Scheduling Directives
for Practical Causality Handling in Synchronous Languages

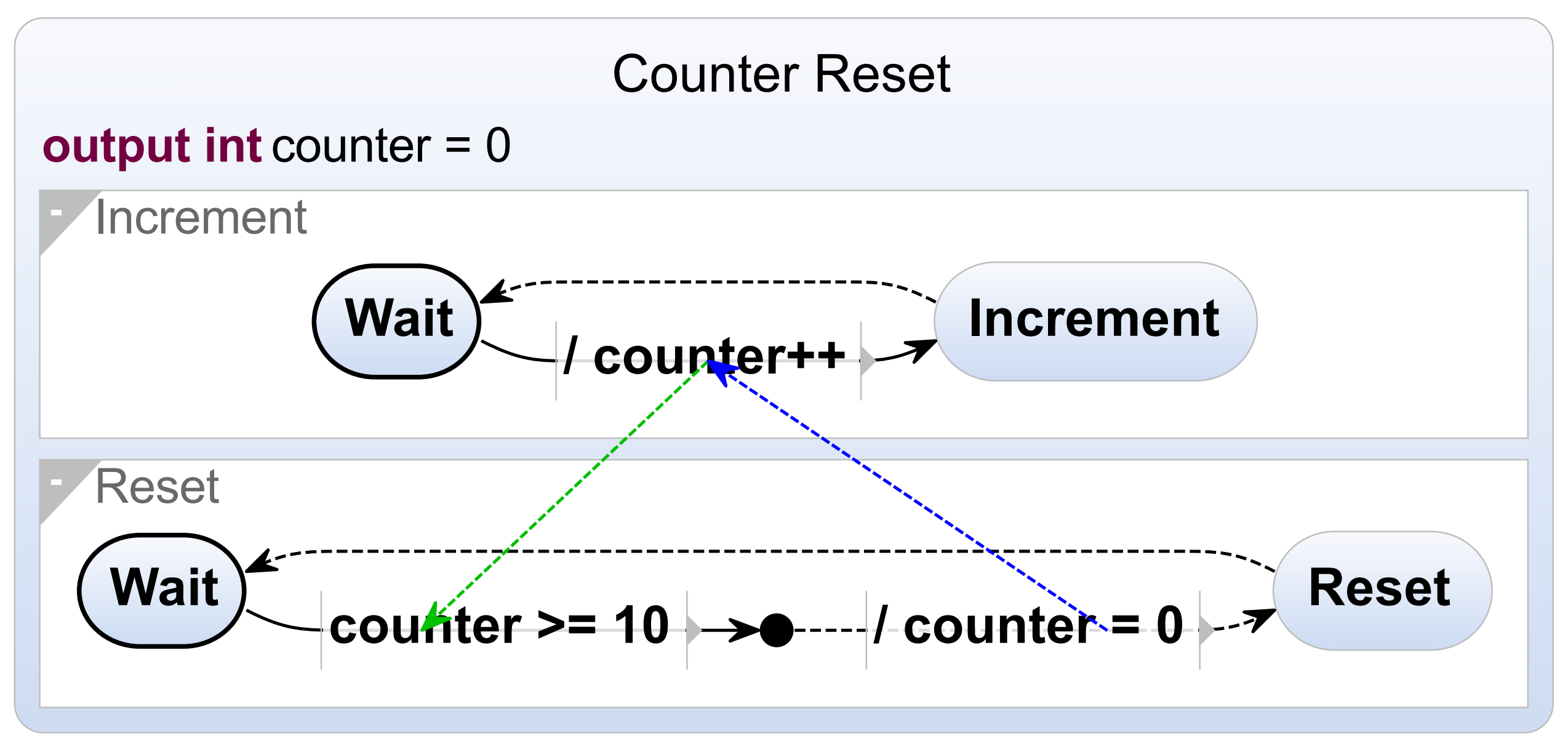
C | A | U

Kiel University

Faculty of Engineering

Department of Computer Science

Not Schedulable



Counter Reset Example in SCCharts [2]
Not schedulable due to cyclic dependencies

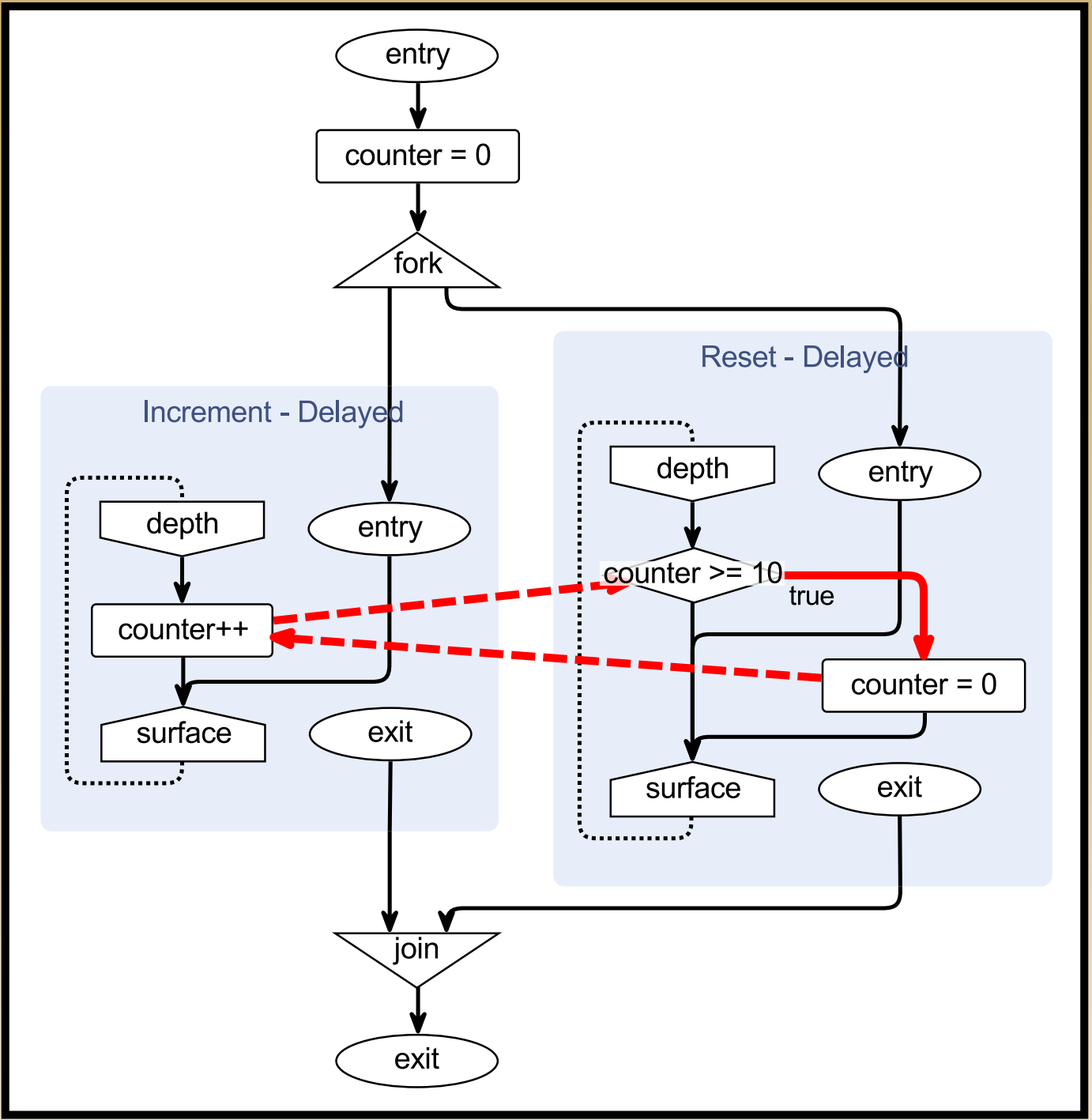
```
scchart CounterReset {
  output int counter = 0

  region Increment:
  initial state Wait
    do counter++
      go to Increment

  state Increment
    immediate go to Wait

  region Reset:
  initial state Wait
    if counter >= 10 go to Do

  connector state Do
    immediate do counter = 0
      go to Reset

  state Reset
    immediate go to Wait
}
```

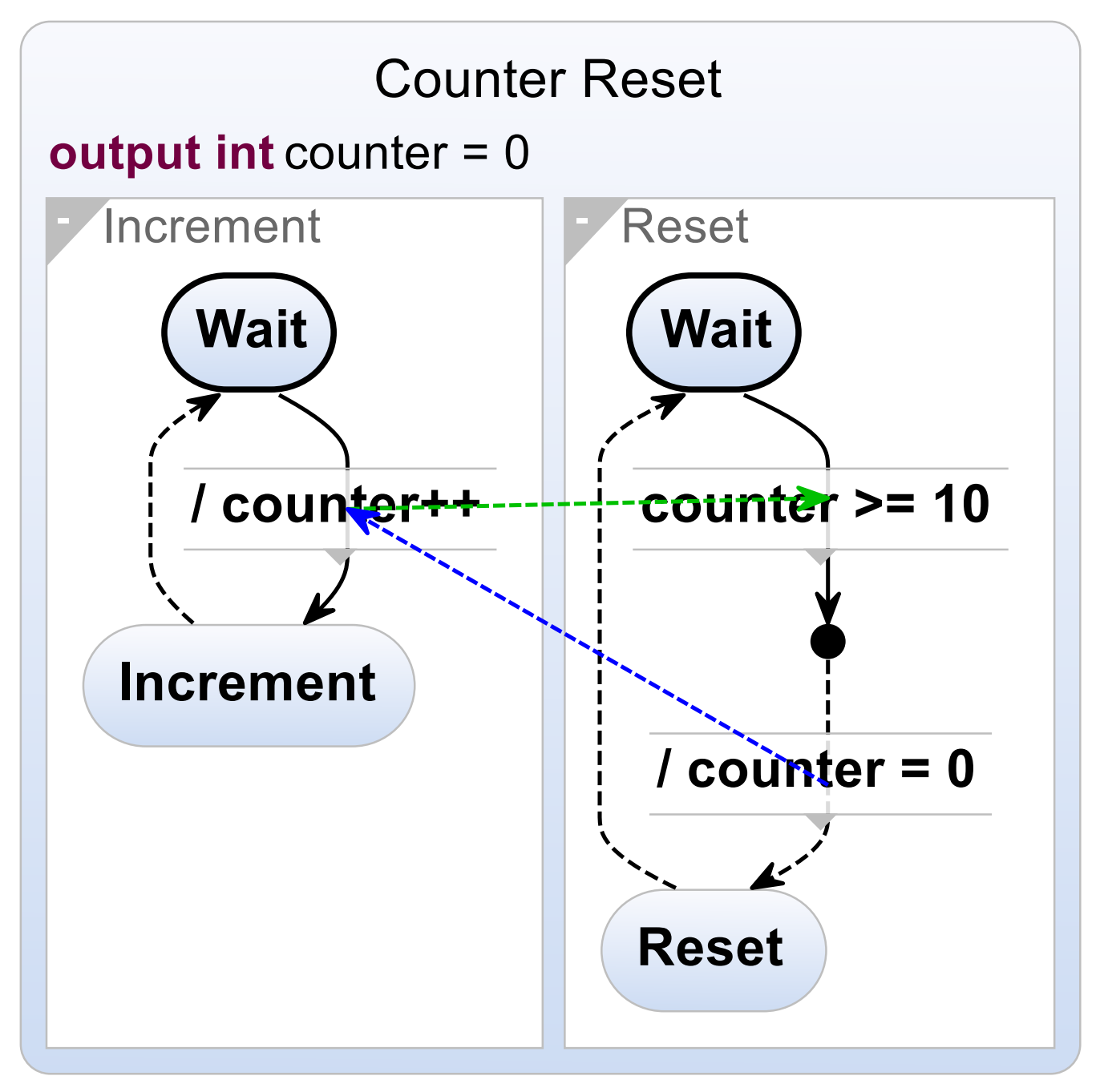
To reconcile concurrency and determinism, synchronous languages follow strictly defined models of computation (MoC). For example, the prominent write-before-read principle demands that a write to some variable *x* must be scheduled before a concurrent read of *x*. This clearly guarantees determinism, but like other scheduling rules comes at the price that a compiler may reject a program because it cannot find a viable schedule. We then say that the program is not causal, and it is the programmers job to fix the program.

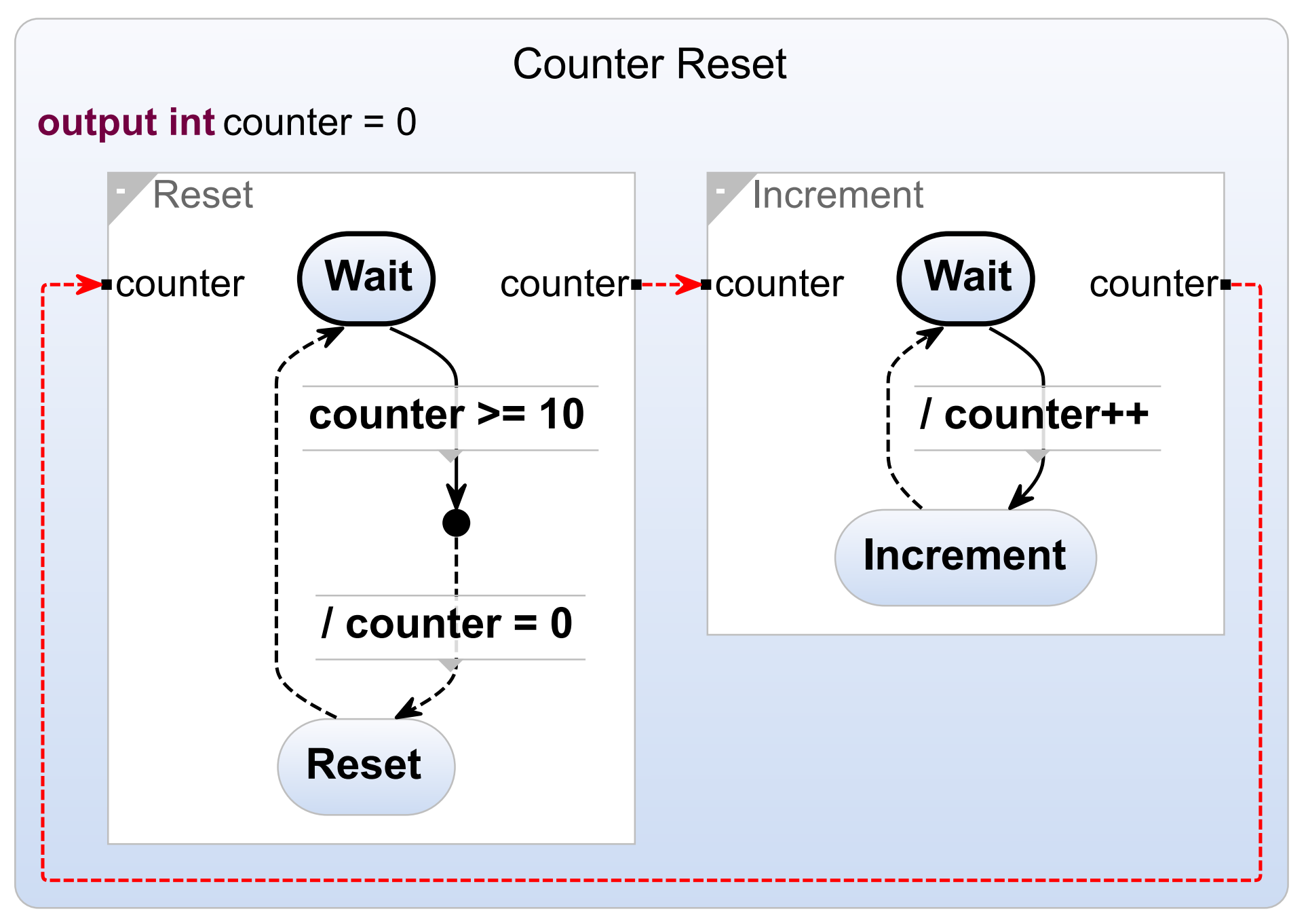
Curing causality issues, in practice, is often easier said than done, due to different reasons.

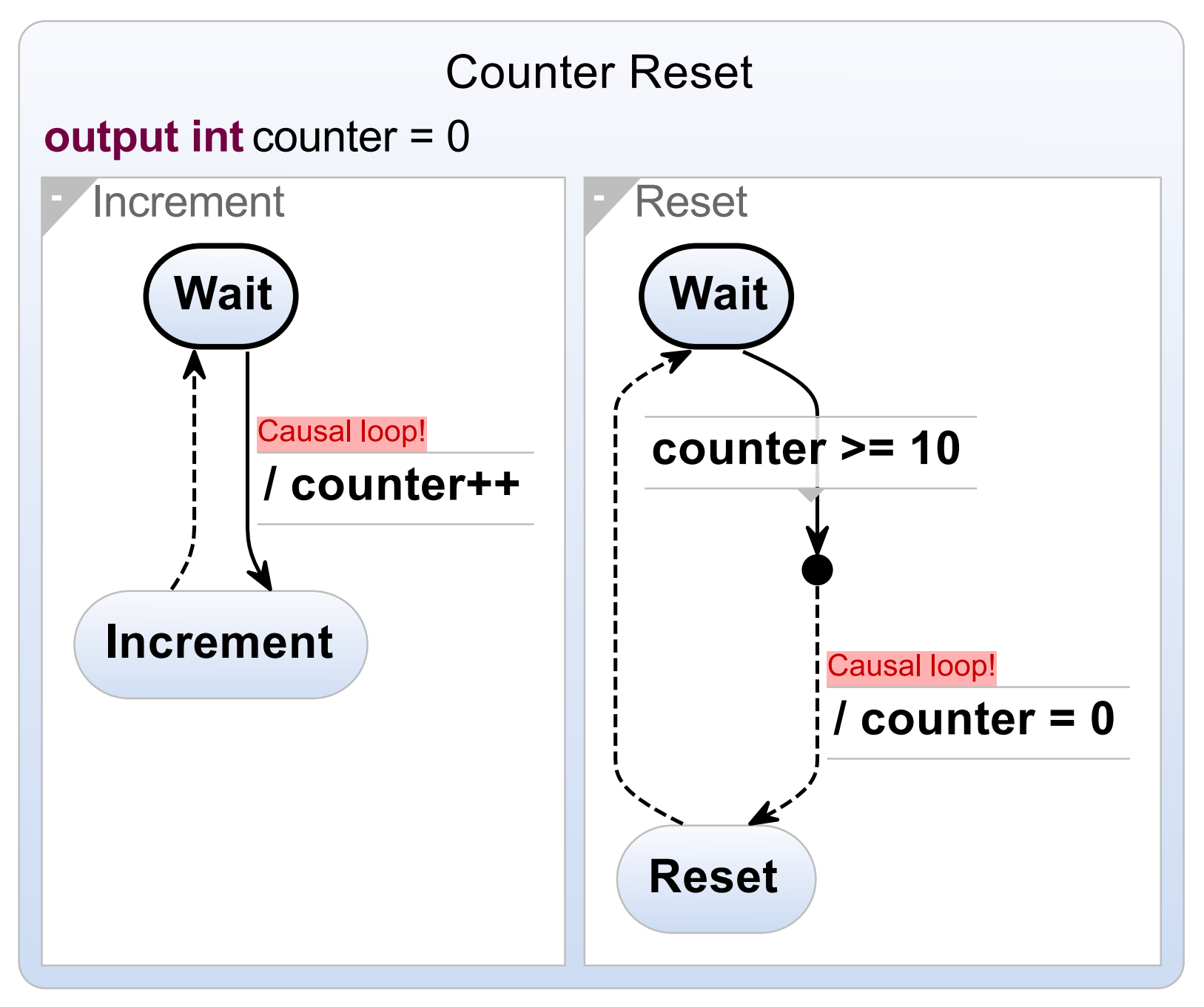
1) Some synchronous MoCs are restrictive in ways that the average programmer may not expect;
2) the compiler's analysis and scheduling abilities may be limited and conservatively reject programs that would indeed be schedulable; and
3) the feedback provided by the compiler may be too limited to be helpful to the programmer.

Guidance









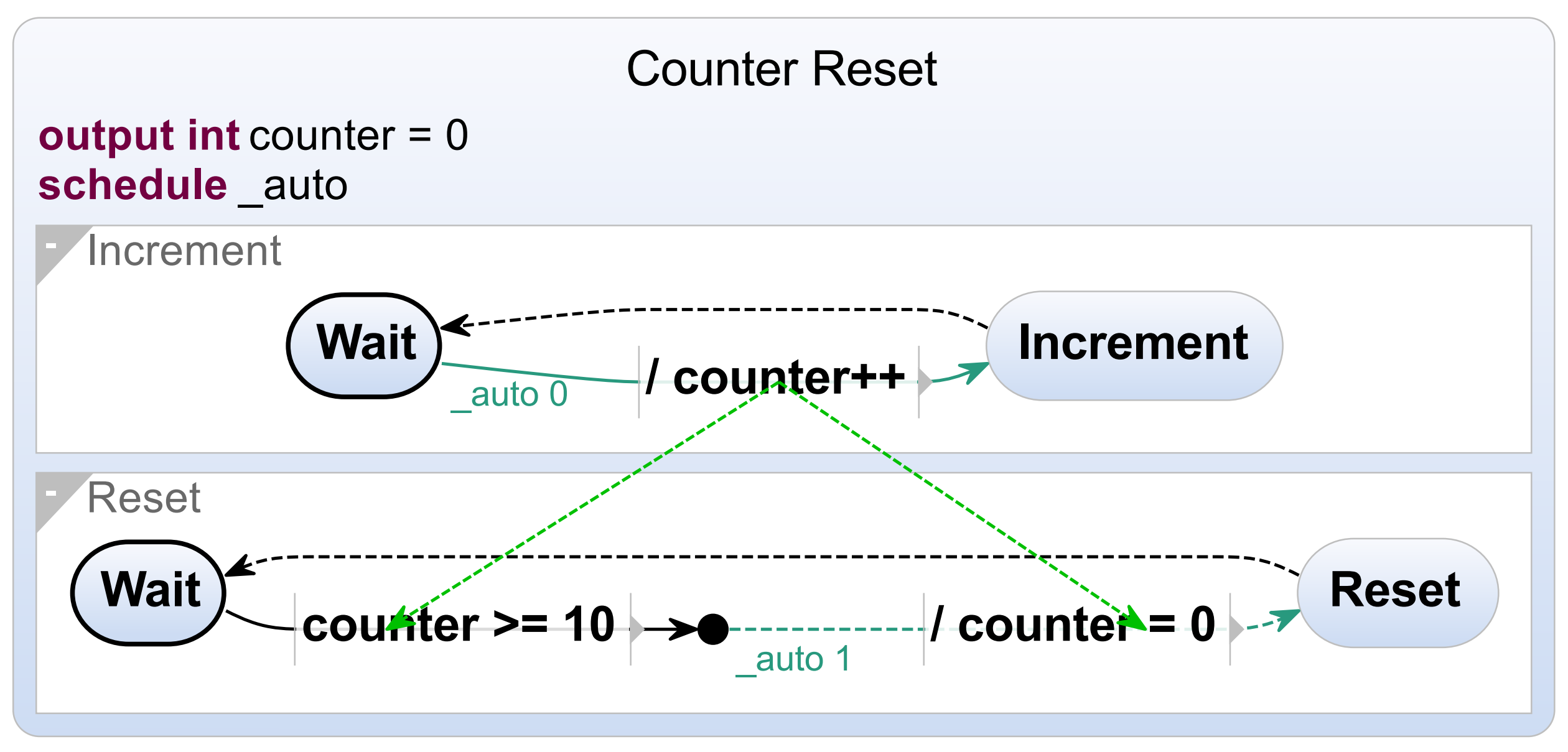
Classically dependency analyses is done on the program dependency graph. It is sometimes hard to map the original model to its control-flow graph representation. Nonetheless, the model is generated during compilation and, hence, available for inspection if necessary.

The data dependency visualization is used to identify individual conflicting data dependencies. The view augments the diagram with data dependencies that originate from variables accesses in the model. The modeler directly interacts with the diagram to add scheduling directives in a user-friendly way.

The induced dataflow view [4] shows communication between concurrent regions. It visualizes the dataflow of the program; even if the underlying model uses the control flow paradigm. The variant thereof depicted here, the causality dataflow view, focusses on identifying data dependency cycles. It highlights relationships similar to the dependency visualization, but on a different granularity. Causal cycles are depicted in red.

A common method of error reporting in tools are simple messages. These messages usually contain additional information about the location of the error to guide the user. In graphical languages, such a message can be embedded directly into the (original) model by annotating it. The KIELER framework [3] allows to create annotated models during compilation to hint at potential problems.

Schedulable



Counter Reset Example in SCCharts
Schedulable due to resolved cyclic dependencies
The name and the index of the scheduling directive is shown as teal transition tail label.

```
scchart CounterReset {
  output int counter = 0
  schedule _auto

  region Increment:
  initial state Wait
    do counter++
      schedule _auto 0
      go to Increment

  state Increment
    immediate go to Wait

  region Reset:
  initial state Wait
    if counter >= 10 go to Do

  connector state Do
    immediate do counter = 0
      schedule _auto 1
      go to Reset

  state Reset
    immediate go to Wait
}
```

If a conflict, induced by the underlying MoC, has been found, the modeler can alter the schedule interactively, either by editing the program or by simply clicking on the appropriate dependency edge.

A scheduling directive (SD) associates a scheduling unit with a named schedule and an index. The scheduling unit may be for example a single statement, or a coarser unit of execution, such as a thread.

A flexible schedule is a schedule that takes all SDs of the model into account. If there exists an SD for two statements, the SD order is used. Otherwise, the MoC determines the order.

For a model that contains scheduling conflicts, we propose to not consider it causally wrong per se, but merely incomplete. When a conflict occurs that leads to an incomplete model, the modeler can complete it with SDs. They can be used directly on different levels of detail and indirectly via model-to-model transformations

The exampe on the right uses a single SD to solve the scheduling conflict. The additional code is depicted in red.

Contact Persons

Steven Smyth,
Alexander Schulz-Rosengarten,
Prof. Dr. Reinhard von Hanxleden

Department of Computer Science,
Kiel University, Germany
Phone: +49 (0) 431 880-7290
ssm@/als@/rvh@informatik.uni-kiel.de
http://rtsys.informatik.uni-kiel.de

[1] S. Smyth, A. Schulz-Rosengarten, R. v. Hanxleden. Practical Causality Handling for Synchronous Languages.
In Proc. of Design, Automation and Test in Europe (DATE '19), Florence, Italy, Mar. 2019.

[2] R. v. Hanxleden, B. Duderstadt, C. Motika, S. Smyth, M. Mendler, J. Aguado, S. Mercer, O. O'Brien.
SCCharts: Sequentially Constructive Statecharts for Safety-Critical Applications.
In Proc. of ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '14), Edinburgh, UK, June 2014.

[3] S. Smyth, A. Schulz-Rosengarten, R. v. Hanxleden. Towards Interactive Compilation Models.
In Proc. of the 8th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA'18), Limassol, Cyprus, Oct. 2018.

[4] N. Wechselberg, A. Schulz-Rosengarten, S. Smyth, R. v. Hanxleden. Augmenting State Models with Data Flow
In Principles of Modeling: Essays Dedicated to Edward A. Lee on the Occasion of His 60th Birthday,
Springer International Publishing, 2018

The KIELER SCCharts Editor is part of

KIELER
The Key to Efficient Modeling

On the web:
http://www.informatik.uni-kiel.de/rtsys/kieler
http://www.sccharts.com