Forum on specification & Design Languages (FDL'20)

FDL stimulates scientific and controversial discussions in a friendly and productive environment.

New trends and traditional topics in the broad fields of embedded/electronics/software systems and languages merge in a lively and cross-discipline research & industrial community.

Calls for Special sessions, Full (8 pp), short (4 pp), and WiP/PhD Forum/Poster (2 pp) papers.

Keynotes: Edward Lee / UC Berkeley,
Manuel Serrano / Inria & Université Côte d'Azur,
Hauke Fuhrmann / Scheidt & Bachmann

7–9 September 2020 | Kiel, Germany

Deadlines:

Special Sessions:	March 22, 2020
Paper Deadline:	**May 29, 2020**
PhD/WiP Deadline:	June 12, 2020
Author Notification:	June 28, 2020
Final Version:	July 19, 2020

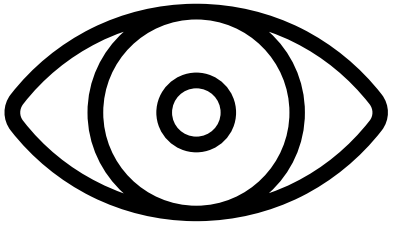
Website: www.fdl-conference.org | Contact: fdl2020@easychair.org

Re FDL'19: Open call for ACM TECS Special Issue on Specification and Design Languages
Deadline: Feb. 1, 2020 (firm)
Contacts: Alain Girault, Reinhard von Hanxleden

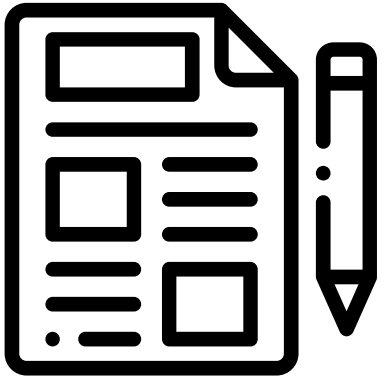
From Lustre to Graphical Dataflow Models

Lena Grimm
Reinhard von Hanxleden
Kiel University

Motivation

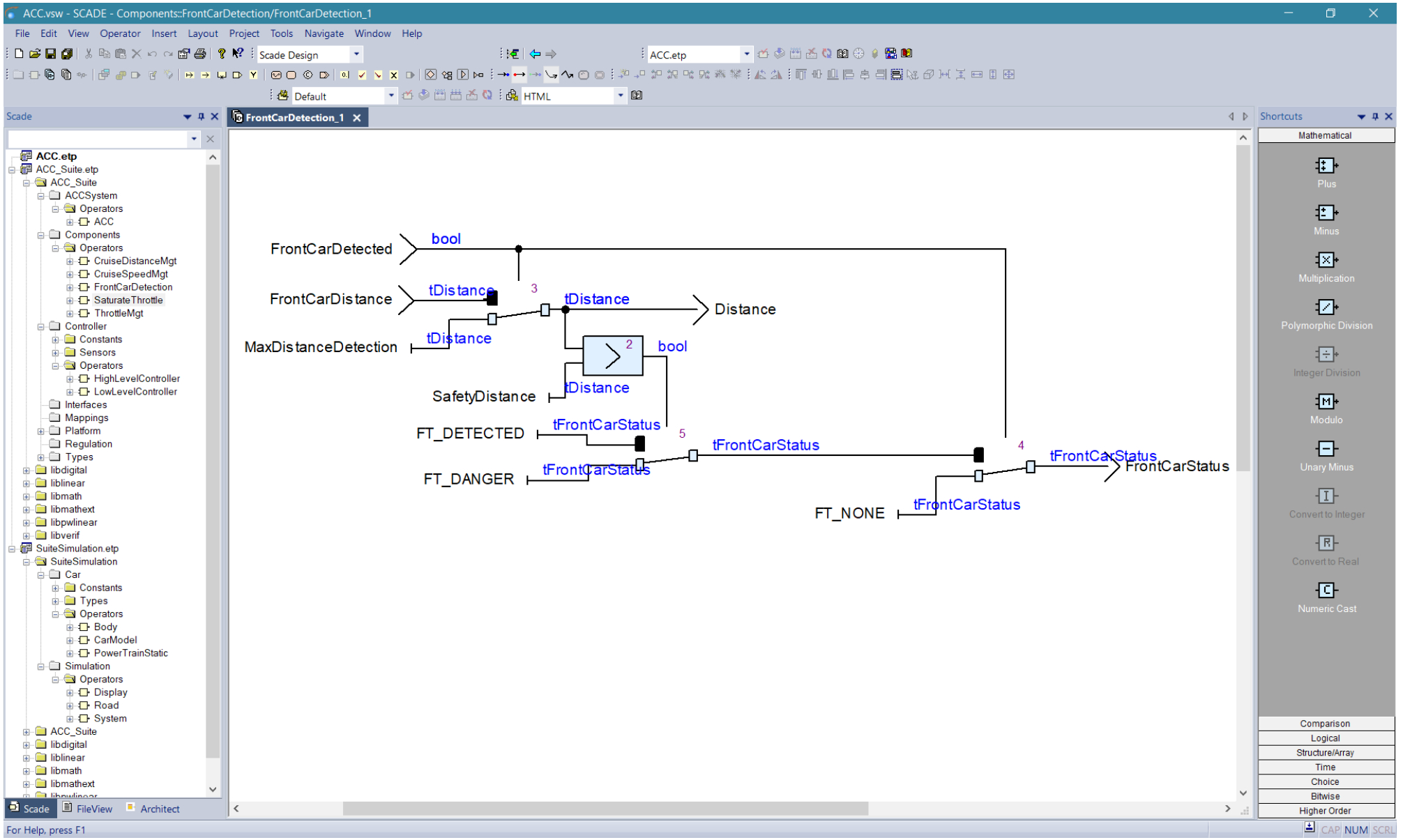


Visual

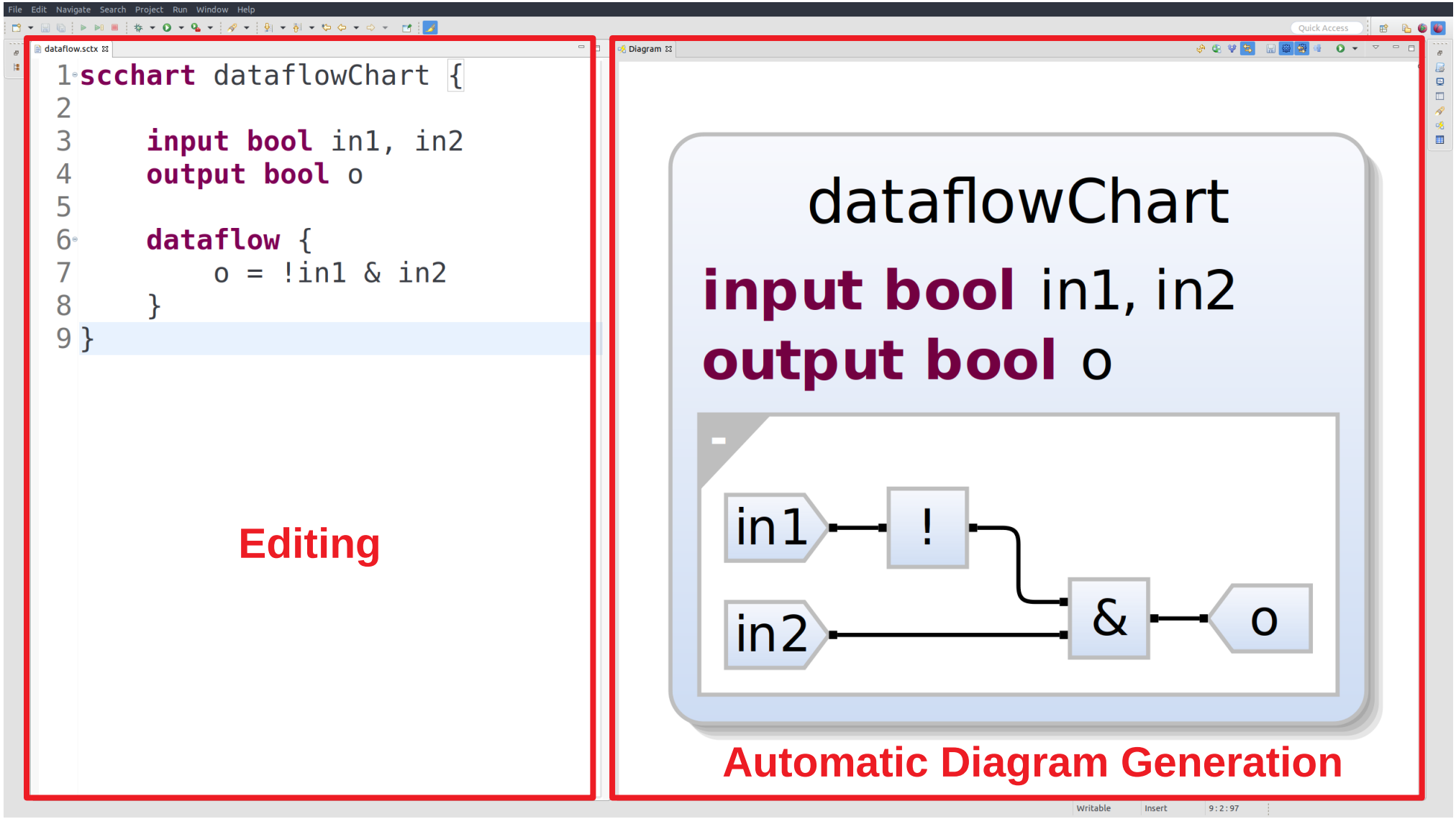


Concepts

SCADE

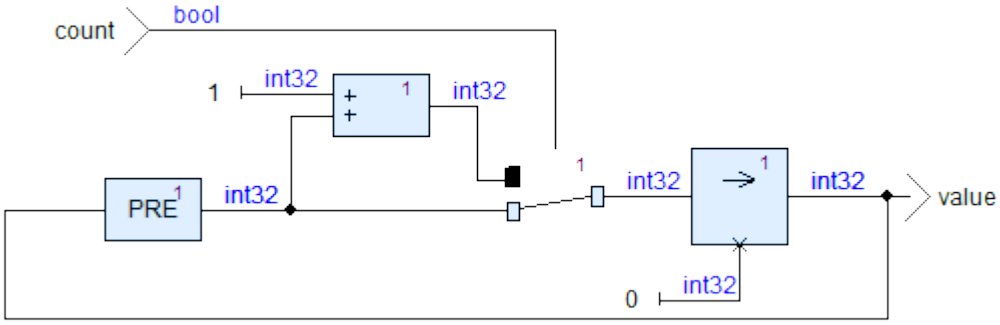

ACC.vsw - SCADE - Components::FrontCarDetection/FrontCarDetection_1
File Edit View Operator Insert Layout Project Tools Navigate Window Help
Scade Design
ACC.etp
Default
HTML
Scade
FrontCarDetection_1
Shortcuts
Mathematical
Plus
Minus
Multiplication
Polymorphic Division
Integer Division
Modulo
Unary Minus
Convert to Integer
Convert to Real
Numeric Cast
Comparison
Logical
Structure/Array
Time
Choice
Bitwise
Higher Order
ACC.etp
ACC_Suite.etp
ACC_Suite
ACCSystem
Operators
ACC
Components
Operators
CruiseDistanceMgt
CruiseSpeedMgt
FrontCarDetection
SaturateThrottle
ThrottleMgt
Controller
Constants
Sensors
Operators
HighLevelController
LowLevelController
Interfaces
Mappings
Platform
Regulation
Types
libdigital
liblinear
libmath
libmathext
libpwlinear
libverif
SuiteSimulation.etp
SuiteSimulation
Car
Constants
Types
Operators
Body
CarModel
PowerTrainStatic
Simulation
Operators
Display
Road
System
ACC_Suite
libdigital
liblinear
libmath
libmathext
FrontCarDetected
bool
FrontCarDistance
tDistance
3
tDistance
Distance
MaxDistanceDetection
tDistance
2
bool
SafetyDistance
tDistance
FT_DETECTED
tFrontCarStatus
5
tFrontCarStatus
4
tFrontCarStatus
FrontCarStatus
FT_DANGER
tFrontCarStatus
FT_NONE
tFrontCarStatus
Scade
FileView
Architect
For Help, press F1
CAP NUM SCRL

KIELER


File Edit Navigate Search Project Run Window Help
dataflow.sctx
1 scchart dataflowChart {
2
3 input bool in1, in2
4 output bool o
5
6 dataflow {
7 o = !in1 & in2
8 }
9 }
Editing
Diagram
dataflowChart
input bool in1, in2
output bool o
in1
!
in2
&
o
Automatic Diagram Generation
Writable
Insert
9:2:97

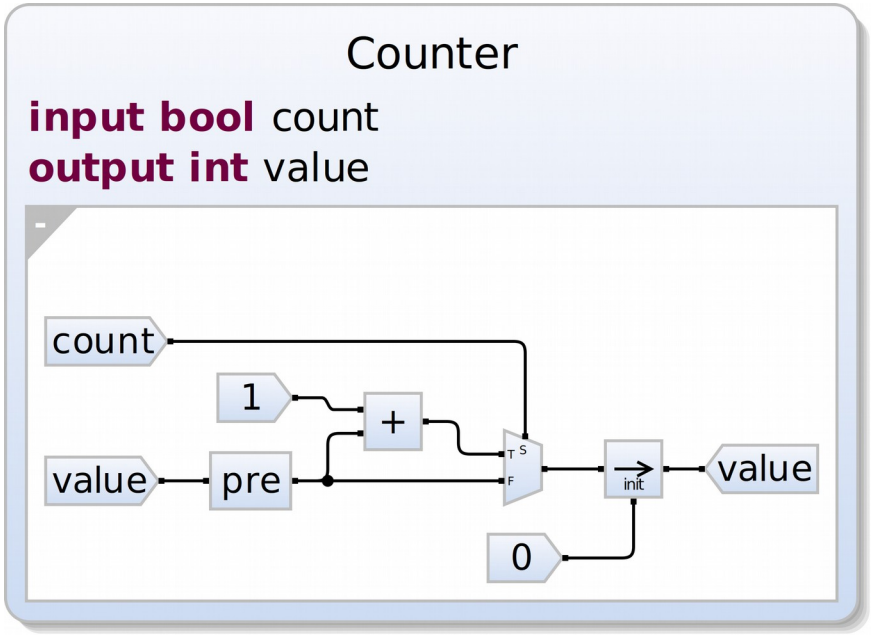
SCADE

```
L1 = count;
value = L7;
L2 = 1;
L3 = pre L7;
L4 = L2 + L3;
L5 = if L1 then (L4) else (L3);
L6 = 0;
L7 = (L6) -> (L5);
```

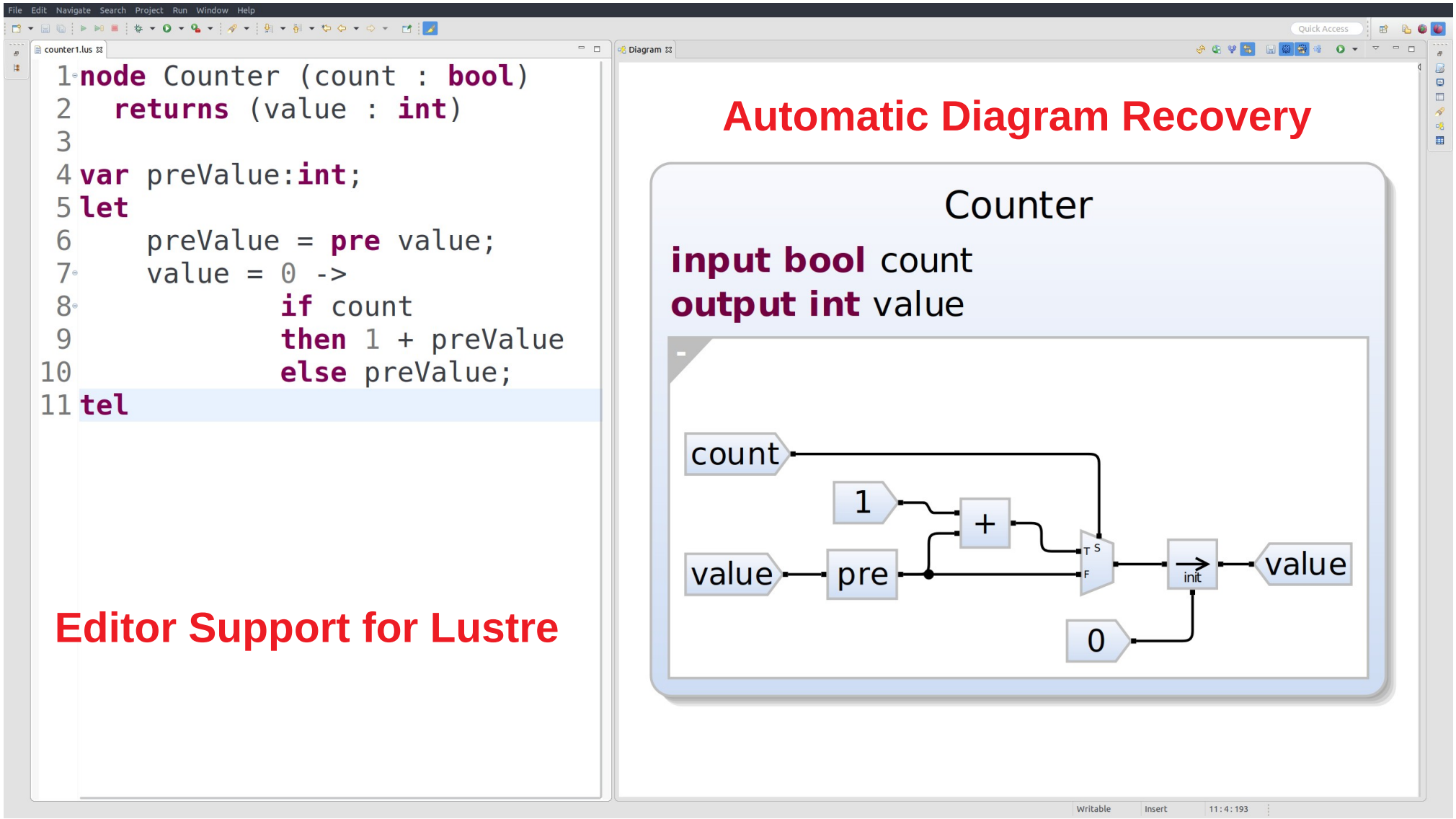


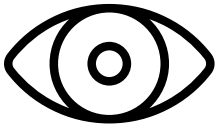
```
node Counter (count:bool) returns (value:int)
var L1:bool;
L2, L3, L4, L5, L6, L7:int;
let
  L1 = count;
  value = L7;
  L2 = 1;
  L3 = pre L7;
  L4 = L2 + L3;
  L5 = if L1 then (L4) else (L3);
  L6 = 0;
  L7 = (L6) -> (L5);
tel
```

KIELER



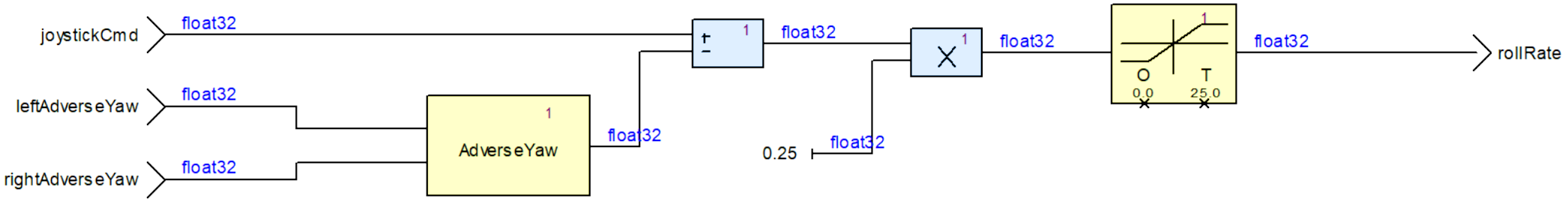
KIELER


File Edit Navigate Search Project Run Window Help
counter1.lus
1 node Counter (count : bool)
2 returns (value : int)
3
4 var preValue:int;
5 let
6 preValue = pre value;
7 value = 0 ->
8 if count
9 then 1 + preValue
10 else preValue;
11 tel
Editor Support for Lustre
Diagram
Automatic Diagram Recovery
Counter
input bool count
output int value
count
1
+
value
pre
T S
F
init
value
0
Writable
Insert
11 : 4 : 193



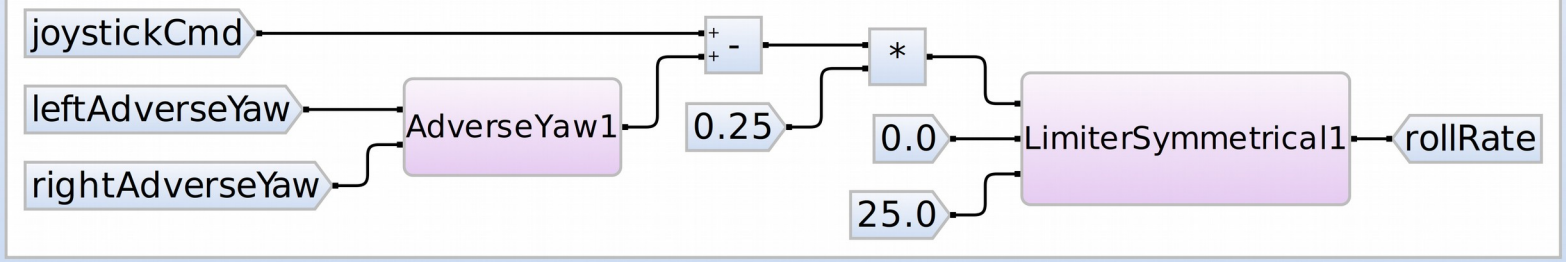
Demo

Larger Example

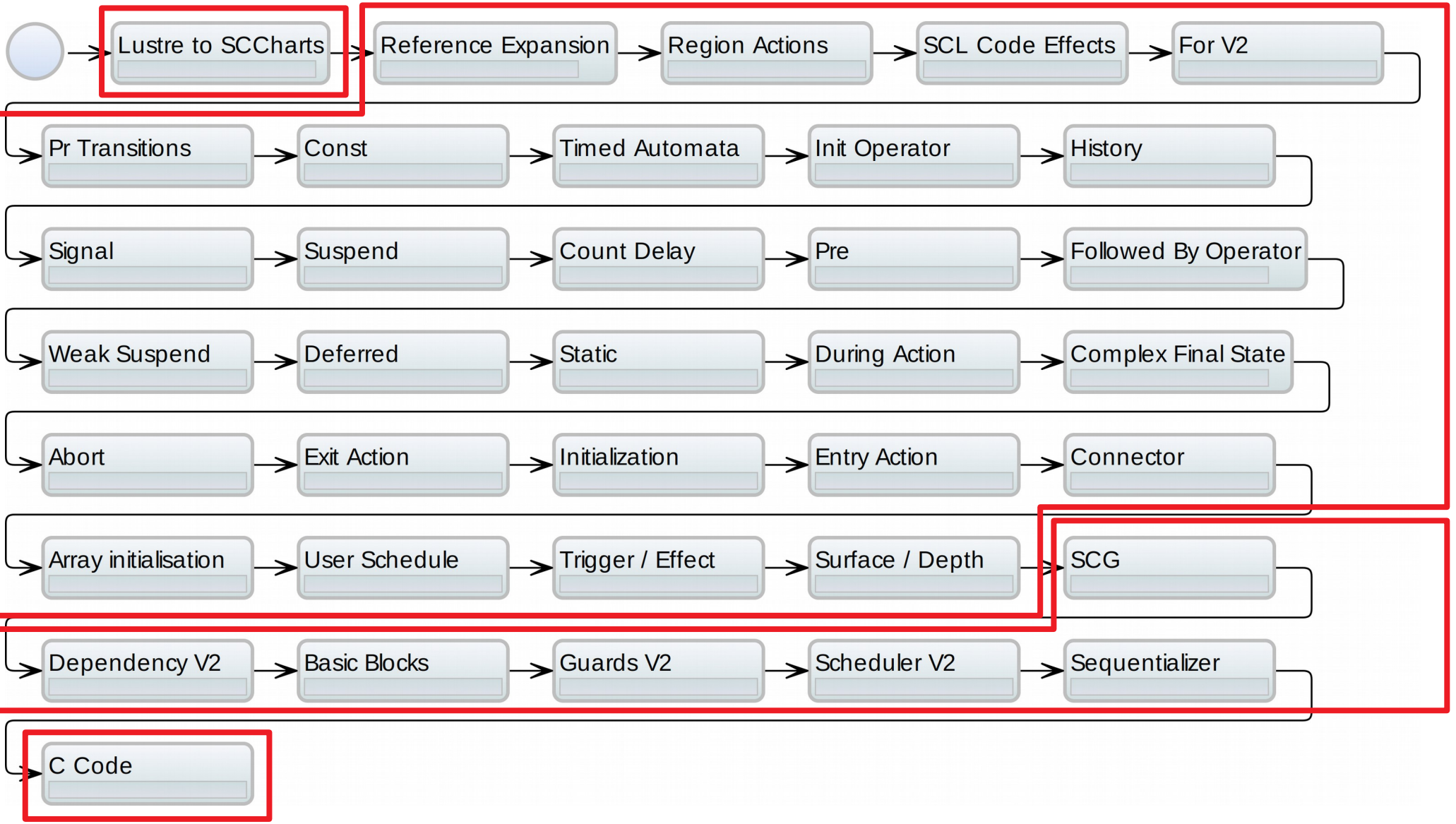


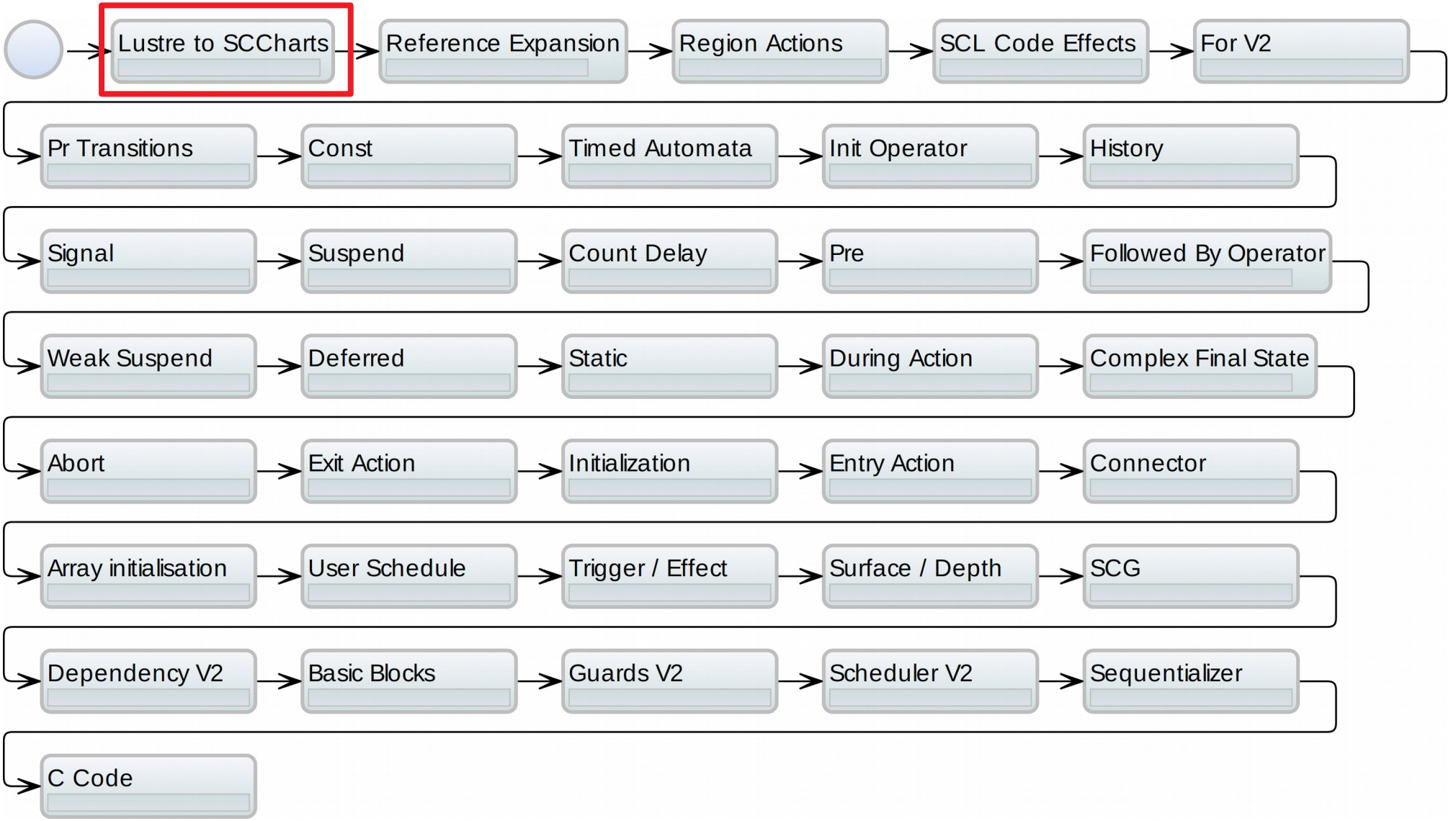
RollRateCalculate

input float joystickCmd
input float leftAdverseYaw
input float rightAdverseYaw
output float rollRate
ref AdverseYaw AdverseYaw1
ref LimiterSymmetrical LimiterSymmetrical1



KIELER Compilation Chain




Lustre to SCCharts
Reference Expansion
Region Actions
SCL Code Effects
For V2
Pr Transitions
Const
Timed Automata
Init Operator
History
Signal
Suspend
Count Delay
Pre
Followed By Operator
Weak Suspend
Deferred
Static
During Action
Complex Final State
Abort
Exit Action
Initialization
Entry Action
Connector
Array initialisation
User Schedule
Trigger / Effect
Surface / Depth
SCG
Dependency V2
Basic Blocks
Guards V2
Scheduler V2
Sequentializer
C Code

Transformation Challenges

Lustre

x	1	2	3	4	5	6	7	8	9
clk	true	false	true	false	false	true	false	true	true
x when clk	1		3			6		8	9

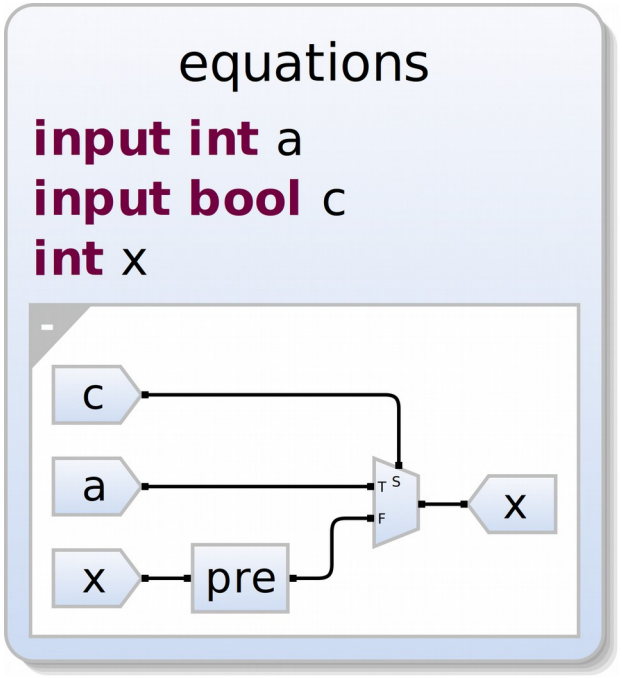
when, current?

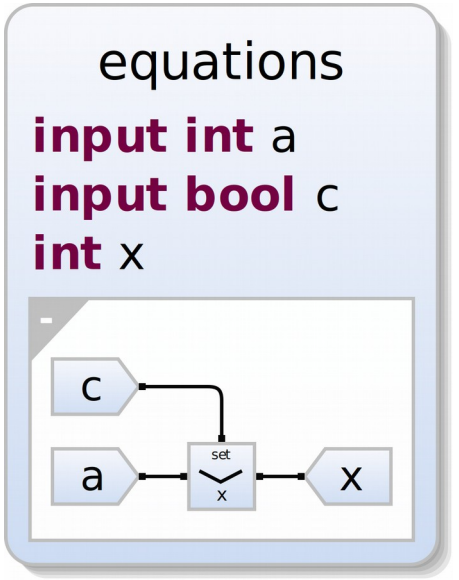
SCCharts

x	1	2	3	4	5	6	7	8	9
clk	true	false	true	false	false	true	false	true	true
clk? x	1	1	3	3	3	6	6	8	9

When Operator

```
node equations(a:int; c:bool)
  returns ();

var x:int when c;
let
  x = a when c;
tel.
```






```
scchart equations {
  input int a
  input bool c
  int x

  dataflow {
    x = c ? a : pre(x)
  }
}
```

```
scchart equations {
  input int a
  input bool c
  int x

  dataflow {
    x = c ? a
  }
}
```

When Operator

clk	true	false	true	false	true	true	false	false	true
x	true	false	false	true	true	false	false	false	true
y	true	false	false	true	false	false	true	true	false

xClk = x when clk	true		false		true	false			true
y when xClk	true				false				false

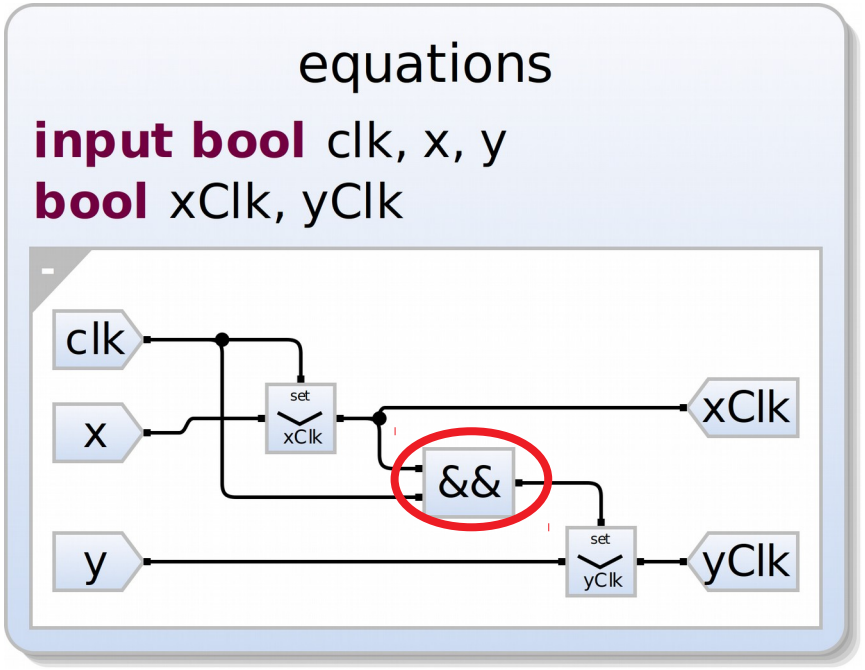
When Operator with Variables I

clk	true	false	true	false	true	true	false	false	true
x	true	false	false	true	true	false	false	false	true
y	true	false	false	true	false	false	true	true	false

xClk = clk? x	true	true	false	false	true	false	false	false	true
xClk? y	true	**false**	false	false	false	false	false	false	false

When Operator

```
node equations(clk,x,y:bool)
  returns ();

var xClk:bool when clk;
    yClk:bool when xClk;
let
  xClk = x when clk;
  yClk = y when xClk;
tel.
```




```
scchart equations {
  input bool clk, x, y
  bool xClk, yClk

  dataflow {
    xClk = clk ? x
    yClk = (xClk && clk) ? y
  }
}
```

When Operator with Variables II

clk	true	false	true	false	true	true	false	false	true
x	true	false	false	true	true	false	false	false	true
y	true	false	false	true	false	false	true	true	false

xClk = clk? x	true	true	false	false	true	false	false	false	true
(clk && xClk)? y	true	true	true	true	false	false	false	false	false

Current

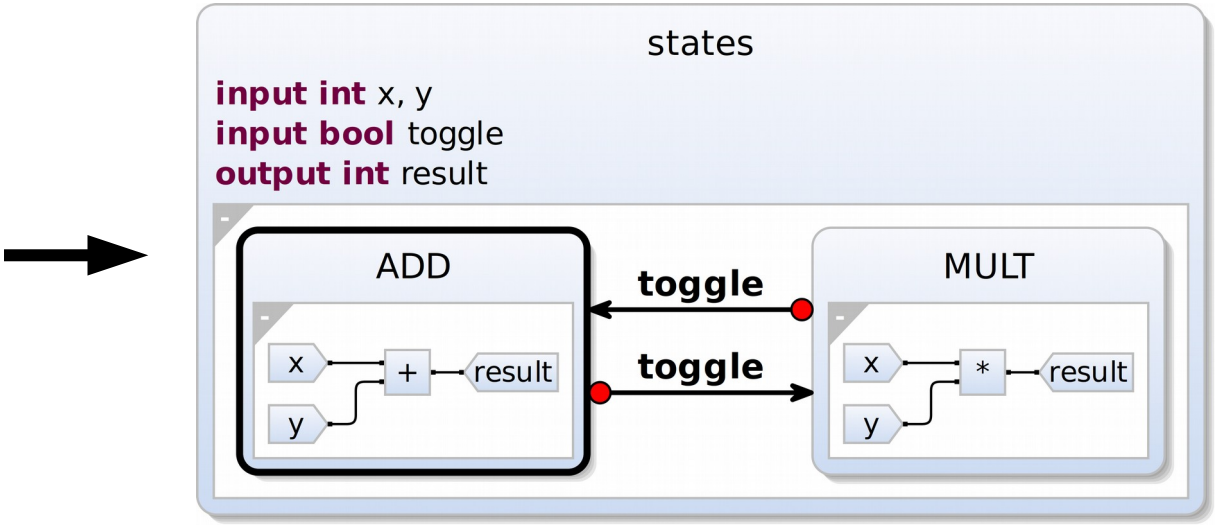
clk	true	false	true	false	true	true	false	false	true
x	true	false	false	true	true	false	false	false	true
y	true	false	false	true	false	false	true	true	false

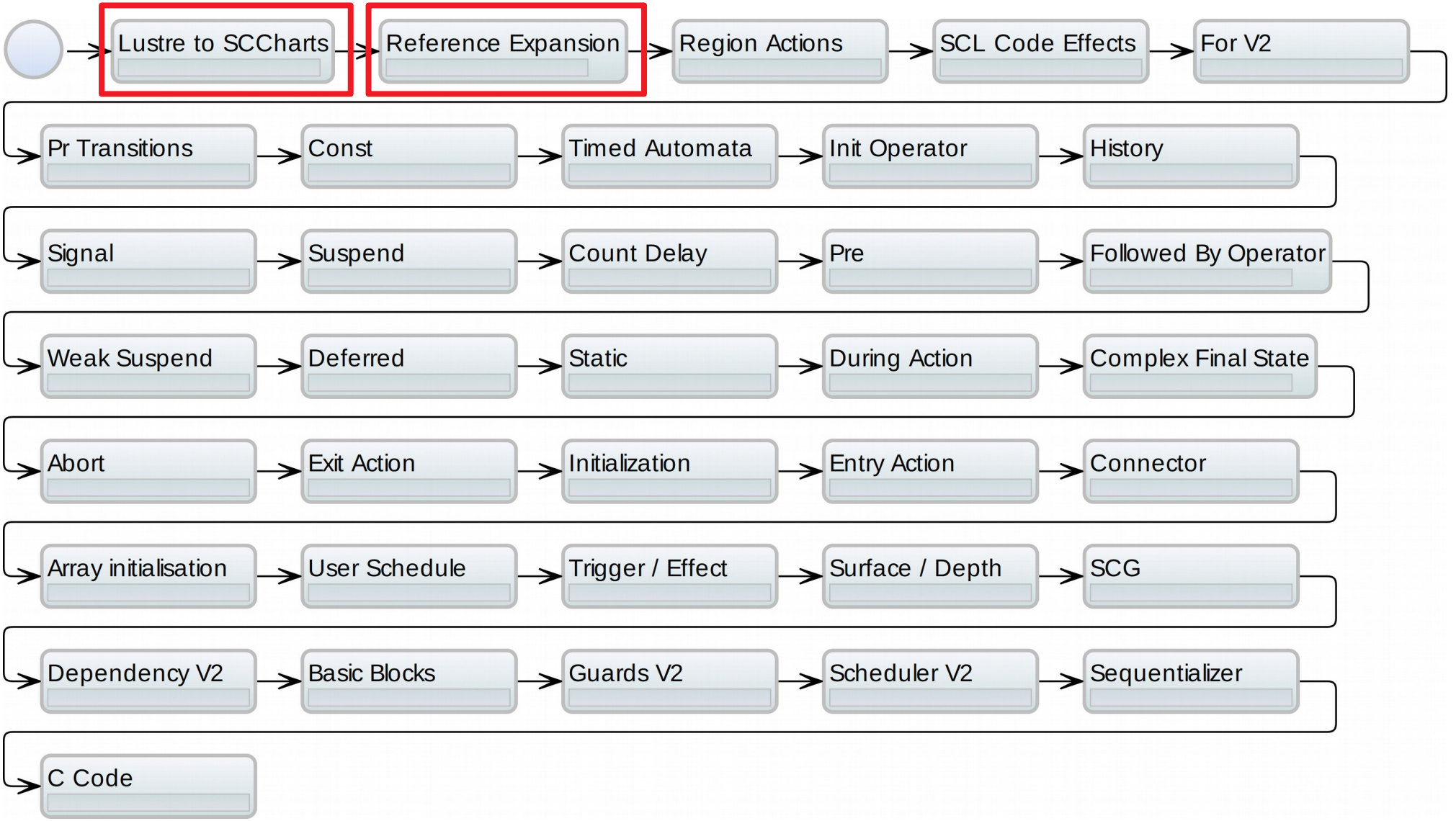
xClk = clk? x	true	true	false	false	true	false	false	false	true
(clk && xClk)? y	true	true	true	true	false	false	false	false	false

Always implicit ‘`current`’ through Variables

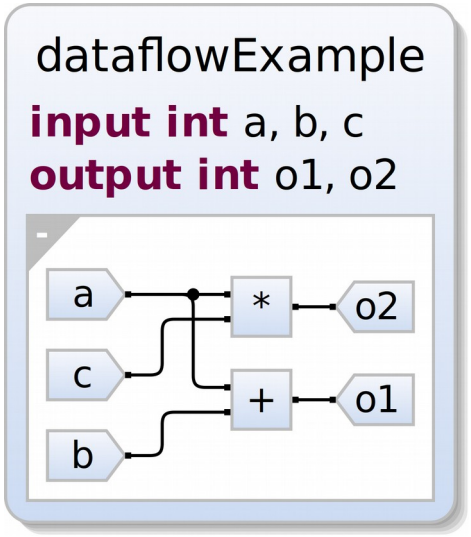
State Extension

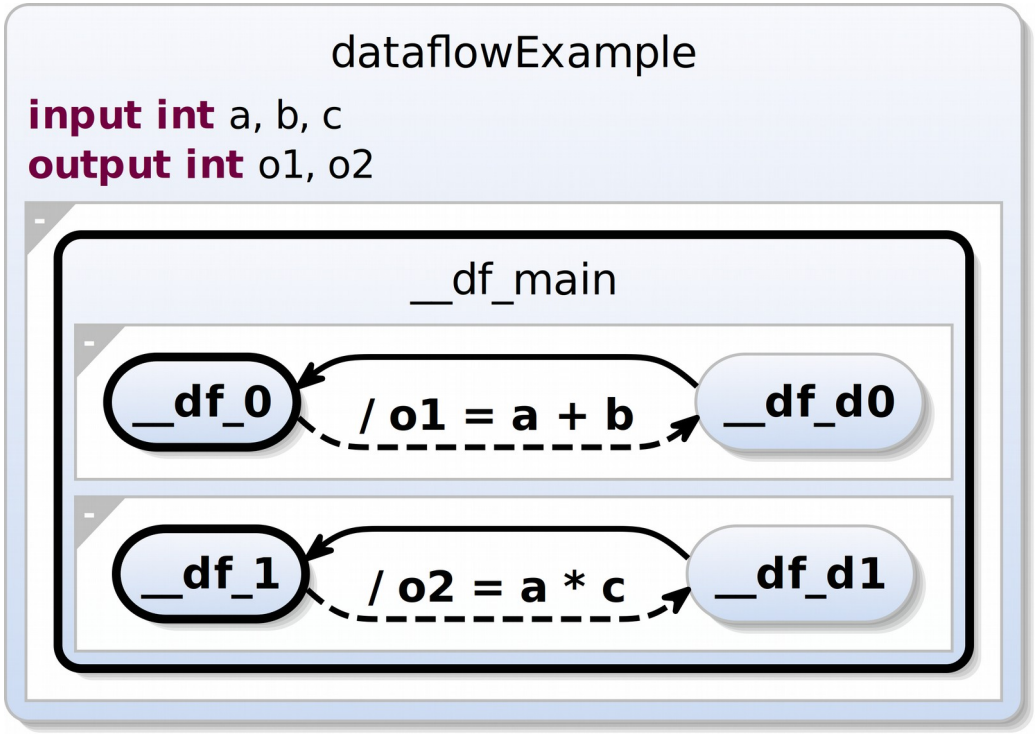
```
node states (x,y:int; toggle:bool)
    returns (result:int);
let
  automaton Calculator

    state ADD
    let
      result = x + y;
    tel
    unless if toggle restart MULT;

    state MULT
    let
      result = x * y;
    tel
    unless if toggle restart ADD;

  returns ..;
tel
```

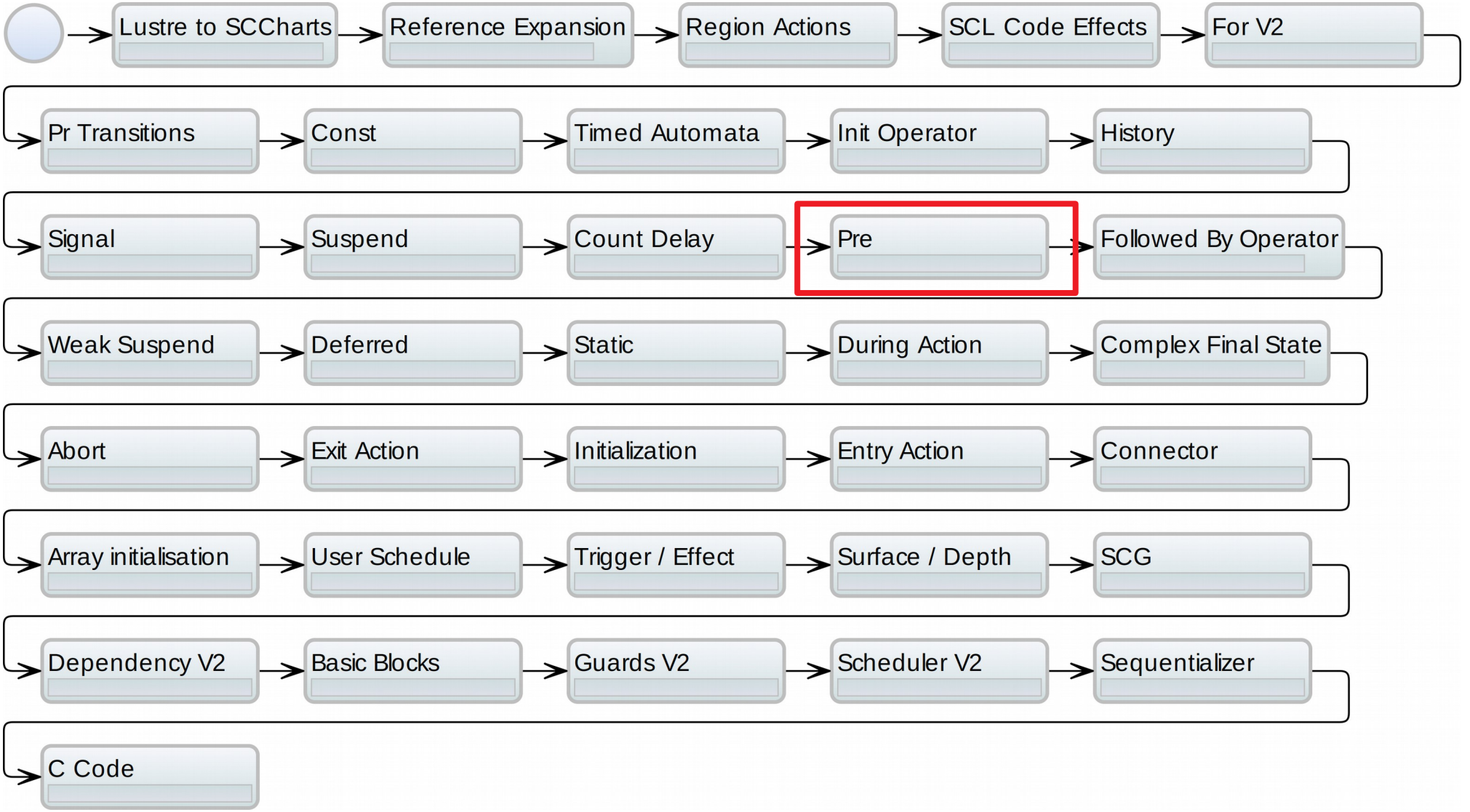



Lustre to SCCharts
Reference Expansion
Region Actions
SCL Code Effects
For V2
Pr Transitions
Const
Timed Automata
Init Operator
History
Signal
Suspend
Count Delay
Pre
Followed By Operator
Weak Suspend
Deferred
Static
During Action
Complex Final State
Abort
Exit Action
Initialization
Entry Action
Connector
Array initialisation
User Schedule
Trigger / Effect
Surface / Depth
SCG
Dependency V2
Basic Blocks
Guards V2
Scheduler V2
Sequentializer
C Code

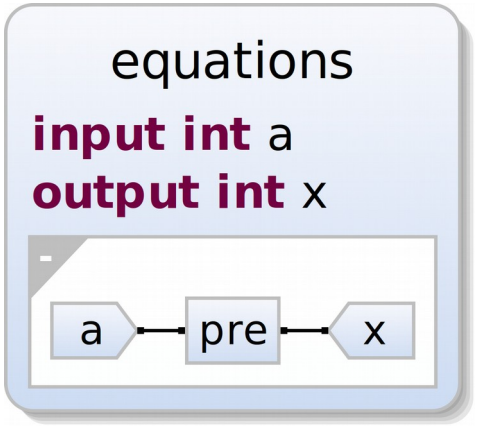
SCCharts Dataflow Semantics

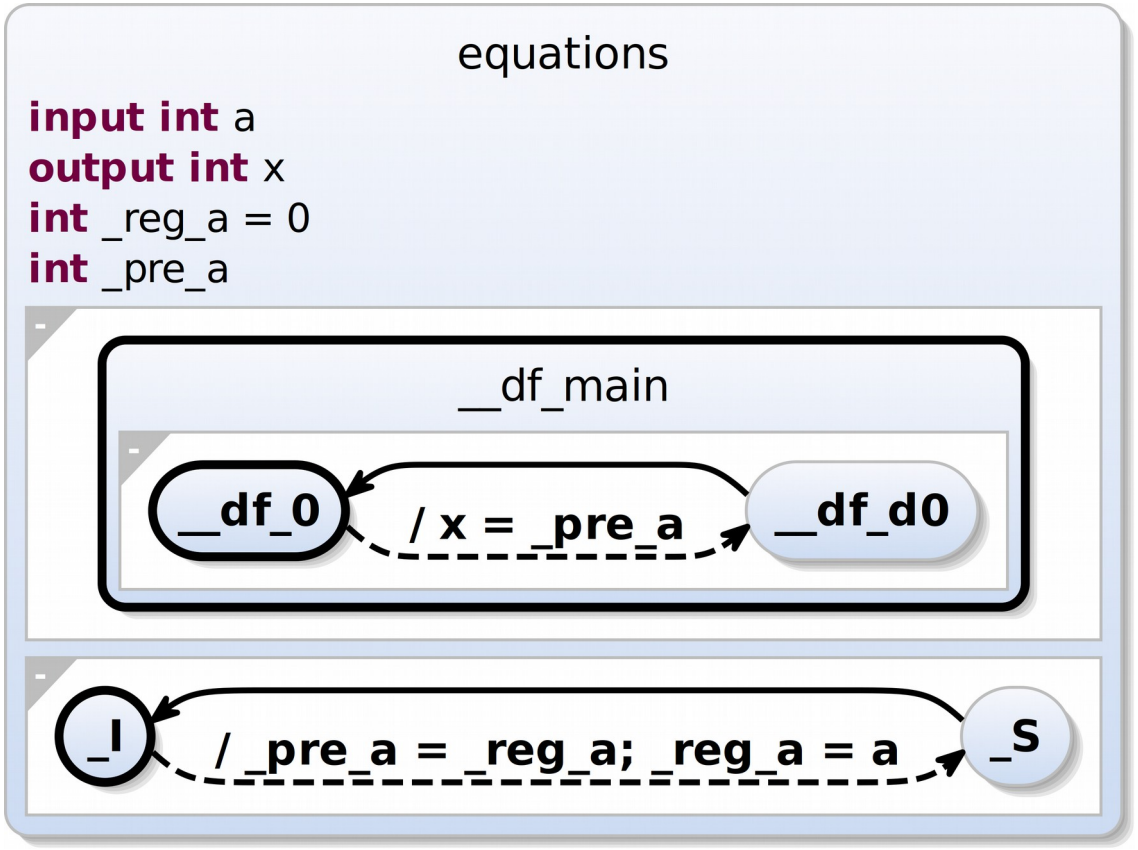





Lustre to SCCharts
Reference Expansion
Region Actions
SCL Code Effects
For V2
Pr Transitions
Const
Timed Automata
Init Operator
History
Signal
Suspend
Count Delay
Pre
Followed By Operator
Weak Suspend
Deferred
Static
During Action
Complex Final State
Abort
Exit Action
Initialization
Entry Action
Connector
Array initialisation
User Schedule
Trigger / Effect
Surface / Depth
SCG
Dependency V2
Basic Blocks
Guards V2
Scheduler V2
Sequentializer
C Code

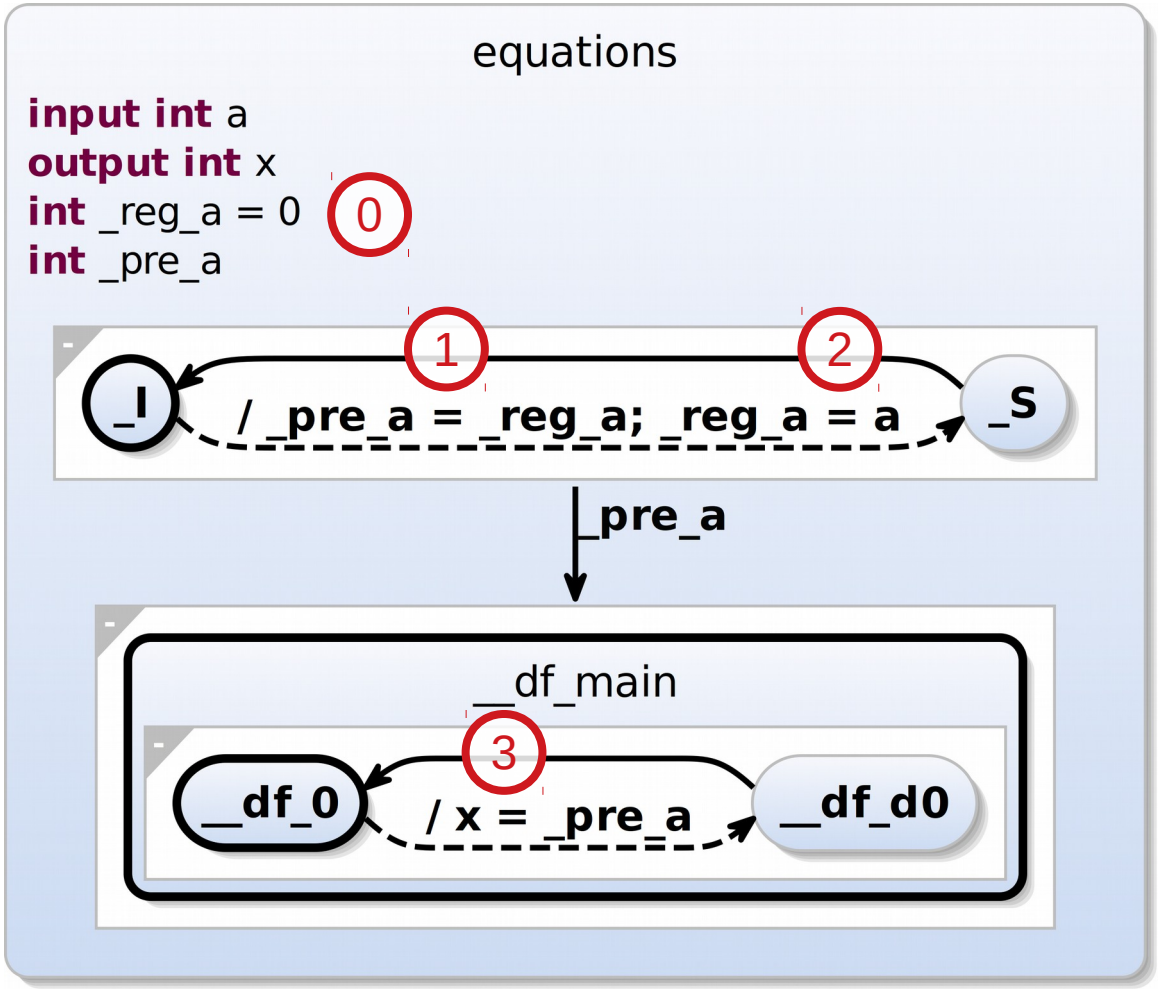
Pre Operator in SCCharts






```
scchart equations {
    input int a
    output int x

    dataflow {
        x = pre(a)
    }
}
```

Pre Operator in SCCharts Induced Dataflow View



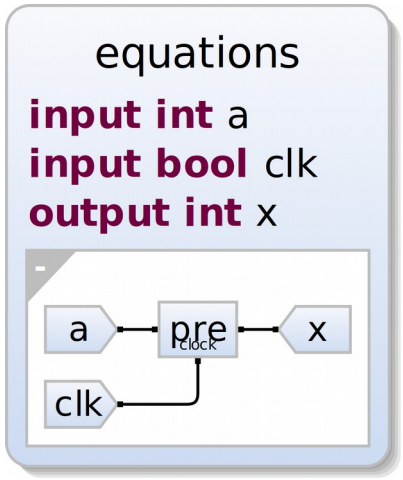
Pre Operator with Clocks

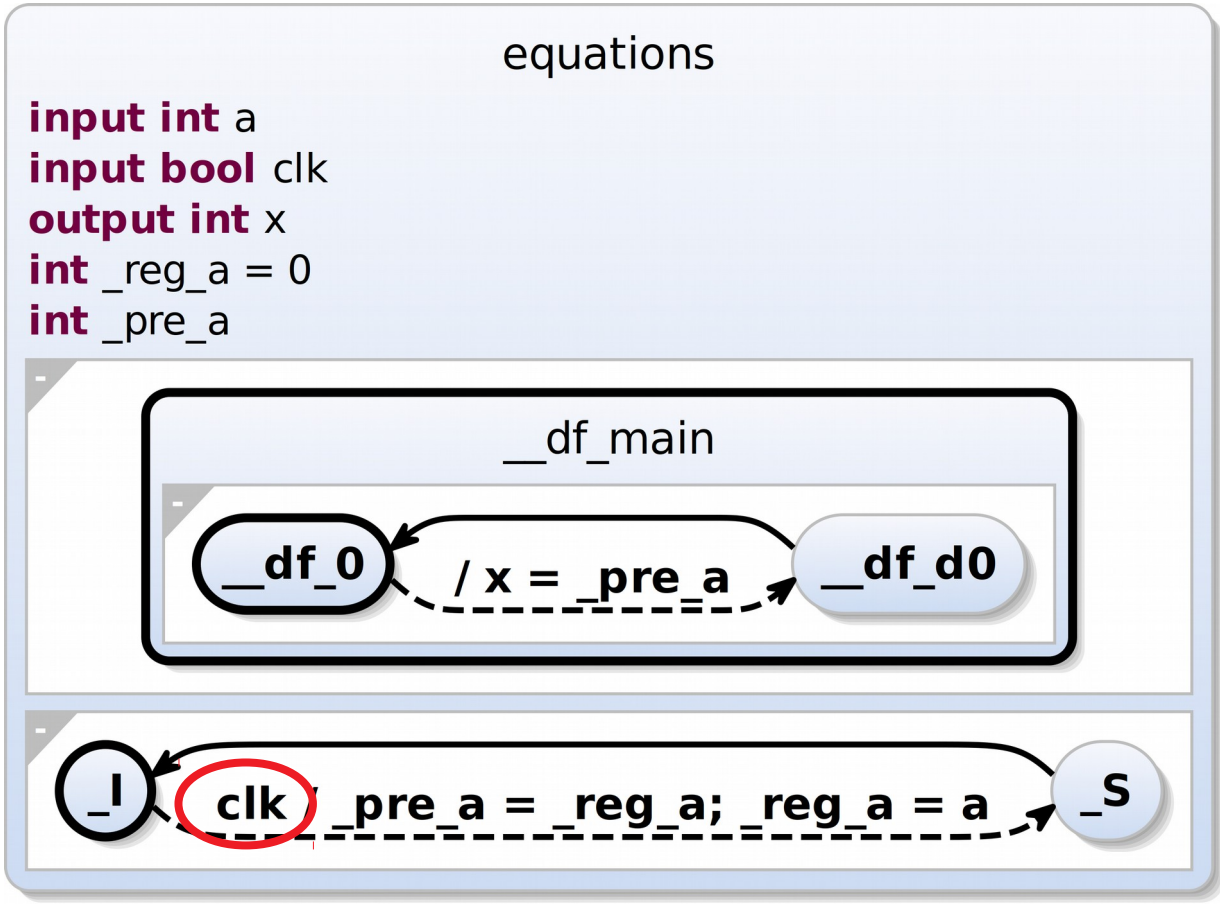
clk	true	false	false	true	true	false	true	false	true
x	1	2	3	4	5	6	7	8	9
xClk = x when clk	1			4	5		7		9
pxClk = pre(xClk)	nil			1	4		5		7
pre(pxClk)	nil			nil	1		4		5

Pre Operator with Clocks and Variables

clk	true	false	false	true	true	false	true	false	true
x	1	2	3	4	5	6	7	8	9
xClk = clk? x	1	1	1	4	5	5	7	7	9
pxClk = pre(xClk)	nil	1	1	1	4	5	5	7	7
pre(pxClk)	nil	nil	1	1 **nil**	1	4	5 **4**	5	7 **5**

Clocked Pre Operator in SCCharts

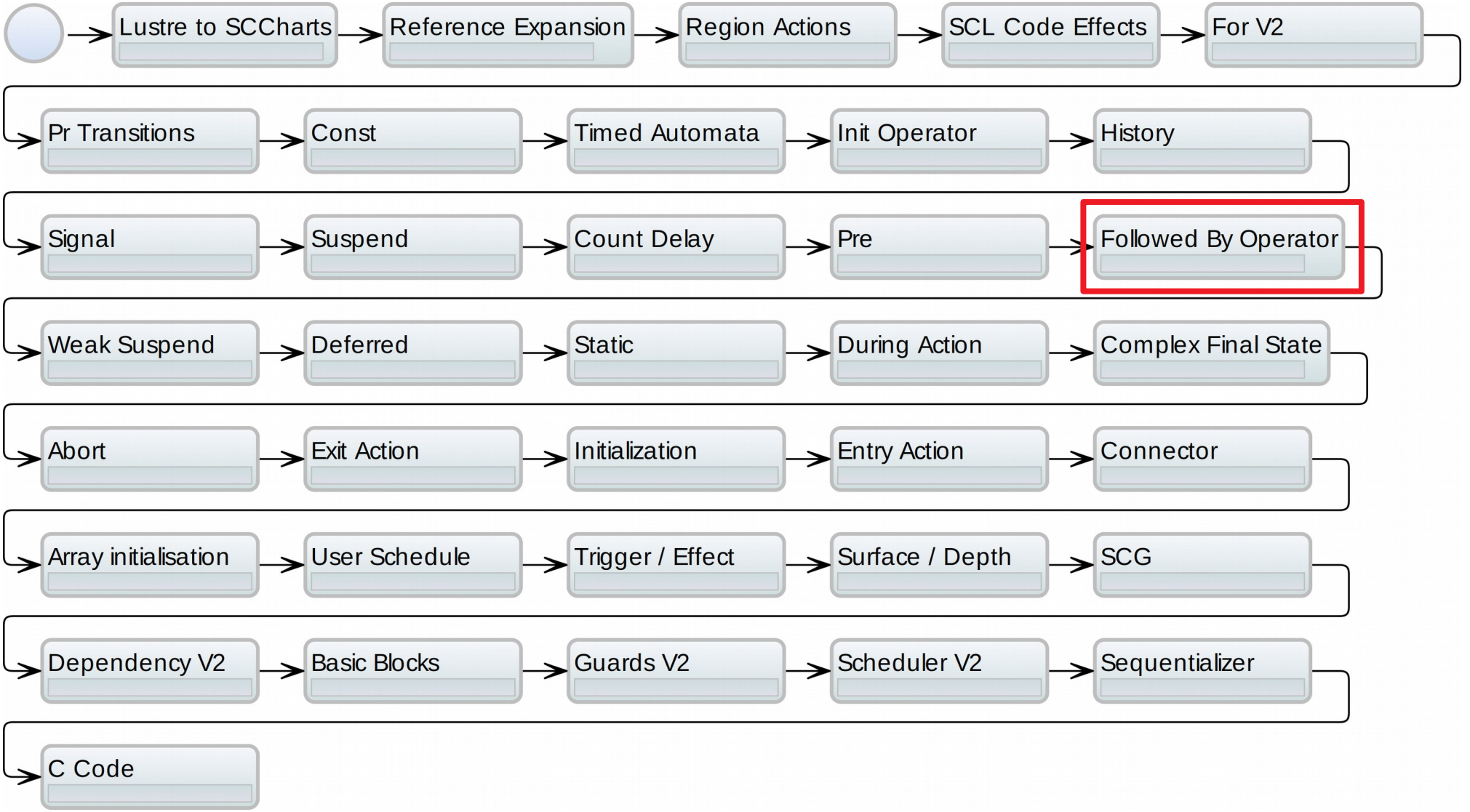




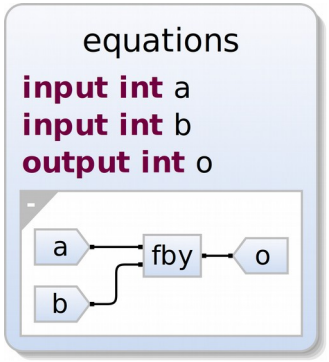

```
scchart equations {
    input int a
    input bool clk
    output int x

    dataflow {
        x = pre(a, clk)
    }
}
```

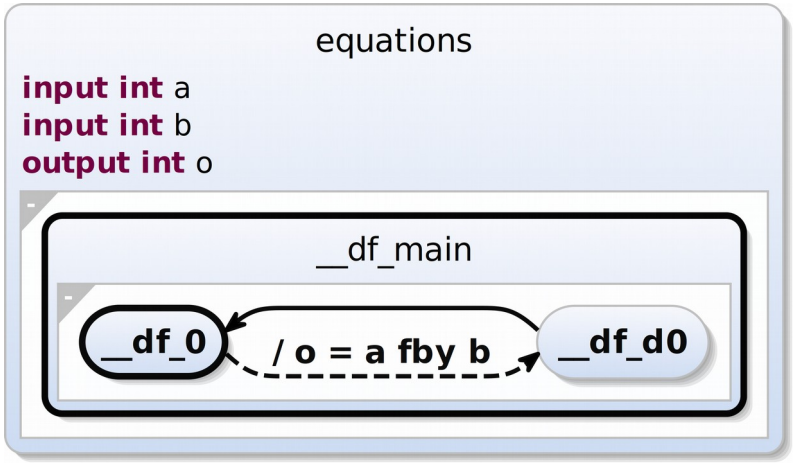
Pre Operator with Clocks and Variables

clk	true	false	false	true	true	false	true	false	true
x	1	2	3	4	5	6	7	8	9
xClk = clk? x	1	1	1	4	5	5	7	7	9
pxClk = pre(xClk, clk)	nil	nil	nil	1	4	4	5	5	7
pre(pxClk, clk)	nil	nil	nil	nil	1	1	4	4	5

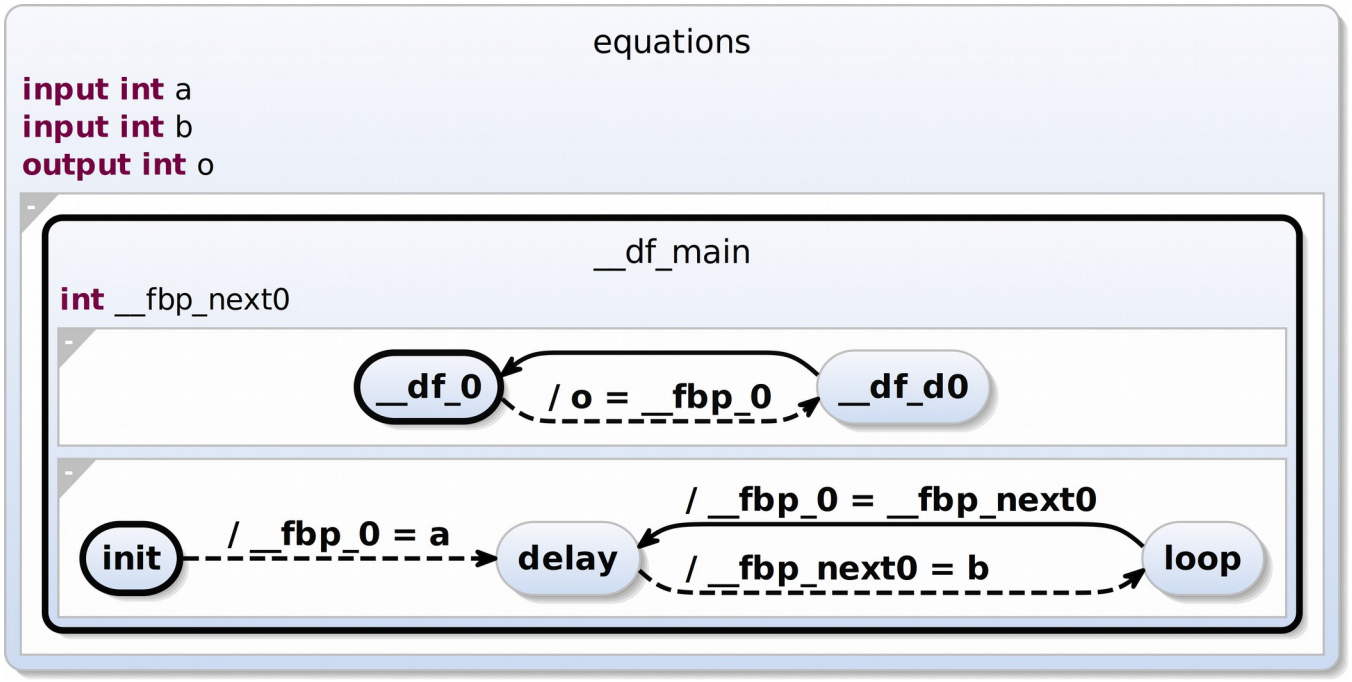

Lustre to SCCharts
Reference Expansion
Region Actions
SCL Code Effects
For V2
Pr Transitions
Const
Timed Automata
Init Operator
History
Signal
Suspend
Count Delay
Pre
Followed By Operator
Weak Suspend
Deferred
Static
During Action
Complex Final State
Abort
Exit Action
Initialization
Entry Action
Connector
Array initialisation
User Schedule
Trigger / Effect
Surface / Depth
SCG
Dependency V2
Basic Blocks
Guards V2
Scheduler V2
Sequentializer
C Code

Fby Transformation

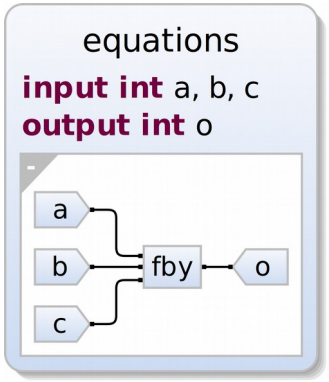


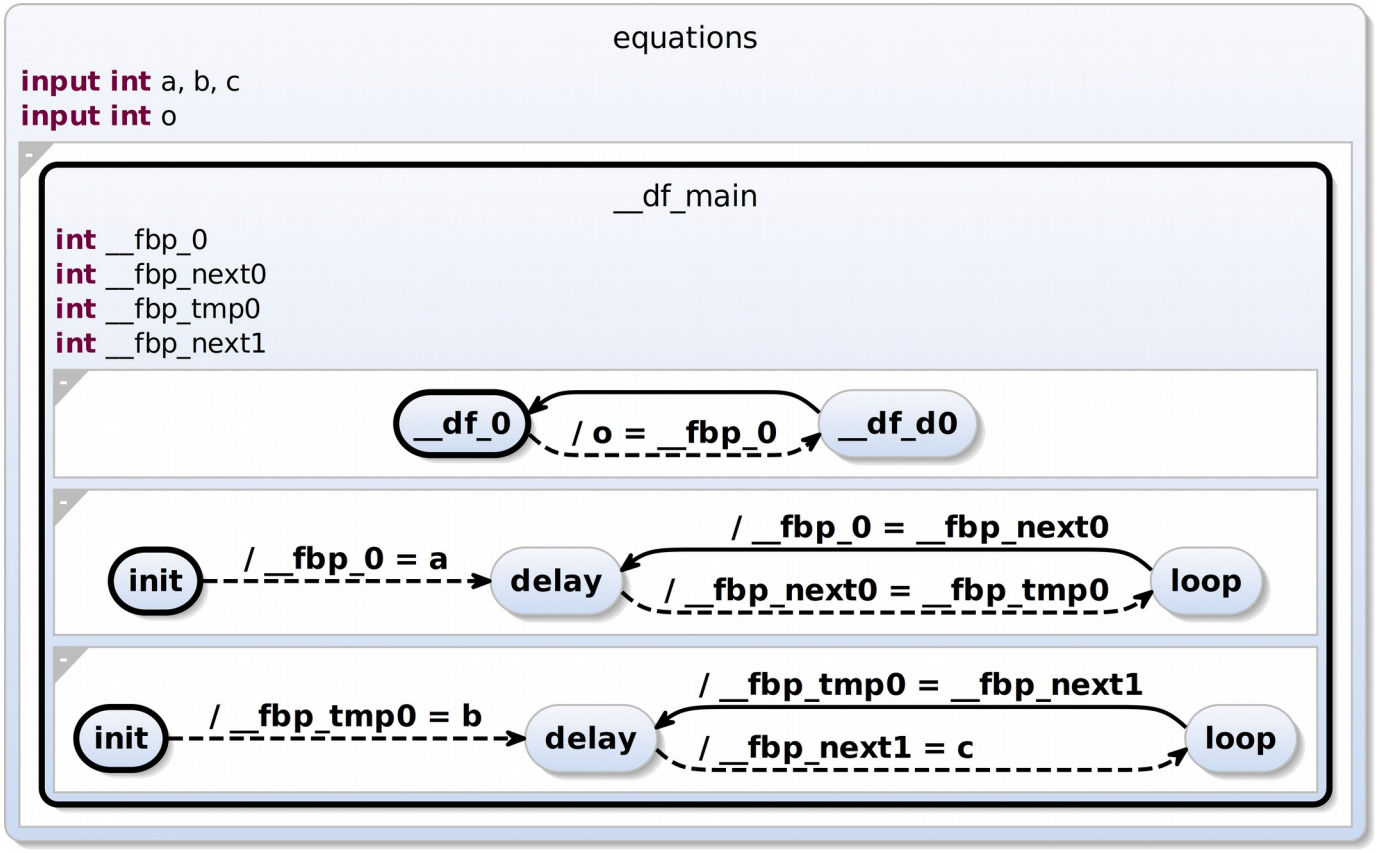


a -> pre(b)



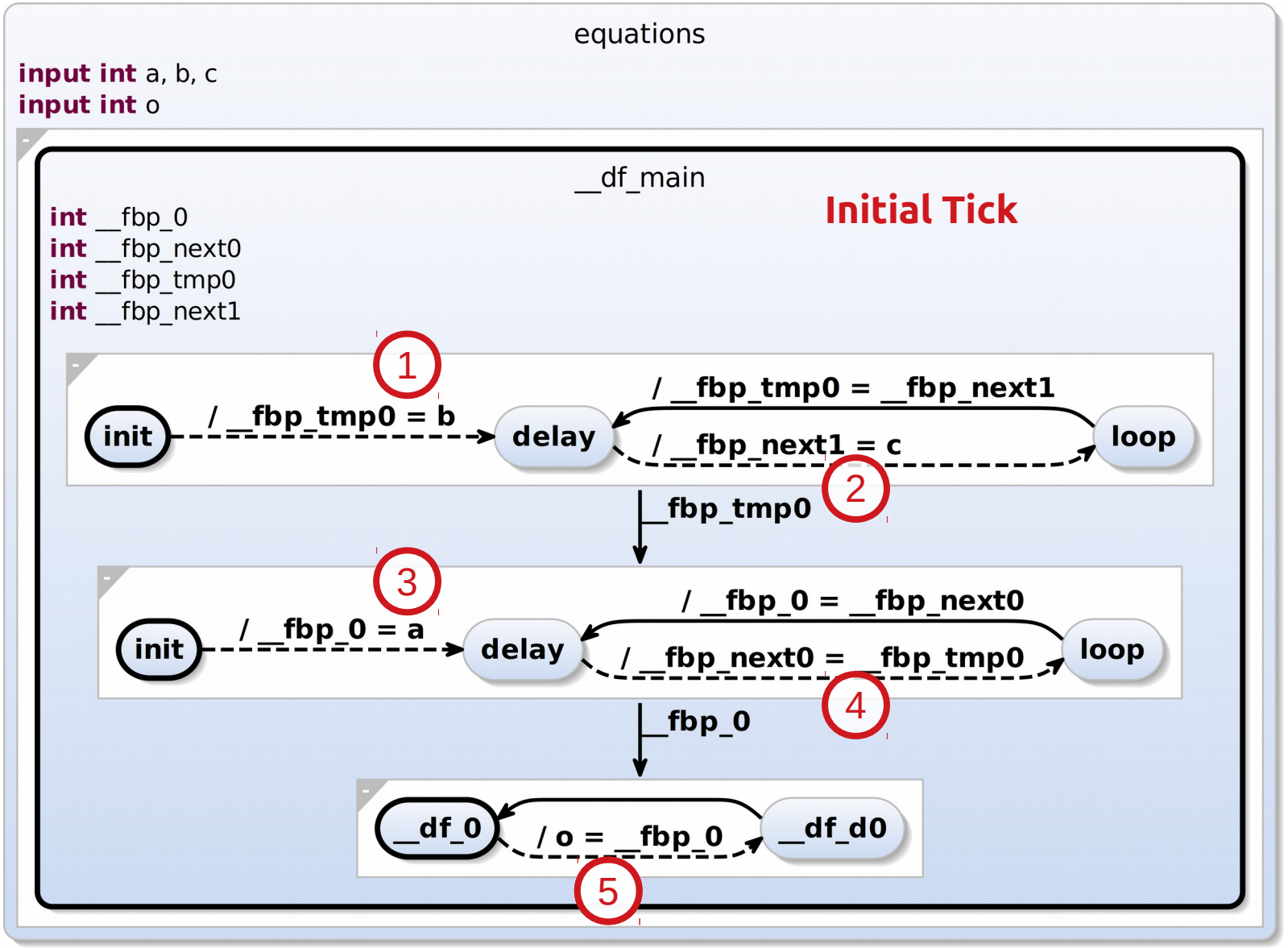
Fby Transformation II






```
scchart equations {
    input int a, b, c
    input int o

    dataflow {
       o = a fby b fby c
    }
}
```

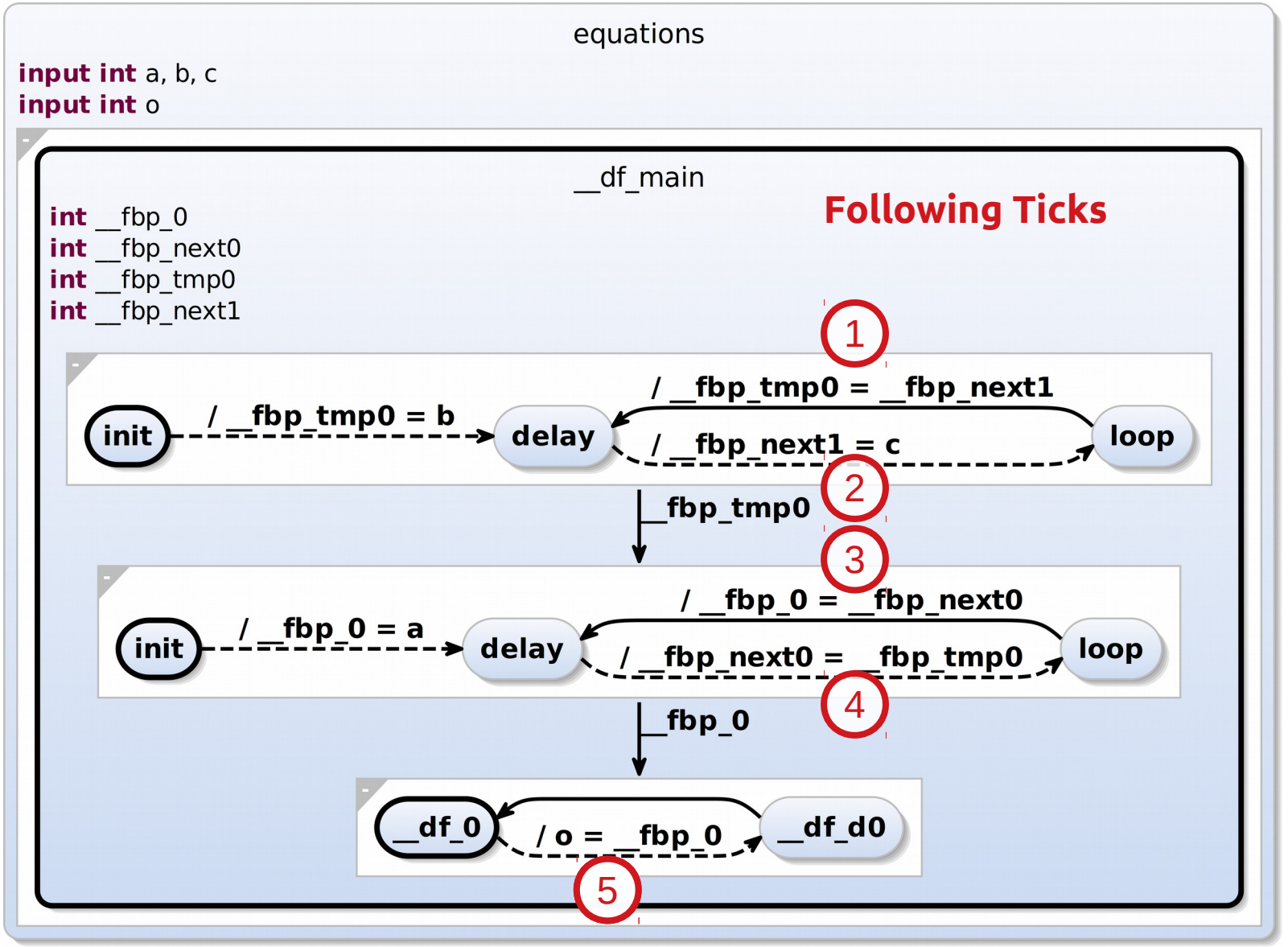
Fby Transformation Induced Dataflow View



b → pre(c)

pre(pre(...))

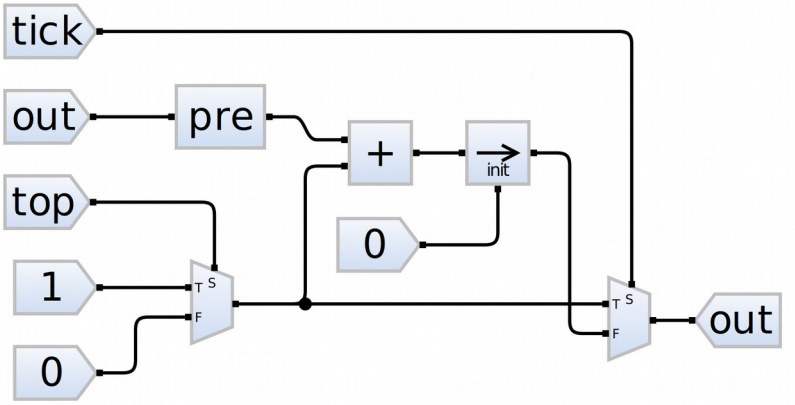
Fby Transformation Induced Dataflow View



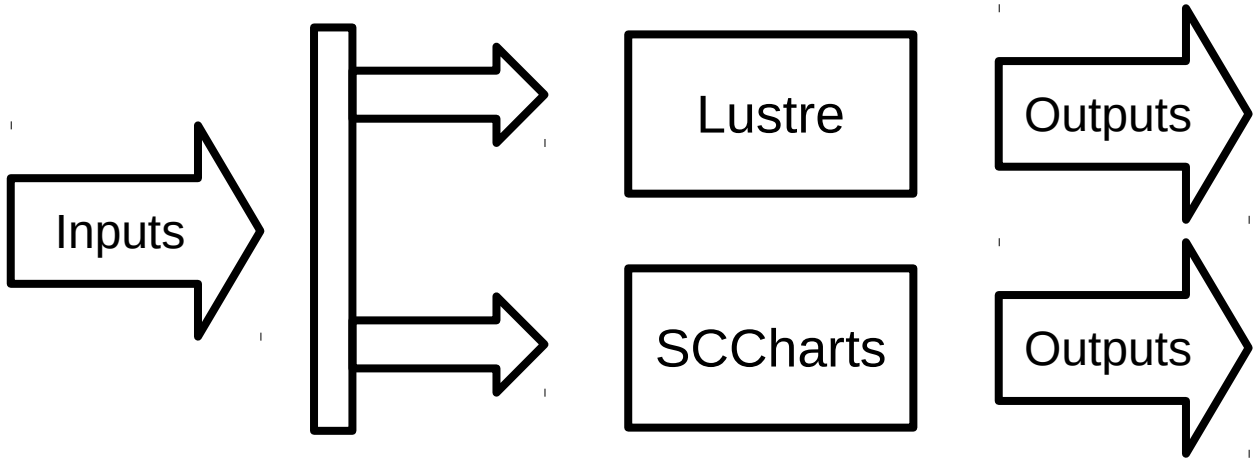
Summary

Model Recovery

```
node counting(tick:bool;top:bool)
     returns (out:int);
var v:int;
let
  v = if top then 1 else 0;
  out = if tick
        then v
        else (0 -> pre out + v);
tel.
```



Behavior Preservation



Thank You!

Download KIELER Nightly Version
https://rtsys.informatik.uni-kiel.de/confluence/display/KIELER/Downloads