3	

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Test.test_1



Information

Extends from Icons.TestModel.

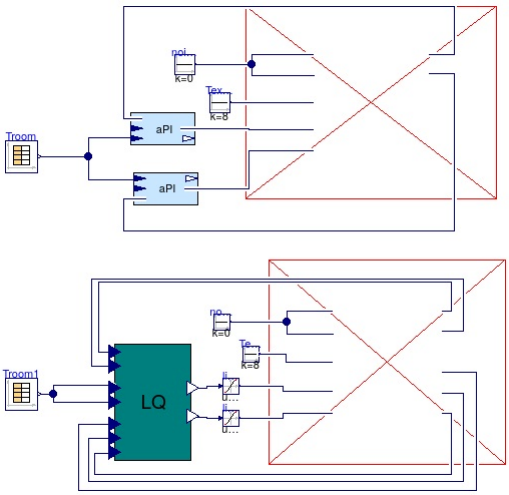
Parameters

Type	Name	Default	Description
Real	Klq[2, 5]	1e5*[9.9993, 0.0004, 0.0000,...	

Modelica definition

```
model test_1
  extends Icons.TestModel;
  parameter Real Klq[2, 5] = 1e5 * [9.9993, 0.0004, 0.0000, 0.0001, 0.0003; 0.0001, 0.0000, 0.0004, 9.9993, 0.0003];
  LQ_Rooms.Base_Classes.LQ_Base lQ_Base(K = Klq);
  Rooms.RoomsElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Nonlinear.Limiter limiter2(uMax = 1000, uMin = 0);
  Modelica.Blocks.Nonlinear.Limiter limiter1(uMax = 1000, uMin = 0);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant Troom1(k = 22);
  Modelica.Blocks.Sources.Constant Troom2(k = 20);
  Modelica.Blocks.Sources.Constant Texternal1(k = 8);
  Modelica.Blocks.Sources.Constant noise1(k = 0);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(CSmax = 1000, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue1(CSmax = 1000, Ti = 8268, K = 38816);
  Rooms.RoomsElectrical rooms1(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.Constant Troom3(k = 22);
  Modelica.Blocks.Sources.Constant Troom4(k = 20);
equation
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(rooms.TWall2, lQ_Base.Vc);
  connect(rooms.TWall1, lQ_Base.Vb);
  connect(rooms.TinsideWall, lQ_Base.Ve);
  connect(rooms.Troom2, lQ_Base.Vd);
  connect(lQ_Base.u1, limiter1.u);
  connect(limiter1.y, rooms.heater1);
  connect(limiter2.y, rooms.heater2);
  connect(lQ_Base.u2, limiter2.u);
  connect(rooms.Troom1, lQ_Base.Va);
  connect(noise.y, rooms.Room1Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(Troom1.y, lQ_Base.Vref1);
  connect(Troom2.y, lQ_Base.Vref2);
  connect(Texternal1.y, rooms1.Texternal);
  connect(noise1.y, rooms1.Room1Noise);
  connect(rooms1.Room2Noise, rooms1.Room1Noise);
  connect(aWPI_analogue.CS, rooms1.heater1);
  connect(aWPI_analogue1.CS, rooms1.heater2);
  connect(rooms1.Troom2, aWPI_analogue1.PV);
  connect(rooms1.Troom1, aWPI_analogue.PV);
  connect(Troom3.y, aWPI_analogue.SP);
  connect(Troom4.y, aWPI_analogue1.SP);
end test_1;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Test.test_2



Information

Extends from Icons.TestModel.

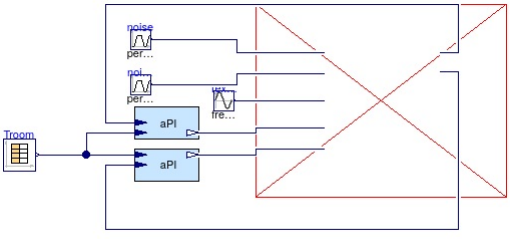
Parameters

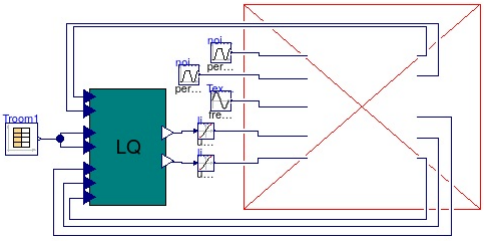
Type	Name	Default	Description
Real	Klq[2, 5]	1e5*[9.9993, 0.0004, 0.0000,...	

Modelica definition

```
model test_2
  extends Icons.TestModel;
  parameter Real Klq[2, 5] = 1e5 * [9.9993, 0.0004, 0.0000, 0.0001, 0.0003; 0.0001, 0.0000, 0.0004, 9.9993, 0.0003];
  LQ_Rooms.Base_Classes.LQ_Base lQ_Base(K = Klq);
  Rooms.RoomsElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Nonlinear.Limiter limiter2(uMax = 1000, uMin = 0);
  Modelica.Blocks.Nonlinear.Limiter limiter1(uMax = 1000, uMin = 0);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant Texternal1(k = 8);
  Modelica.Blocks.Sources.Constant noise1(k = 0);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(CSmax = 1000, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue1(CSmax = 1000, Ti = 8268, K = 38816);
  Rooms.RoomsElectrical rooms1(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.CombiTimeTable Troom(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 18
  Modelica.Blocks.Sources.CombiTimeTable Troom1(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 1
equation
  connect(rooms.Room2Noise, rooms.Room1Noise);
  connect(rooms.TWall2, lQ_Base.Vc);
  connect(rooms.TWall1, lQ_Base.Vb);
  connect(rooms.TinsideWall, lQ_Base.Ve);
  connect(rooms.Troom2, lQ_Base.Vd);
  connect(lQ_Base.u1, limiter1.u);
  connect(limiter1.y, rooms.heater1);
  connect(limiter2.y, rooms.heater2);
  connect(lQ_Base.u2, limiter2.u);
  connect(rooms.Troom1, lQ_Base.Va);
  connect(noise.y, rooms.Room1Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(Texternal1.y, rooms1.Texternal);
  connect(noise1.y, rooms1.Room1Noise);
  connect(rooms1.Room2Noise, rooms1.Room1Noise);
  connect(aWPI_analogue.CS, rooms1.heater1);
  connect(aWPI_analogue1.CS, rooms1.heater2);
  connect(rooms1.Troom2, aWPI_analogue1.PV);
  connect(rooms1.Troom1, aWPI_analogue.PV);
  connect(Troom.y[1], aWPI_analogue.SP);
  connect(aWPI_analogue1.SP, aWPI_analogue.SP);
  connect(Troom1.y[1], lQ_Base.Vref1);
  connect(lQ_Base.Vref2, lQ_Base.Vref1);
end test_2;
```

EEB.CaseStudies.DEIB.BuildingLevel.ElectricEquivalentStudy.Test.test_3





Information

Extends from Icons.TestModel.

Parameters

Type	Name	Default	Description
Real	Klq[2, 5]	1e5*[9.9993, 0.0004, 0.0000,...	

Modelica definition

```
model test_3
  extends Icons.TestModel;
  parameter Real Klq[2, 5] = 1e5 * [9.9993, 0.0004, 0.0000, 0.0001, 0.0003; 0.0001, 0.0000, 0.0004, 9.9993, 0.0003];
  Real Ec(start = 0) "Total energy consuption with IMC";
  Real Ec1(start = 0) "Total energy consuption with LQ";
  Modelica.Blocks.Sources.Sine Texternal(freqHz = 1 / 86400, phase = -pi / 2, offset = 8, amplitude = 3);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(CSmax = 1000, CSmin = 0, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue1(CSmax = 1000, CSmin = 0, Ti = 8268, K = 38816);
  Modelica.Blocks.Sources.Trapezoid noise(period = 3600 * 24, rising = 300, falling = 300, startTime = 6e5, width = 720, amplitude = 250);
  Modelica.Blocks.Sources.Trapezoid noise1(period(displayUnit = "h") = 3600 * 24, rising = 300, falling = 300, startTime = 1e6, width = 720, amplitude = 300);
  Modelica.Blocks.Sources.CombiTimeTable Troom(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 18
  Rooms.RoomsElectrical rooms(Air1(v(start = 0)), Air2(v(start = 0)));
  LQ_Rooms.Base_Classes.LQ_Base lQ_Base(K = Klq);
  Rooms.RoomsElectrical rooms1(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Nonlinear.Limiter limiter2(uMax = 1000, uMin = 0);
  Modelica.Blocks.Nonlinear.Limiter limiter1(uMax = 1000, uMin = 0);
  Modelica.Blocks.Sources.CombiTimeTable Troom1(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 1
  Modelica.Blocks.Sources.Sine Texternal1(freqHz = 1 / 86400, phase = -pi / 2, offset = 8, amplitude = 3);
  Modelica.Blocks.Sources.Trapezoid noise2(period = 3600 * 24, rising = 300, falling = 300, startTime = 6e5, width = 720, amplitude = 250);
  Modelica.Blocks.Sources.Trapezoid noise3(period(displayUnit = "h") = 3600 * 24, rising = 300, falling = 300, startTime = 1e6, width = 720, amplitude = 300);
equation
  if time < 1382400 then
    der(Ec) = 0;
    der(Ec1) = 0;
  elseif time > 1468800 then
    der(Ec) = 0;
    der(Ec1) = 0;
  else
    der(Ec) = (rooms.heater1 + rooms.heater2) / 1000 / 3600;
    der(Ec1) = (rooms.heater1 + rooms.heater2) / 1000 / 3600;
  end if;
  connect(Troom.y[1], aWPI_analogue.SP);
  connect(aWPI_analogue1.SP, aWPI_analogue.SP);
  connect(noise.y, rooms.Room1Noise);
  connect(noise1.y, rooms.Room2Noise);
  connect(Texternal.y, rooms.Texternal);
  connect(aWPI_analogue.CS, rooms.heater1);
  connect(aWPI_analogue1.CS, rooms.heater2);
  connect(rooms.Troom2, aWPI_analogue1.PV);
  connect(rooms.Troom1, aWPI_analogue.PV);
  connect(rooms1.TWall2, lQ_Base.Vc);
  connect(rooms1.TWall1, lQ_Base.Vb);
  connect(rooms1.TinsideWall, lQ_Base.Ve);
  connect(rooms1.Troom2, lQ_Base.Vd);
  connect(lQ_Base.u1, limiter1.u);
  connect(limiter1.y, rooms1.heater1);
  connect(limiter2.y, rooms1.heater2);
  connect(lQ_Base.u2, limiter2.u);
  connect(rooms1.Troom1, lQ_Base.Va);
  connect(Troom1.y[1], lQ_Base.Vref1);
  connect(lQ_Base.Vref2, lQ_Base.Vref1);
  connect(Texternal1.y, rooms1.Texternal);
  connect(noise2.y, rooms1.Room1Noise);
  connect(noise3.y, rooms1.Room2Noise);
end test_3;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy

Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
OfficeBuilding	

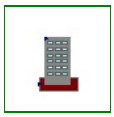
EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding

Information

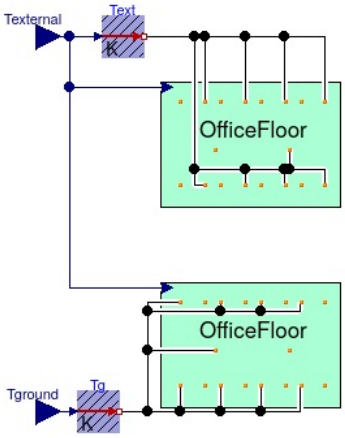
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
BaseClasses	
OfficeBuilding	
Test	
CaseStudy_Building	



EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.OfficeBuilding



Parameters

Type	Name	Default	Description
Integer	n	5	number of floor (min 2)

Connectors

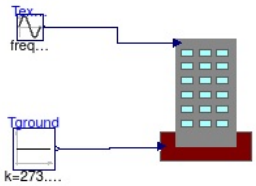
Type	Name	Description
input RealInput	Texternal	
input RealInput	Tground	

Modelica definition

```
model OfficeBuilding
  BaseClasses.OfficeFloor floor[n - 2];
  parameter Integer n = 5 "number of floor (min 2)";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Tg;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature Text;
  BaseClasses.OfficeFloor Ground(gammaFloor = 0.31, gammaFloor_c = 0.31);
  BaseClasses.OfficeFloor Top(gammaCeiling = 5, gammaFloor_c = 5);
  Modelica.Blocks.Interfaces.RealInput Texternal;
  Modelica.Blocks.Interfaces.RealInput Tground;
equation
  if n > 2 then
    connect(floor[1].floor_r1, Ground.ceiling_r1);
    connect(floor[1].floor_r2, Ground.ceiling_r2);
    connect(floor[1].floor_r3, Ground.ceiling_r3);
    connect(floor[1].floor_r4, Ground.ceiling_r4);
    connect(floor[1].floor_r5, Ground.ceiling_r5);
    connect(floor[1].floor_r6, Ground.ceiling_r6);
    connect(floor[1].floor_r7, Ground.ceiling_r7);
    connect(floor[1].floor_r8, Ground.ceiling_r8);
    connect(floor[1].floor_corridor, Ground.ceiling_corridor);
    connect(floor[n - 2].ceiling_r1, Top.floor_r1);
    connect(floor[n - 2].ceiling_r2, Top.floor_r2);
    connect(floor[n - 2].ceiling_r3, Top.floor_r3);
    connect(floor[n - 2].ceiling_r4, Top.floor_r4);
    connect(floor[n - 2].ceiling_r5, Top.floor_r5);
    connect(floor[n - 2].ceiling_r6, Top.floor_r6);
    connect(floor[n - 2].ceiling_r7, Top.floor_r7);
    connect(floor[n - 2].ceiling_r8, Top.floor_r8);
    connect(floor[n - 2].ceiling_corridor, Top.floor_corridor);
    for i in 1:n - 3 loop
      connect(floor[i].Texternal, Texternal);
      connect(floor[i].ceiling_r1, floor[i + 1].floor_r1);
      connect(floor[i].ceiling_r2, floor[i + 1].floor_r2);
      connect(floor[i].ceiling_r3, floor[i + 1].floor_r3);
```

```
      connect(floor[i].ceiling_r4, floor[i + 1].floor_r4);
      connect(floor[i].ceiling_r5, floor[i + 1].floor_r5);
      connect(floor[i].ceiling_r6, floor[i + 1].floor_r6);
      connect(floor[i].ceiling_r7, floor[i + 1].floor_r7);
      connect(floor[i].ceiling_r8, floor[i + 1].floor_r8);
      connect(floor[i].ceiling_corridor, floor[i + 1].floor_corridor);
    end for;
    connect(floor[n - 2].Texternal, Texternal);
  else
    connect(Ground.ceiling_r1, Top.floor_r1);
    connect(Ground.ceiling_r2, Top.floor_r1);
    connect(Ground.ceiling_r3, Top.floor_r1);
    connect(Ground.ceiling_r4, Top.floor_r1);
    connect(Ground.ceiling_r5, Top.floor_r1);
    connect(Ground.ceiling_r6, Top.floor_r1);
    connect(Ground.ceiling_r7, Top.floor_r1);
    connect(Ground.ceiling_r8, Top.floor_r1);
    connect(Ground.ceiling_corridor, Top.floor_corridor);
  end if;
  connect(Text.T, Texternal);
  connect(Top.Texternal, Texternal);
  connect(Ground.Texternal, Texternal);
  connect(Ground.floor_r5, Tg.port);
  connect(Ground.floor_r1, Tg.port);
  connect(Ground.floor_corridor, Tg.port);
  connect(Ground.floor_r6, Tg.port);
  connect(Ground.floor_r7, Tg.port);
  connect(Ground.floor_r8, Tg.port);
  connect(Top.ceiling_r1, Text.port);
  connect(Top.ceiling_r2, Text.port);
  connect(Top.ceiling_r3, Text.port);
  connect(Top.ceiling_r4, Text.port);
  connect(Top.ceiling_r5, Text.port);
  connect(Top.ceiling_r6, Text.port);
  connect(Top.ceiling_r7, Text.port);
  connect(Top.ceiling_r8, Text.port);
  connect(Ground.floor_r2, Tg.port);
  connect(Ground.floor_r3, Tg.port);
  connect(Ground.floor_r4, Tg.port);
  connect(Top.ceiling_corridor, Text.port);
  connect(Tg.T, Tground);
end OfficeBuilding;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.CaseStudy_Building



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model CaseStudy_Building
  extends Icons.CaseStudyModel;
  OfficeBuilding officeBuilding(n = 10);
  Modelica.Blocks.Sources.Constant Tground(k = 273.15 + 12);
  Modelica.Blocks.Sources.Sine Texternal(freqHz = 1 / 86400, phase = -pi / 2, amplitude = 4, offset = 273.15 + 34);
equation
  connect(Tground.y, officeBuilding.Tground);
  connect(Texternal.y, officeBuilding.Texternal);
end CaseStudy_Building;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses

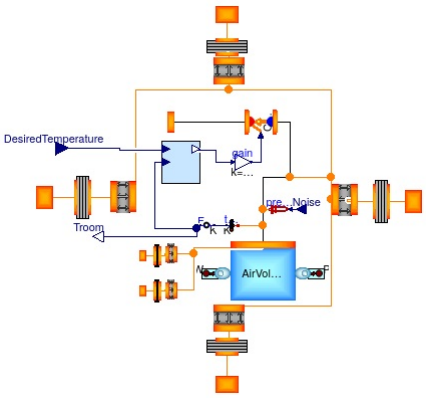
Information

Extends from Modelica.Icons.BasesPackage (Icon for packages containing base classes).

Package Content

Name	Description
OfficeRoom	
OfficeFloor	
IMC_Tuning	

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses.OfficeRoom



Parameters

Type	Name	Default	Description
Area	A13	15	wall1 and wall3 surface [m2]
Area	A24	15	wall2 and wall4 surface [m2]
Area	Af	25	floor surface [m2]
Area	Ac	25	ceiling surface [m2]
Real	K	-3.5	Gain
Time	Ti	220	Integral time [s]
Volume	Vroom	5*5*3	volume [m3]
Length	s1	0.2	wall_1 thickness [m]
Length	s2	0.2	wall_2 thickness [m]
Length	s3	0.2	wall_3 thickness [m]
Length	s4	0.2	wall_4 thickness [m]
Length	sf	0.2	floor thickness [m]
Length	sc	0.2	ceiling thickness [m]
Integer	n1	2	number of layers of wall_1
Integer	n2	2	number of layers of wall_2
Integer	n3	2	number of layers of wall1_3
Integer	n4	2	number of layers of wall1_4
Integer	nf	2	number of layers of floor
Integer	nc	2	number of layers of ceiling

Connectors

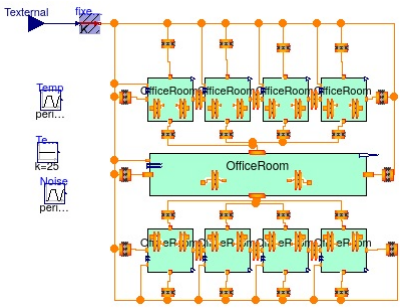
Type	Name	Description
HeatPort	Wall1	
HeatPort	Wall3	
HeatPort	Wall4	
HeatPort	Wall2	
input RealInput	DesiredTemperature	
HeatPort	Texternal	
input RealInput	Noise	Peolpe+PC
HeatPort	Ceiling	
HeatPort	Floor	
output RealOutput	Troom	

Modelica definition

```
model OfficeRoom
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall(ro = 1500, cp = 920, lambda = 0.2, A = A13, s = s1, n = n1);
  Components.BaseComponents.Air.Volumes.AirVolume airVolume(Tstart = 273.15 + 25, V = Vroom);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca6(gamma = 3.5, S = A13);
  Interfaces.Thermal.HeatPort Wall1;
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Interfaces.Thermal.HeatPort Wall3;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall1(ro = 1500, cp = 920, lambda = 0.2, A = A13, s = s3, n = n3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca5(gamma = 3.5, S = A13);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall13(ro = 1500, cp = 920, lambda = 0.2, A = A24, s = s4, n = n4);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca3(gamma = 3.5, S = A24);
  Interfaces.Thermal.HeatPort Wall4;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall12(ro = 1500, cp = 920, lambda = 0.2, A = A24, s = s2, n = n2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(gamma = 3.5, S = A24);
  Interfaces.Thermal.HeatPort Wall2;
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPcool heatPump_ConstantCOPcool(constCOPcool = 3, Wmax = 1500);
  Modelica.Blocks.Interfaces.RealInput DesiredTemperature;
  Modelica.Blocks.Math.Gain gain(k = 1 / 3);
  Interfaces.Thermal.HeatPort Texternal;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow prescribedHeatFlow;
  Modelica.Blocks.Interfaces.RealInput Noise "Peolpe+PC";
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(Ti = Ti, K = K, CSmin = 0);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor4;
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin1;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall14(ro = 1500, cp = 920, lambda = 0.2, s = sc, A = Ac, n = nc);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall15(ro = 1500, cp = 920, lambda = 0.2, s = sf, A = Af, n = nf);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca2(gamma = 3.5, S = Af);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca4(gamma = 3.5, S = Ac);
  Interfaces.Thermal.HeatPort Ceiling;
  Interfaces.Thermal.HeatPort Floor;
```

```
  parameter SI.Area A13 = 15 "wall1 and wall3 surface";
  parameter SI.Area A24 = 15 "wall2 and wall4 surface";
  parameter SI.Area Af = 25 "floor surface";
  parameter SI.Area Ac = 25 "ceiling surface";
  parameter Real K = -3.5 "Gain";
  parameter SI.Time Ti = 220 "Integral time";
  Modelica.Blocks.Interfaces.RealOutput Troom;
  parameter SI.Volume Vroom = 5 * 5 * 3 "volume";
  parameter SI.Length s1 = 0.2 "wall_1 thickness";
  parameter SI.Length s2 = 0.2 "wall_2 thickness";
  parameter SI.Length s3 = 0.2 "wall_3 thickness";
  parameter SI.Length s4 = 0.2 "wall_4 thickness";
  parameter SI.Length sf = 0.2 "floor thickness";
  parameter SI.Length sc = 0.2 "ceiling thickness";
  parameter Integer n1 = 2 "number of layers of wall_1";
  parameter Integer n2 = 2 "number of layers of wall_2";
  parameter Integer n3 = 2 "number of layers of wall1_3";
  parameter Integer n4 = 2 "number of layers of wall1_4";
  parameter Integer nf = 2 "number of layers of floor";
  parameter Integer nc = 2 "number of layers of ceiling";
  Components.BaseComponents.Air.Sources.AirSource_wTX_fixed airSource_fixed_wTX(w0 = 0, T0 = 273.15 + 25);
equation
  connect(airVolume.air_flange2, airSink_P_fixed.air_flange);
  connect(convSca2Sca6.ss2, layeredWall1.side2);
  connect(layeredWall1.side1, convSca2Sca5.ss1);
  connect(layeredWall3.side1, convSca2Sca3.ss1);
  connect(convSca2Sca1.ss2, layeredWall2.side2);
  connect(convSca2Sca5.ss2, airVolume.heatPort);
  connect(convSca2Sca1.ss1, airVolume.heatPort);
  connect(convSca2Sca6.ss1, airVolume.heatPort);
  connect(convSca2Sca3.ss2, airVolume.heatPort);
  connect(gain.y, heatPump_ConstantCOPcool.cmd01);
  connect(prescribedHeatFlow.port, airVolume.heatPort);
  connect(temperatureSensor4.T, FromKelvin1.Kelvin);
  connect(gain.u, aWPI_analogue.CS);
  connect(prescribedHeatFlow.Q_flow, Noise);
  connect(temperatureSensor4.port, airVolume.heatPort);
  connect(layeredWall1.side2, Wall3);
  connect(Wall2, layeredWall2.side1);
  connect(layeredWall.side1, Wall1);
  connect(layeredWall3.side2, Wall4);
  connect(layeredWall4.side1, Ceiling);
  connect(layeredWall5.side1, Floor);
  connect(layeredWall5.side2, convSca2Sca2.ss2);
  connect(layeredWall4.side2, convSca2Sca4.ss2);
  connect(convSca2Sca4.ss1, airVolume.heatPort);
  connect(convSca2Sca2.ss1, airVolume.heatPort);
  connect(Troom, FromKelvin1.Celsius);
  connect(DesiredTemperature, aWPI_analogue.SP);
  connect(aWPI_analogue.PV, FromKelvin1.Celsius);
  connect(heatPump_ConstantCOPcool.hotPort, Texternal);
  connect(heatPump_ConstantCOPcool.coldPort, airVolume.heatPort);
  connect(airVolume.air_flange1, airSource_fixed_wTX.air_flange);
end OfficeRoom;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses.OfficeFloor



Parameters

Type	Name	Default	Description
Area	A13	15	Office wall1 and wall3 surface [m2]
Area	A24	15	Office wall2 and wall4 surface [m2]
Area	Af	25	Office floor surface [m2]
Area	Ac	25	Office ceiling surface [m2]
Area	A13_c	15	Office wall1 and wall3 surface [m2]
Area	A24_c	60	Office wall2 and wall4 surface [m2]
Area	Af_c	100	Office floor surface [m2]
Area	Ac_c	100	Office ceiling surface [m2]
CoefficientOfHeatTransfer	gammaFloor	4	heat transfer coefficient office floor [W/(m2.K)]
CoefficientOfHeatTransfer	gammaCeiling	4	heat transfer coefficient office ceiling [W/(m2.K)]
CoefficientOfHeatTransfer	gammaFloor_c	4	heat transfer coefficient corridor floor [W/(m2.K)]
CoefficientOfHeatTransfer	gammaCeiling_c	4	heat transfer coefficient corridor ceiling [W/(m2.K)]

Connectors

Type	Name	Description
HeatPort	floor_r1	
HeatPort	ceiling_r1	
HeatPort	floor_r2	
HeatPort	ceiling_r2	
HeatPort	floor_r3	
HeatPort	ceiling_r3	
HeatPort	floor_r4	
HeatPort	ceiling_r4	
HeatPort	floor_r5	
HeatPort	ceiling_r5	
HeatPort	floor_r6	
HeatPort	ceiling_r6	
HeatPort	floor_r7	
HeatPort	ceiling_r7	
HeatPort	floor_r8	
HeatPort	ceiling_r8	
HeatPort	floor_corridor	

HeatPort	ceiling_corridor	
input RealInput	Texternal	

Modelica definition

```
model OfficeFloor
  BaseClasses.OfficeRoom officeRoom1(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s1 = 0.4, s2 = 0.4, n1 = 4, n2 = 4);
  BaseClasses.OfficeRoom officeRoom2(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s2 = 0.4, n2 = 4);
  BaseClasses.OfficeRoom officeRoom3(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s2 = 0.4, n2 = 4);
  BaseClasses.OfficeRoom officeRoom4(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s2 = 0.4, s3 = 0.4, n2 = 4, n3 = 4);
  BaseClasses.OfficeRoom officeRoom5(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s1 = 0.4, s4 = 0.4, n1 = 4, n4 = 4);
  BaseClasses.OfficeRoom officeRoom6(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s4 = 0.4, n4 = 4);
  BaseClasses.OfficeRoom officeRoom7(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s4 = 0.4, n4 = 4);
  BaseClasses.OfficeRoom officeRoom8(A13 = A13, A24 = A24, Af = Af, Ac = Ac, s3 = 0.4, s4 = 0.4, n3 = 4, n4 = 4);
  BaseClasses.OfficeRoom Corridor(A13 = A13_c, A24 = A24_c, Af = Af_c, Ac = Ac_c, s1 = 0.4, s3 = 0.4, n1 = 4, n3 = 4, Vroom = 20 * 5 * 3, K = -11.2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca2(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca3(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca4(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca5(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca6(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca7(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca8(S = A24);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca(S = A13);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca9(S = A13_c);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca10(S = A13);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca11(S = A13);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca12(S = A13_c);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca13(S = A13);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature;
  Modelica.Blocks.Sources.Trapezoid Noise(rising(displayUnit = "h") = 3600, width = 10 * 3600, falling(displayUnit = "h") = 3600, period = 3600 * 24, startTime = 86400 + 7 * 3600, ampl
  Interfaces.Thermal.HeatPort floor_r1;
  Interfaces.Thermal.HeatPort ceiling_r1;
  Interfaces.Thermal.HeatPort floor_r2;
  Interfaces.Thermal.HeatPort ceiling_r2;
  Interfaces.Thermal.HeatPort floor_r3;
  Interfaces.Thermal.HeatPort ceiling_r3;
  Interfaces.Thermal.HeatPort floor_r4;
  Interfaces.Thermal.HeatPort ceiling_r4;
  Interfaces.Thermal.HeatPort floor_r5;
  Interfaces.Thermal.HeatPort ceiling_r5;
  Interfaces.Thermal.HeatPort floor_r6;
  Interfaces.Thermal.HeatPort ceiling_r6;
  Interfaces.Thermal.HeatPort floor_r7;
  Interfaces.Thermal.HeatPort ceiling_r7;
  Interfaces.Thermal.HeatPort floor_r8;
  Interfaces.Thermal.HeatPort ceiling_r8;
  Interfaces.Thermal.HeatPort floor_corridor;
  Interfaces.Thermal.HeatPort ceiling_corridor;
  parameter SI.Area A13 = 15 "Office wall1 and wall3 surface";
  parameter SI.Area A24 = 15 "Office wall2 and wall4 surface";
  parameter SI.Area Af = 25 "Office floor surface";
  parameter SI.Area Ac = 25 "Office ceiling surface";
  parameter SI.Area A13_c = 15 "Office wall1 and wall3 surface";
  parameter SI.Area A24_c = 60 "Office wall2 and wall4 surface";
  parameter SI.Area Af_c = 100 "Office floor surface";
  parameter SI.Area Ac_c = 100 "Office ceiling surface";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca14(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca15(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca16(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca17(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca18(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca19(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca20(S = A24, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca21(S = A24, gamma = 2);
  Modelica.Blocks.Interfaces.RealInput Texternal;
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca22(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca23(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca24(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca25(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca26(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca27(S = A13, gamma = 2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca28(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca29(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca30(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca31(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca32(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca33(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca34(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca35(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca36(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca37(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca38(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca39(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca40(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca41(S = Ac, gamma = gammaCeiling);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca42(S = Af, gamma = gammaFloor);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca43(S = Ac, gamma = gammaCeiling);
  parameter SI.CoefficientOfHeatTransfer gammaFloor = 4 "heat transfer coefficient office floor";
  parameter SI.CoefficientOfHeatTransfer gammaCeiling = 4 "heat transfer coefficient office ceiling";
  parameter SI.CoefficientOfHeatTransfer gammaFloor_c = 4 "heat transfer coefficient corridor floor";
  parameter SI.CoefficientOfHeatTransfer gammaCeiling_c = 4 "heat transfer coefficient corridor ceiling";
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca44(S = Af_c, gamma = gammaFloor_c);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca45(S = Ac_c, gamma = gammaCeiling_c);
  Modelica.Blocks.Sources.Trapezoid Temp(width(displayUnit = "h") = 43200, period(displayUnit = "h") = 86400, offset = 25, startTime = 3600 * 7, amplitude = -0.5, rising(displayUnit =
  Modelica.Blocks.Sources.Constant TempCorridor(k = 25);
equation
  connect(Temp.y, officeRoom1.DesiredTemperature);
  connect(Temp.y, officeRoom2.DesiredTemperature);
  connect(Temp.y, officeRoom3.DesiredTemperature);
  connect(Temp.y, officeRoom4.DesiredTemperature);
  connect(Temp.y, officeRoom5.DesiredTemperature);
  connect(Temp.y, officeRoom6.DesiredTemperature);
  connect(Temp.y, officeRoom7.DesiredTemperature);
  connect(Temp.y, officeRoom8.DesiredTemperature);
  connect(TempCorridor.y, Corridor.DesiredTemperature);
  connect(Noise.y, officeRoom1.Noise);
  connect(Noise.y, officeRoom2.Noise);
  connect(Noise.y, officeRoom3.Noise);
  connect(Noise.y, officeRoom4.Noise);
  connect(Noise.y, officeRoom5.Noise);
  connect(Noise.y, officeRoom6.Noise);
  connect(Noise.y, officeRoom7.Noise);
  connect(Noise.y, officeRoom8.Noise);
  connect(Noise.y, Corridor.Noise);
  connect(convSca2Sca2.ss1, fixedTemperature.port);
  connect(convSca2Sca1.ss1, fixedTemperature.port);
  connect(convSca2Sca3.ss1, fixedTemperature.port);
  connect(convSca2Sca4.ss1, fixedTemperature.port);
  connect(convSca2Sca11.ss1, fixedTemperature.port);
  connect(convSca2Sca.ss2, fixedTemperature.port);
  connect(convSca2Sca9.ss2, fixedTemperature.port);
  connect(convSca2Sca10.ss2, fixedTemperature.port);
  connect(convSca2Sca6.ss2, fixedTemperature.port);
  connect(convSca2Sca5.ss2, fixedTemperature.port);
  connect(convSca2Sca7.ss2, fixedTemperature.port);
  connect(convSca2Sca8.ss2, fixedTemperature.port);
```

```
  connect(convSca2Sca13.ss1, fixedTemperature.port);
  connect(convSca2Sca12.ss1, fixedTemperature.port);
  connect(officeRoom1.Texternal, fixedTemperature.port);
  connect(Corridor.Texternal, fixedTemperature.port);
  connect(officeRoom6.Texternal, fixedTemperature.port);
  connect(officeRoom7.Texternal, fixedTemperature.port);
  connect(officeRoom8.Texternal, fixedTemperature.port);
  connect(officeRoom5.Texternal, fixedTemperature.port);
  connect(officeRoom1.Wall4, convSca2Sca14.ss1);
  connect(officeRoom2.Wall4, convSca2Sca15.ss1);
  connect(officeRoom3.Wall4, convSca2Sca16.ss1);
  connect(officeRoom4.Wall4, convSca2Sca17.ss1);
  connect(officeRoom5.Wall4, convSca2Sca18.ss2);
  connect(convSca2Sca19.ss2, officeRoom6.Wall4);
  connect(convSca2Sca20.ss2, officeRoom7.Wall4);
  connect(convSca2Sca21.ss2, officeRoom8.Wall4);
  connect(convSca2Sca18.ss1, Corridor.Wall4);
  connect(convSca2Sca19.ss1, Corridor.Wall4);
  connect(convSca2Sca20.ss1, Corridor.Wall4);
  connect(convSca2Sca21.ss1, Corridor.Wall4);
  connect(convSca2Sca14.ss2, Corridor.Wall2);
  connect(convSca2Sca15.ss2, Corridor.Wall2);
  connect(convSca2Sca16.ss2, Corridor.Wall2);
  connect(convSca2Sca17.ss2, Corridor.Wall2);
  connect(fixedTemperature.T, Texternal);
  connect(officeRoom1.Wall3, convSca2Sca22.ss2);
  connect(convSca2Sca22.ss1, officeRoom2.Wall1);
  connect(officeRoom2.Wall3, convSca2Sca23.ss2);
  connect(convSca2Sca23.ss1, officeRoom3.Wall1);
  connect(officeRoom3.Wall3, convSca2Sca24.ss2);
  connect(convSca2Sca24.ss1, officeRoom4.Wall1);
  connect(officeRoom5.Wall3, convSca2Sca25.ss2);
  connect(convSca2Sca25.ss1, officeRoom6.Wall1);
  connect(officeRoom6.Wall3, convSca2Sca26.ss2);
  connect(convSca2Sca26.ss1, officeRoom7.Wall1);
  connect(officeRoom7.Wall3, convSca2Sca27.ss2);
  connect(convSca2Sca27.ss1, officeRoom8.Wall1);
  connect(floor_r1, convSca2Sca28.ss2);
  connect(convSca2Sca28.ss1, officeRoom1.Floor);
  connect(officeRoom1.Ceiling, convSca2Sca29.ss2);
  connect(convSca2Sca29.ss1, ceiling_r1);
  connect(floor_r2, convSca2Sca32.ss2);
  connect(officeRoom2.Ceiling, convSca2Sca33.ss2);
  connect(convSca2Sca33.ss1, ceiling_r2);
  connect(convSca2Sca32.ss1, officeRoom2.Floor);
  connect(floor_r3, convSca2Sca34.ss2);
  connect(convSca2Sca34.ss1, officeRoom3.Floor);
  connect(officeRoom3.Ceiling, convSca2Sca35.ss2);
  connect(convSca2Sca35.ss1, ceiling_r3);
  connect(officeRoom4.Floor, convSca2Sca36.ss1);
  connect(floor_r4, convSca2Sca36.ss2);
  connect(officeRoom4.Ceiling, convSca2Sca37.ss2);
  connect(convSca2Sca37.ss1, ceiling_r4);
  connect(officeRoom8.Ceiling, convSca2Sca43.ss2);
  connect(convSca2Sca43.ss1, ceiling_r8);
  connect(officeRoom8.Floor, convSca2Sca42.ss1);
  connect(floor_r8, convSca2Sca42.ss2);
  connect(convSca2Sca31.ss1, ceiling_r7);
  connect(officeRoom7.Ceiling, convSca2Sca31.ss2);
  connect(convSca2Sca30.ss1, officeRoom7.Floor);
  connect(floor_r7, convSca2Sca30.ss2);
  connect(convSca2Sca41.ss1, ceiling_r6);
  connect(convSca2Sca41.ss2, officeRoom6.Ceiling);
  connect(officeRoom6.Floor, convSca2Sca40.ss1);
  connect(convSca2Sca40.ss2, floor_r6);
  connect(convSca2Sca39.ss1, ceiling_r5);
  connect(convSca2Sca39.ss2, officeRoom5.Ceiling);
  connect(convSca2Sca38.ss1, officeRoom5.Floor);
  connect(floor_r5, convSca2Sca38.ss2);
  connect(floor_corridor, convSca2Sca44.ss2);
  connect(convSca2Sca44.ss1, Corridor.Floor);
  connect(Corridor.Ceiling, convSca2Sca45.ss2);
  connect(convSca2Sca45.ss1, ceiling_corridor);
  connect(convSca2Sca2.ss2, officeRoom1.Wall2);
  connect(convSca2Sca1.ss2, officeRoom2.Wall2);
  connect(convSca2Sca3.ss2, officeRoom3.Wall2);
  connect(convSca2Sca4.ss2, officeRoom4.Wall2);
  connect(convSca2Sca11.ss2, officeRoom4.Wall3);
  connect(convSca2Sca12.ss2, Corridor.Wall3);
  connect(convSca2Sca13.ss2, officeRoom8.Wall3);
  connect(officeRoom8.Wall2, convSca2Sca8.ss1);
  connect(officeRoom7.Wall2, convSca2Sca7.ss1);
  connect(officeRoom6.Wall2, convSca2Sca5.ss1);
  connect(officeRoom5.Wall2, convSca2Sca6.ss1);
  connect(convSca2Sca10.ss1, officeRoom5.Wall1);
  connect(convSca2Sca9.ss1, Corridor.Wall1);
  connect(convSca2Sca.ss1, officeRoom1.Wall1);
  connect(officeRoom2.Texternal, fixedTemperature.port);
  connect(officeRoom3.Texternal, fixedTemperature.port);
  connect(officeRoom4.Texternal, fixedTemperature.port);
end OfficeFloor;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses.IMC_Tuning

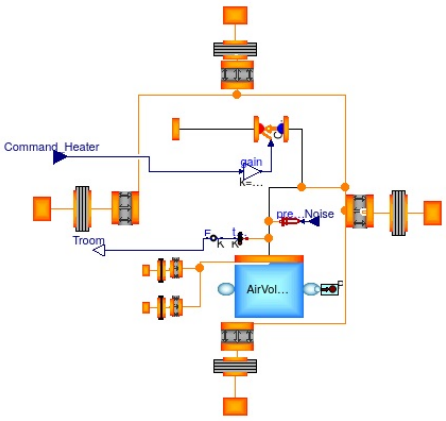
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.CaseStudyModel.

Package Content

Name	Description
Room_without_PI	
IMC_Tuning	

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses.IMC_Tuning.Room_without_PI



Parameters

Type	Name	Default	Description
Area	A13	15	wall1 and wall3 surface [m2]
Area	A24	15	wall2 and wall4 surface [m2]
Area	Af	25	floor surface [m2]
Area	Ac	25	ceiling surface [m2]
Volume	Vroom	5*5*3	volume [m3]
Length	s1	0.2	wall_1 thickness [m]
Length	s2	0.2	wall_2 thickness [m]
Length	s3	0.2	wall_3 thickness [m]
Length	s4	0.2	wall_4 thickness [m]
Length	sf	0.2	floor thickness [m]
Length	sc	0.2	ceiling thickness [m]
Integer	n1	2	number of layers of wall_1
Integer	n2	2	number of layers of wall_2
Integer	n3	2	number of layers of wall1_3
Integer	n4	2	number of layers of wall1_4
Integer	nf	2	number of layers of floor
Integer	nc	2	number of layers of ceiling

Connectors

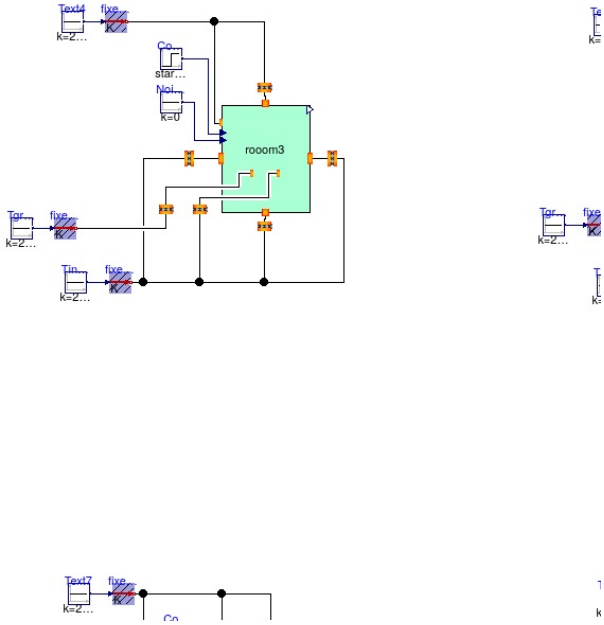
Type	Name	Description
HeatPort	Wall1	
HeatPort	Wall3	
HeatPort	Wall4	
HeatPort	Wall2	
input RealInput	Command_Heater	
HeatPort	Texternal	
input RealInput	Noise	Peolpe+PC
HeatPort	Ceiling	
HeatPort	Floor	
output RealOutput	Troom	

Modelica definition

```
model Room_without_PI
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall1(ro = 1500, cp = 920, lambda = 0.2, A = A13, s = s1, n = n1);
  Components.BaseComponents.Air.Volumes.AirVolume airVolume(Tstart = 273.15 + 25, V = Vroom);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca6(gamma = 3.5, S = A13);
  Interfaces.Thermal.HeatPort Wall1;
  Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink_P_fixed;
  Interfaces.Thermal.HeatPort Wall3;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall1(ro = 1500, cp = 920, lambda = 0.2, A = A13, s = s3, n = n3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca5(gamma = 3.5, S = A13);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall3(ro = 1500, cp = 920, lambda = 0.2, A = A24, s = s4, n = n4);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca3(gamma = 3.5, S = A24);
  Interfaces.Thermal.HeatPort Wall4;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall2(ro = 1500, cp = 920, lambda = 0.2, A = A24, s = s2, n = n2);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(gamma = 3.5, S = A24);
  Interfaces.Thermal.HeatPort Wall2;
  Components.BaseComponents.HVAC.HeatPumps.HeatPump_ConstantCOPcool heatPump_ConstantCOPcool(Wmax = 1500, constCOPcool = 3);
  Modelica.Blocks.Interfaces.RealInput Command_Heater;
  Modelica.Blocks.Math.Gain gain(k = 1 / 3);
  Interfaces.Thermal.HeatPort Texternal;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow prescribedHeatFlow;
  Modelica.Blocks.Interfaces.RealInput Noise "Peolpe+PC";
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor temperatureSensor4;
```

```
  Modelica.Thermal.HeatTransfer.Celsius.FromKelvin FromKelvin1;
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall4(ro = 1500, cp = 920, lambda = 0.2, s = sc, A = Ac, n = nc);
  Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous layeredWall5(ro = 1500, cp = 920, lambda = 0.2, s = sf, A = Af, n = nf);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca2(gamma = 3.5, S = Af);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca4(gamma = 3.5, S = Ac);
  Interfaces.Thermal.HeatPort Ceiling;
  Interfaces.Thermal.HeatPort Floor;
  parameter SI.Area A13 = 15 "wall1 and wall3 surface";
  parameter SI.Area A24 = 15 "wall2 and wall4 surface";
  parameter SI.Area Af = 25 "floor surface";
  parameter SI.Area Ac = 25 "ceiling surface";
  Modelica.Blocks.Interfaces.RealOutput Troom;
  parameter SI.Volume Vroom = 5 * 5 * 3 "volume";
  parameter SI.Length s1 = 0.2 "wall_1 thickness";
  parameter SI.Length s2 = 0.2 "wall_2 thickness";
  parameter SI.Length s3 = 0.2 "wall_3 thickness";
  parameter SI.Length s4 = 0.2 "wall_4 thickness";
  parameter SI.Length sf = 0.2 "floor thickness";
  parameter SI.Length sc = 0.2 "ceiling thickness";
  parameter Integer n1 = 2 "number of layers of wall_1";
  parameter Integer n2 = 2 "number of layers of wall_2";
  parameter Integer n3 = 2 "number of layers of wall1_3";
  parameter Integer n4 = 2 "number of layers of wall1_4";
  parameter Integer nf = 2 "number of layers of floor";
  parameter Integer nc = 2 "number of layers of ceiling";
equation
  connect(airVolume.air_flange2, airSink_P_fixed.air_flange);
  connect(convSca2Sca6.ss2, layeredWall.side2);
  connect(layeredWall1.side1, convSca2Sca5.ss1);
  connect(layeredWall3.side1, convSca2Sca3.ss1);
  connect(convSca2Sca1.ss2, layeredWall2.side2);
  connect(convSca2Sca5.ss2, airVolume.heatPort);
  connect(convSca2Sca1.ss1, airVolume.heatPort);
  connect(convSca2Sca6.ss1, airVolume.heatPort);
  connect(convSca2Sca3.ss2, airVolume.heatPort);
  connect(gain.y, heatPump_ConstantCOPcool.cmd01);
  connect(prescribedHeatFlow.port, airVolume.heatPort);
  connect(temperatureSensor4.T, FromKelvin1.Kelvin);
  connect(prescribedHeatFlow.Q_flow, Noise);
  connect(temperatureSensor4.port, airVolume.heatPort);
  connect(layeredWall1.side2, Wall3);
  connect(Wall2, layeredWall2.side1);
  connect(layeredWall1.side1, Wall1);
  connect(layeredWall3.side2, Wall4);
  connect(layeredWall4.side1, Ceiling);
  connect(layeredWall5.side1, Floor);
  connect(layeredWall5.side2, convSca2Sca2.ss2);
  connect(layeredWall4.side2, convSca2Sca4.ss2);
  connect(convSca2Sca4.ss1, airVolume.heatPort);
  connect(convSca2Sca2.ss1, airVolume.heatPort);
  connect(Troom, FromKelvin1.Celsius);
  connect(heatPump_ConstantCOPcool.hotPort, Texternal);
  connect(heatPump_ConstantCOPcool.coldPort, airVolume.heatPort);
  connect(gain.u, Command_Heater);
end Room_without_PI;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.BaseClasses.IMC_Tuning.IMC_Tuning



Information

Extends from Icons.CaseStudyModel.

Modelica definition

```
model IMC_Tuning
  extends Icons.CaseStudyModel;
  Modelica.Blocks.Sources.Constant Noise(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature;
  Modelica.Blocks.Sources.Constant Text(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature1;
  Modelica.Blocks.Sources.Constant Tinside(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca4(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca5(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca3(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca11(S = 15);
```

```
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca12(S = 15);
  Modelica.Blocks.Sources.Constant Noise1(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature2;
  Modelica.Blocks.Sources.Constant Text1(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater1(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom1(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature3;
  Modelica.Blocks.Sources.Constant Tinside1(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca2(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca6(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca7(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca8(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca9(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca10(S = 15);
  Modelica.Blocks.Sources.Constant Noise2(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature4;
  Modelica.Blocks.Sources.Constant Text2(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater2(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI Corridor(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4, A24 = 20 * 3, Af = 20 * 5, Ac = 20 * 5, Vroom = 20 * 5 * 3);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature5;
  Modelica.Blocks.Sources.Constant Tinside2(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca13(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca14(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca15(gamma = 3.5, S = 20 * 3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca16(S = 15, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca17(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca18(gamma = 3.5, S = 20 * 3);
  Modelica.Blocks.Sources.Constant Noise3(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature6;
  Modelica.Blocks.Sources.Constant Text3(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater3(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom2(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature7;
  Modelica.Blocks.Sources.Constant Tinside3(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca19(S = 25, gamma = 0.31);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca20(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca21(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca22(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca23(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca24(S = 15);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature8;
  Modelica.Blocks.Sources.Constant Tground(k = 273.15 + 12);
  Modelica.Blocks.Sources.Constant Noise4(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature9;
  Modelica.Blocks.Sources.Constant Text4(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater4(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom3(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature10;
  Modelica.Blocks.Sources.Constant Tinside4(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca25(S = 25, gamma = 0.31);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca26(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca27(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca28(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca29(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca30(S = 15);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature11;
  Modelica.Blocks.Sources.Constant Tground1(k = 273.15 + 12);
  Modelica.Blocks.Sources.Constant Noise5(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature12;
  Modelica.Blocks.Sources.Constant Text5(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater5(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI Corridor1(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4, A24 = 20 * 3, Af = 20 * 5, Ac = 20 * 5, Vroom = 20 * 5 * 3);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature13;
  Modelica.Blocks.Sources.Constant Tinside5(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca31(S = 100, gamma = 0.31);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca32(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca33(gamma = 3.5, S = 20 * 3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca34(S = 15, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca35(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca36(gamma = 3.5, S = 20 * 3);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature14;
  Modelica.Blocks.Sources.Constant Tground2(k = 273.15 + 12);
  Modelica.Blocks.Sources.Constant Noise6(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature15;
  Modelica.Blocks.Sources.Constant Text6(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater6(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom4(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature16;
  Modelica.Blocks.Sources.Constant Tinside6(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca37(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca38(S = 25, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca39(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca40(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca41(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca42(S = 15);
  Modelica.Blocks.Sources.Constant Noise7(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature17;
  Modelica.Blocks.Sources.Constant Text7(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater7(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI rooom5(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature18;
  Modelica.Blocks.Sources.Constant Tinside7(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca43(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca44(S = 25, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca45(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca46(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca47(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca48(S = 15);
  Modelica.Blocks.Sources.Constant Noise8(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature19;
  Modelica.Blocks.Sources.Constant Text8(k = 273.15 + 32);
  Modelica.Blocks.Sources.Step CommandHeater8(offset = 0.1, height = 0.01, startTime = 1e7);
  Room_without_PI Corridor2(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4, A24 = 20 * 3, Af = 20 * 5, Ac = 20 * 5, Vroom = 20 * 5 * 3);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature20;
  Modelica.Blocks.Sources.Constant Tinside8(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca49(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca50(gamma = 5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca51(gamma = 3.5, S = 20 * 3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca52(S = 15, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca53(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca54(gamma = 3.5, S = 20 * 3);
equation
  connect(fixedTemperature.T, Text.y);
  connect(Noise.y, rooom.Noise);
  connect(rooom.Texternal, fixedTemperature.port);
  connect(fixedTemperature1.T, Tinside.y);
  connect(CommandHeater.y, rooom.Command_Heater);
  connect(rooom.Floor, convSca2Sca4.ss1);
  connect(rooom.Ceiling, convSca2Sca5.ss1);
  connect(rooom.Wall1, convSca2Sca11.ss2);
  connect(convSca2Sca11.ss1, fixedTemperature.port);
```

```
connect(convSca2Sca12.ss1, fixedTemperature.port);
connect(convSca2Sca12.ss2, rooom.Wall2);
connect(convSca2Sca1.ss1, rooom.Wall3);
connect(rooom.Wall4, convSca2Sca3.ss1);
connect(convSca2Sca3.ss2, fixedTemperature1.port);
connect(convSca2Sca5.ss2, fixedTemperature1.port);
connect(convSca2Sca4.ss2, fixedTemperature1.port);
connect(convSca2Sca1.ss2, fixedTemperature1.port);
connect(fixedTemperature2.T, Text1.y);
connect(Noise1.y, rooom1.Noise);
connect(rooom1.Texternal, fixedTemperature2.port);
connect(fixedTemperature3.T, Tinside1.y);
connect(CommandHeater1.y, rooom1.Command_Heater);
connect(rooom1.Floor, convSca2Sca2.ss1);
connect(rooom1.Ceiling, convSca2Sca6.ss1);
connect(rooom1.Wall1, convSca2Sca9.ss2);
connect(convSca2Sca10.ss1, fixedTemperature2.port);
connect(convSca2Sca10.ss2, rooom1.Wall2);
connect(convSca2Sca8.ss1, rooom1.Wall3);
connect(rooom1.Wall4, convSca2Sca7.ss1);
connect(convSca2Sca7.ss2, fixedTemperature3.port);
connect(convSca2Sca6.ss2, fixedTemperature3.port);
connect(convSca2Sca2.ss2, fixedTemperature3.port);
connect(convSca2Sca8.ss2, fixedTemperature3.port);
connect(convSca2Sca9.ss1, fixedTemperature3.port);
connect(fixedTemperature4.T, Text2.y);
connect(Noise2.y, Corridor.Noise);
connect(Corridor.Texternal, fixedTemperature4.port);
connect(fixedTemperature5.T, Tinside2.y);
connect(CommandHeater2.y, Corridor.Command_Heater);
connect(Corridor.Floor, convSca2Sca13.ss1);
connect(Corridor.Ceiling, convSca2Sca14.ss1);
connect(Corridor.Wall1, convSca2Sca17.ss2);
connect(convSca2Sca17.ss1, fixedTemperature4.port);
connect(convSca2Sca18.ss2, Corridor.Wall2);
connect(convSca2Sca16.ss1, Corridor.Wall3);
connect(Corridor.Wall4, convSca2Sca15.ss1);
connect(convSca2Sca15.ss2, fixedTemperature5.port);
connect(convSca2Sca14.ss2, fixedTemperature5.port);
connect(convSca2Sca13.ss2, fixedTemperature5.port);
connect(convSca2Sca16.ss2, fixedTemperature4.port);
connect(convSca2Sca18.ss1, fixedTemperature5.port);
connect(fixedTemperature6.T, Text3.y);
connect(Noise3.y, rooom2.Noise);
connect(rooom2.Texternal, fixedTemperature6.port);
connect(fixedTemperature7.T, Tinside3.y);
connect(CommandHeater3.y, rooom2.Command_Heater);
connect(rooom2.Floor, convSca2Sca19.ss1);
connect(rooom2.Ceiling, convSca2Sca20.ss1);
connect(rooom2.Wall1, convSca2Sca23.ss2);
connect(convSca2Sca23.ss1, fixedTemperature6.port);
connect(convSca2Sca24.ss1, fixedTemperature6.port);
connect(convSca2Sca24.ss2, rooom2.Wall2);
connect(convSca2Sca22.ss1, rooom2.Wall3);
connect(rooom2.Wall4, convSca2Sca21.ss1);
connect(convSca2Sca21.ss2, fixedTemperature7.port);
connect(convSca2Sca22.ss2, fixedTemperature7.port);
connect(fixedTemperature8.T, Tground.y);
connect(convSca2Sca19.ss2, fixedTemperature8.port);
connect(convSca2Sca20.ss2, fixedTemperature7.port);
connect(fixedTemperature9.T, Text4.y);
connect(Noise4.y, rooom3.Noise);
connect(rooom3.Texternal, fixedTemperature9.port);
connect(fixedTemperature10.T, Tinside4.y);
connect(CommandHeater4.y, rooom3.Command_Heater);
connect(rooom3.Floor, convSca2Sca25.ss1);
connect(rooom3.Ceiling, convSca2Sca26.ss1);
connect(rooom3.Wall1, convSca2Sca29.ss2);
connect(convSca2Sca30.ss1, fixedTemperature9.port);
connect(convSca2Sca30.ss2, rooom3.Wall2);
connect(convSca2Sca28.ss1, rooom3.Wall3);
connect(rooom3.Wall4, convSca2Sca27.ss1);
connect(convSca2Sca27.ss2, fixedTemperature10.port);
connect(convSca2Sca26.ss2, fixedTemperature10.port);
connect(convSca2Sca28.ss2, fixedTemperature10.port);
connect(convSca2Sca29.ss1, fixedTemperature10.port);
connect(fixedTemperature11.T, Tground1.y);
connect(convSca2Sca25.ss2, fixedTemperature11.port);
connect(fixedTemperature12.T, Text5.y);
connect(Noise5.y, Corridor1.Noise);
connect(Corridor1.Texternal, fixedTemperature12.port);
connect(fixedTemperature13.T, Tinside5.y);
connect(CommandHeater5.y, Corridor1.Command_Heater);
connect(Corridor1.Floor, convSca2Sca31.ss1);
connect(Corridor1.Ceiling, convSca2Sca32.ss1);
connect(Corridor1.Wall1, convSca2Sca35.ss2);
connect(convSca2Sca35.ss1, fixedTemperature12.port);
connect(convSca2Sca36.ss2, Corridor1.Wall2);
connect(convSca2Sca34.ss1, Corridor1.Wall3);
connect(Corridor1.Wall4, convSca2Sca33.ss1);
connect(convSca2Sca33.ss2, fixedTemperature13.port);
connect(convSca2Sca32.ss2, fixedTemperature13.port);
connect(convSca2Sca34.ss2, fixedTemperature12.port);
connect(convSca2Sca36.ss1, fixedTemperature13.port);
connect(fixedTemperature14.T, Tground2.y);
connect(convSca2Sca31.ss2, fixedTemperature14.port);
connect(fixedTemperature15.T, Text6.y);
connect(Noise6.y, rooom4.Noise);
connect(rooom4.Texternal, fixedTemperature15.port);
connect(fixedTemperature16.T, Tinside6.y);
connect(CommandHeater6.y, rooom4.Command_Heater);
connect(rooom4.Floor, convSca2Sca37.ss1);
connect(rooom4.Ceiling, convSca2Sca38.ss1);
connect(rooom4.Wall1, convSca2Sca41.ss2);
connect(convSca2Sca41.ss1, fixedTemperature15.port);
connect(convSca2Sca42.ss1, fixedTemperature15.port);
connect(convSca2Sca42.ss2, rooom4.Wall2);
connect(convSca2Sca40.ss1, rooom4.Wall3);
connect(rooom4.Wall4, convSca2Sca39.ss1);
connect(convSca2Sca39.ss2, fixedTemperature16.port);
connect(convSca2Sca37.ss2, fixedTemperature16.port);
connect(convSca2Sca40.ss2, fixedTemperature16.port);
connect(fixedTemperature17.T, Text7.y);
connect(Noise7.y, rooom5.Noise);
connect(rooom5.Texternal, fixedTemperature17.port);
connect(fixedTemperature18.T, Tinside7.y);
connect(CommandHeater7.y, rooom5.Command_Heater);
connect(rooom5.Floor, convSca2Sca43.ss1);
connect(rooom5.Ceiling, convSca2Sca44.ss1);
connect(rooom5.Wall1, convSca2Sca47.ss2);
connect(convSca2Sca48.ss1, fixedTemperature17.port);
```

```
  connect(convSca2Sca48.ss2, rooom5.Wall2);
  connect(convSca2Sca46.ss1, rooom5.Wall3);
  connect(rooom5.Wall4, convSca2Sca45.ss1);
  connect(convSca2Sca45.ss2, fixedTemperature18.port);
  connect(convSca2Sca43.ss2, fixedTemperature18.port);
  connect(convSca2Sca46.ss2, fixedTemperature18.port);
  connect(convSca2Sca47.ss1, fixedTemperature18.port);
  connect(fixedTemperature19.T, Text8.y);
  connect(Noise8.y, Corridor2.Noise);
  connect(Corridor2.Texternal, fixedTemperature19.port);
  connect(fixedTemperature20.T, Tinside8.y);
  connect(CommandHeater8.y, Corridor2.Command_Heater);
  connect(Corridor2.Floor, convSca2Sca49.ss1);
  connect(Corridor2.Ceiling, convSca2Sca50.ss1);
  connect(Corridor2.Wall1, convSca2Sca53.ss2);
  connect(convSca2Sca53.ss1, fixedTemperature19.port);
  connect(convSca2Sca54.ss2, Corridor2.Wall2);
  connect(convSca2Sca52.ss1, Corridor2.Wall3);
  connect(Corridor2.Wall4, convSca2Sca51.ss1);
  connect(convSca2Sca51.ss2, fixedTemperature20.port);
  connect(convSca2Sca49.ss2, fixedTemperature20.port);
  connect(convSca2Sca52.ss2, fixedTemperature19.port);
  connect(convSca2Sca54.ss1, fixedTemperature20.port);
  connect(convSca2Sca38.ss2, fixedTemperature15.port);
  connect(convSca2Sca44.ss2, fixedTemperature17.port);
  connect(convSca2Sca50.ss2, fixedTemperature19.port);
end IMC_Tuning;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.Test

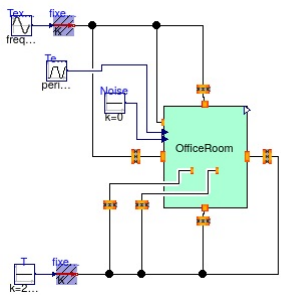
Information

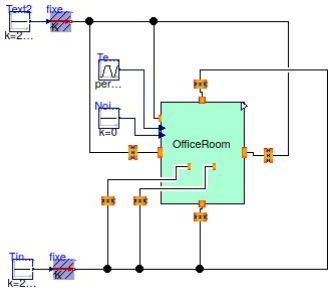
Extends from Modelica.Icons.Package (Icon for standard packages), Icons.TestModel.

Package Content

Name	Description
test_rooms	
test_floor	

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.Test.test_rooms





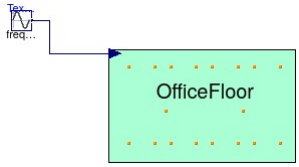
Information

Extends from Icons.TestModel.

Modelica definition

```
model test_rooms
  extends Icons.TestModel;
  Modelica.Blocks.Sources.Constant Noise(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature;
  BaseClasses.OfficeRoom rooom(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4, Ti = 140, K = -3.5);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature1;
  Modelica.Blocks.Sources.Constant T(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca4(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca5(gamma = 3.5, S = 25);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca3(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca1(S = 15, gamma = 3.5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca11(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca12(S = 15);
  Modelica.Blocks.Sources.Trapezoid Temp1(amplitude = -1, width(displayUnit = "h") = 43200, startTime = 86400, rising(displayUnit = "h") = 7200, falling(displayUnit = "h") = 7200, peri
  Modelica.Blocks.Sources.Sine Texternal(freqHz = 1 / 86400, phase = -pi / 2, amplitude = 4, offset = 273.15 + 36);
  Modelica.Blocks.Sources.Constant Noise2(k = 0) "Noise due to People and PC inside the room";
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature4;
  Modelica.Blocks.Sources.Constant Text2(k = 273.15 + 36);
  BaseClasses.OfficeRoom Corridor(n1 = 4, n2 = 4, n3 = 4, n4 = 4, nf = 4, nc = 4, A24 = 20 * 3, Af = 20 * 5, Ac = 20 * 5, Vroom = 20 * 5 * 3, K = -11.2);
  Modelica.Thermal.HeatTransfer.Sources.PrescribedTemperature fixedTemperature5;
  Modelica.Blocks.Sources.Constant Tinside2(k = 273.15 + 26);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca13(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca14(gamma = 3.5, S = 100);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca15(gamma = 3.5, S = 20 * 3);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca16(S = 15, gamma = 5);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca17(S = 15);
  Components.BaseComponents.Thermal.HeatTransfer.Convection_SS convSca2Sca18(gamma = 3.5, S = 20 * 3);
  Modelica.Blocks.Sources.Trapezoid Temp2(amplitude = -1, width(displayUnit = "h") = 43200, startTime = 86400, rising(displayUnit = "h") = 7200, falling(displayUnit = "h") = 7200, peri
equation
  connect(Noise.y, rooom.Noise);
  connect(rooom.Texternal, fixedTemperature.port);
  connect(fixedTemperature1.T, T.y);
  connect(rooom.Floor, convSca2Sca4.ss1);
  connect(rooom.Ceiling, convSca2Sca5.ss1);
  connect(rooom.Wall1, convSca2Sca11.ss2);
  connect(convSca2Sca11.ss1, fixedTemperature.port);
  connect(convSca2Sca12.ss1, fixedTemperature.port);
  connect(convSca2Sca12.ss2, rooom.Wall2);
  connect(convSca2Sca1.ss1, rooom.Wall3);
  connect(rooom.Wall4, convSca2Sca3.ss1);
  connect(convSca2Sca3.ss2, fixedTemperature1.port);
  connect(convSca2Sca5.ss2, fixedTemperature1.port);
  connect(convSca2Sca4.ss2, fixedTemperature1.port);
  connect(convSca2Sca1.ss2, fixedTemperature1.port);
  connect(Temp1.y, rooom.DesiredTemperature);
  connect(fixedTemperature.T, Texternal.y);
  connect(fixedTemperature4.T, Text2.y);
  connect(Noise2.y, Corridor.Noise);
  connect(Corridor.Texternal, fixedTemperature4.port);
  connect(fixedTemperature5.T, Tinside2.y);
  connect(Corridor.Floor, convSca2Sca13.ss1);
  connect(Corridor.Ceiling, convSca2Sca14.ss1);
  connect(Corridor.Wall1, convSca2Sca17.ss2);
  connect(convSca2Sca17.ss1, fixedTemperature4.port);
  connect(convSca2Sca18.ss2, Corridor.Wall2);
  connect(convSca2Sca16.ss1, Corridor.Wall3);
  connect(Corridor.Wall4, convSca2Sca15.ss1);
  connect(convSca2Sca15.ss2, fixedTemperature5.port);
  connect(convSca2Sca14.ss2, fixedTemperature5.port);
  connect(convSca2Sca13.ss2, fixedTemperature5.port);
  connect(convSca2Sca16.ss2, fixedTemperature4.port);
  connect(convSca2Sca18.ss1, fixedTemperature5.port);
  connect(Temp2.y, Corridor.DesiredTemperature);
end test_rooms;
```

EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.OfficeBuilding.Test.test_floor



Information

Extends from Icons.TestModel.

Modelica definition

```
model test_floor
  extends Icons.TestModel;
  BaseClasses.OfficeFloor officeFloor;
  Modelica.Blocks.Sources.Sine Texternal1(freqHz = 1 / 86400, phase = -pi / 2, offset = 273.15 + 36, amplitude = 1);
equation
  connect(Texternal1.y, officeFloor.Texternal);
end test_floor;
```

EEB.CaseStudies.DEIB.BuildingLevel.Test

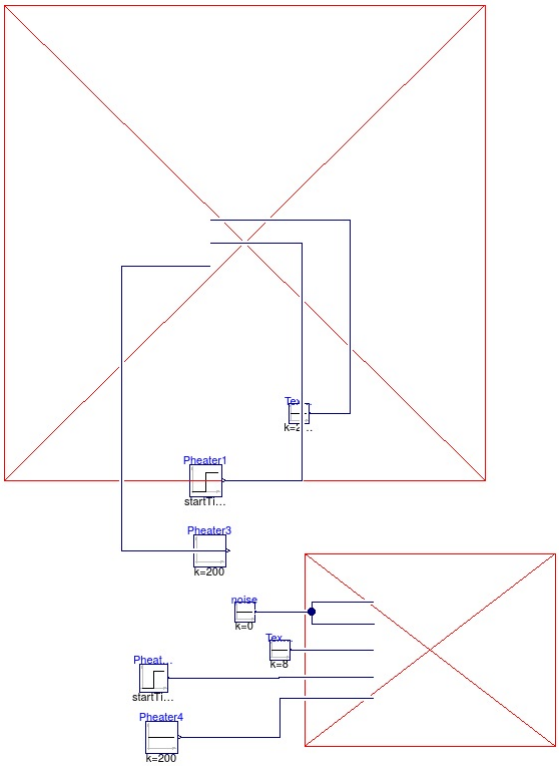
Information

Extends from Modelica.Icons.Package (Icon for standard packages), Icons.TestModel.

Package Content

Name	Description
test_1	
test_2	

EEB.CaseStudies.DEIB.BuildingLevel.Test.test_1



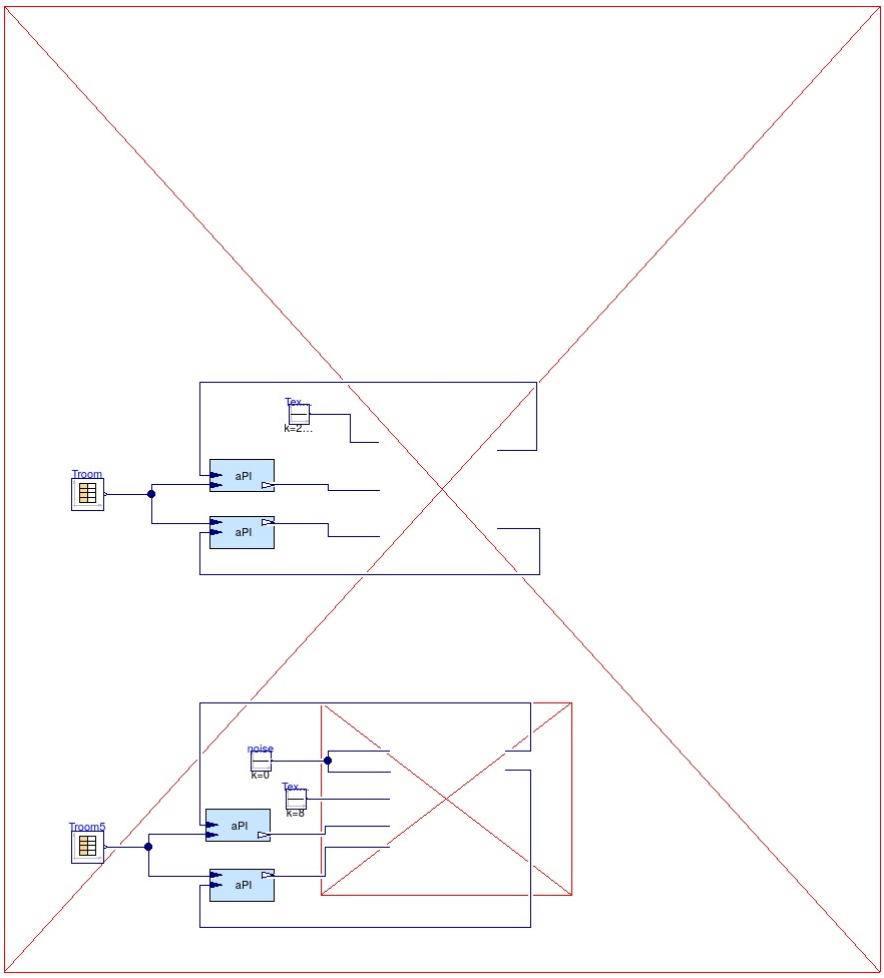
Information

Extends from Icons.TestModel.

Modelica definition

```
model test_1
  extends Icons.TestModel;
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  ElectricEquivalentStudy.Rooms.RoomsElectrical RoomsElectric(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.Constant Texternal1(k = 273.15 + 8);
  Modelica.Blocks.Sources.Step Pheater1(height = 100, startTime = 6e6, offset = 300);
  Modelica.Blocks.Sources.Step Pheater2(height = 100, offset = 300, startTime = 6e6);
  Modelica.Blocks.Sources.Constant Pheater3(k = 200);
  Modelica.Blocks.Sources.Constant Pheater4(k = 200);
  EEB.CaseStudies.DEIB.BuildingLevel.ThermalManagementStudy.Rooms.RoomsThermalSimplified
                                                                    RoomsThermal(Tstart = 273.15 + 0, wr = 0);
equation
  connect(Pheater3.y, RoomsThermal.heater2);
  connect(Pheater1.y, RoomsThermal.heater1);
  connect(Texternal1.y, RoomsThermal.Texternal);
  connect(Texternal.y, RoomsElectric.Texternal);
  connect(noise.y, RoomsElectric.Room1Noise);
  connect(RoomsElectric.Room2Noise, RoomsElectric.Room1Noise);
  connect(Pheater2.y, RoomsElectric.heater1);
  connect(Pheater4.y, RoomsElectric.heater2);
end test_1;
```

EEB.CaseStudies.DEIB.BuildingLevel.Test.test_2



Information

Extends from Icons.TestModel.

Modelica definition

```
model test_2
  extends Icons.TestModel;
  Modelica.Blocks.Sources.Constant Texternal(k = 8);
  Modelica.Blocks.Sources.Constant noise(k = 0);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue(CSmax = 1000, Ti = 5682, K = 27450);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue1(CSmax = 1000, Ti = 8268, K = 38816);
  ElectricEquivalentStudy.Rooms.RoomsElectrical RoomsElectric(Air1(v(start = 0)), Air2(v(start = 0)));
  Modelica.Blocks.Sources.Constant Texternal1(k = 273.15 + 8);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue2(Ti = 5682, K = 27450, CSmax = 1000);
  Controllers.Blocks.Analogue.AWPI_1dof aWPI_analogue3(Ti = 8268, K = 38816, CSmax = 1000);
  ThermalManagementStudy.Rooms.RoomsThermalSimplified RoomsThermalSimplified(Tstart = 273.15 + 0, wr = 0);
  Modelica.Blocks.Sources.CombiTimeTable Troom(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 18
  Modelica.Blocks.Sources.CombiTimeTable Troom5(columns = {2}, extrapolation = Modelica.Blocks.Types.Extrapolation.Periodic, table = [0, 20; 5 * 3600, 20; 8 * 3600, 25; 17 * 3600, 25; 1
equation
  connect(Texternal.y, RoomsElectric.Texternal);
  connect(noise.y, RoomsElectric.Room1Noise);
  connect(RoomsElectric.Room2Noise, RoomsElectric.Room1Noise);
  connect(aWPI_analogue.CS, RoomsElectric.heater1);
  connect(aWPI_analogue1.CS, RoomsElectric.heater2);
  connect(RoomsElectric.Troom2, aWPI_analogue1.PV);
  connect(RoomsElectric.Troom1, aWPI_analogue.PV);
  connect(Troom.y[1], aWPI_analogue2.SP);
  connect(aWPI_analogue3.SP, aWPI_analogue2.SP);
  connect(Troom5.y[1], aWPI_analogue.SP);
  connect(aWPI_analogue1.SP, aWPI_analogue.SP);
  connect(Texternal1.y, RoomsThermalSimplified.Texternal);
  connect(RoomsThermalSimplified.Troom1, aWPI_analogue2.PV);
  connect(RoomsThermalSimplified.Troom2, aWPI_analogue3.PV);
  connect(RoomsThermalSimplified.heater1, aWPI_analogue2.CS);
  connect(RoomsThermalSimplified.heater2, aWPI_analogue3.CS);
end test_2;
```

EEB.InternalTests

Information

Extends from Modelica.Icons.InternalPackage (Icon for an internal package (indicating that the package should not be directly utilized by user)).

Package Content

Name	Description
ExampleMoistAir2	
LabForSeminars	

EEB.InternalTests.ExampleMoistAir2

Parameters

Type	Name	Default	Description
Volume	V	2	[m3]
MassFraction	Xa	0.005	[1]
ThermalConductance	G	50	[W/K]

Modelica definition

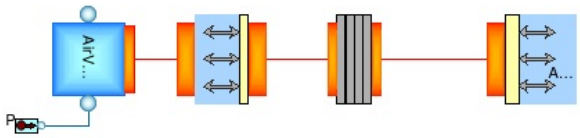
```
model ExampleMoistAir2
  Media.Substances.MoistAir air;
  parameter Modelica.SIunits.Volume V=2;
  parameter Modelica.SIunits.MassFraction Xa=0.005;
  parameter Modelica.SIunits.ThermalConductance G=50;
  Modelica.SIunits.Mass M;
  Modelica.SIunits.Power Q;
  Modelica.SIunits.Energy E;
  Modelica.SIunits.Temperature Ta(start=273.15+20);
  Modelica.SIunits.Pressure pa(start=100000);
  Modelica.SIunits.Temperature Te;
equation

  E = M*(air.h-pa/air.d);
  M = V*air.d;
  der(M) = 0;
  der(E) = Q - G*(Ta-Te);

  Ta = air.T;
  pa = air.p;

  air.X = Xa;
  Te = 273.15+20;
  Q = 100;

end ExampleMoistAir2;
```

EEB.InternalTests.LabForSeminars

Package Content

Name	Description
Example_002	
Example_003	
Example_004	
Example_005	
Example_006	
Example_001	
ExampleElec1	
Example_007	

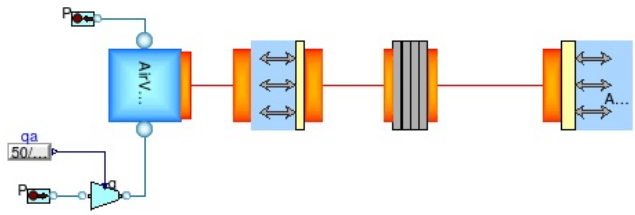
EEB.InternalTests.LabForSeminars.Example_002



Modelica definition

```
model Example_002
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource;
equation
  connect(a2w.wall, MLwall.side1);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
  connect(airSource.air_flange, air.air_flange2);
  connect(air.heatPort, a2w.air);
end Example_002;
```

EEB.InternalTests.LabForSeminars.Example_003

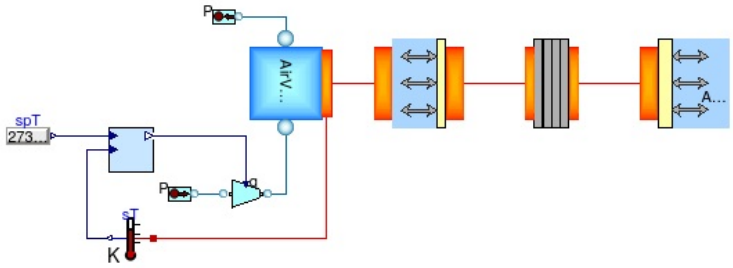


Modelica definition

```
model Example_003
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource(T0 = 301.15);
EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
Modelica.Blocks.Sources.RealExpression qa(y = 50 / 3600);
equation
  connect(qa.y, fan.iq);
  connect(fan.air_flange2, air.air_flange2);
```

```
  connect(airSource.air_flange, fan.air_flange1);
  connect(airSink.air_flange, air.air_flange1);
  connect(air.heatPort, a2w.air);
  connect(a2w.wall, MLwall.side1);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
end Example_003;
```

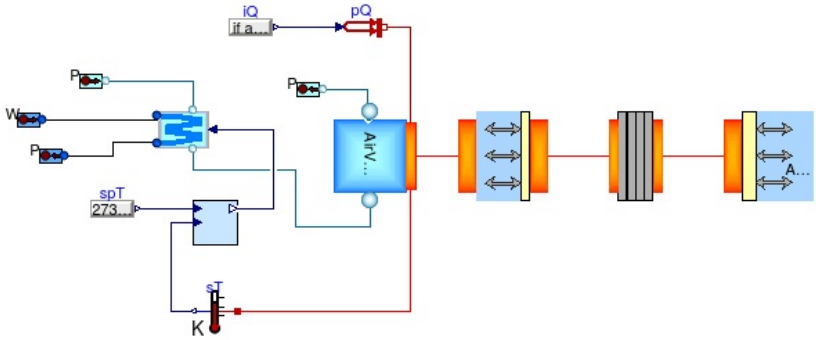
EEB.InternalTests.LabForSeminars.Example_004



Modelica definition

```
model Example_004
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource(T0 = 301.15);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression spT(y = 273.25 + 25);
EEB.Controllers.Blocks.Analogue.AWPI_1dof PI_T(CSmax = 300 / 3600, CSmin = 0, K = 10, Ti = 100);
Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sT;
equation
  connect(sT.T, PI_T.PV);
  connect(air.heatPort, sT.port);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
  connect(PI_T.CS, fan.iq);
  connect(spT.y, PI_T.SP);
  connect(fan.air_flange2, air.air_flange2);
  connect(airSource.air_flange, fan.air_flange1);
  connect(airSink.air_flange, air.air_flange1);
  connect(air.heatPort, a2w.air);
  connect(a2w.wall, MLwall.side1);
end Example_004;
```

EEB.InternalTests.LabForSeminars.Example_005

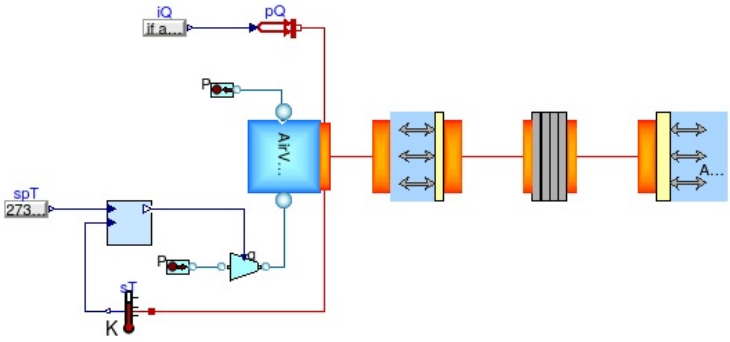


Modelica definition

```
model Example_005
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
```

```
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource(T0 = 301.15);
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression spT(y = 273.25 + 25);
  EEB.Controllers.Blocks.Analogue.AWPI_1dof PI_T(CSmax = 1, CSmin = 0, K = 10, Ti = 100);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sT;
Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pQ;
Modelica.Blocks.Sources.RealExpression iQ(y = if ambient_settings.hod > 8 and ambient_settings.hod < 17 then 50 else 0);
EEB.Components.AggregateComponents.Heating.FanCoil fanCoil1;
Components.BaseComponents.Water.Sources.WaterSource_WT_fixed wsrc(W0 = 1);
Components.BaseComponents.Water.Sinks.WaterSink_P_fixed wsnk;
equation
  connect(fanCoil1.air_flange1, air.air_flange2);
  connect(fanCoil1.icmd01, PI_T.CS);
  connect(wsnk.water_flange, fanCoil1.water_flange2);
  connect(wsrc.water_flange, fanCoil1.water_flange1);
  connect(airSource.air_flange, fanCoil1.air_flange2);
  connect(iQ.y, pQ.Q_flow);
  connect(pQ.port, air.heatPort);
  connect(air.heatPort, sT.port);
  connect(sT.T, PI_T.PV);
  connect(spT.y, PI_T.SP);
  connect(airSink.air_flange, air.air_flange1);
  connect(a2w.wall, MLwall.side1);
  connect(air.heatPort, a2w.air);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
end Example_005;
```

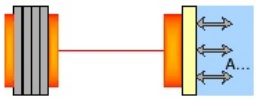
EEB.InternalTests.LabForSeminars.Example_006



Modelica definition

```
model Example_006
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                        convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource(T0 = 301.15);
  EEB.Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression spT(y = 273.25 + 25);
  EEB.Controllers.Blocks.Analogue.AWPI_1dof PI_T(CSmax = 300 / 3600, CSmin = 0, K = 10, Ti = 100);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sT;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pQ;
  Modelica.Blocks.Sources.RealExpression iQ(y = if ambient_settings.hod > 8 and ambient_settings.hod < 17 then 50 else 0);
equation
  connect(iQ.y, pQ.Q_flow);
  connect(pQ.port, air.heatPort);
  connect(air.heatPort, sT.port);
  connect(sT.T, PI_T.PV);
  connect(spT.y, PI_T.SP);
  connect(PI_T.CS, fan.iq);
  connect(airSink.air_flange, air.air_flange1);
  connect(airSource.air_flange, fan.air_flange1);
  connect(fan.air_flange2, air.air_flange2);
  connect(a2w.wall, MLwall.side1);
  connect(air.heatPort, a2w.air);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
end Example_006;
```

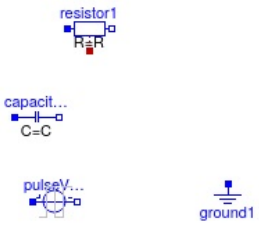
EEB.InternalTests.LabForSeminars.Example_001



Modelica definition

```
model Example_001
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15,Tstart = 298.15, n = 100, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
equation
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
end Example_001;
```

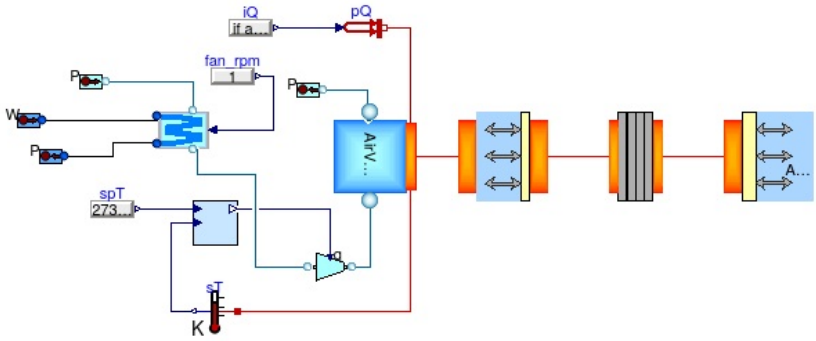
EEB.InternalTests.LabForSeminars.ExampleElec1



Modelica definition

```
model ExampleElec1
  Modelica.Electrical.Analog.Basic.Ground ground1;
  Modelica.Electrical.Analog.Basic.Capacitor capacitor1;
  Modelica.Electrical.Analog.Basic.Resistor resistor1;
  Modelica.Electrical.Analog.Sources.PulseVoltage pulseVoltage1;
end ExampleElec1;
```

EEB.InternalTests.LabForSeminars.Example_007



Modelica definition

```
model Example_007
  EEB.Components.BaseComponents.Envelope.SolidMultilayer_Homogeneous MLwall(A = 15, Tstart = 298.15, n = 10, ro(displayUnit = "kg/m3"));
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2Ext_Clarke
                                                                    convection_Wall2Ext_Clarke1;
  inner EEB.BoundaryConditions.AmbientConditions ambient_settings(Ta_avg = 293.15);
  EEB.Components.BaseComponents.Air.Volumes.AirVolume air(Tstart = 298.15, V = 100);
  EEB.Components.BaseComponents.Thermal.HeatTransfer.Convection_Wall2air_BRIS a2w;
  EEB.Components.BaseComponents.Air.Sources.AirSource_pTX_fixed airSource(T0 = 301.15);
  EEB.Components.BaseComponents.Air.Sinks.AirSink_P_fixed airSink;
  Modelica.Blocks.Sources.RealExpression spT(y = 273.25 + 25);
  EEB.Controllers.Blocks.Analogue.AWPI_1dof PI_T(CSmax = 300 / 3600, CSmin = 0, K = 10, Ti = 100);
  Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor sT;
  Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow pQ;
  Modelica.Blocks.Sources.RealExpression iQ(y = if ambient_settings.hod > 8 and ambient_settings.hod < 17 then 50 else 0);
  EEB.Components.AggregateComponents.Heating.FanCoil fanCoil1;
  Components.BaseComponents.Water.Sources.WaterSource_WT_fixed wsrc(W0 = 1);
  Components.BaseComponents.Water.Sinks.WaterSink_P_fixed wsnk;
```

```
  Modelica.Blocks.Sources.RealExpression fan_rpm(y = 1);
  Components.BaseComponents.Air.Movers.AirPrescribedFlowRate_Volume fan;
equation
  connect(PI_T.CS, fan.iq);
  connect(fan.air_flange2, air.air_flange2);
  connect(fanCoil1.air_flange1, fan.air_flange1);
  connect(fan_rpm.y, fanCoil1.icmd01);
  connect(wsnk.water_flange, fanCoil1.water_flange2);
  connect(wsrc.water_flange, fanCoil1.water_flange1);
  connect(airSource.air_flange, fanCoil1.air_flange2);
  connect(iQ.y, pQ.Q_flow);
  connect(pQ.port, air.heatPort);
  connect(air.heatPort, sT.port);
  connect(sT.T, PI_T.PV);
  connect(spT.y, PI_T.SP);
  connect(airSink.air_flange, air.air_flange1);
  connect(a2w.wall, MLwall.side1);
  connect(air.heatPort, a2w.air);
  connect(MLwall.side2, convection_Wall2Ext_Clarke1.wall);
end Example_007;
```
