A Timing Model for Synchronous Language Implementations in Simulink

Timothy Bourke and Arcot Sowmya

School of Computer Science and Engineering

University of New South Wales, Sydney

and

National ICT Australia

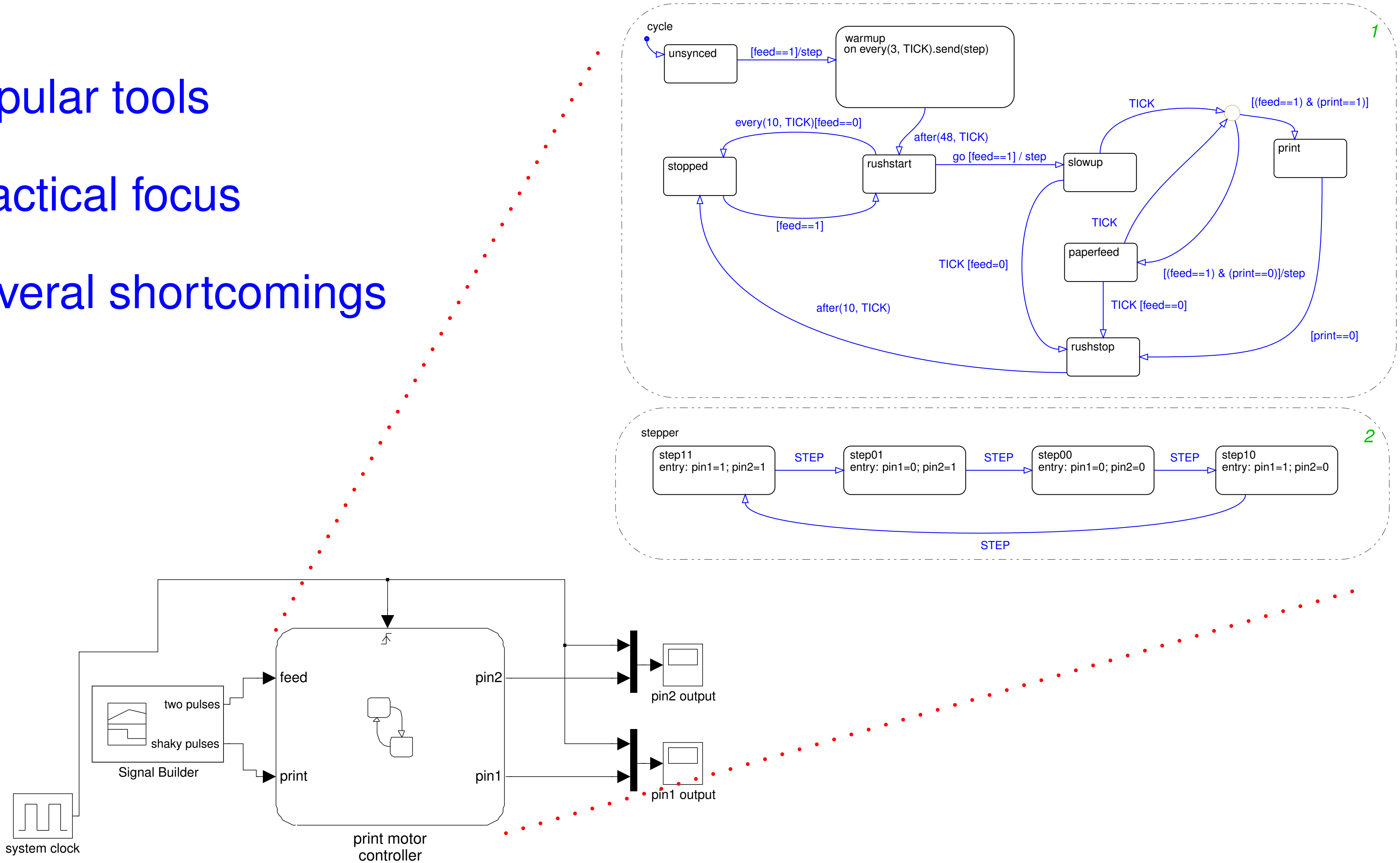
`tbourke@cse.unsw.edu.au`

Outline

- ⇒ Simulink and Stateflow
- An Argos block
- Timing Model
- Embedding within Simulink
- Concluding remarks

Simulink and Stateflow

- Popular tools
- Practical focus
- Several shortcomings



Simulation ⇝ Model-driven Development

Reasoning about Stateflow designs is complicated:

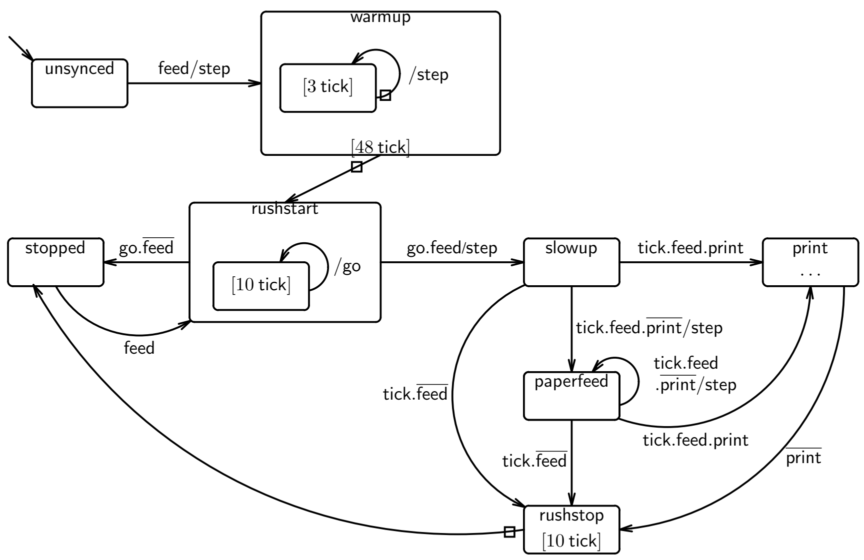
1. intricate ordering rules
2. queued event processing
3. stacking of communications
4. implicit assumption of synchrony

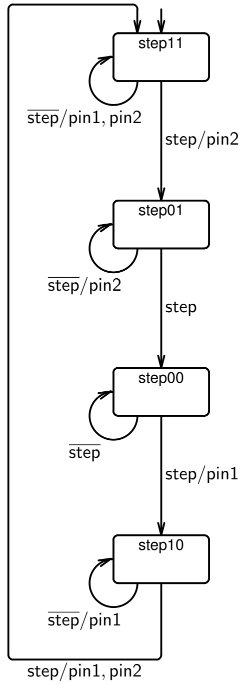
Synchronous languages have *better* underlying models

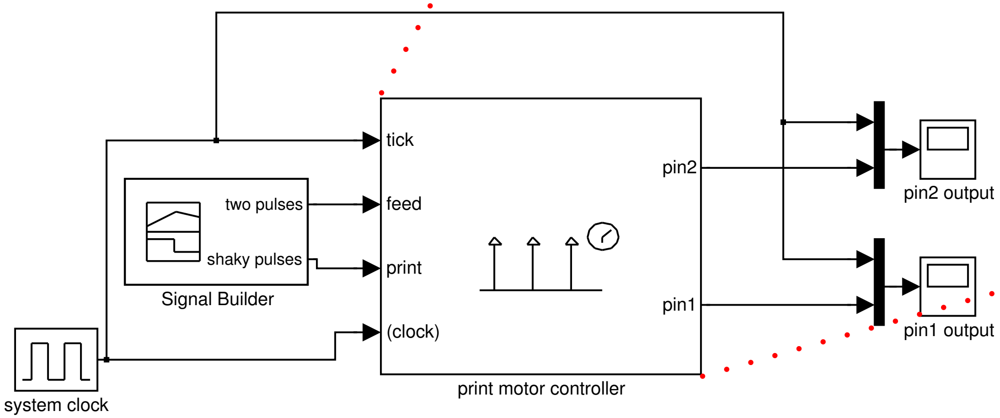
(assumption)

An Argos [Mar91, MR01] block: syncblock [BS05]

- Our first attempt at combining synchronous languages and Simulink.
- Simulate with Argos controllers.







aside: [CCM$^+$03, SSC$^+$04]

- *integrate* rather than *extract*
- *simulate* sync. programs

syncblock: simulating embedded controllers

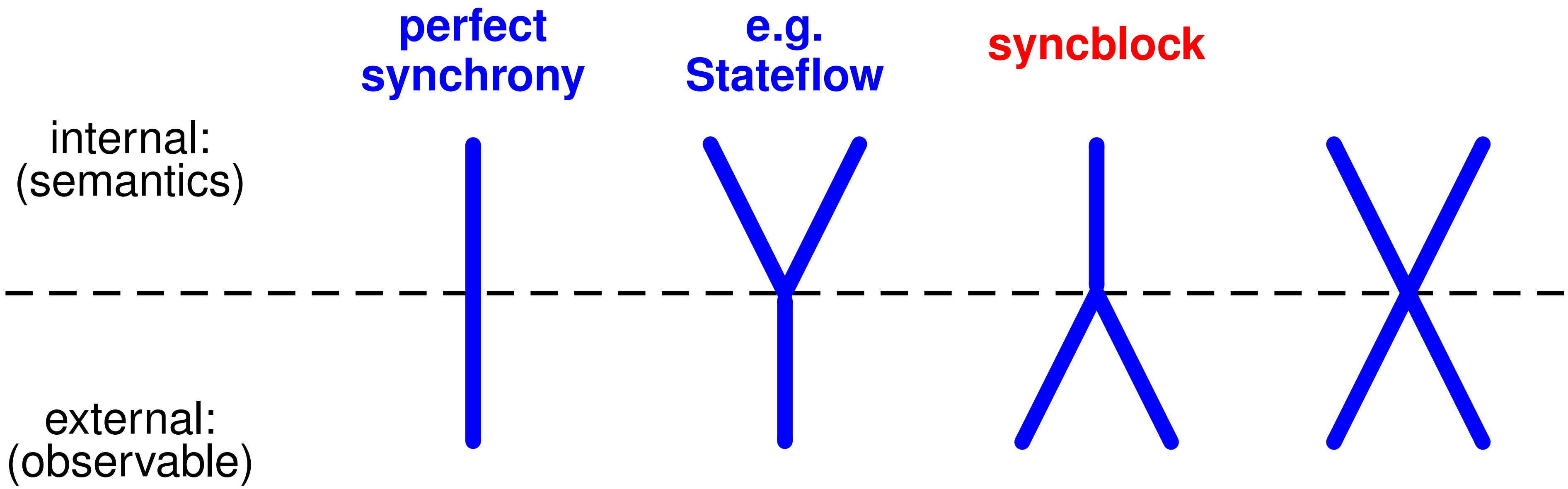
- **Original Prototype**: perfect synchrony
  - Block outputs appear simultaneously with inputs.
  - i.e. in the same Simulink step.

- *But*, Simulink normally models timing detail.

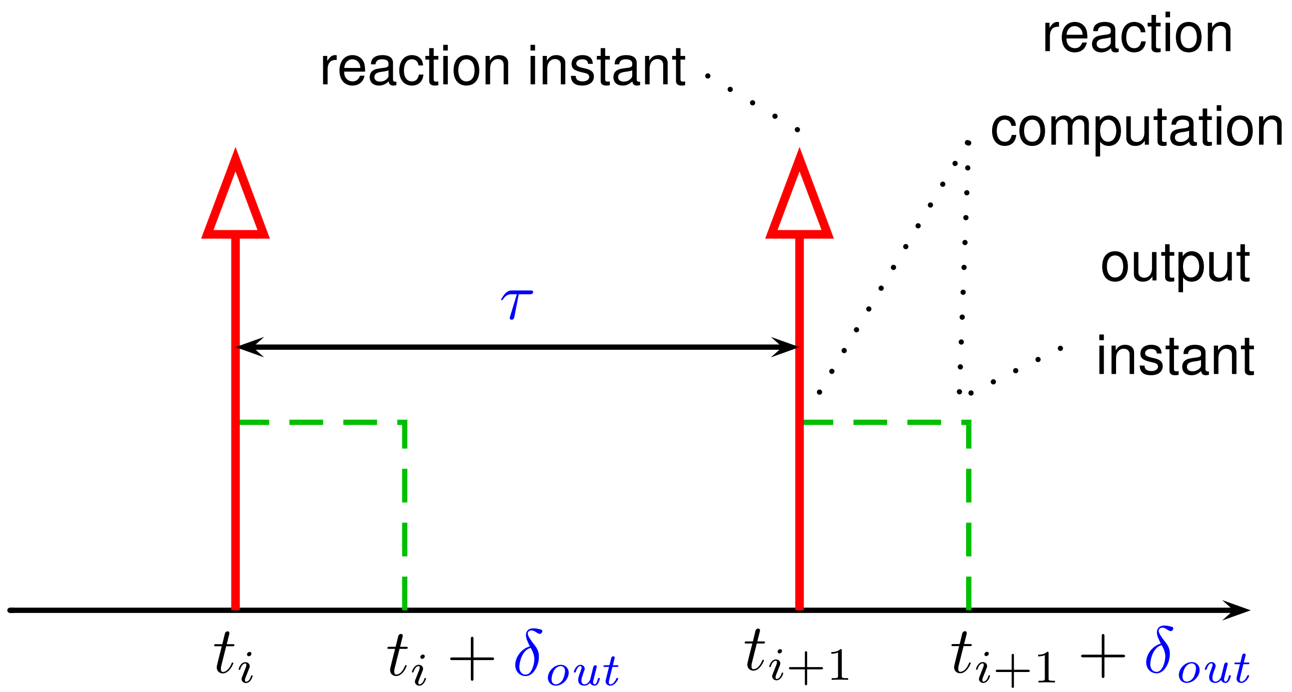
consider: dedicated embedded controllers

aim: provide simulation runs with low-level timing detail.

syncblock: simulating synchrony



- **Revised approach**: simulate implementation delays.
  - Internally: synchronous semantics.
  - Externally: delay between inputs and outputs.
- Necessary to latch inputs and outputs, and to schedule reactions.
  - Effectively modelling part of the platform (if abstractly).



Idealised parameters

aside: TAXYS [STY03]

- event-driven or sample-driven: mode
- Delay between input and output: δ_{out}
- Minimum pause between reactions: τ
- Program $+$ Limitations $=$ Simulation (block)

  $=$ Implementation (model)

Outline

- $\surd$ Simulink and Stateflow
- $\surd$ An Argos block
- $\Rightarrow$ Timing Model
- Embedding within Simulink
- Concluding remarks

Transformation to Timed Automata

fix: $A_B = \langle S, s_0, I, O, T \rangle$ $\quad \tau \in \mathbb{Q}_0^+$

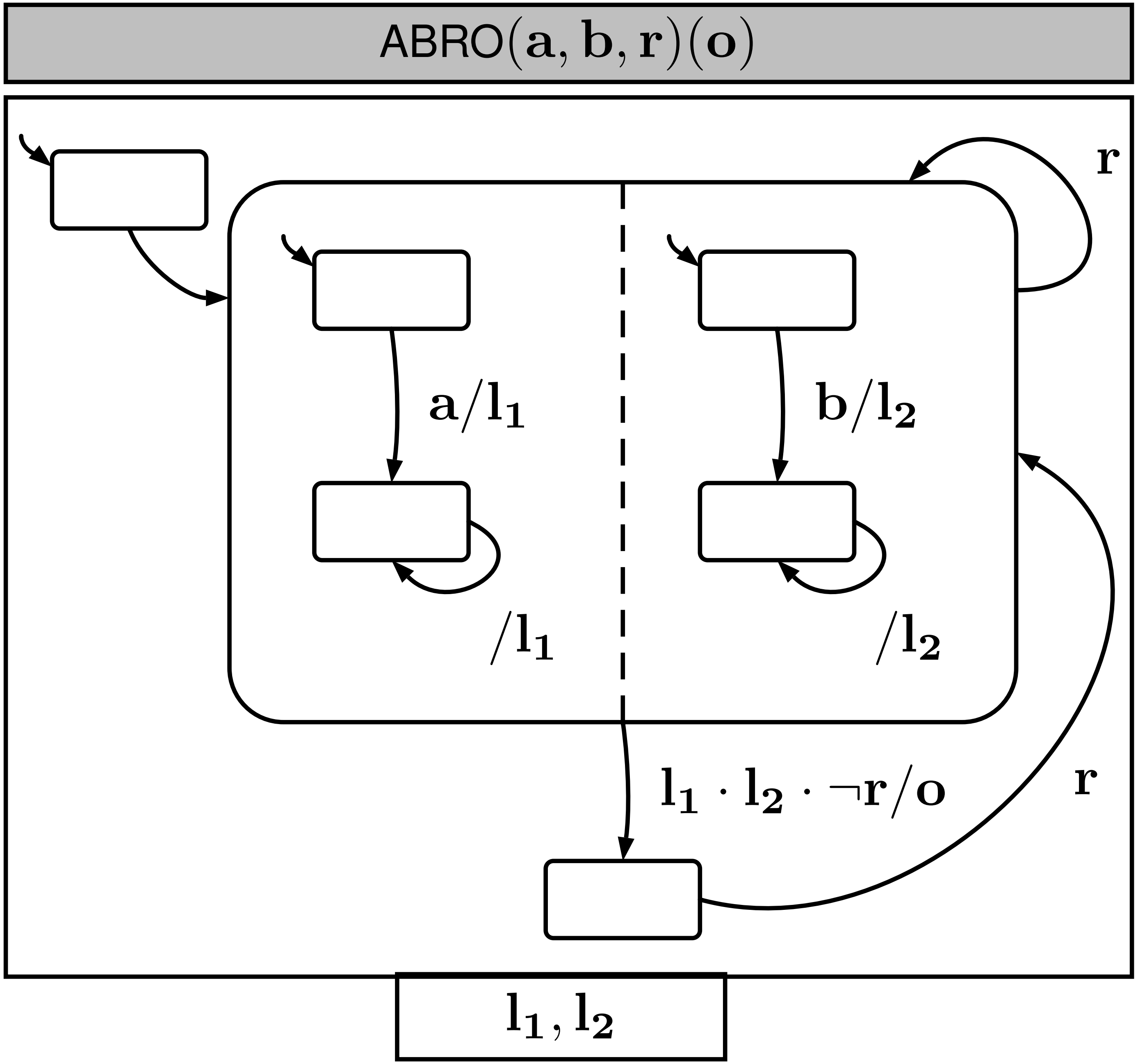
$trigger \in \{sample, event\}$ $\quad \delta_{out} \in \mathbb{Q}_0^+$

requiring: $\delta_{out} \leq \tau$ $\quad trigger = event \vee \tau > 0$

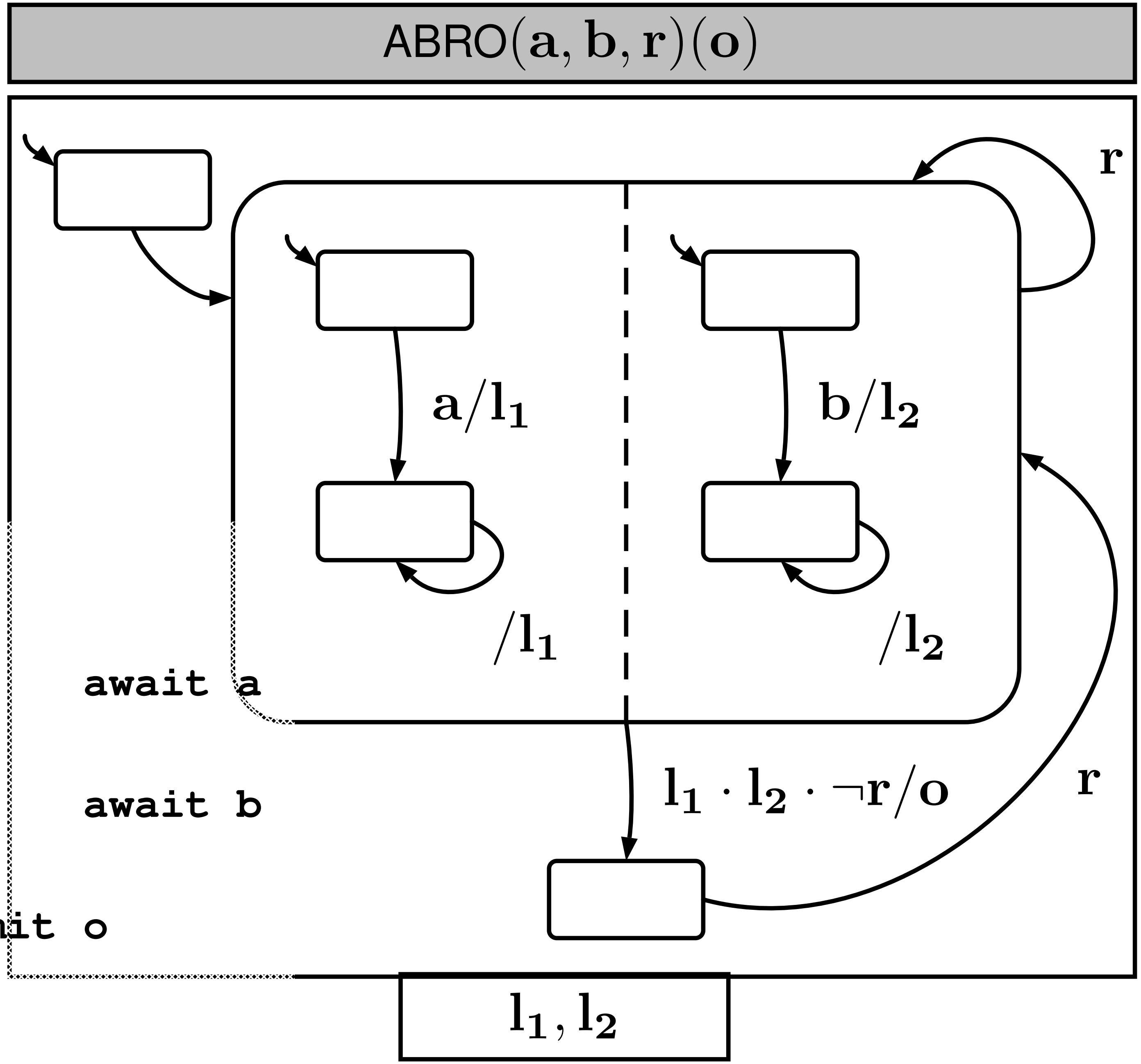
then define: $A_{\tau,\delta_{out}}^{trigger} = \langle \Sigma, L, L_0, C, E \rangle$: [AD94]

- $\Sigma = I \mathbin{\dot\cup} O \mathbin{\dot\cup} \{\texttt{react}\}$
- $L = (S \mathbin{\dot\cup} \{\texttt{startup}\}) \times \mathcal{P}(I) \times \mathcal{P}(O) \times \mathbb{B}$
- $L_0 = \{(\texttt{startup}, \emptyset, \emptyset, \mathsf{ff})\}$
- $C = \{x\}$
- E is the smallest set defined by the conjunction of 9 transition rules.

(almost) ABRO [Ber00]: in Argos

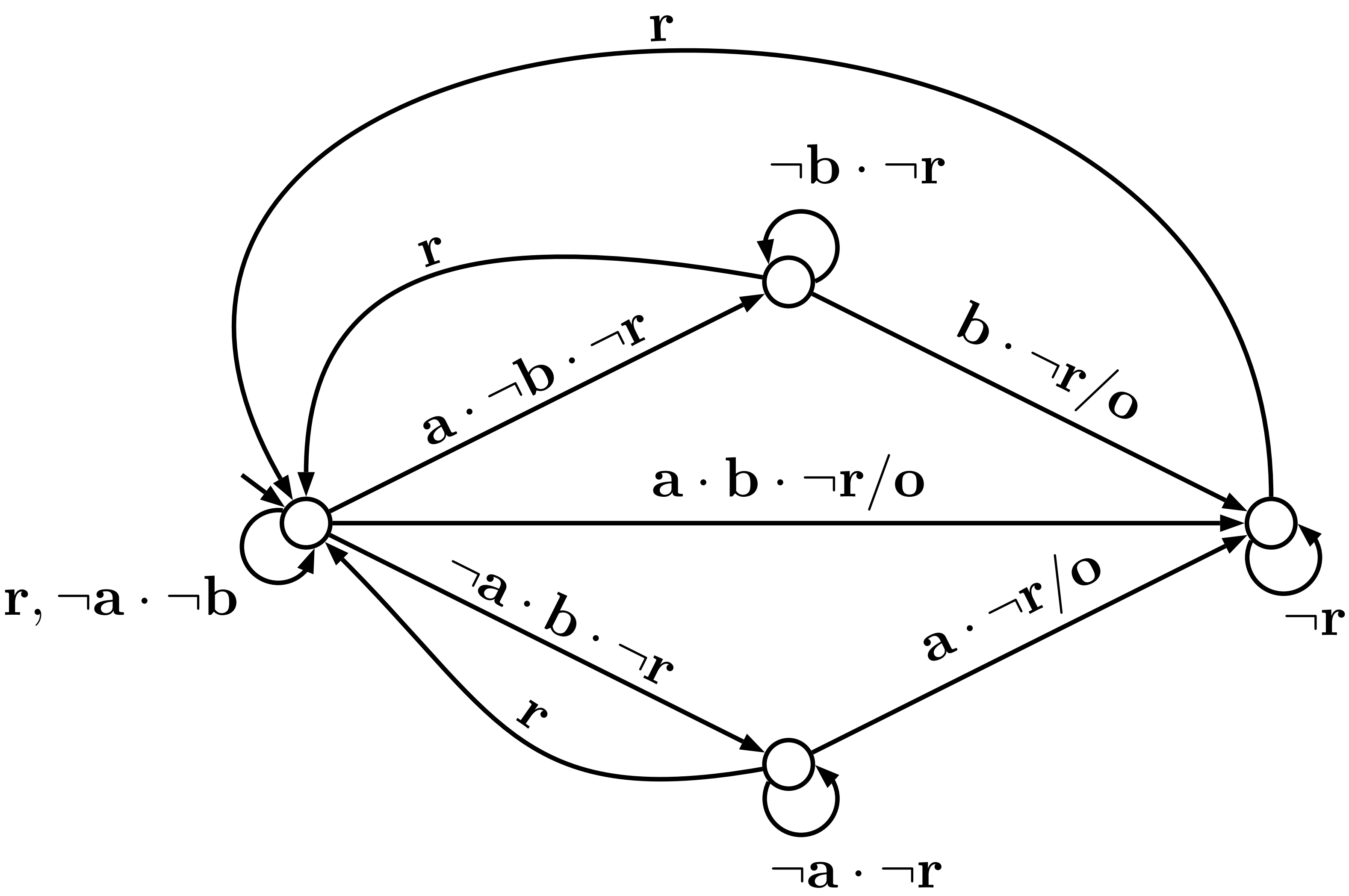


(almost) ABRO [Ber00]: in Argos

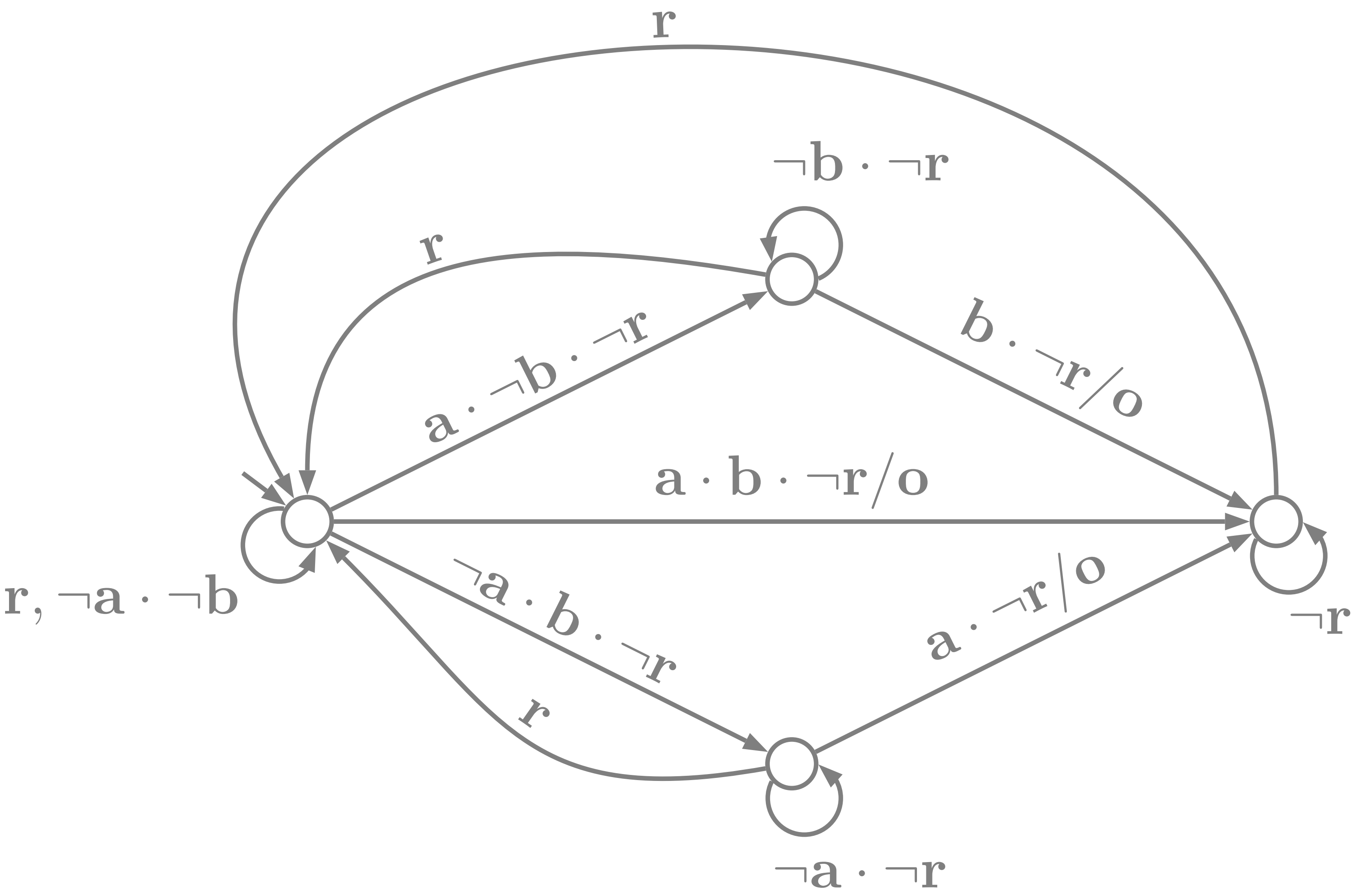



```
loop
     [
          await a
     ||
          await b
     ];
     emit o
each r
```

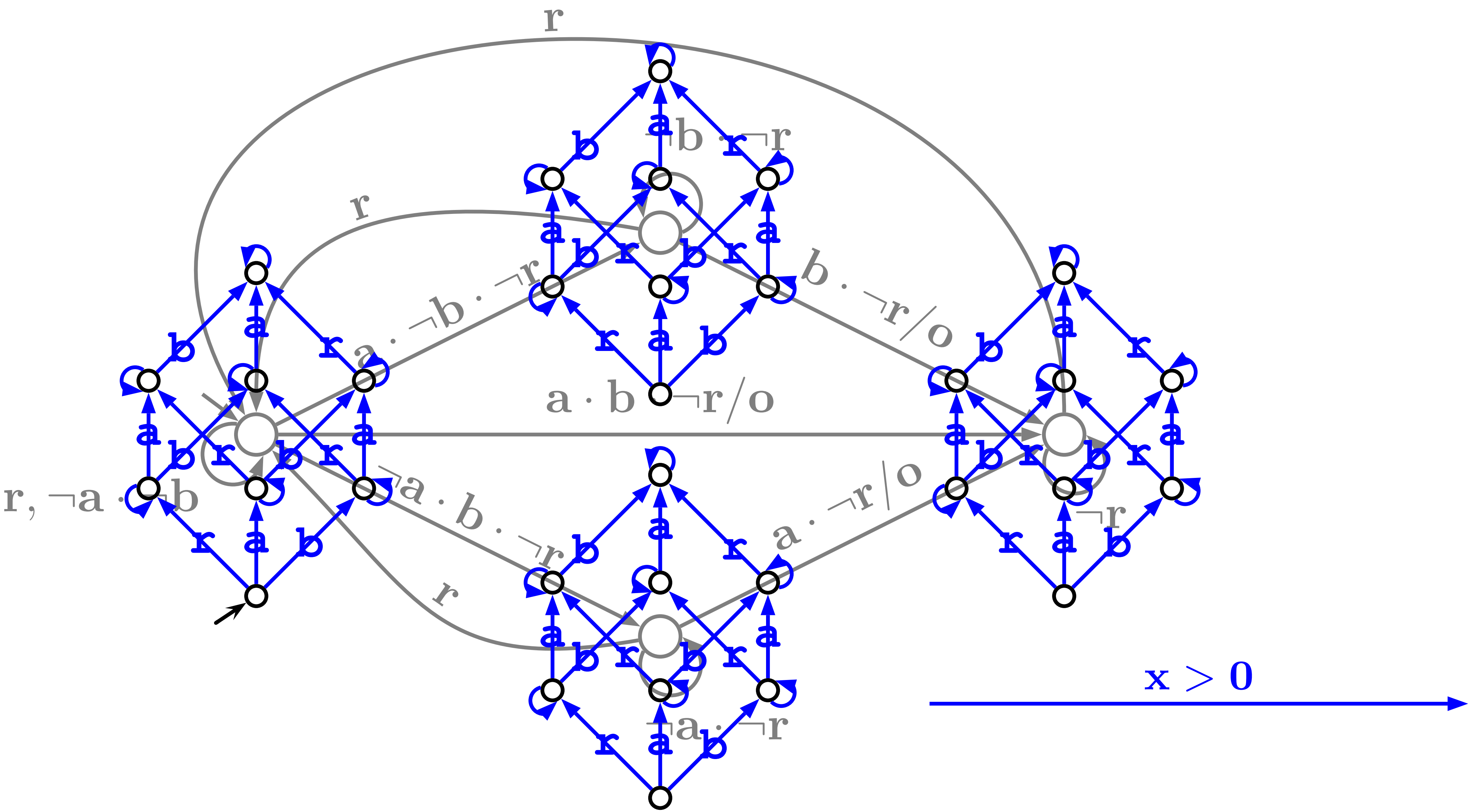
(almost) ABRO: Labelled Transition System



(almost) ABRO: Timed Transition System

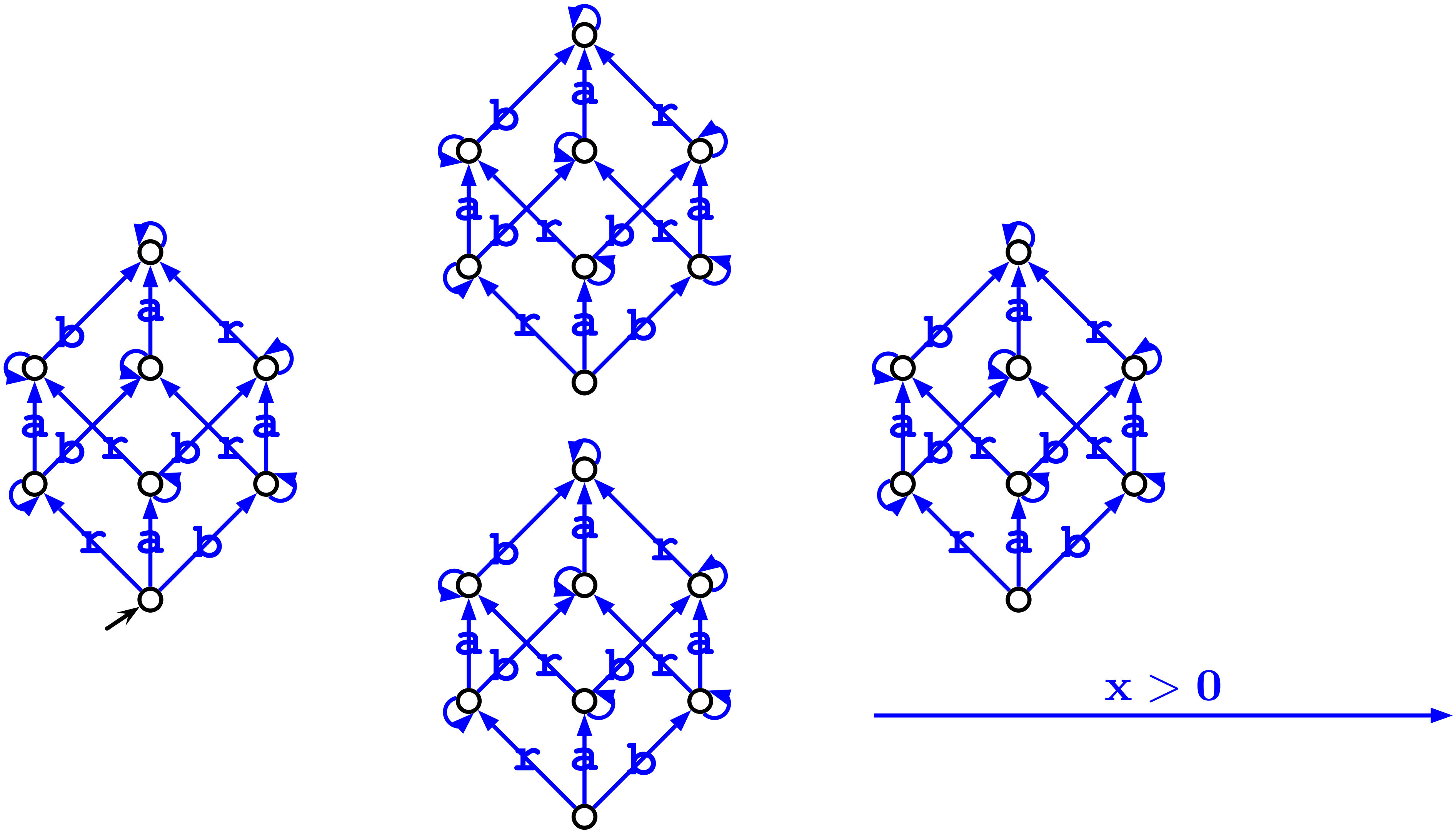


(almost) ABRO: Timed Transition System



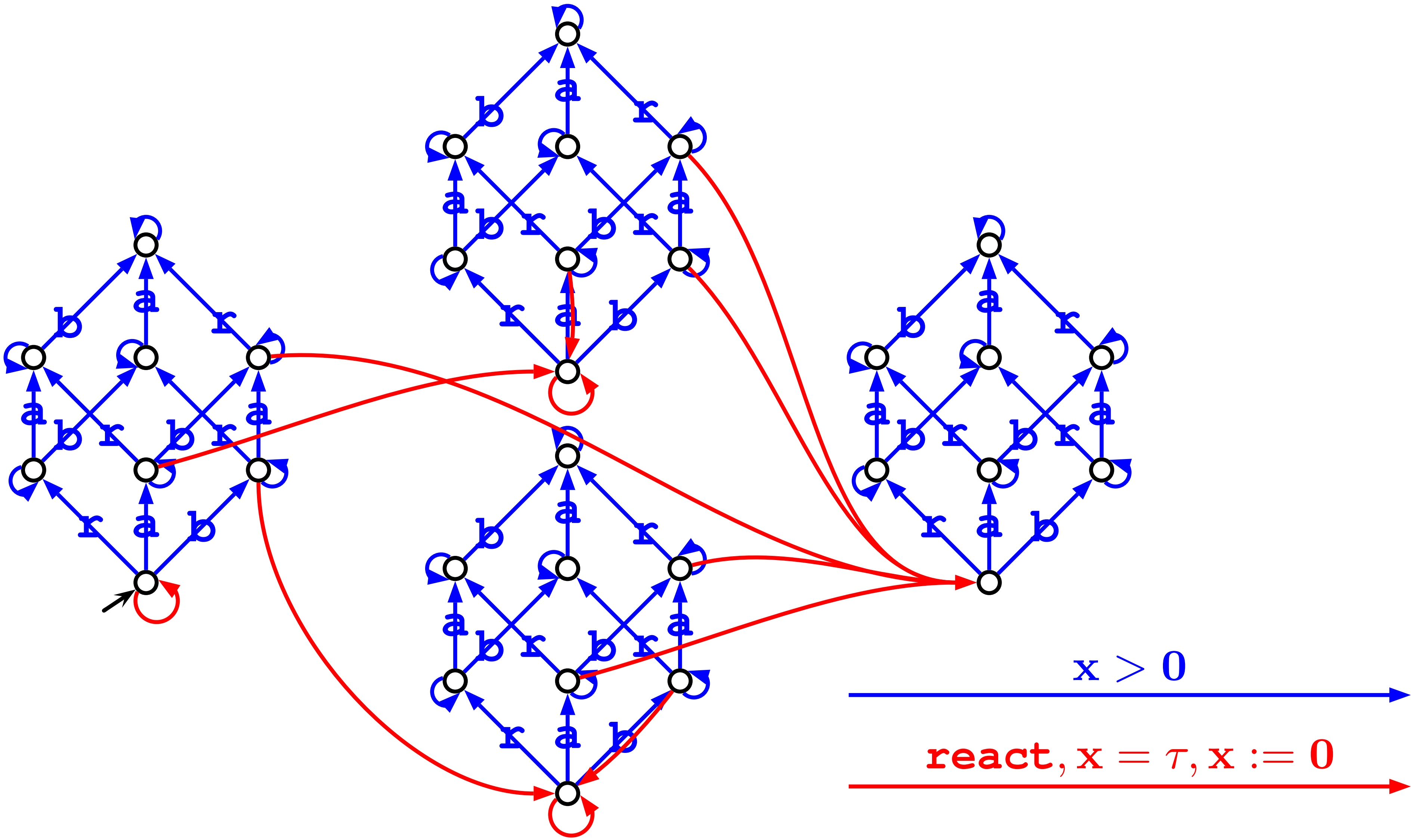
(almost) ABRO: Timed Transition System

$trigger = sample$



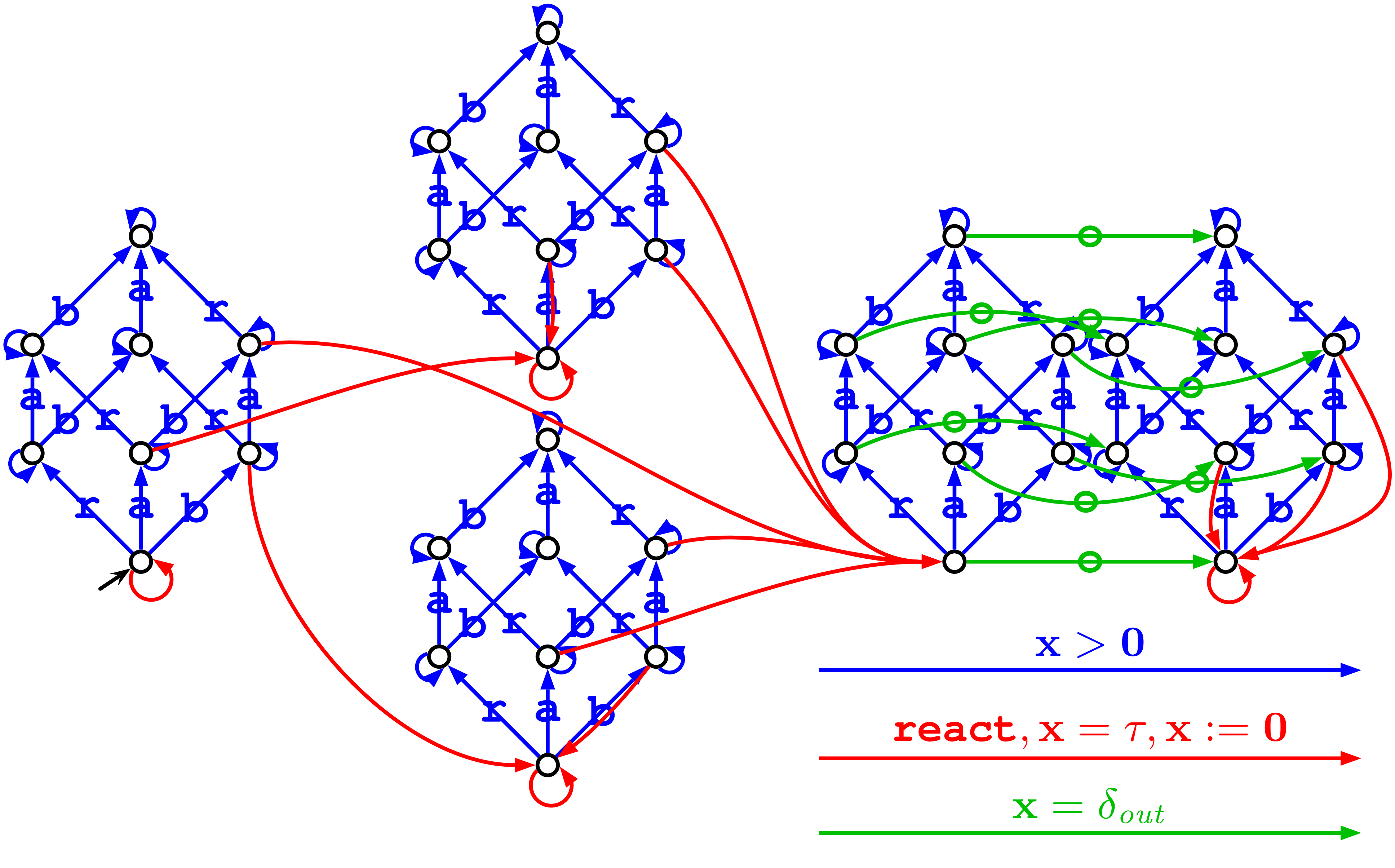
(almost) ABRO: Timed Transition System

$trigger = sample$



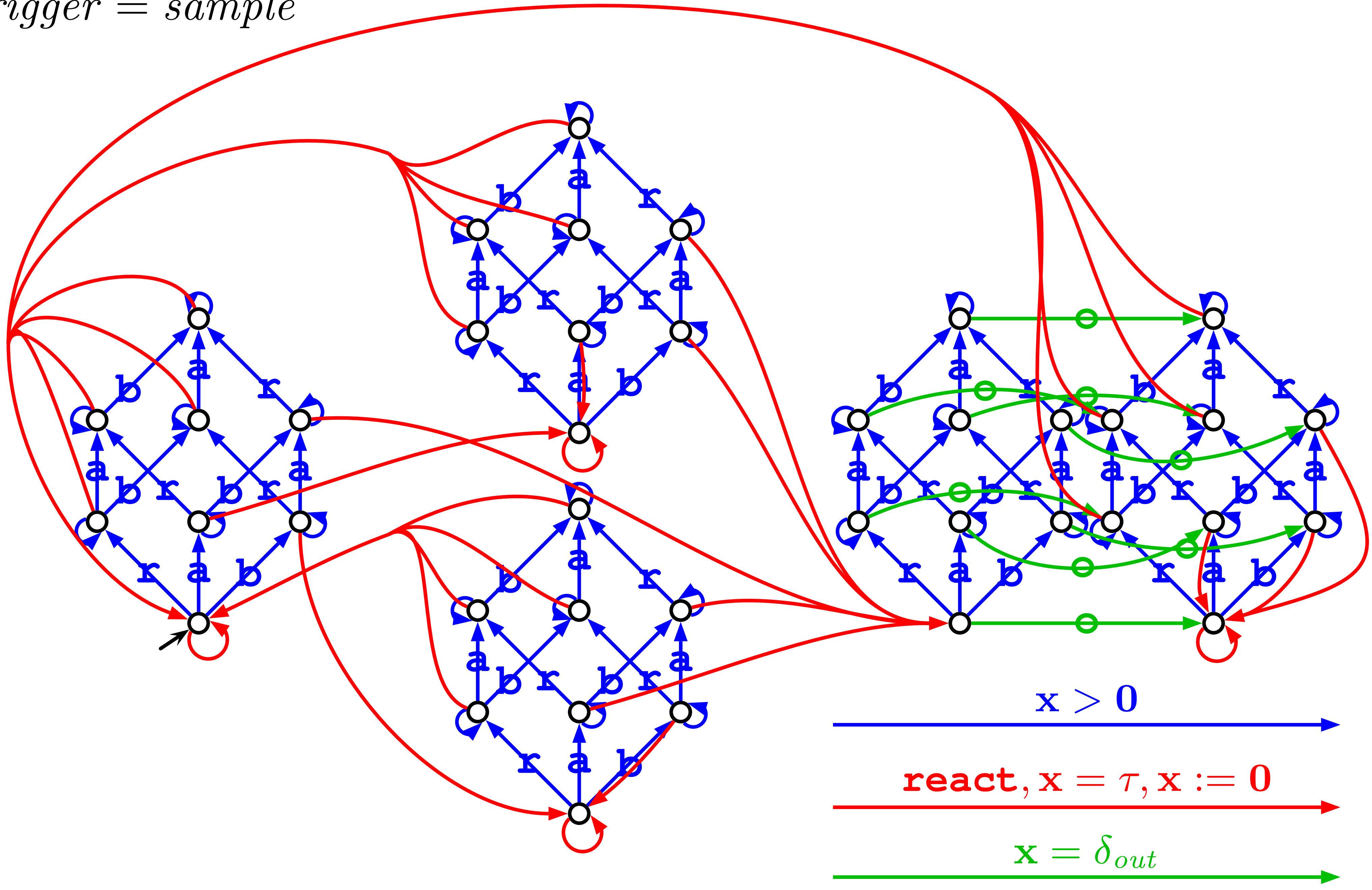
(almost) ABRO: Timed Transition System

$trigger = sample$



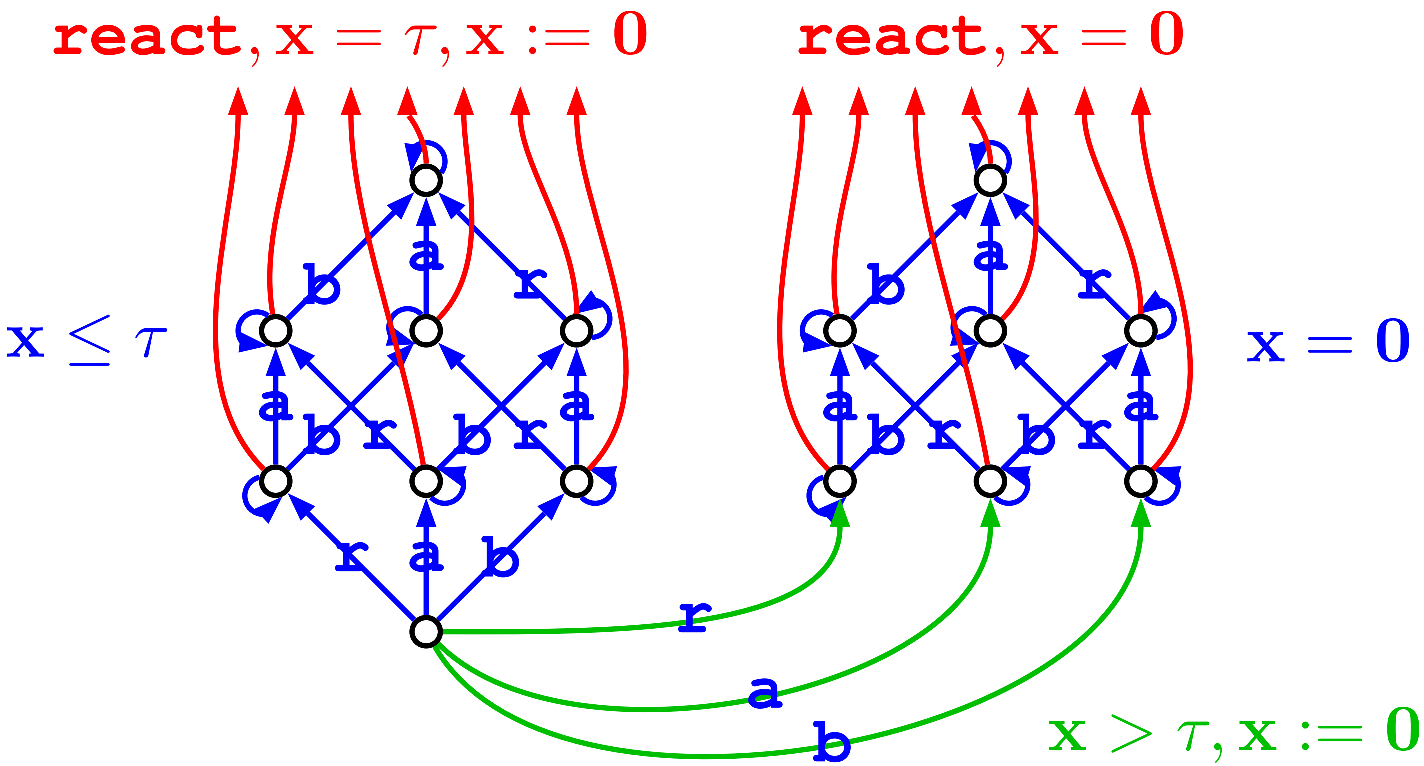
(almost) ABRO: Timed Transition System

$trigger = sample$



Event-driven triggering

$$\textit{trigger} = \textit{event}$$

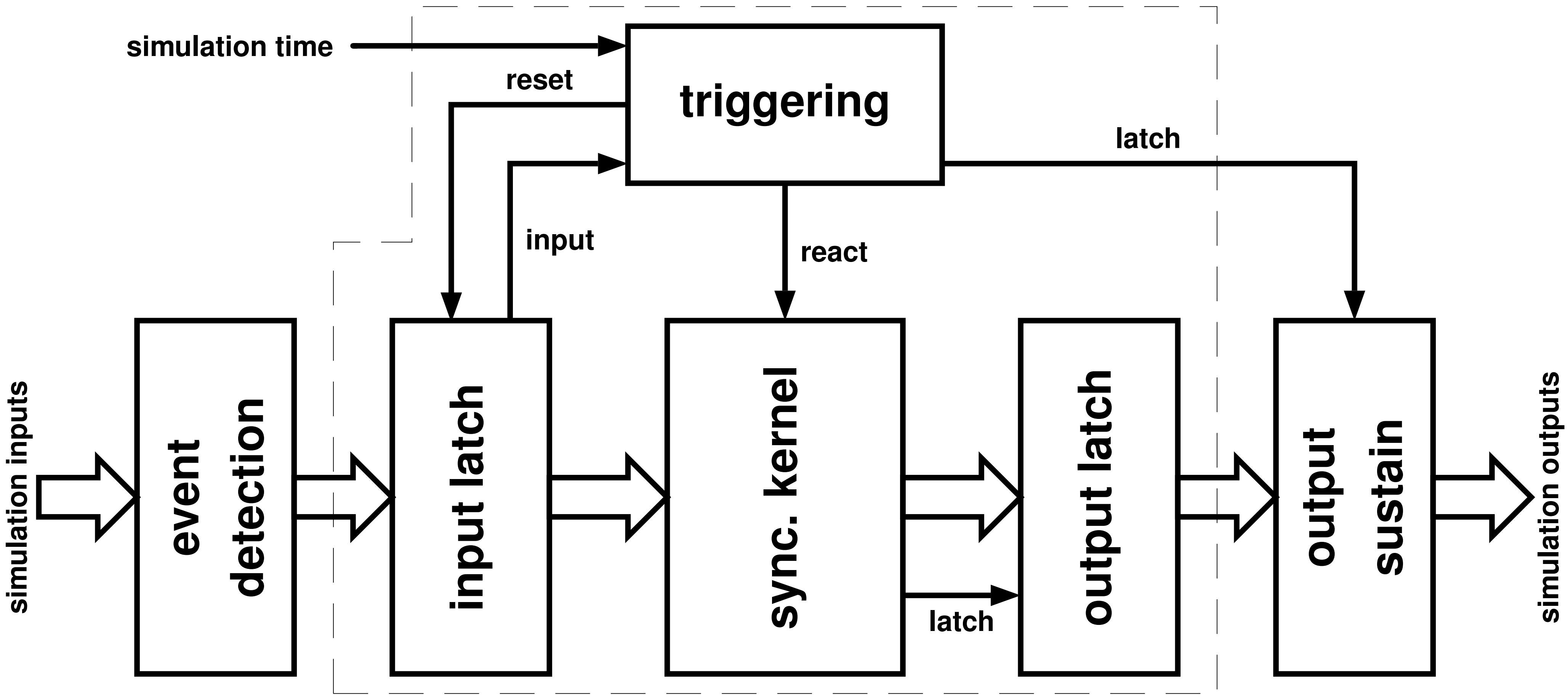


- Input events during a reaction must wait until $x = \tau$
- Otherwise, they trigger a reaction *urgently* [BST97]

Outline

- $\surd$ Simulink and Stateflow
- $\surd$ An Argos block
- $\surd$ Timing Model
- $\Rightarrow$ Embedding within Simulink
- Concluding remarks

Embedding within Simulink



One block or many?

Embedding within Simulink

Adopt a semantics for Simulink

- Simulation Engine
- Intent of models

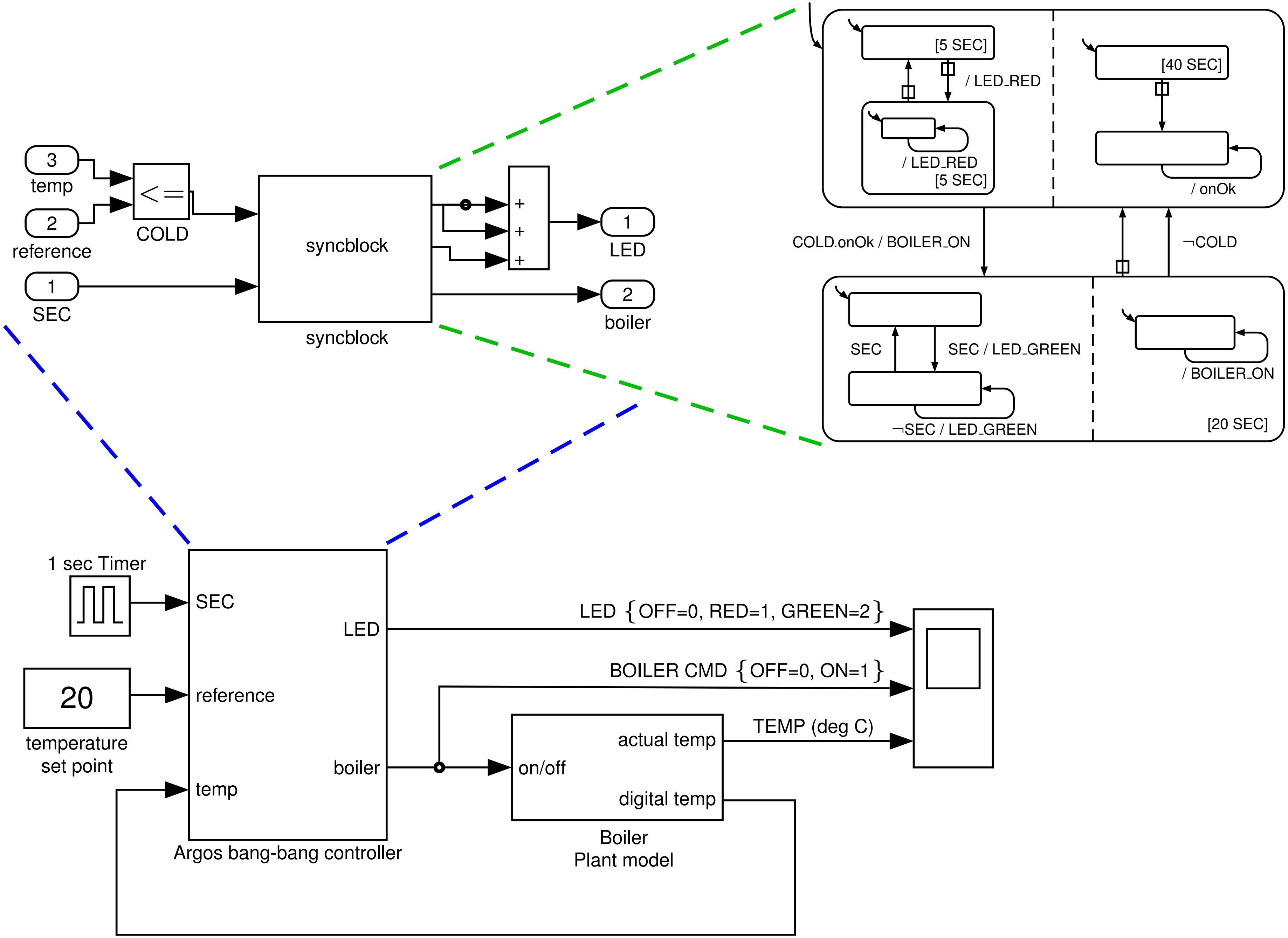
Translate models

e.g. to Lustre

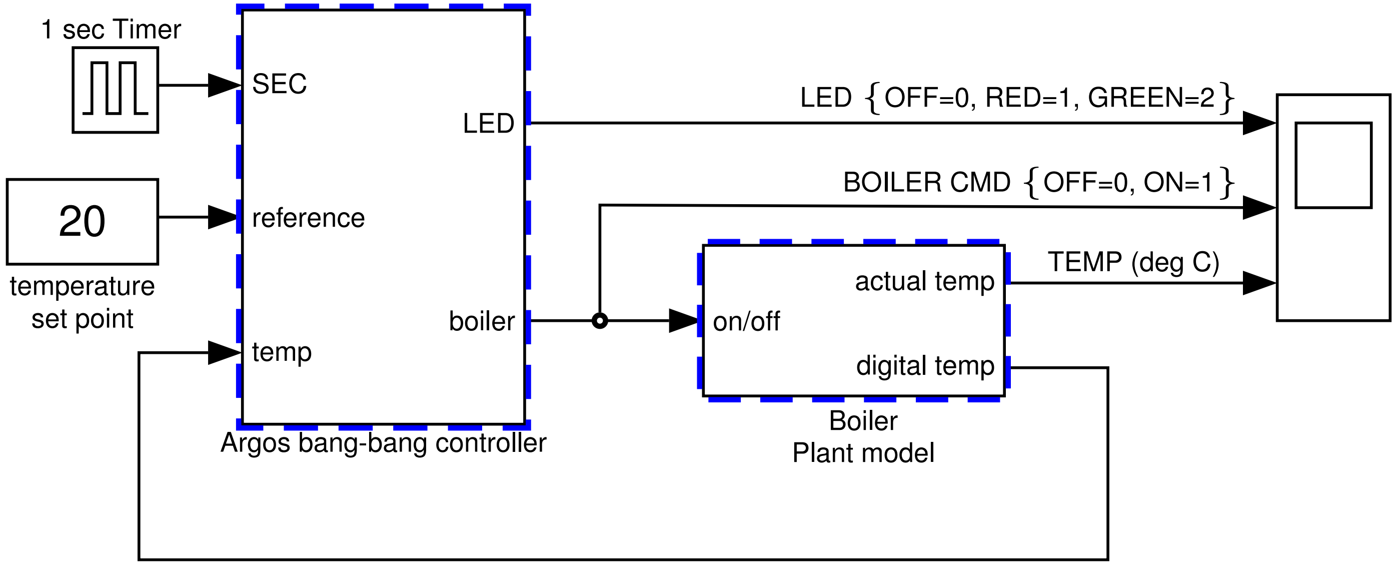
Interactions of block

mix conceptual and low-level operations

Mathworks Bang-bang temperature controller

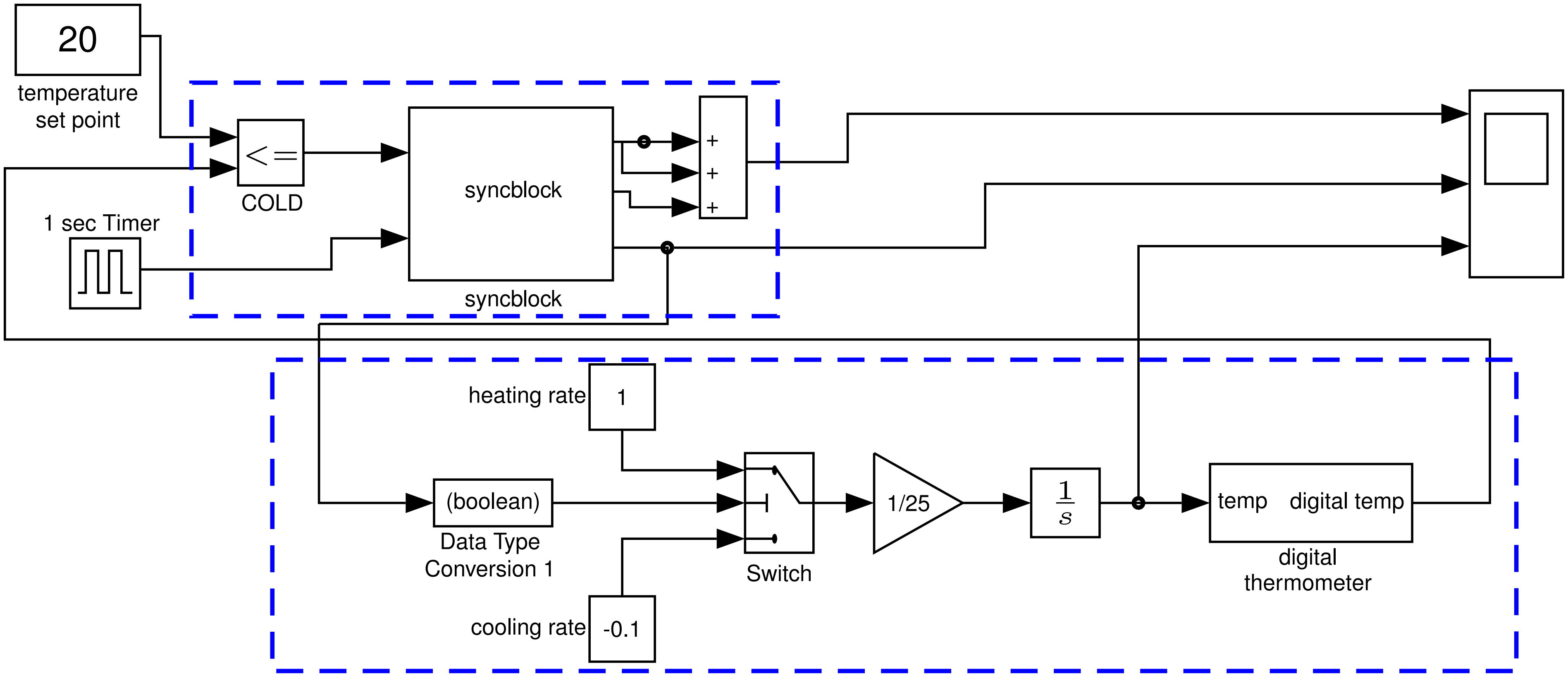


Simulink simulation engine: initialization



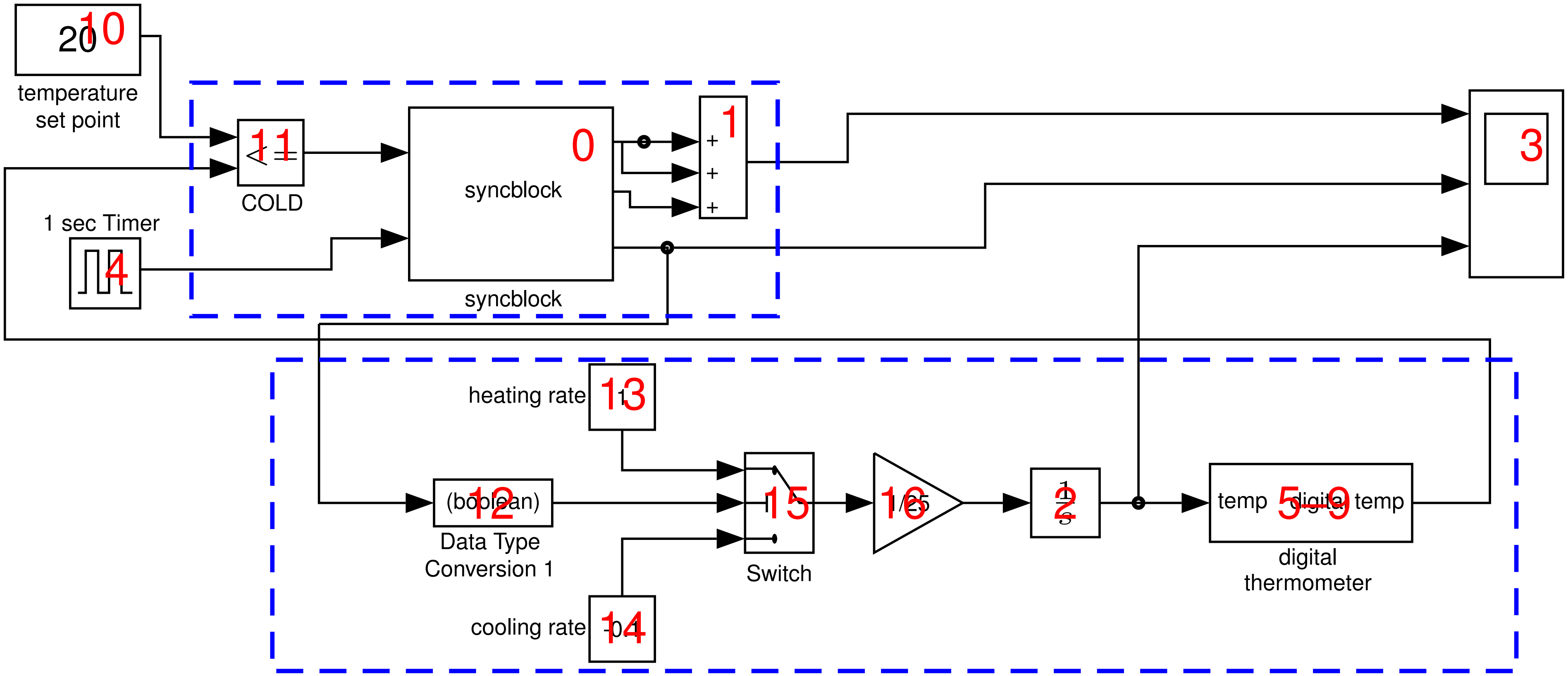
1. Flatten model

Simulink simulation engine: initialization



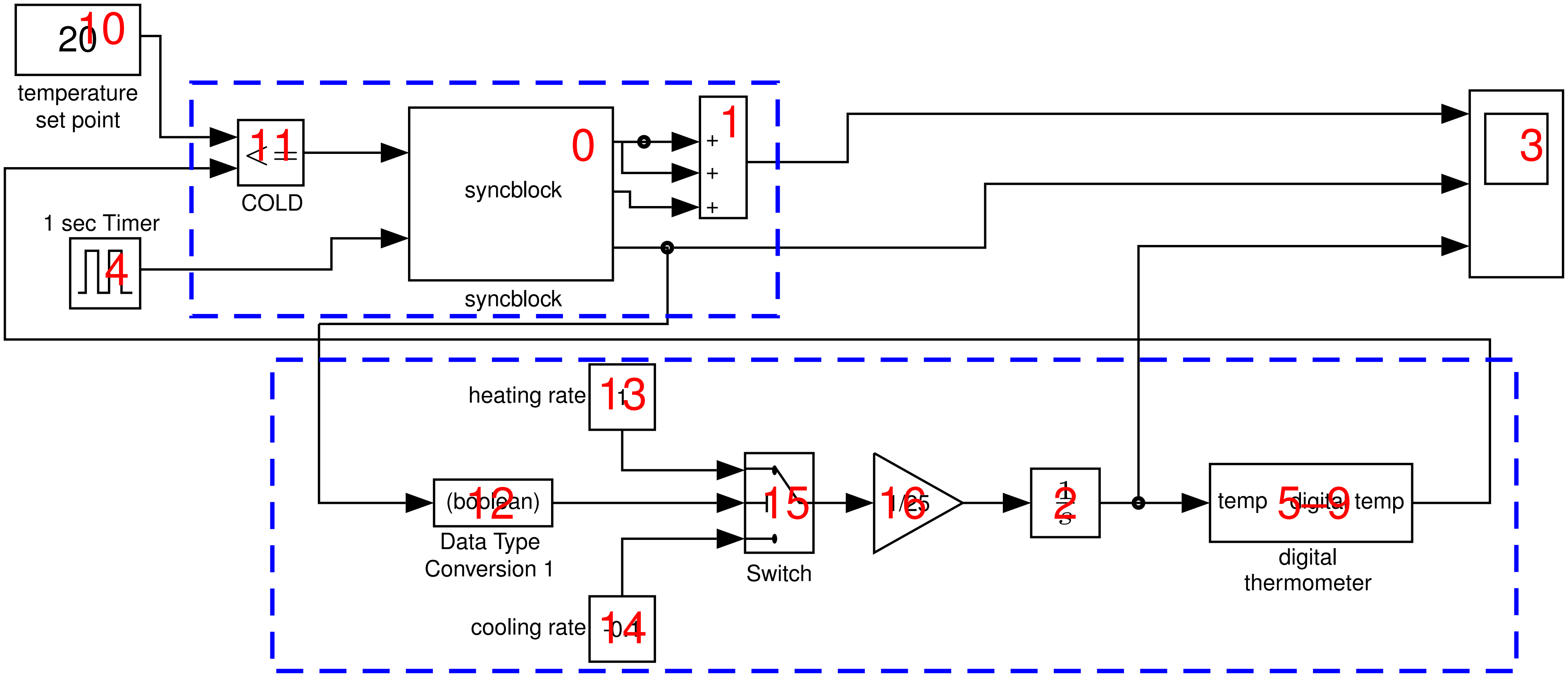
1. Flatten model

Simulink simulation engine: initialization



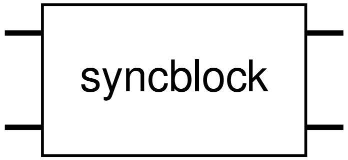
1. Flatten model
2. Order by signal dependencies.

Simulink simulation engine: initialization



1. Flatten model
2. Order by signal dependencies.
3. Start at $t = 0$.
5. Visit each block—maybe several times.
6. Increase t—depends on *solver*.
7. repeat from step 5

Behaviour of syncblock



$$y = f_o(t, x, u) \qquad \textit{outputs}$$

$$x'_d = f_u(t, x, u) \qquad \textit{update}$$

x_c *previous clock value*

x_{t_p} *previous sample time*

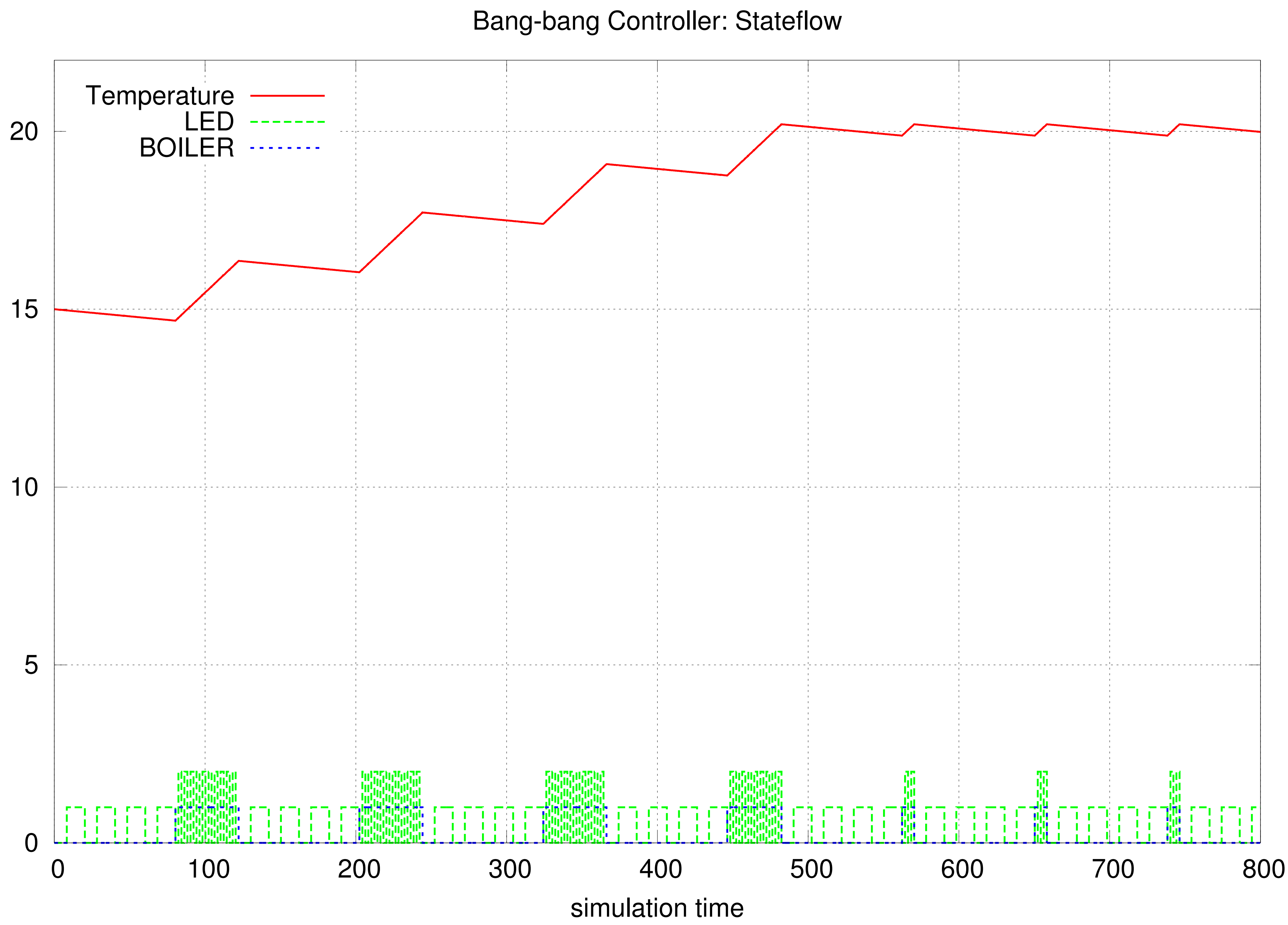
- Two predicates: $react$ and $emit$.
- Instants of interest:
  - sample-driven:

$\delta_{out} = \tau$	$[\tau, 0]$
otherwise	$[\tau, 0]$ and $[\tau, \delta_{out}]$

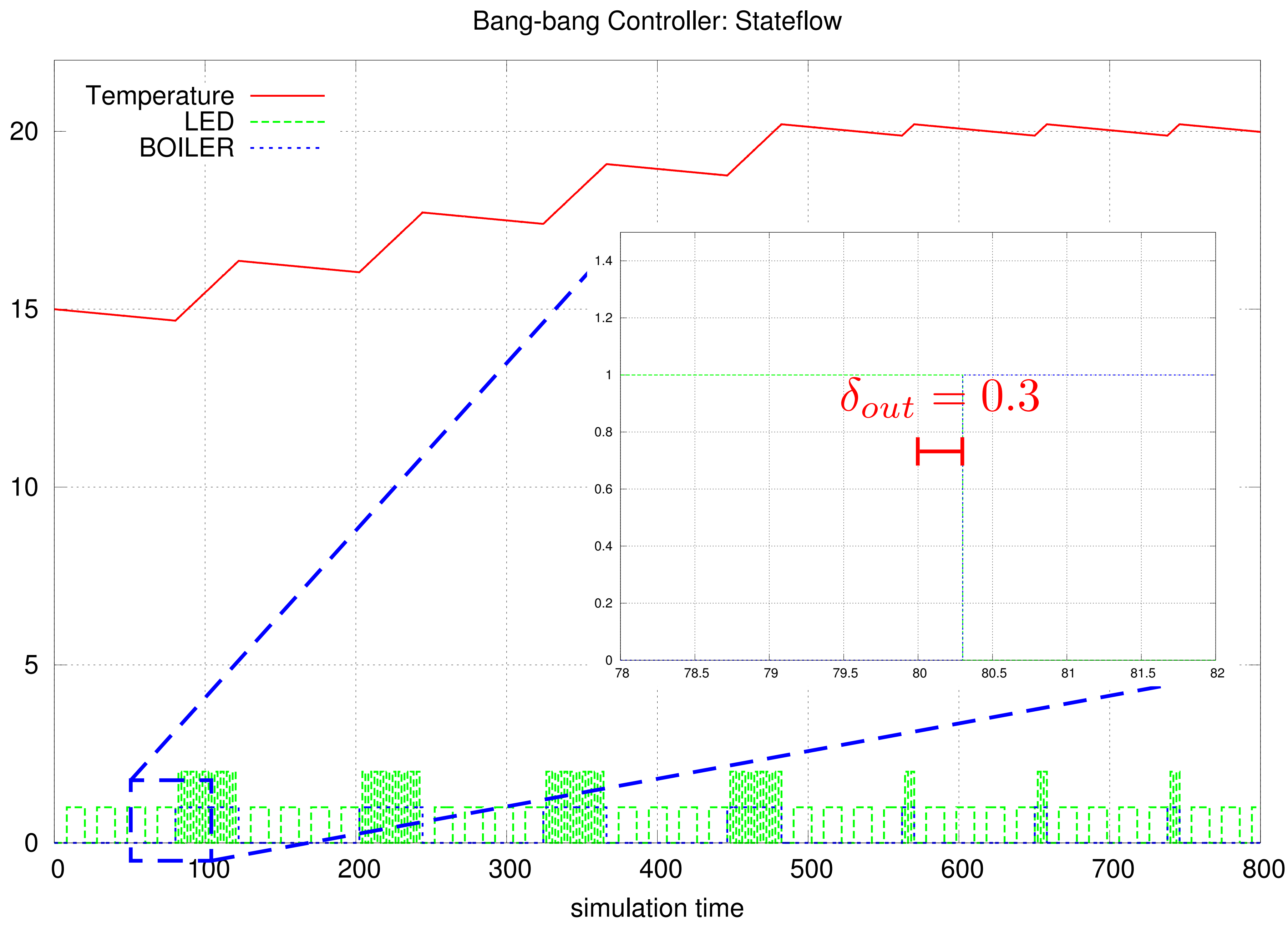
  - event-driven:

$\tau = 0$	inherited
otherwise	zero-crossings

Effect of parameters



Effect of parameters



Summary

$\surd$ Simulink and Stateflow

$\surd$ An Argos block

$\surd$ Timing Model

$\surd$ Embedding within Simulink

$\Rightarrow$ Concluding remarks

- Working prototype uses Argos.
- Timed automata framework clarifies implementation.
- Looking for case-studies to evaluate utility.

Concluding remarks

References

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994.

[Ber00] Gérard Berry. *The Esterel v5 Language Primer*. Ecole des Mines and INRIA, version 5.92 edition, June 2000.

[BS05] T. Bourke and A. Sowmya. Formal models in industry standard tools: An Argos block within Simulink. In Francis E.H. Tay, editor, *Int. J. Software Engineering and Knowledge Engineering: Selected Papers from the 2005 International Conference on Embedded and Hybrid Systems*, volume 15, pages 389–395, Singapore, April 2005. World Scientific.

[BST97] Sébastien Bornot, Joseph Sifakis, and Stavros Tripakis. Modeling urgency in timed systems. In Willem P. de Roever, Hans Langmaack, and Amir Pnueli, editors, *International Symp. Compositionality: The Significant Difference (COMPOS '97)*, volume 1536 of *Lecture Notes in Computer Science*, pages 103–129, Bad Malente, Germany, September 1997. Springer-Verlag.

[CCM$^+$03] Paul Caspi, Adrian Curic, Aude Maignan, Christos Sofronis, Stavros Tripakis, and Peter Niebert. From Simulink to SCADE/Lustre to TTA: a layered approach for distributed embedded applications. In *Proc. 2003 ACM SIGPLAN Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES '03)*, pages 153–162. ACM, ACM Press, 2003.

[Mar91] F. Maraninchi. The Argos language: Graphical representation of automata and description of reactive systems. In *Proc. IEEE Workshop on Visual Languages*, pages 254–259, October 1991.

[MR01] Florence Maraninchi and Yann Rémond. Argos: an automaton-based synchronous language. *Computer Languages*, 27(1–3):61–92, 2001.

[SSC+04] N. Scaife, C. Sofronis, P. Caspi, S. Tripakis, and F. Maraninchi. Defining and translating a “safe” subset of Simulink/Stateflow into Lustre. In G. Buttazzo, editor, *Proc. 4th ACM International Conference on Embedded Software (EMSOFT’04)*, pages 259–268, Pisa, Italy, September 2004. ACM, ACM Press.

[STY03] Joseph Sifakis, Stavros Tripakis, and Sergio Yovine. Building models of real-time systems from application software. *Proc. IEEE*, 91(1):100–111, January 2003.