Exploiting Structural Dynamism in FHM: Modelling of Ideal Diodes

EUROSIM 2010: Session on Physical Modelling and Control

Henrik Nilsson and George Giorgidze

School of Computer Science

University of Nottingham

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:
  - Simplifies the language

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:
  - Simplifies the language
  - Facilitates efficient implementation

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:
  - Simplifies the language
  - Facilitates efficient implementation
- But this assumption is very ***limiting*** for ***hybrid modelling***; even simple systems often cannot be simulated:

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:
  - Simplifies the language
  - Facilitates efficient implementation
- But this assumption is very ***limiting*** for ***hybrid modelling***; even simple systems often cannot be simulated:
  - Breaking pendulum

The Assumption of Fixed Causality

- Current main-stream non-causal modelling and simulation languages, like Modelica, are ***designed*** and ***implemented*** assuming causality remains fixed during simulation:
  - Simplifies the language
  - Facilitates efficient implementation
- But this assumption is very ***limiting*** for ***hybrid modelling***; even simple systems often cannot be simulated:
  - Breaking pendulum
  - Ideal diodes in various configurations.

This Talk

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:
    - in particular, causality allowed to change

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:
    - in particular, causality allowed to change
    - even more drastic changes possible.

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:
    - in particular, causality allowed to change
    - even more drastic changes possible.
- Application to modelling with ideal diodes:

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:
    - in particular, causality allowed to change
    - even more drastic changes possible.
- Application to modelling with ideal diodes:
  - Half-wave rectifier with in-line inductor

This Talk

- Introduction to ***Functional Hybrid Modelling (FHM)***:
  - Novel approach to non-causal, hybrid modelling and simulation.
  - Designed and implemented ***assuming*** an evolving system:
    - in particular, causality allowed to change
    - even more drastic changes possible.
- Application to modelling with ideal diodes:
  - Half-wave rectifier with in-line inductor
  - Full-wave rectifier

FHM in a Nutshell

FHM in a Nutshell

- A ***functional approach*** to modelling and simulation of (physical) systems.

FHM in a Nutshell

- A ***functional approach*** to modelling and simulation of (physical) systems.
- Two-level design:
  - ***equational level*** for modelling components
  - ***functional level*** for spatial and temporal composition of components

FHM in a Nutshell

- A ***functional approach*** to modelling and simulation of (physical) systems.
- Two-level design:
  - ***equational level*** for modelling components
  - ***functional level*** for spatial and temporal composition of components
- ***Non-causal modelling***: undirected equations.

FHM in a Nutshell

- A ***functional approach*** to modelling and simulation of (physical) systems.
- Two-level design:
  - ***equational level*** for modelling components
  - ***functional level*** for spatial and temporal composition of components
- ***Non-causal modelling***: undirected equations.
- ***First-class models*** ($=$ systems of equations) at the functional level.

FHM in a Nutshell

- A ***functional approach*** to modelling and simulation of (physical) systems.
- Two-level design:
  - ***equational level*** for modelling components
  - ***functional level*** for spatial and temporal composition of components
- ***Non-causal modelling***: undirected equations.
- ***First-class models*** ($=$ systems of equations) at the functional level.
- Equation systems allowed to ***evolve*** over time.

Functional?

"Functional" as in ***Pure Functional Programming***:

- ***Declarative*** programming paradigm
- Programs are pure functions: ***no side effects***.
- Not just functions on "numbers": arguments and results may be arbitrary types, including:
  - ***functions***
  - ***models*** = systems of equations
- Both functions and models are thus ***first-class entities***.

Different?

Is FHM very different from current non-causal languages like Modelica?

Different?

Is FHM very different from current non-causal languages like Modelica?

Yes, in some ways, but:

Different?

Is FHM very different from current non-causal languages like Modelica?

Yes, in some ways, but:

- FHM ***implementation techniques*** could be used in the implementations of existing non-causal languages to improve their support for systems with evolving structure.

Different?

Is FHM very different from current non-causal languages like Modelica?

Yes, in some ways, but:

- FHM ***implementation techniques*** could be used in the implementations of existing non-causal languages to improve their support for systems with evolving structure.
- FHM could be viewed as a ***core language***:
  - semantics
  - compilation target

Prototype Hydra Implementation (1)

The current FHM instance is called ***Hydra***:

Prototype Hydra Implementation (1)

The current FHM instance is called ***Hydra***:

- Embedding in ***Haskell***.

Prototype Hydra Implementation (1)

The current FHM instance is called ***Hydra***:

- Embedding in ***Haskell***.
- Model ***transformed*** to form suitable for simulation, then ***JIT compiled to native code*** by an embedded compiler.

Prototype Hydra Implementation (1)

The current FHM instance is called ***Hydra***:

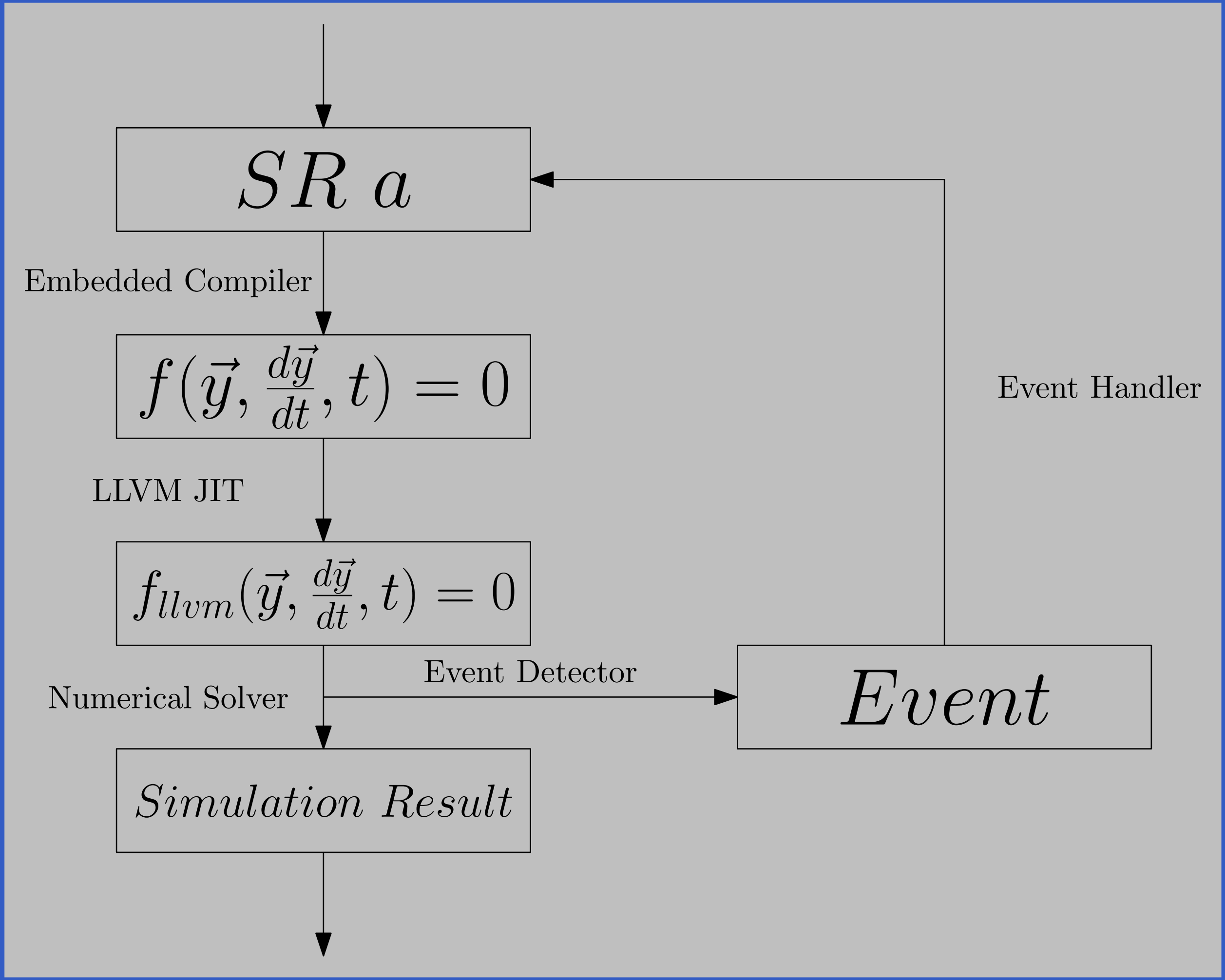
- Embedding in ***Haskell***.
- Model ***transformed*** to form suitable for simulation, then ***JIT compiled to native code*** by an embedded compiler.
- State-of-the art ***numerical solvers from SUNDIALS*** suite (from LLNL) used for simulation and event detection.

Prototype Hydra Implementation (1)

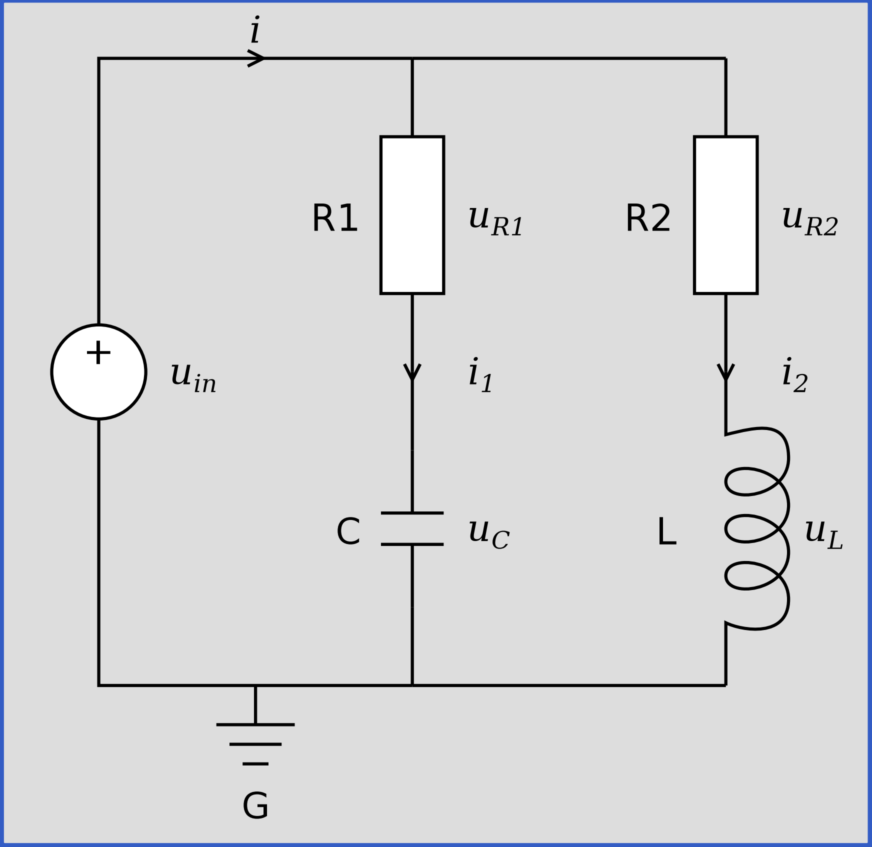
The current FHM instance is called ***Hydra***:

- Embedding in ***Haskell***.
- Model ***transformed*** to form suitable for simulation, then ***JIT compiled to native code*** by an embedded compiler.
- State-of-the art ***numerical solvers from SUNDIALS*** suite (from LLNL) used for simulation and event detection.
- Transformation and compilation ***repeated*** when system structure changes at events.

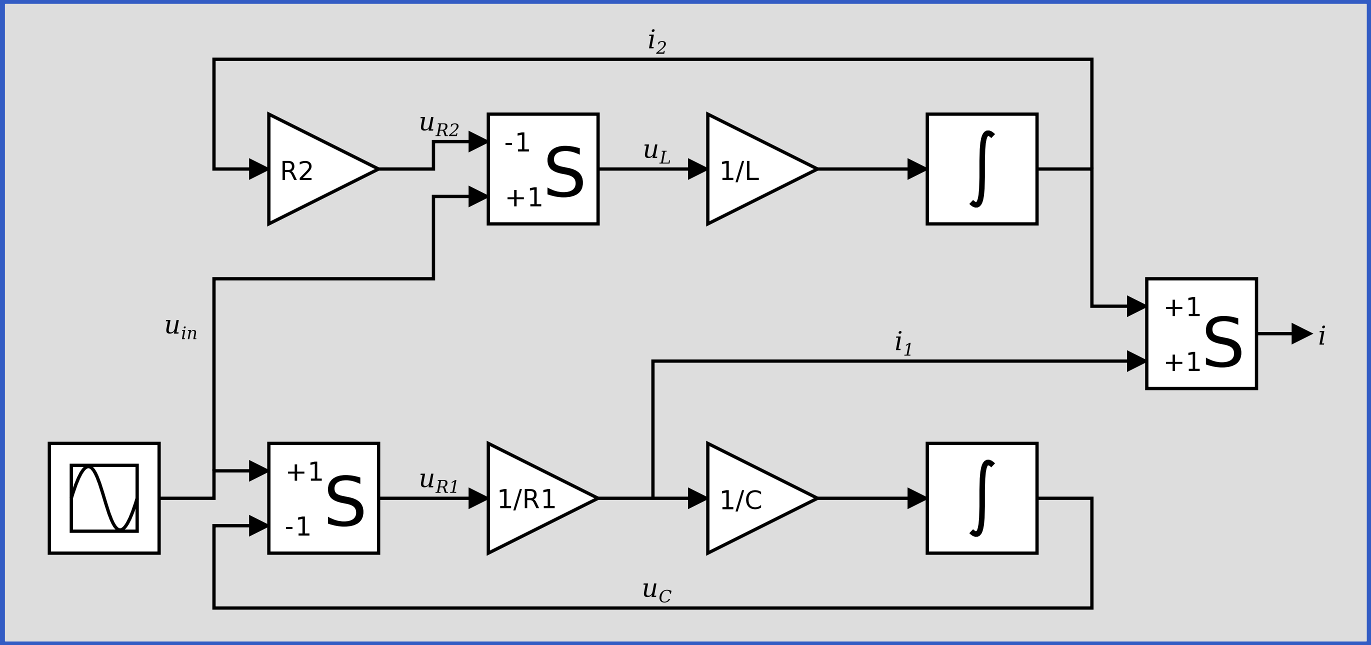
Prototype Hydra Implementation (2)



Example: A Simple Circuit



Simple Circuit: Causal Model



$$u_{R_2} = R_2 i_2$$

$$u_L = u_{in} - u_{R_2}$$

$$i_2{}' = \frac{u_L}{L}$$

$$u_{R_1} = u_{in} - u_C$$

$$i_1 = \frac{u_{R_1}}{R_1}$$

$$u_C{}' = \frac{i_1}{C}$$

$$i = i_1 + i_2$$

Simple Circuit: Non-Causal Model (1)

Non-causal resistor model:

$$\begin{aligned} v_p - v_n &= u \\ i_p + i_n &= 0 \\ R i_p &= u \end{aligned}$$

Non-causal inductor model:

$$\begin{aligned} v_p - v_n &= u \\ i_p + i_n &= 0 \\ L {i_p}' &= u \end{aligned}$$

Note the commonality: can be factored out as a separate ***two pin*** component.

Simple Circuit: Non-Causal Model (2)

A non-causal model of the entire circuit is created by ***instantiating*** the component models: copy the equations and rename the variables.

The instantiated components are then ***composed*** by adding connection equations according to Kirchhoff's laws, e.g.:

$$\begin{aligned} v_{R_1,n} &= v_{C,p} \\ i_{R_1,n} + i_{C,p} &= 0 \end{aligned}$$

Simple Circuit in FHM (1)

$$twoPin :: \textsf{SR}\ (\texttt{Pin}, \texttt{Pin}, \texttt{Voltage})$$
$$twoPin = \textbf{sigrel}\ (p, n, u)\ \textbf{where}$$
$$p.v - n.v = u$$
$$p.i + n.i = 0$$

Simple Circuit in FHM (1)

Record describing an electrical connection with fields v for voltage and i for current.

$twoPin$ :: SR (`Pin`, `Pin`, `Voltage`)

$twoPin =$ **sigrel** (p, n, u) **where**

$$p.v - n.v = u$$

$$p.i + n.i = 0$$

Simple Circuit in FHM (1)

Record describing an electrical connection with fields v for voltage and i for current.

$twoPin$ :: SR (Pin, Pin, Voltage)
$twoPin$ = **sigrel** (p, n, u) **where**

$$p.v - n.v = u$$
$$p.i + n.i = 0$$

(Partial) model represented by relation over 5 time-varying entities, i.e. signals.

Simple Circuit in FHM (1)

Record describing an electrical connection with fields v for voltage and i for current.

$twoPin$ :: SR (`Pin`, `Pin`, `Voltage`)
$twoPin$ = **sigrel** (p, n, u) **where**

$$p.v - n.v = u$$
$$p.i + n.i = 0$$

(Partial) model represented by relation over 5 time-varying entities, i.e. signals.

(Note: Somewhat idealised syntax compared with present implementation.)

Simple Circuit in FHM (2)

$$
\begin{aligned}
&resistor :: \texttt{Resistance} \rightarrow \mathsf{SR}\,(\texttt{Pin}, \texttt{Pin})\\
&resistor\ r = \textbf{sigrel}\ (p, n)\ \textbf{where}\\
&\qquad\qquad twoPin \diamond (p, n, u)\\
&\qquad\qquad r \cdot p.i = u
\end{aligned}
$$

Simple Circuit in FHM (2)

Parametrised model represented by ***function*** mapping parameters to a model. Note: first class models!

$$resistor :: \texttt{Resistance} \rightarrow \mathsf{SR}\,(\texttt{Pin}, \texttt{Pin})$$
$$resistor\ r = \textbf{sigrel}\ (p, n)\ \textbf{where}$$
$$twoPin \diamond (p, n, u)$$
$$r \cdot p.i = u$$

Simple Circuit in FHM (2)

Parametrised model represented by ***function*** mapping parameters to a model. Note: first class models!

$$resistor :: \texttt{Resistance} \rightarrow \texttt{SR}\,(\texttt{Pin}, \texttt{Pin})$$
$$resistor\ r = \textbf{sigrel}\ (p, n)\ \textbf{where}$$
$$twoPin \diamond (p, n, u)$$
$$r \cdot p.i = u$$

Signal relation ***application*** allows modular construction of models from component models.

Simple Circuit in FHM (3)

Inductors and capacitors are modelled similarly:

$$
\begin{aligned}
&\mathit{inductor} :: \texttt{Inductance} \rightarrow \texttt{SR}\,(\texttt{Pin}, \texttt{Pin})\\
&\mathit{inductor}\ l = \textbf{sigrel}\ (p, n)\ \textbf{where}\\
&\qquad\qquad \mathit{twoPin} \diamond (p, n, u)\\
&\qquad\qquad l \cdot \textbf{der}(p.i) = u
\end{aligned}
$$

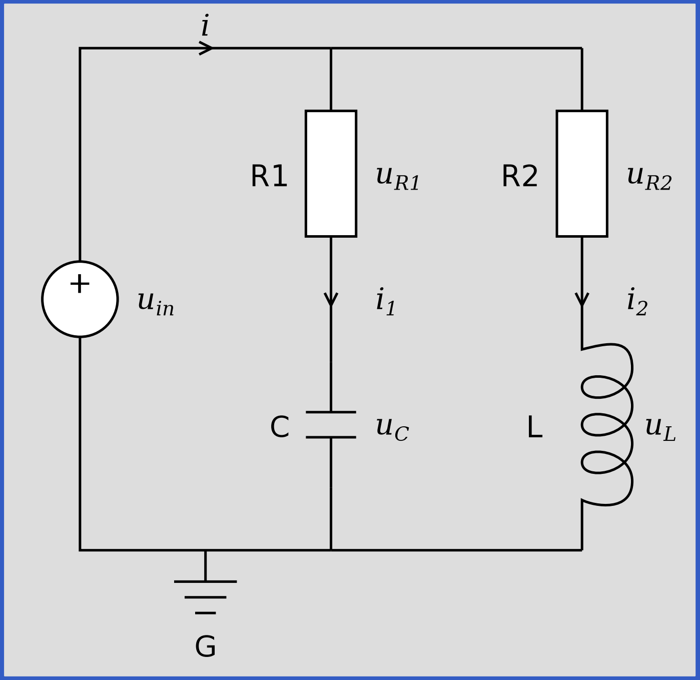
$$
\begin{aligned}
&\mathit{capacitor} :: \texttt{Capacitance} \rightarrow \texttt{SR}\,(\texttt{Pin}, \texttt{Pin})\\
&\mathit{capacitor}\ c = \textbf{sigrel}\ (p, n)\ \textbf{where}\\
&\qquad\qquad \mathit{twoPin} \diamond (p, n, u)\\
&\qquad\qquad c \cdot \textbf{der}(u) = p.i
\end{aligned}
$$

Simple Circuit in FHM (3)

$simpleCircuit$ :: `SR Current`
$simpleCircuit =$ **sigrel** i **where**

$$
\begin{aligned}
&resistor(1000) \diamond (r1p, r1n)\\
&resistor(2200) \diamond (r2p, r2n)\\
&capacitor(0.00047) \diamond (cp, cn)\\
&inductor(0.01) \diamond (lp, ln)\\
&vSourceAC(12) \diamond (acp, acn)\\
&ground \diamond gp\\
&\ldots
\end{aligned}
$$

Simple Circuit in FHM (4)



...

connect acp, $r1p$, $r2p$

connect $r1n$, cp

connect $r2n$, lp

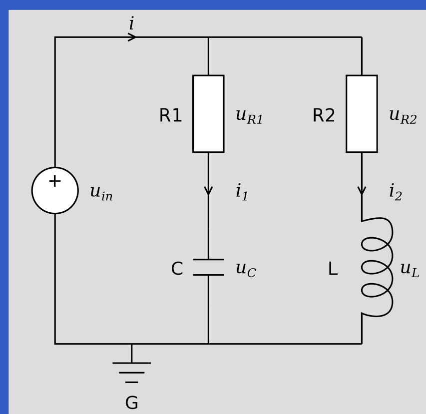
connect acn, cn, ln, gp

$i = r1p.i + r2p.i$

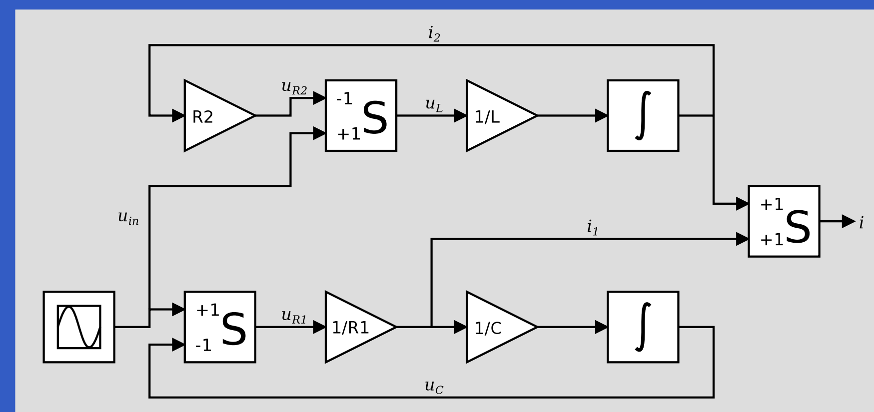
Notes on the Causal Model (1)

Notes on the Causal Model (1)

- Topology of causal model and modelled system do not agree:

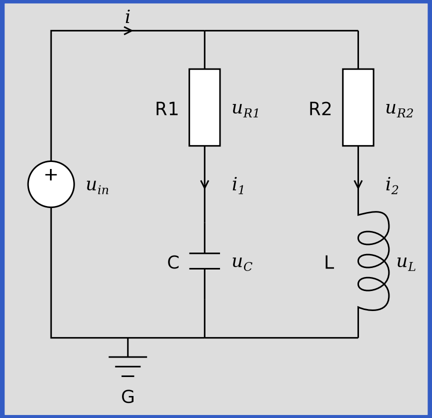


vs.

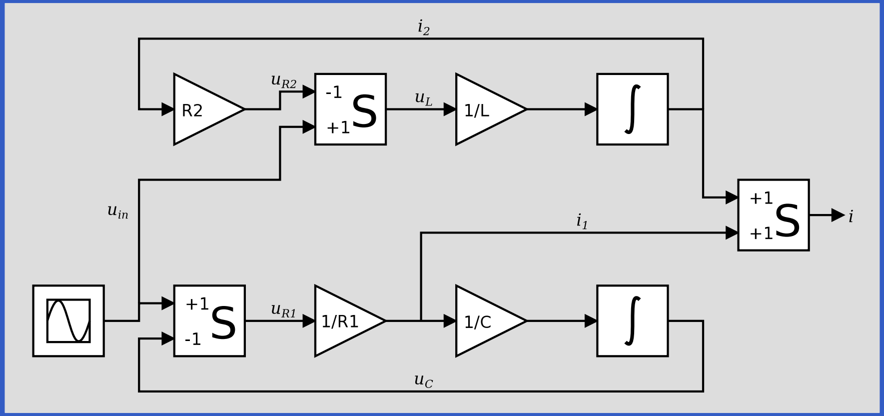


Notes on the Causal Model (1)

- Topology of causal model and modelled system do not agree:



vs.



- A small change in the modelled system can lead to large changes in the model.

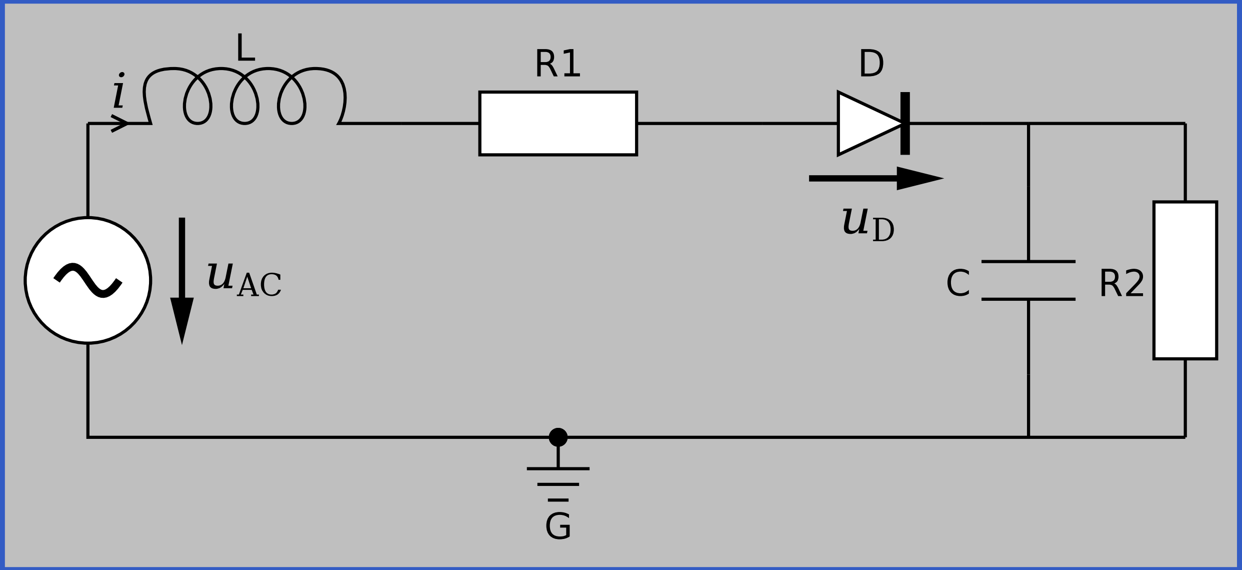
Notes on the Causal Model (2)

- In non-causal modelling, user need not worry about causality, but the ***simulator*** may well exploit structural properties like causality for e.g. efficient simulation.

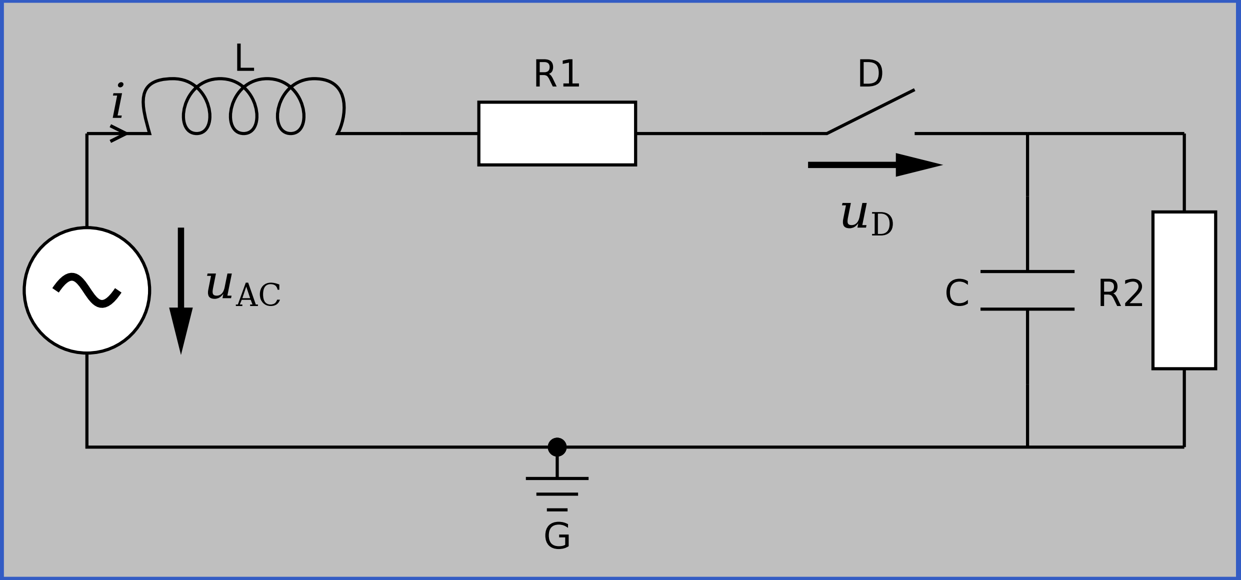
Notes on the Causal Model (2)

- In non-causal modelling, user need not worry about causality, but the ***simulator*** may well exploit structural properties like causality for e.g. efficient simulation.
- Once-off exploitation of ***any*** structural properties will preclude significant dynamic structural changes.

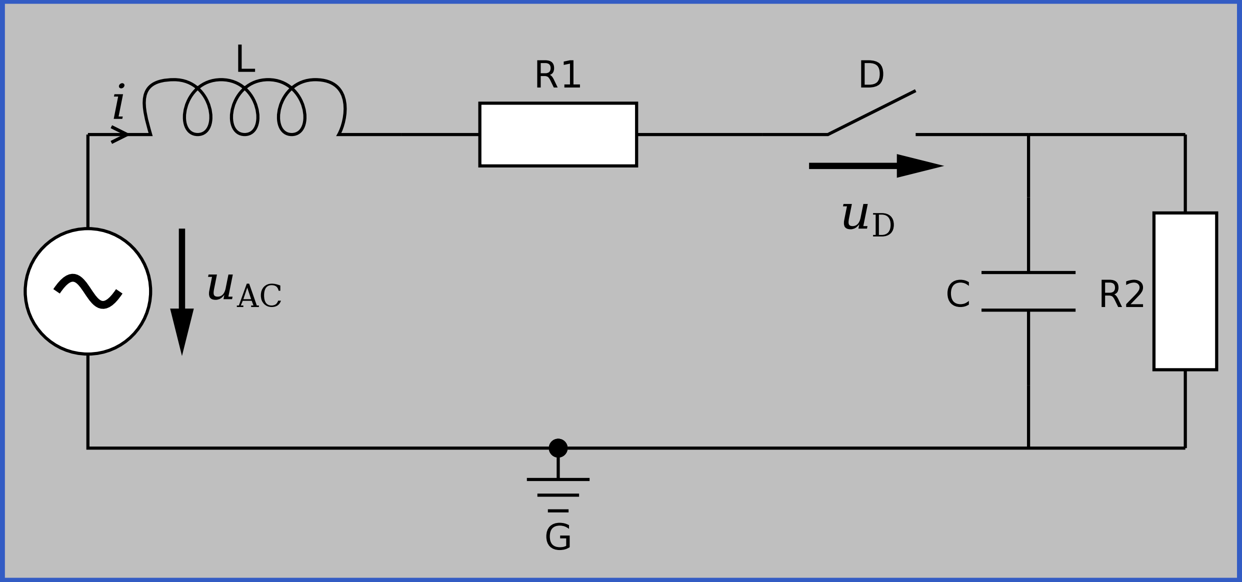
Example: Ideal Diodes (1)



Example: Ideal Diodes (2)



Example: Ideal Diodes (2)



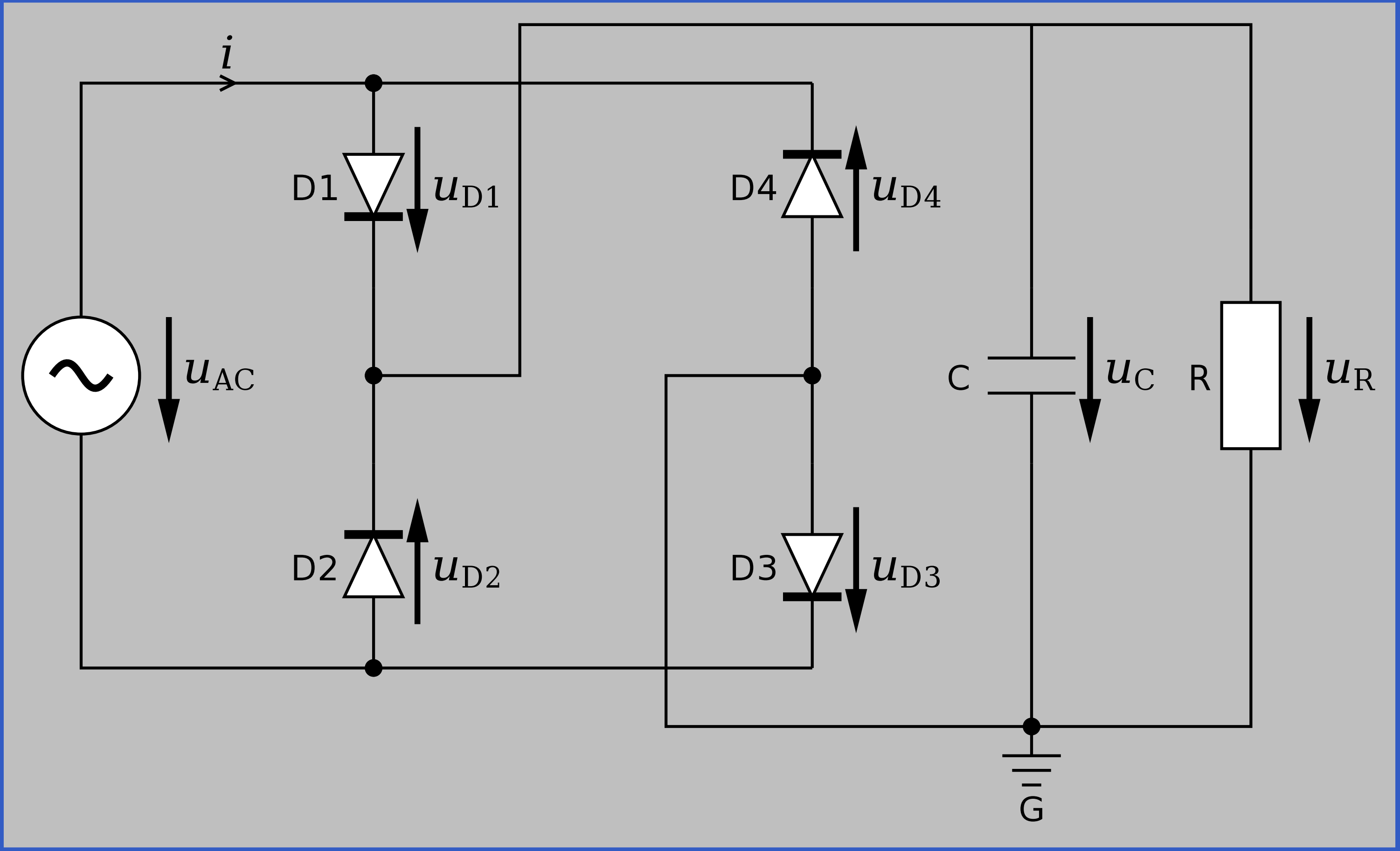
The in-line inductor means that an assumption of fixed causality will cause ***simulation to fail*** with a division by zero when the switch opens.

Example: Ideal Diodes (3)

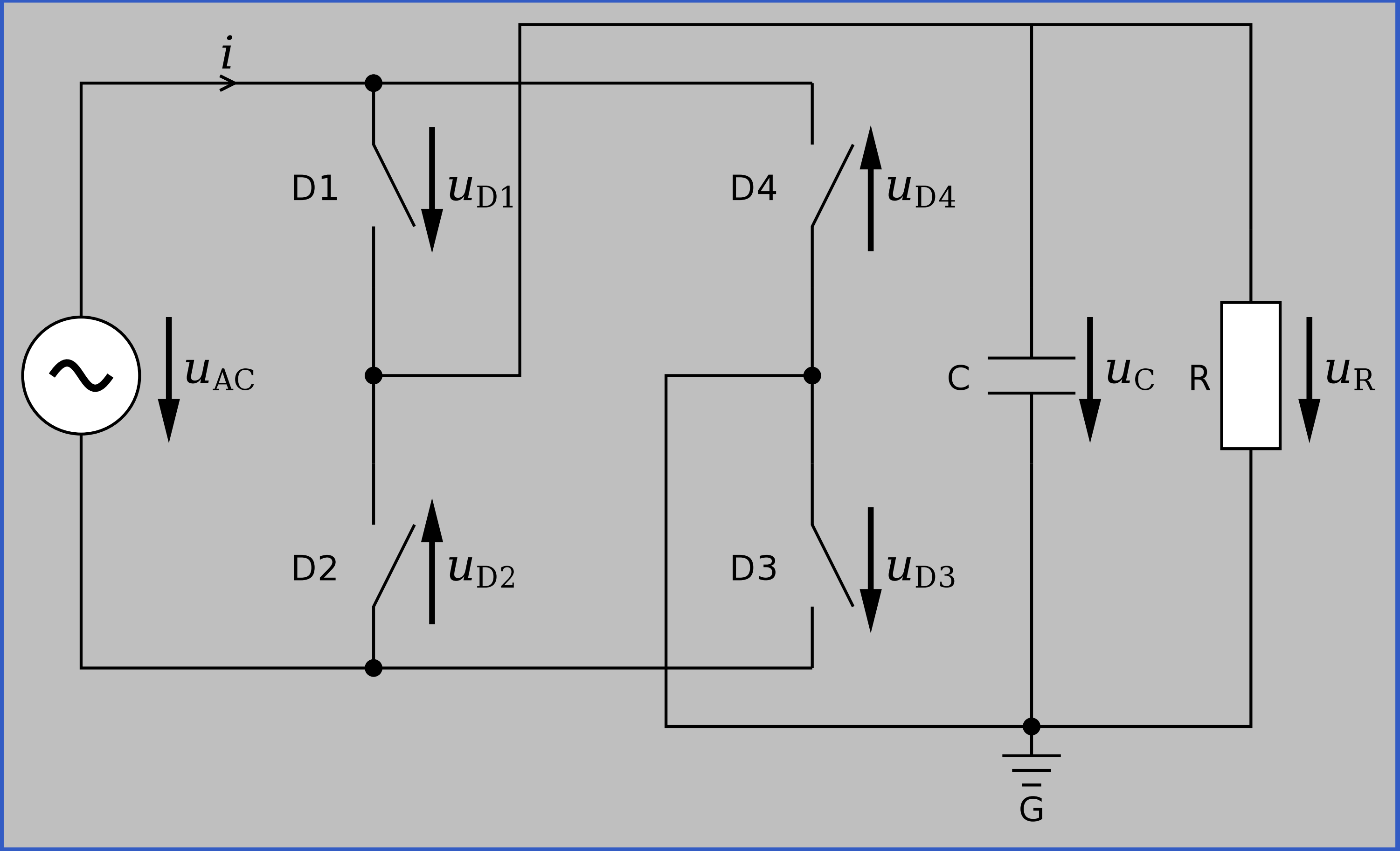
$$
\begin{aligned}
&icDiode :: \text{SR}\,(\texttt{Pin}, \texttt{Pin})\\
&icDiode = \textbf{sigrel}\ (p, n)\ \textbf{where}\\
&\qquad twoPin \diamond (p, n, u)\\
&\qquad \textbf{initially}; \textbf{when}\ p.v - n.v > 0 \ \Rightarrow\\
&\qquad\quad u = 0\\
&\qquad \textbf{when}\ p.i < 0 \ \Rightarrow\\
&\qquad\quad p.i = 0
\end{aligned}
$$

(Note: again, syntax somewhat idealised compared with present implementation.)

Example: Ideal Diodes (4)



Example: Ideal Diodes (5)



Example: Ideal Diodes (6)

To simulate the full-wave rectifier:

- The diode model has to be extended to allow expressing the voltage over the diodes always pairwise equal:

$$
\begin{aligned}
&\mathit{icDiode} :: \text{SR}\ (\texttt{Pin}, \texttt{Pin}, \texttt{Voltage})\\
&\mathit{icDiode} = \textbf{sigrel}\ (p, n, u)\ \textbf{where}\\
&\qquad\qquad \mathit{twoPin} \diamond (p, n, u)\\
&\qquad\qquad \textbf{initially};\ \textbf{when}\ p.v - n.v > 0\ \Rightarrow\\
&\qquad\qquad\quad u = 0\\
&\qquad\qquad \textbf{when}\ p.i < 0\ \Rightarrow\\
&\qquad\qquad\quad p.i = 0
\end{aligned}
$$

Example: Ideal Diodes (7)

- Redundant, semantically identical equations needs to be eliminated ("constant propagation" suffice in this case).
- End result is a fairly compositional model.
- No separate formalism, such as state charts, for controlling the switching.
- No need to worry about the here $2^4 = 16$ (and, in general, 2^n) possible modes: each mode computed on demand.
- No domain-specific assumptions built into the language itself.

Conclusions

- Assuming unchanging structural properties like causality severely limits what hybrid models can be simulated.
- Avoiding this restriction allows a number of challenging systems to be modelled and simulated in a straightforward manner.
- Of course not the whole story; many challenging problems remain: e.g., state transfer between structural configurations, chattering . . .