The Dymola Bond Graph Library

• In this class, we shall deal with some issues relating to the construction of the *Dymola Bond Graph Library*.

• The design principles are explained, and some further features of the *Dymola* modeling framework are shown.

• We shall introduce the concept of model wrapping as implemented in the bond graph library.

• An example of an electronic circuit simulation completes the presentation.
Table of Contents

- Across and through variables
- Gyro-bonds
- Graphical bond-graph modeling
- Bond-graph connectors
- A-causal and causal bonds
- Junctions
- Element models
- Model wrapping
- Bond-graph electrical library
- Wrapped resistor model
- Bipolar junction transistor
- Inverter Circuit
Across and Through Variables

- **Dymola** offers two types of variables, the *across variables* and the *through variables*.
- In a **Dymola node**, across variables are set equal across all connections to the node, whereas through variables add up to zero.
- Consequently, if we equate *across variables* with *efforts*, and *through variables* with *flows*, **Dymola nodes** correspond exactly to the **0-junctions** of our bond graphs.
Gyro-bonds

• In my modeling book, I exploited this similarity by implementing the bonds as twisted wires (as null-modems).

• By requesting furthermore that:
  ♦ 0- and 1-junctions must always toggle. No two junctions of the same gender may be connected by a bond.
  ♦ All elements must always be attached to 0-junctions, never to 1-junctions.

both the 0-junctions and the 1-junctions can be implemented as Dymola nodes.
Gyro-bonds II

\[ \begin{array}{c}
\text{L} & 1 & \text{R} & 1 \\
\text{B} & 2 & \text{B} & 2 \\
\text{C} & 0 & \text{C} & 0 \\
\end{array} \] \Rightarrow \begin{array}{c}
\text{L} & 1 & \text{R} & 1 \\
\text{B} & 2 & \text{B} & 2 \\
\text{C} & 0 & \text{C} & 0 \\
\end{array} \] \Rightarrow \begin{array}{c}
\text{L} & 1 & \text{G} & 1 \\
\text{B} & 2 & \text{B} & 2 \\
\text{C} & 0 & \text{C} & 0 \\
\end{array} \]
Graphical Bond Graph Modeling I

• For graphical bond-graph modeling, these additional rules may, however, be too constraining.

• For example, thermal systems often exhibit 0-junctions with many bonds attached. It must be possible to split these 0-junctions into a series of separate 0-junctions connected by bonds, so that the number of bonds attached at any one junction can be kept sufficiently small.
Graphical Bond Graph Modeling II

• For this reason, the graphical bond graph modeling of *Dymola* defines both *efforts* and *flows* as *across variables*.

• Consequently, the *junctions* will have to be programmed explicitly. They can no longer be implemented as *Dymola nodes*. 
The Bond Graph Connectors I

- The directional variable, $d$, is a third across variable made available as part of the bond-graph connector, which is depicted as a grey dot.
The A-Causal Bond “Model”

- The model of a bond can now be constructed by dragging two of the bond-graph connectors into the diagram window. They are named $\text{BondCon1}$ and $\text{BondCon2}$.

Icon window

Place the text “\%name” in the icon window to get the name of the model displayed upon invocation.
The Bond Graph Connectors II

- **Dymola** variables are usually a-causal. However, they can be made causal by declaring them explicitly in a causal form.
- Two additional bond-graph connectors have been defined. The **e-connector** treats the **effort** as an **input**, and the **flow** as an **output**.

- The **f-connector** treats the **flow** as **input** and the **effort** as **output**.
The Causal Bond “Blocks”

Using these connectors, causal bond blocks can be defined.
The $f$-connector is used at the side of the causality stroke.
The $e$-connector is used at the other side.
The causal connectors are only used in the context of the bond blocks. Everywhere else, the normal bond-graph connectors are to be used.
The Junctions I

- The junctions can now be programmed. Let us look at a 0-junction with three bond attachments.

\[
\begin{align*}
    e[2] &= e[1]; \\
    e[3] &= e[2]; \\
\end{align*}
\]
The ThreePortZero partial model drags the three bond connectors into the diagram window, and packs the individual bond variables into two vectors.
The Element Models

- Let us now look at the bond-graphic element models. The bond graph capacitor may serve as an example.

Add text “\( C = \%C \)” to icon window.
Model Wrapping

- Although it is possible to model physical systems manually down to the bond graph level, this may not always be convenient.
- The bond graph interface is the lowermost graphical interface that is still fully object-oriented.
- The interface is important as it keeps the distance between the lowermost graphical layer and the equation layer as small as possible.
- Higher level graphical layers can be built easily on top of the bond graph layer for enhanced convenience.
The Bond Graph Electrical Library

- It is possible to wrap any other object-oriented graphical modeling paradigm around the bond graph methodology.
- This was done with the analog electrical library that forms part of the standard library of Modelica.
- A new analog electrical library was created as part of the bond graph library.
- In this new library, the bottom layer graphical models were wrapped around a yet lower level bond graph layer.
The Wrapped Resistor Model

The Spice-style resistor model has a thermal port carrying the heat generated by the resistor.

Icon window

The \textit{wrapper models} convert the connectors between the three domains: electrical, thermal, and bond graph.

Diagram window
The Wrapped Resistor Model II
The Wrapped Resistor Model III

```
partial model SpiceTwoPort "Two-port interface for Spice-style models"

extends Bondlib.Electrical.Analog.Interfaces.OnePort;

parameter MODELC.A.SUnits.Temperature RMW=300.15 "reference temperature";
end SpiceTwoPort;
```

```
partial model OnePort "OnePort partial model of analog electrical library"

Modelica.SIunits.Voltage v "Voltage drop between the two pins";
Modelica.SIunits.Current i "Current flowing from pin p to pin n";

equation
  v = p.v - n.v;
  i = p.i;
end OnePort;
```
The Wrapped Resistor Model IV

Diagram window

Parameter window
The Wrapped Resistor Model V

- RS
- Spice
- R=1

```
model RS "Wrapped resistor model"
   extends Interfaces.SpiceTwoPort;
   parameter Real R (unit="1/R") = 1 "Resistance at reference temperature";
   parameter Real T1 (unit="1/K") = 0 "Linear temperature coefficient";
   parameter Real T2 (unit="1/K") = 0 "Quadratic temperature coefficient";
   parameter Real A = "Relative area occupied by resistor";
   Modelica.SIunits.Resistance Rval "Resistance value at circuit temperature";
   equation
   if R > 0 then
     Rval = R + T1*DT + T2*DT^2 + T3*DT^3; else
     Rval = 0;
   end if;
   e1 = Rval*R1;
   e2 = Rval*R2;
end RS;
```
The Bipolar Junction Transistor

Icon window

Diagram window
The Bipolar Junction Transistor II
The Bipolar Junction Transistor III
The Bipolar Junction Transistor IV
The Bipolar Junction Transistor V

Spice
The Bipolar Junction Transistor VI

```
model DJS "Spice-style junction diode model for bipolar transistors"
    extends Interfaces.SpicetwoPort; 
constant Modelica.SIunits.Entropy k=Modelica.Constants.k 
    "Boltzmann’s constant"; 
constant Modelica.SIunits.ElectricCharge q=1.6021892e-19 "Electron charge"; 
constant Real GapCl=7.022e-4 "First bandgap correction factor Silicon"; 
constant Real GapC2=1108.0 "Second bandgap correction factor Silicon"; 
parameter Modelica.SIunits.Current IS=1e-15 
    "Saturation current at reference temperature"; 
parameter Modelica.SIunits.Voltage VO=1.15 
    "Energy gap for temperature effect on saturation current"; 
parameter Real N=1 "Current emission coefficient"; 
parameter Real XTI=3 "Saturation current temperature exponent"; 
parameter Real Area=1 "Relative area occupied by diode"; 
parameter Integer Level=2 
    "Transistor modeling level (Ebers-Moll = 1; Gummel-Poon = 2)"; 
parameter Real XMn=100 "if x < XMn, the exp(x) function is linearized"; 
parameter Real XMax=40 "if x > XMax, the exp(x) function is linearized"; 
Modelica.SIunits.Voltage Vt "Thermal voltage"; 
Modelica.SIunits.Current ISval "Saturation current at device temperature"; 
Modelica.SIunits.Energy E0val "Activation energy at device temperature";
protected 
    parameter Real ISMin = exp(ISval); 
    parameter Real ISMax = exp(ISMax); 
    Real ISTemp "Temperature quotient"; 
    Real et; 

equation
    /* Compute thermal voltage as function of temperature */
    Vt = k*q2/q; 
    et = el/(N*Vt);
    /* Compute temperature dependence of saturation current */
    RTemp = e2/Toma; 
    ISval = IS + GapCl*el2*e2/|e2 + GapC2|; 
    ISval = IS*exp(|RTemp - 1|*ISval/Vt + XTI*ln|RTemp|); 
    /* Compute diode characteristic */
    if Level==2 then 
        /* Gummel-Poon model */
        fl = ISval*Area*(if et < XMn then ISMin*/(et - XMn + 1) - 1 
                    else 
                    if et > XMax then ISMax*/(et - XMax + 1) - 1 
                    else exp(et - 1); 
        else 
            /* Ebers-Moll model */
            fl = ISval*(if et < XMn then ISMin*/(et - XMn + 1) - 1 
                         else 
                         if et > XMax then ISMax*/(et - XMax + 1) - 1 
                         else exp(et - 1); 
    end if;
    /* Compute heat flow */
    f2 = 0;
end DJS;
```
Inverter Circuit II

Initial number of equations

Final number of equations

Simulation Time
Simulation Results
References


- Cellier, F.E. (2007), The Dymola Bond-Graph Library, Version 2.3.