Synchronous Design and Verification of Critical Embedded Systems using SCADE and Esterel

Gérard Berry

Chief Scientist

www.esterel-technologies.com
Gerard.Berry@esterel-technologies.com

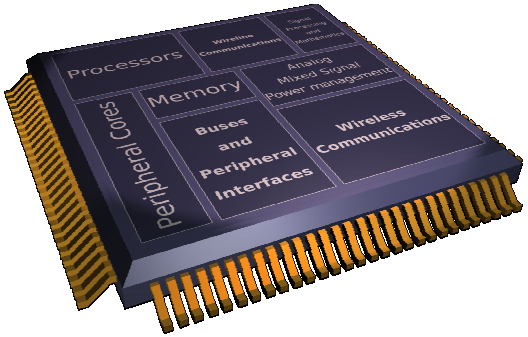
Esterel Technologies - Industries Served

Esterel Studio™

Specification-to-RTL of **hardware IP designs**

• rigorous & unambiguous executable specifications

• automatically-generated efficient RTL / C code



SCADE Suite™

De-facto Standard for **Safety-critical avionics embedded software**

• DO-178B Level A certified systems

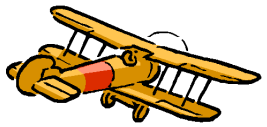
• automatically-generated C code

SCADE Drive™

Safety-critical automotive embedded software

• code generator certified by TUV - IEC 61508 standard

Applications and Constraints



flight-control, engines, brakes, fuel, power, climate
safety-critical => certification

trajectory, attitude, image, telecom
mission-critical => very high quality

telephone, audio, TV, DVD, games
business critical => time-to market + quality

pacemakers, diabet control, robot surgeons
life-critical => TBD (!)

SCADE Suite in Aerospace & Defense Applications

- More than 50 companies using SCADE for
  - Flight control systems
  - Power management
  - Reconfiguration management
  - Autopilots
  - Engine control systems
  - Braking systems
  - Fuel management
  - Cockpit display and alarm management

EUROCOPTER – EC145

Dassault Aviation - Rafale

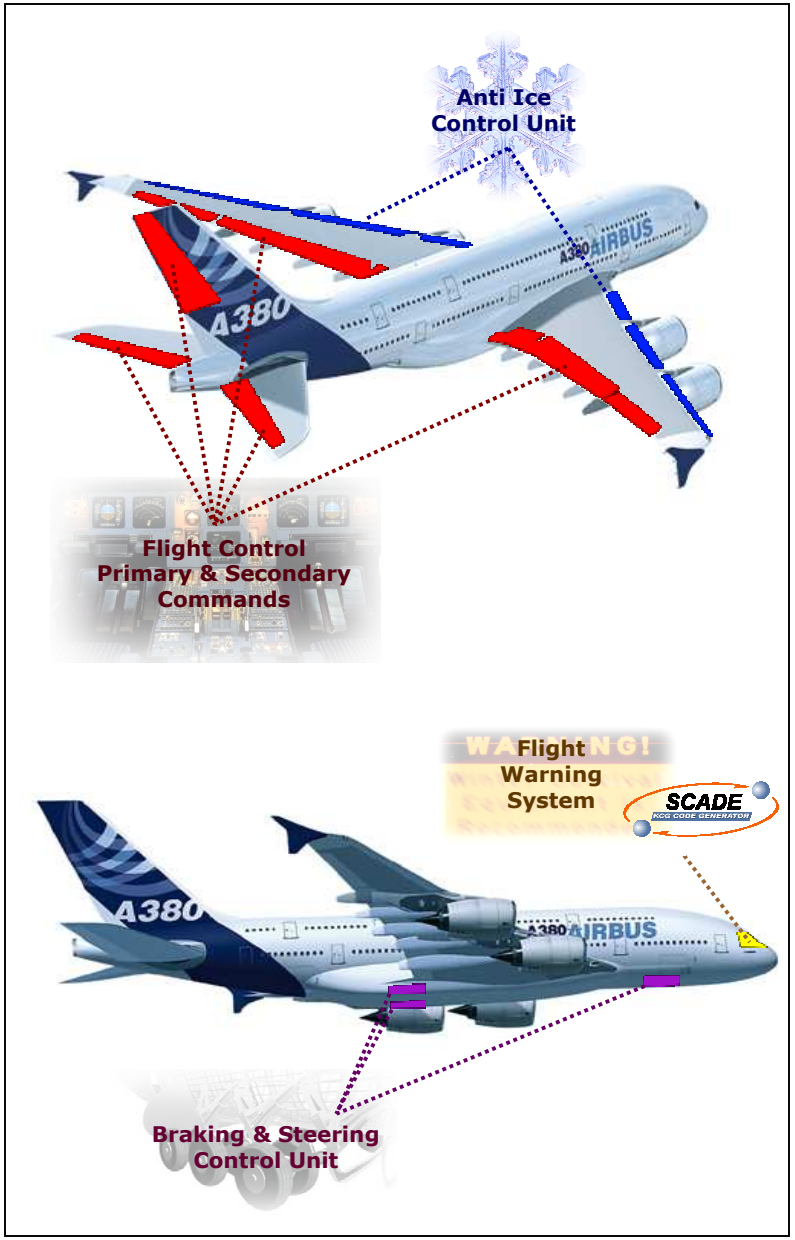
Dassault Aviation – Falcon 7X

AIRBUS – A340-600 & A380

Aeroengines by Snecma
©Snecma/Studio Pons

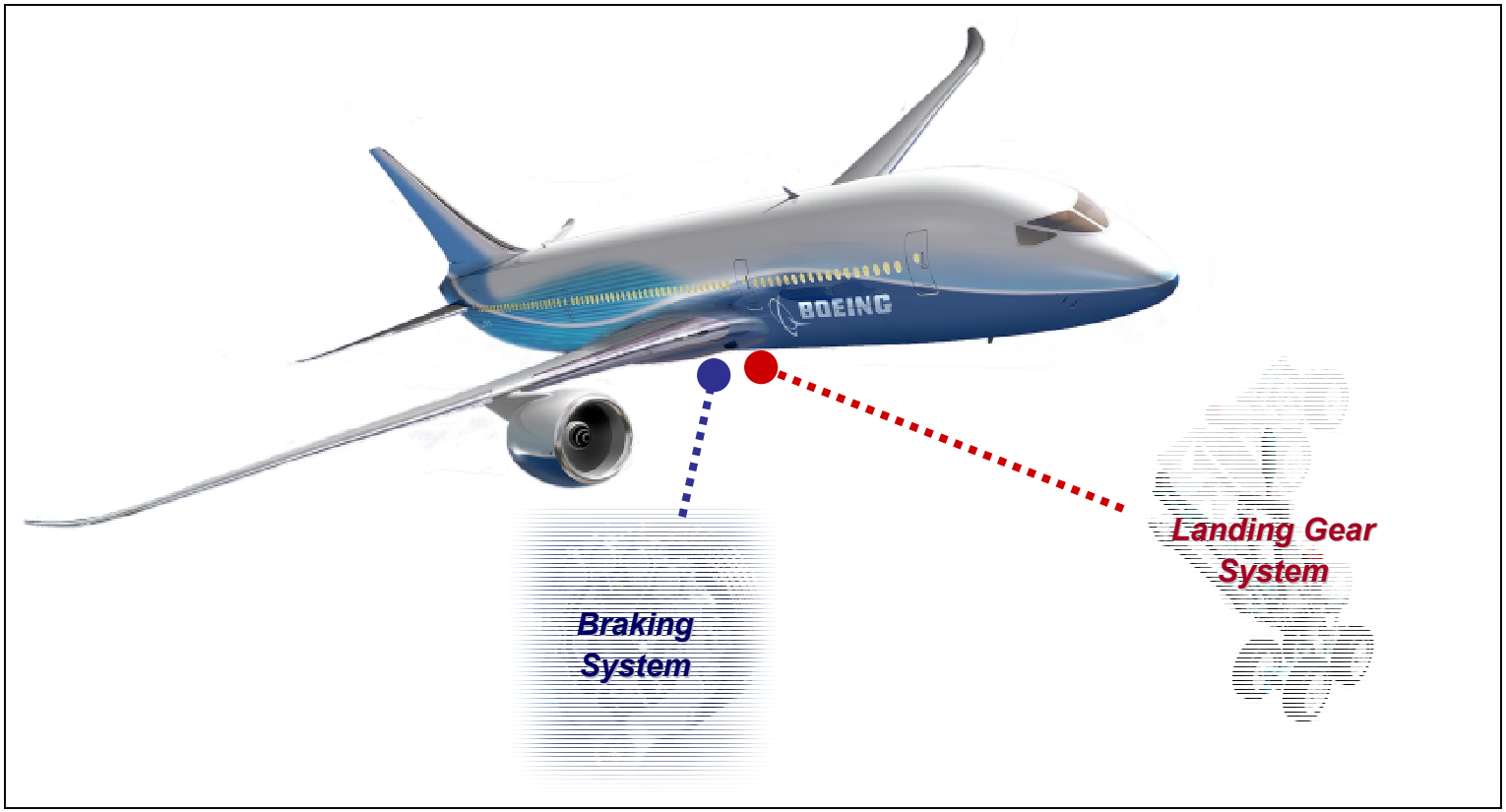
US Air Force - F16

SCADE Suite in the A380



- SCADE = Airbus corporate standard for all new airplanes developments
  - Flight Control system
  - Flight Warning system
  - Electrical Load Management system
  - Anti Icing system
  - Braking and Steering system
  - Cockpit Display system
  - Part of ATSU (Board / Ground comms)
  - FADEC (Engine Control)
  - EIS2 : Specification GUI Cockpit:
    - PFD : Primary Flight Display
    - ND : Navigation Display
    - EWD : Engine Warning Display
    - SD : System Display

SCADE Suite in the 787



- SCADE is present in the following Boeing 787 systems:
  - Landing Gear System *(Smiths Aerospace)*
  - Braking System *(Messier Bugatti)*

SCADE in Automotive & Land Systems Applications

- Automotive & 2-Wheelers:
  - Airbags
  - Braking Systems, ABS & ESP
  - Steering
  - Chassis & Suspension Systems
  - Restraining systems
  - Engine regulation
  - X-By-Wire applications
- Heavy Duty Land systems:
  - Cranes
  - Tractors
  - Tanks
  - Earth Moving Machines
  - Trucks
  - Construction equipment
  - Mining machines, etc…





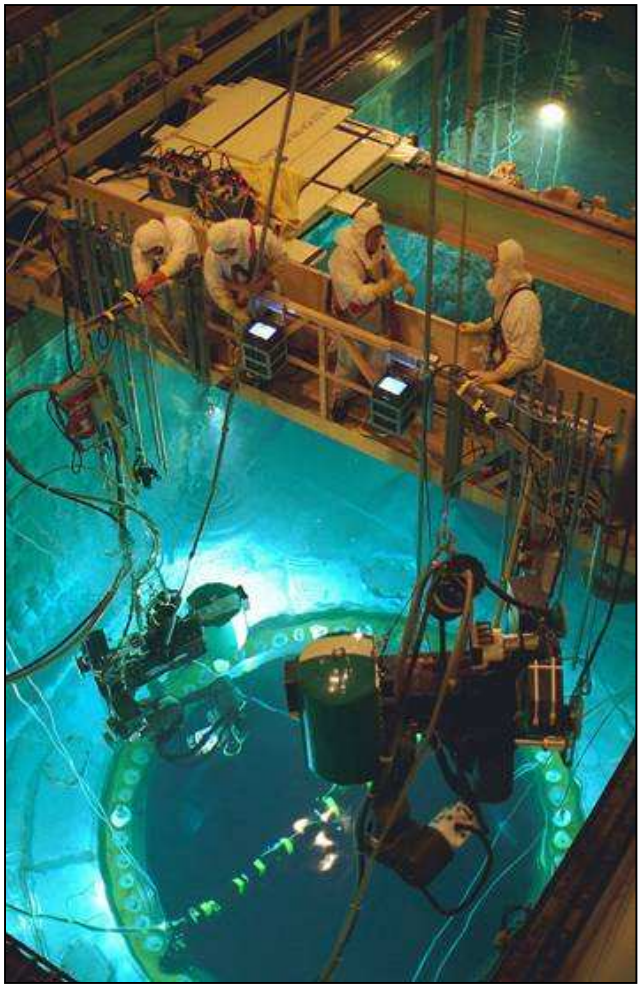


SCADE in Rail Transportation Applications

- Interlocking systems control
- Signaling
- Ground stations
- Automatic Train Operations
- Train Control Systems
- Critical Graphics Displays
- Level Crossings
- Safe Platforms

SCADE Nuclear I&C Applications

- Reactor Protection Systems:
  - Reactor limitation system (1E)
  - Trip processing & Emergency shutdown (1E)
  - Safety actuation (1E)
- Nuclear Instrumentation Systems:
  - Power measurement system
  - Neutron detectors (1E)
  - Pressurizer heating controllers
  - Neutron instrumentation systems (1E)
  - Boron meters
- Other Safety Systems
  - Safety valve control system
  - Rod control systems
  - Diesel sequencing system (1E)
  - Rod position instrumentation systems

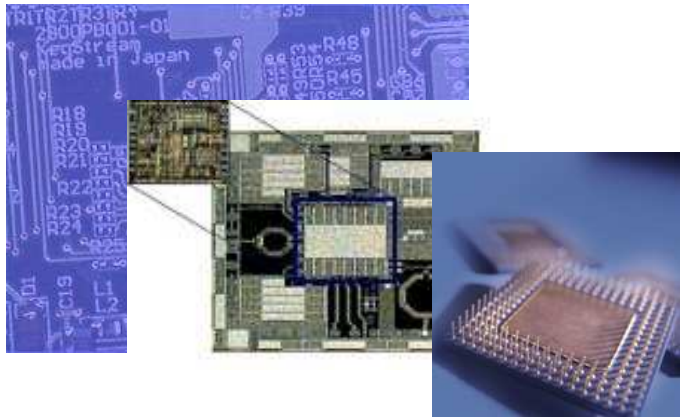


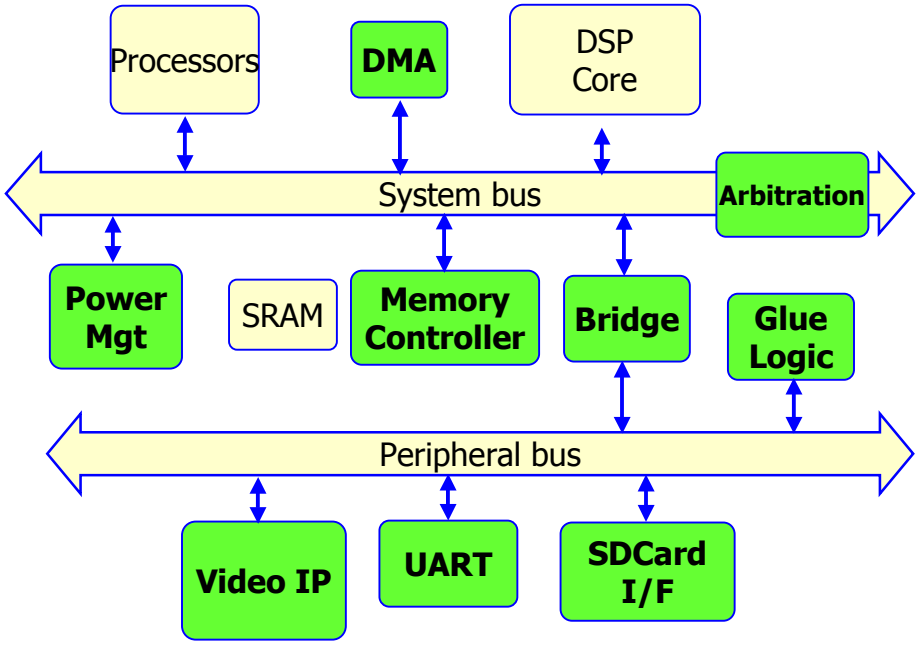
The Esterel Consortium

- In 2001 Esterel Technologies formed a consortium of leading Semiconductor companies
  - Early adopters of Esterel Studio™
  - Best practice sharing about project use and design flow integration
  - Collaborative specification of the main product features and roadmap
  - Attended by academic partners for scientific advise
  - Attended by Esterel Studio offer partners
  - Support to the IEEE standardization process of the Esterel language

Esterel Application Targets

- Bus interfaces and peripheral controllers
  - Bus bridges
  - Serial ATA
  - Flash cards
  - Video controllers
- Processor modeling and synthesis
  - Instruction Set Architecture
  - Complex instruction and data cache
  - Arbiters
  - On-chip power management
  - DMA
  - Interrupt control
- Communication IPs
  - Radio
  - Fast serial links (UART, Aurora)
  - Bluetooth / Ethernet Controller





Messages to Users

1. Specification of dynamics cannot be accurate when written on static paper
2. Animated executable specifications key to reuse, inter-teams communication, what-if studies, etc
3. Once such specs are available, why recoding?
4. Any safe model-based spec-to-implementation path must be based on formal methods and tools
   - hierarchical behavior description
   - languages with formal semantics
   - formal compiling algorithms
   - formal verification techniques
5. Formal verification is a design tool usable at all design steps, not only in validation phase

A short history of synchrony

A short history of synchrony

- 1982-1985 : first ideas, languages, and semantics
  - Esterel : Berry – Marmorat - Rigault, Sophia-Antipolis
  - Lustre : Caspi – Halbwachs, Grenoble
  - Signal : Benveniste – Le Guernic, Rennes

A short history of synchrony

- 1982-1985 : first ideas, languages, and semantics
  - Esterel : Berry – Marmorat - Rigault, Sophia-Antipolis
  - Lustre : Caspi – Halbwachs, Grenoble
  - Signal : Benveniste – Le Guernic, Rennes

Computer Science
Control Theory

A short history of synchrony

- 1982-1985 : first ideas, languages, and semantics
  - Esterel : Berry – Marmorat - Rigault, Sophia-Antipolis
  - Lustre : Caspi – Halbwachs, Grenoble
  - Signal : Benveniste – Le Guernic, Rennes

  Computer Science
  Control Theory

- 1985-1998 : more languages, semantics, compiling & verification
  - SyncCharts (André), Reactive C (Boussinot), TCC (Saraswat), etc.
  - causality analysis (Berry, Gonthier, Shiple)
  - links to dataflow (Ptolemy), to hardware (Vuillemin), etc.
  - formal optimization & verification techniques (Madre & Coudert, Touati)

  First industrialization of Esterel by ILOG for Dassault Aviation / Thales
  Formal verification (BDDs) on landing gear, Displays, etc in 1991
  Creation of SCADE by Verilog for Schneider and Airbus

- 1998 –2001 : more research, maturation, industrial projects

  S. Edwards, Synopsys

  Shyamasundar, TIFR, Ramesh, IIT Mumbai

  Saraswat, Xerox

  K. Schneider, Karlsruhe / KaisersLautern

  ...

  applications: avionics, nuclear plant safety, telecom, robotics, etc.

- 2001-2007 : industrial expansion

  Development of Esterel v7 for hardware circuit design

  Creation of the Esterel Consortium

  Massive usage of SCADE in certified avionics embedded systems

  Growing usage of SCADE in railways and automotive industries

- 1998 –2001 : more research, maturation, industrial projects

  S. Edwards, Synopsys

  Shyamasundar, TIFR, Ramesh, IIT Mumbai

  Saraswat, Xerox

  K. Schneider, Karlsruhe / KaisersLautern

  ...

  applications: avionics, nuclear plant safety, telecom, robotics, etc.

- 2001-2007 : industrial expansion

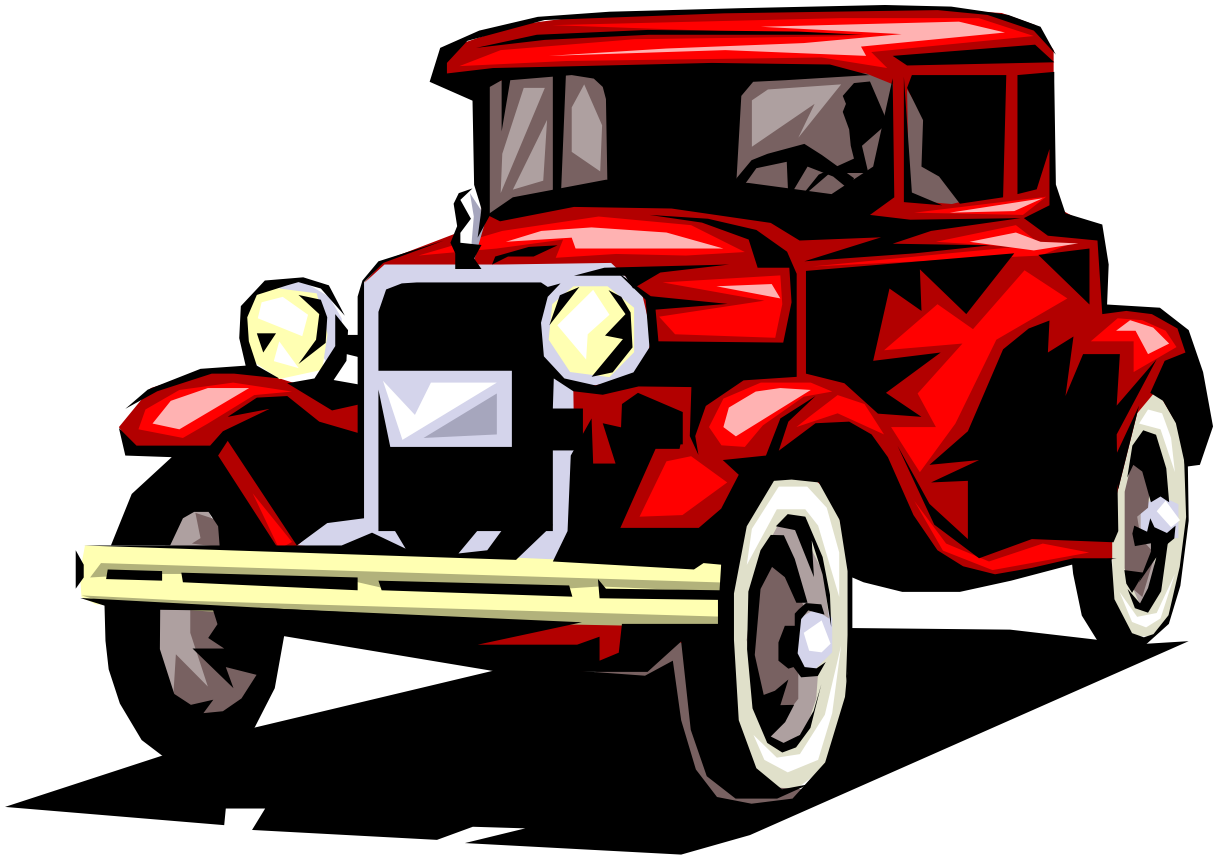
  Development of Esterel v7 for hardware circuit design

  Creation of the Esterel Consortium

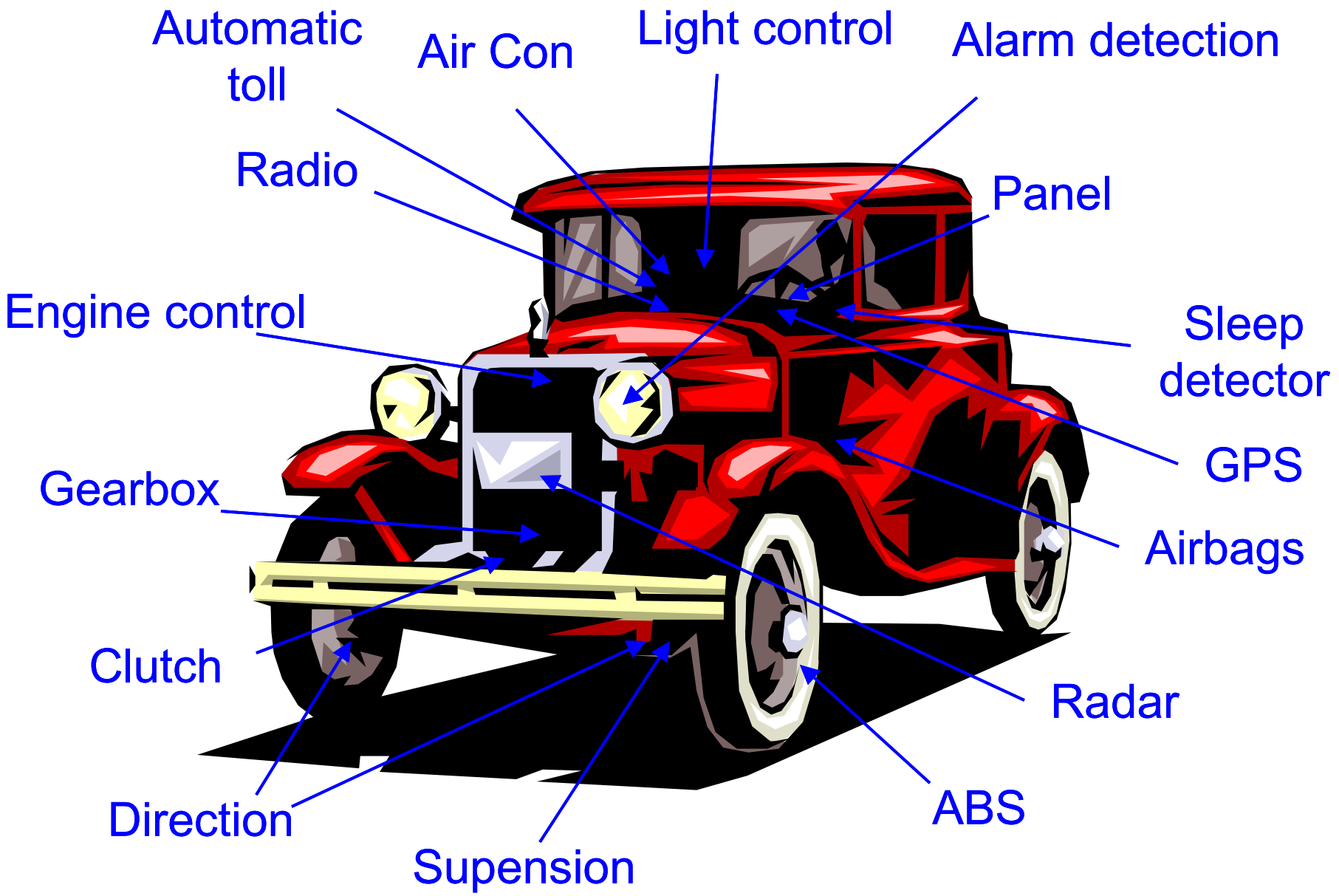
  Massive usage of SCADE in certified avionics embedded systems

  Growing usage of SCADE in railways and automotive industries

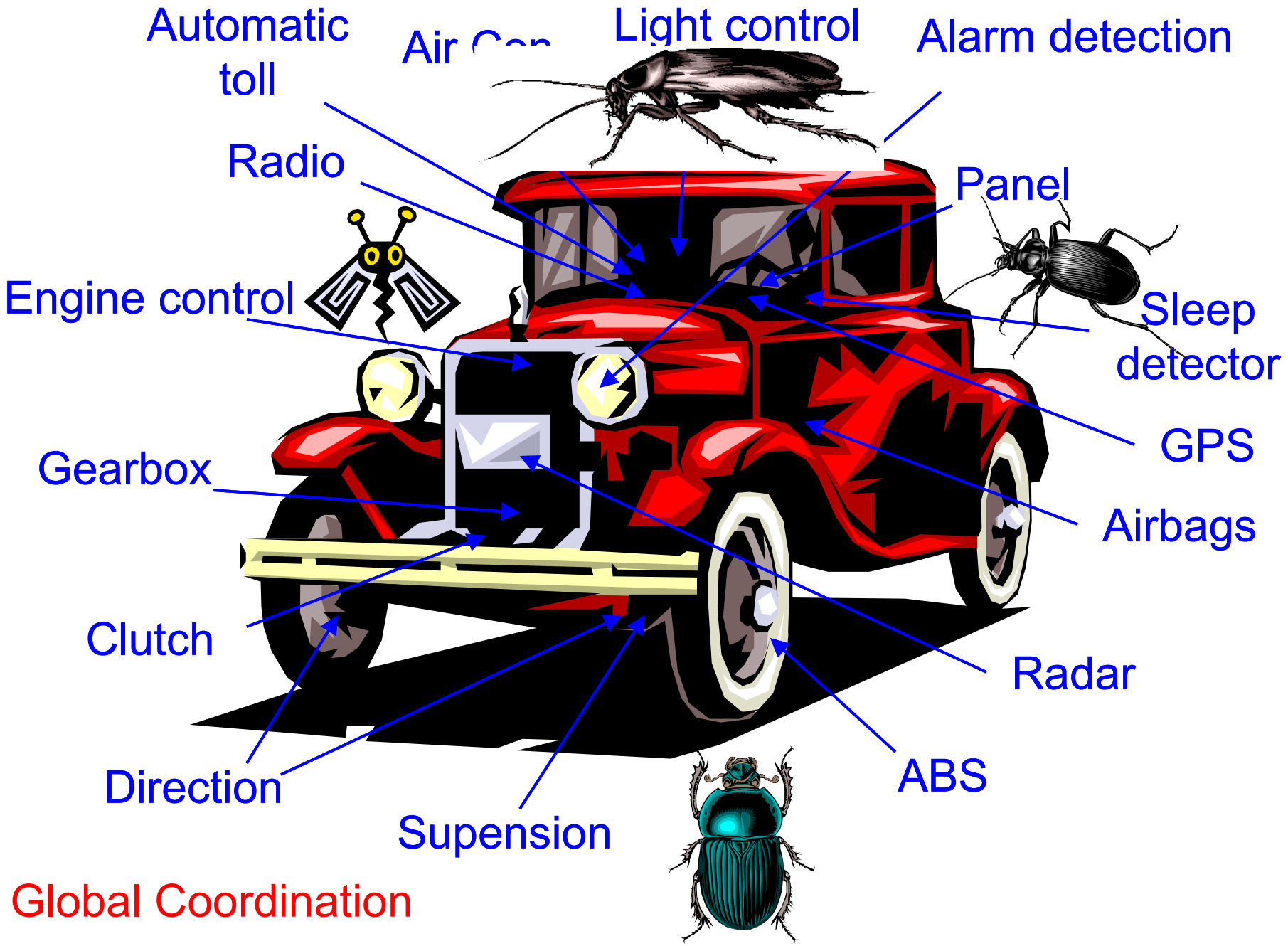
Still few researchers working on this key subject!



Global Coordination



Global Coordination

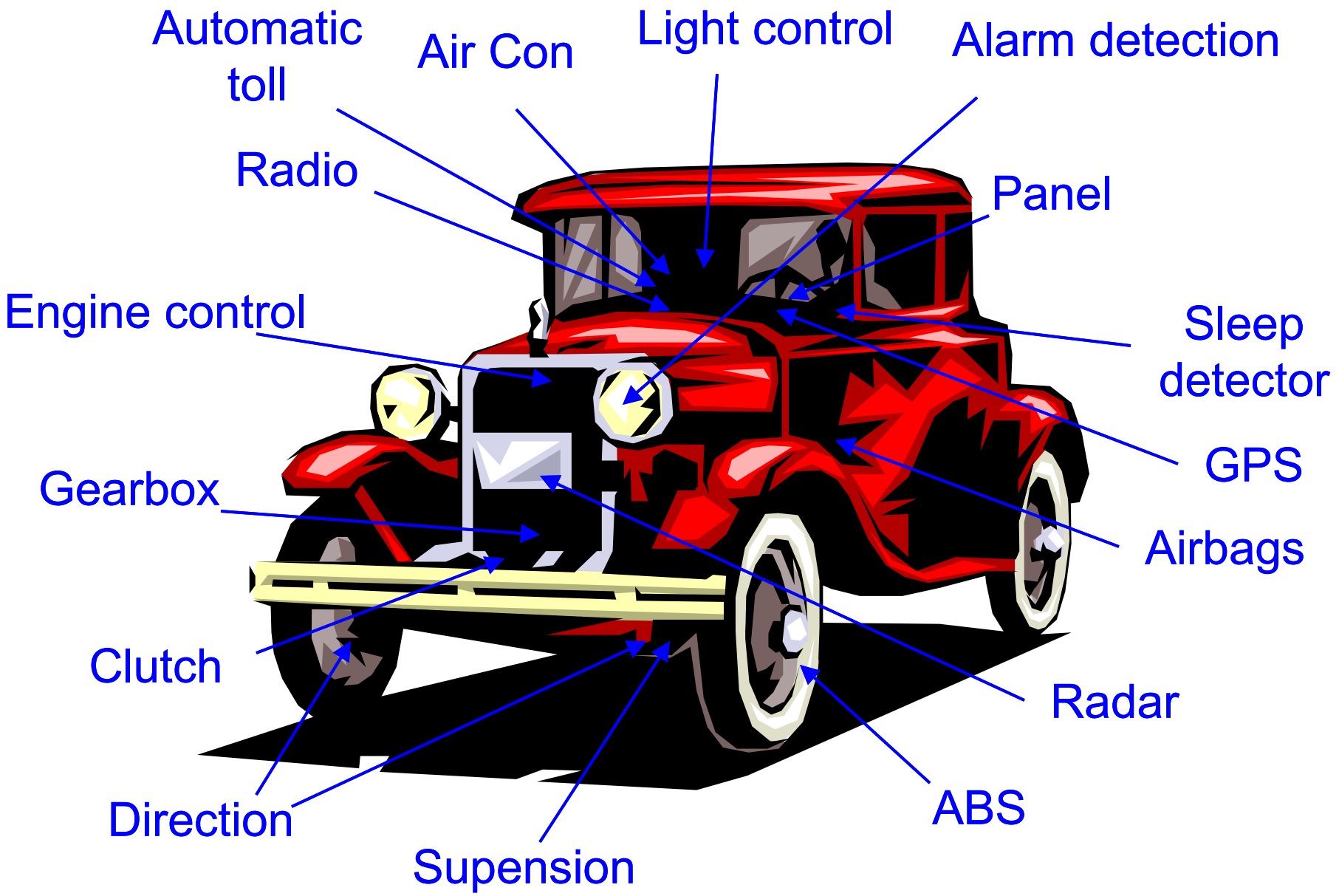


How to avoid or control bugs?

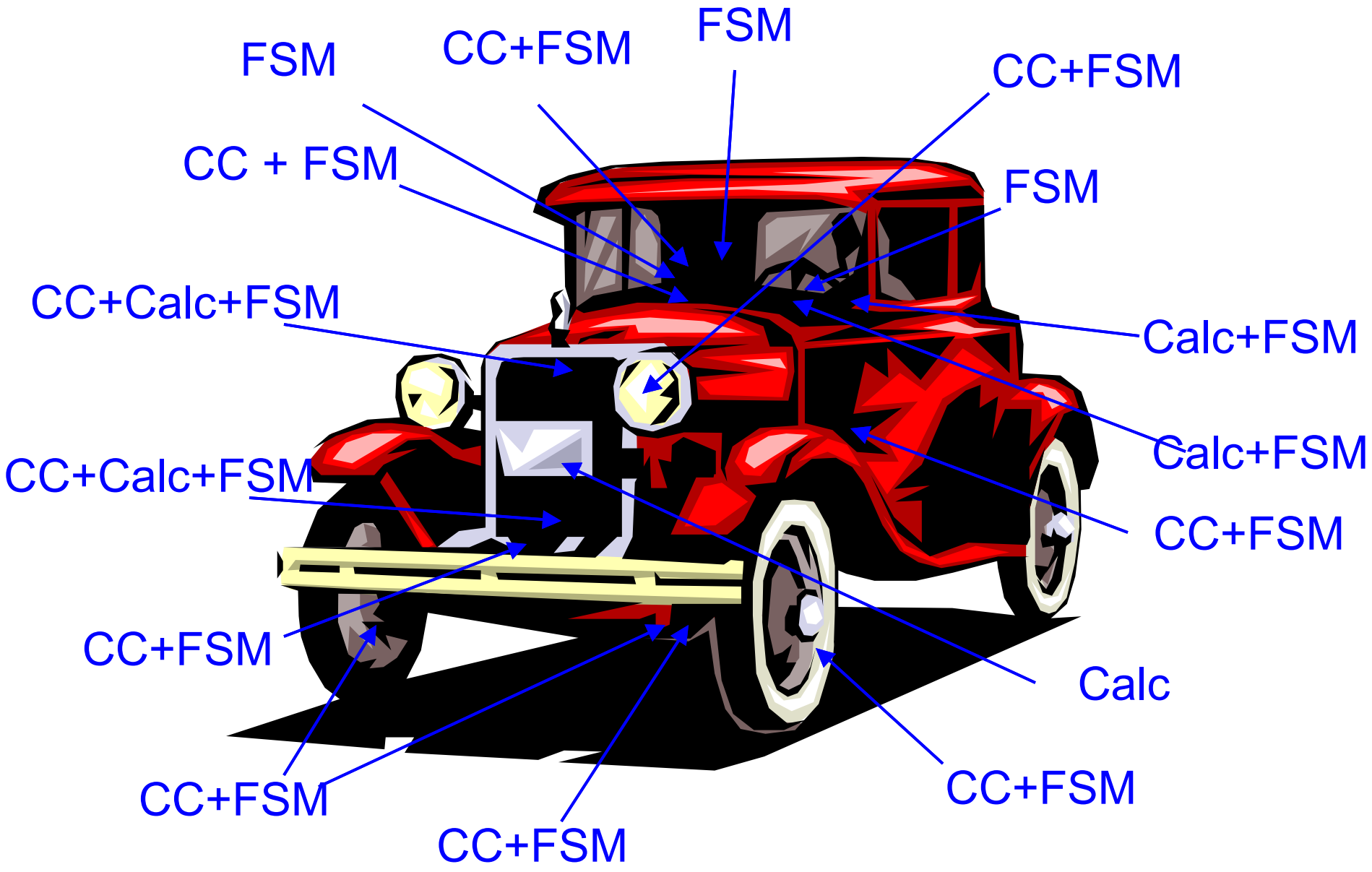
- Traditional : better verification by fancier simulation
- Next step : better design
  - better and more reusable specifications
  - simpler computation models, formalisms, semantics
  - reduce architect / designer distance
  - reduce hardware / software distance
- Mandatory: better tooling
  - synthesis from high-level descriptions
  - formal property verification / program equivalence
  - certified libraries

Embedded Modules Anatomy

- CC : continuous control, signal processing
  - differential equations, digital filtering
  - specs and simulation with Matlab / Scilab
- FSM : finite state machines (automata)
  - discrete control, protocols, security, displays, etc.
  - flat or hierarchical FSMs
- Calc : heavy calculations
  - navigation, encryption, image processing
  - C + libraries
- Web : HMI, audio / video
  - user interaction / audio / vidéo
  - data flow networks, Java



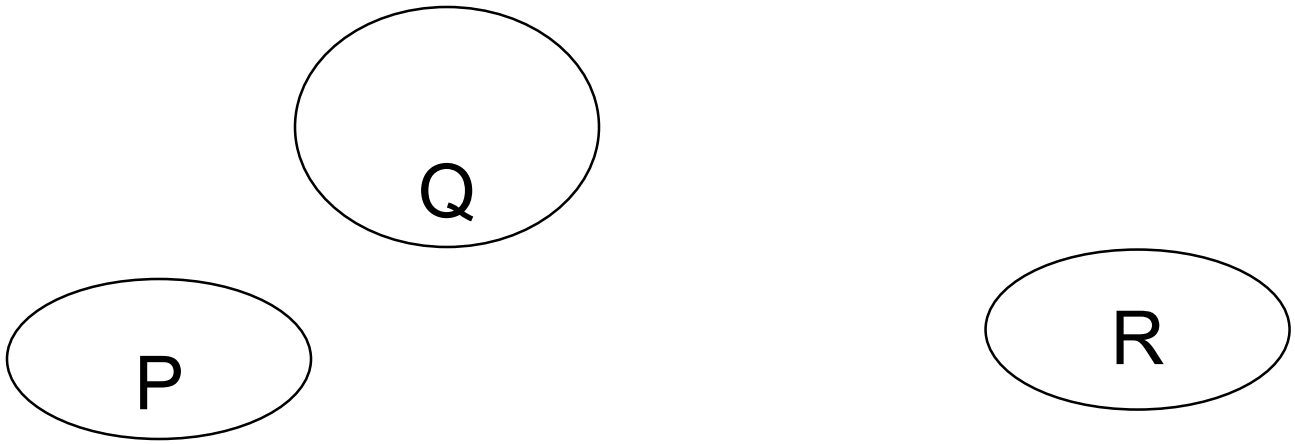
Global Coordination



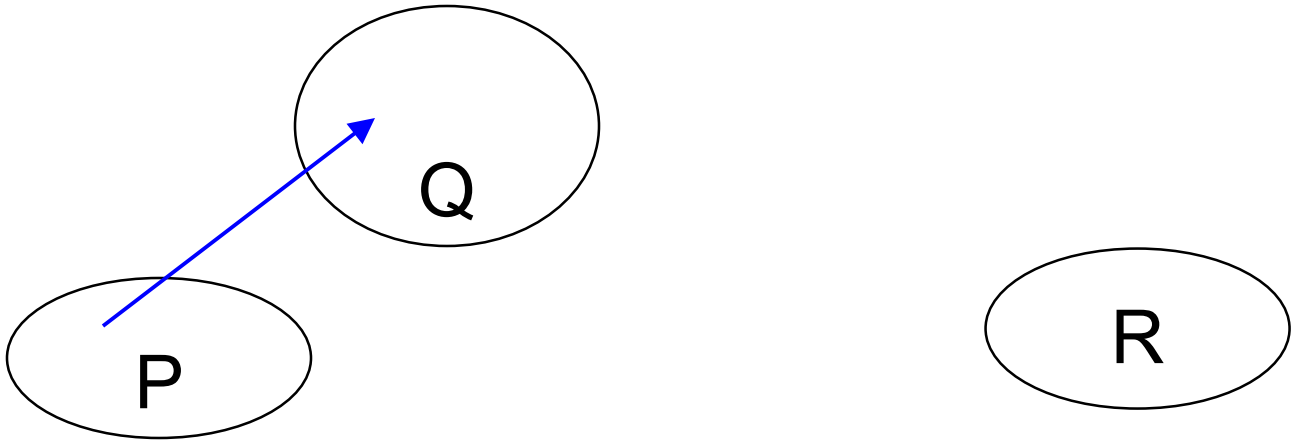
Global Coordination : Calc+CC+FSM

Key Computation Principles

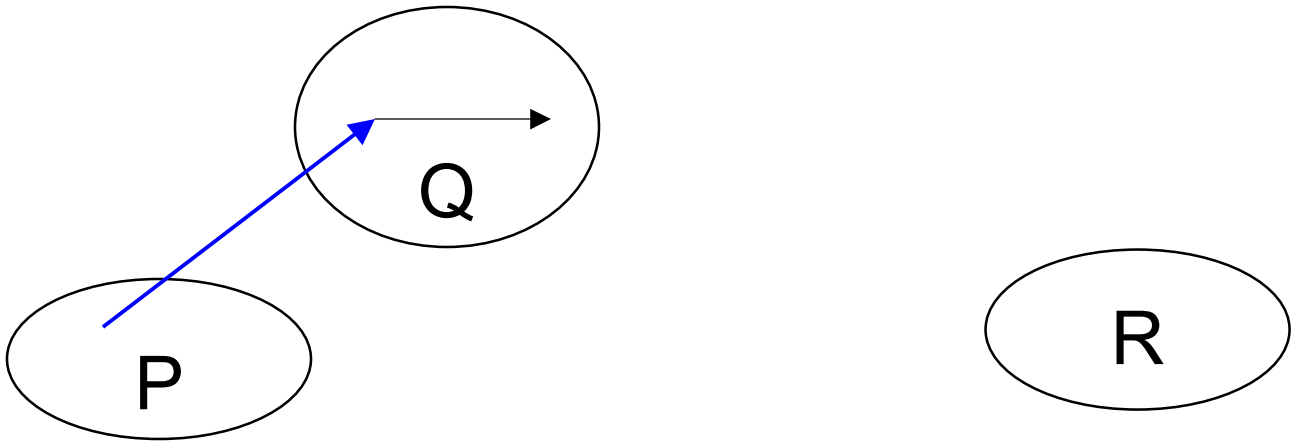
- Concurrency is fundamental
  - implicit in CC, audio / video, protocols, etc.
  - also mandatory for Web and Calc
- Determinism is fundamental
  - implicit for CC and FSM
  - who would drive a non-deterministic car?
  - can be relaxed for Web, infotainment, etc.
- Physical distribution becomes fundamental
  - separation of functions, links between them
  - redundancy for fault-tolerance
  - global time needed for distributed control



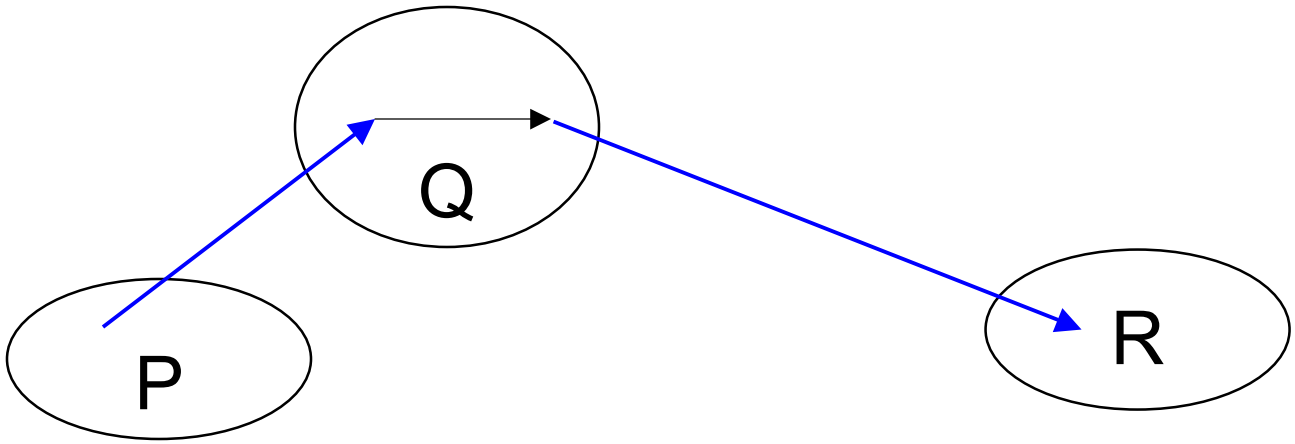
Concurrency : the compositionality principle



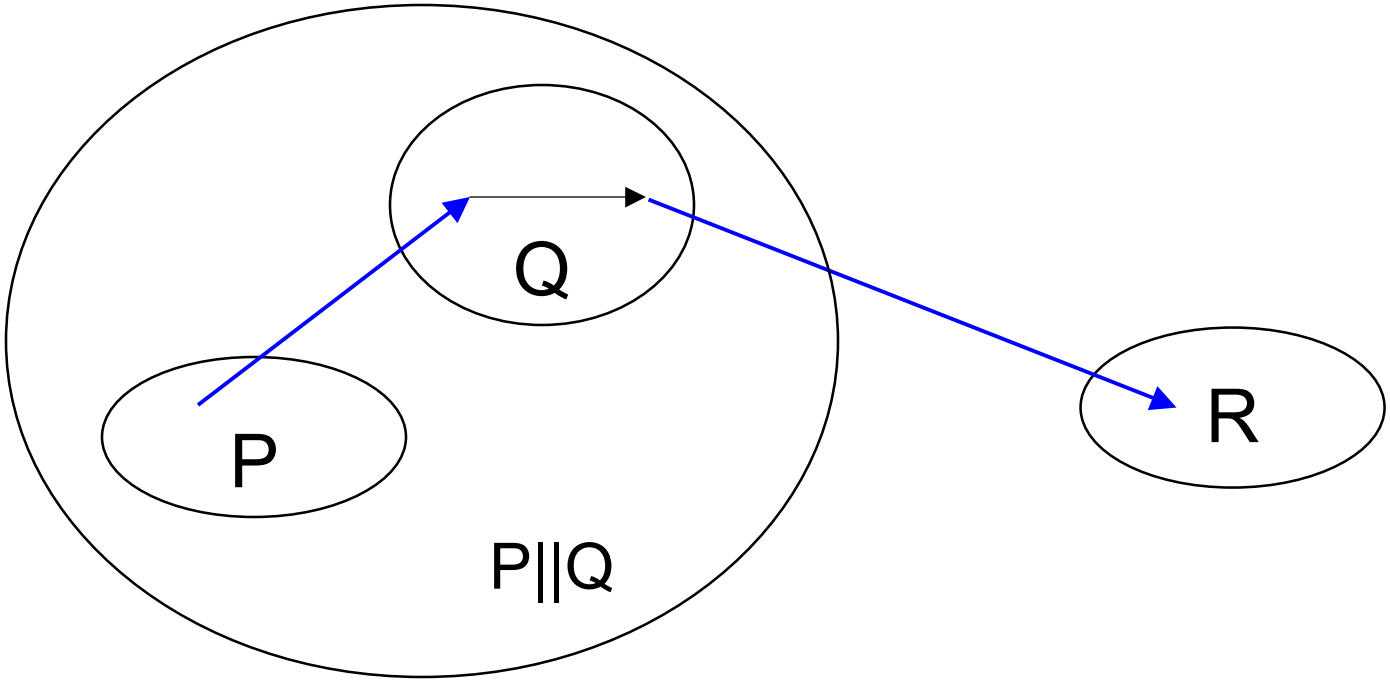
Concurrency : the compositionality principle



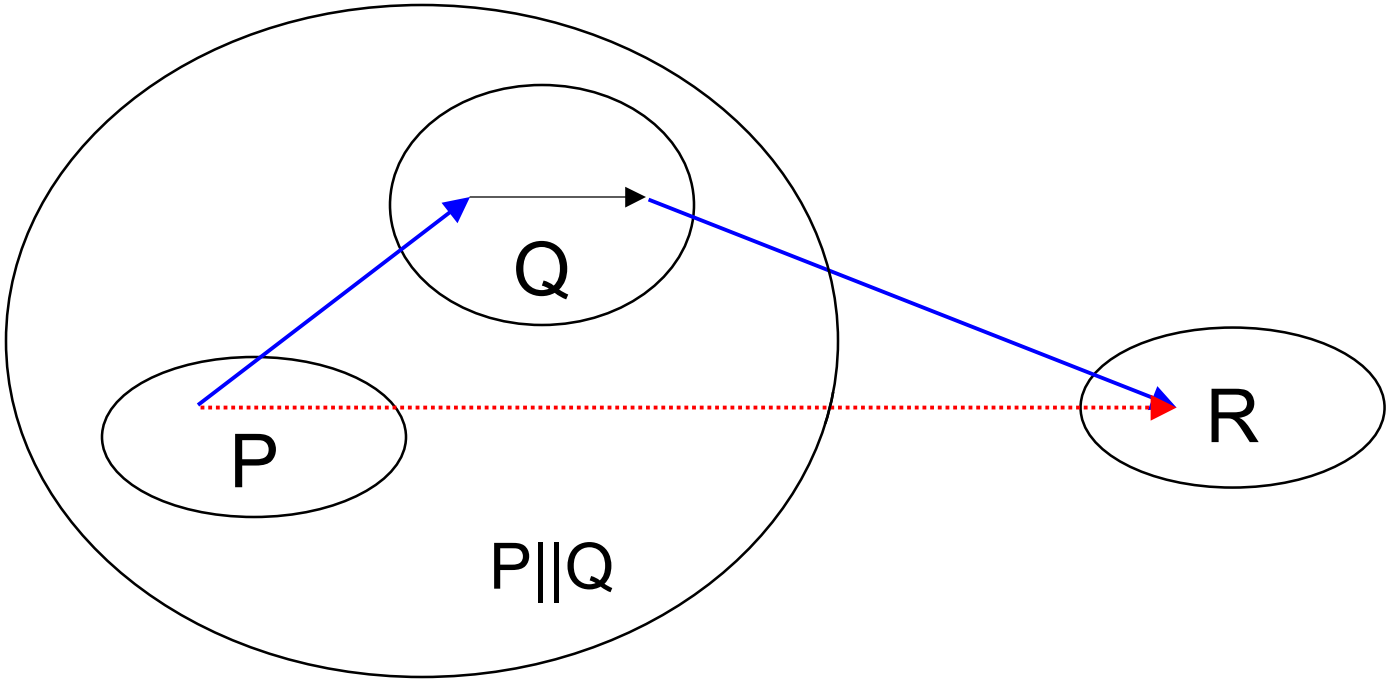
Concurrency : the compositionality principle



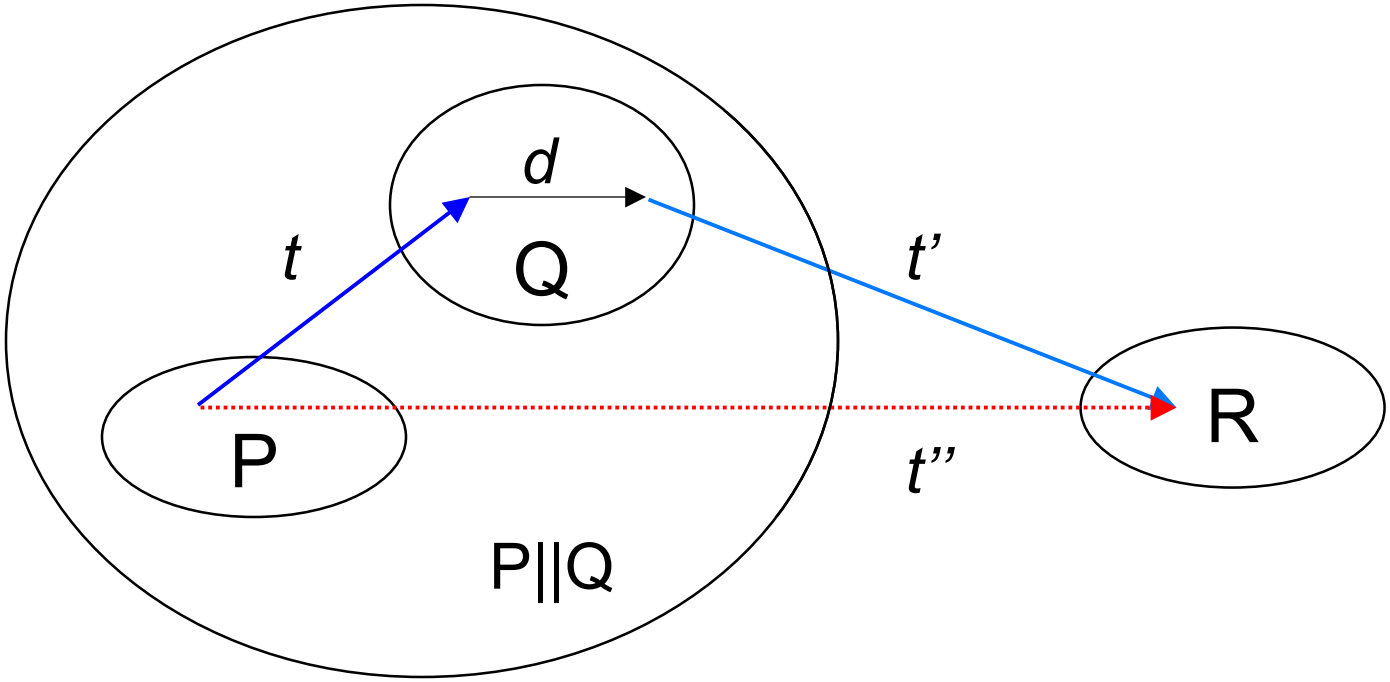
Concurrency : the compositionality principle

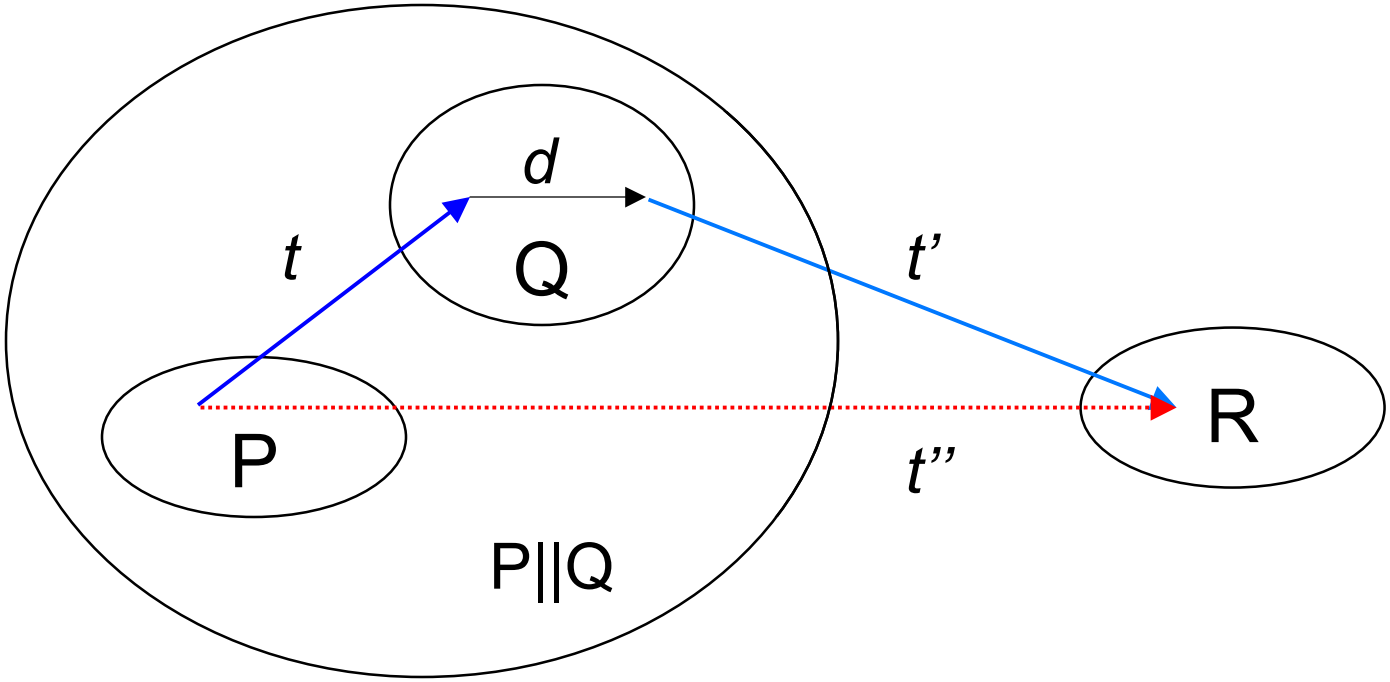


Concurrency : the compositionality principle

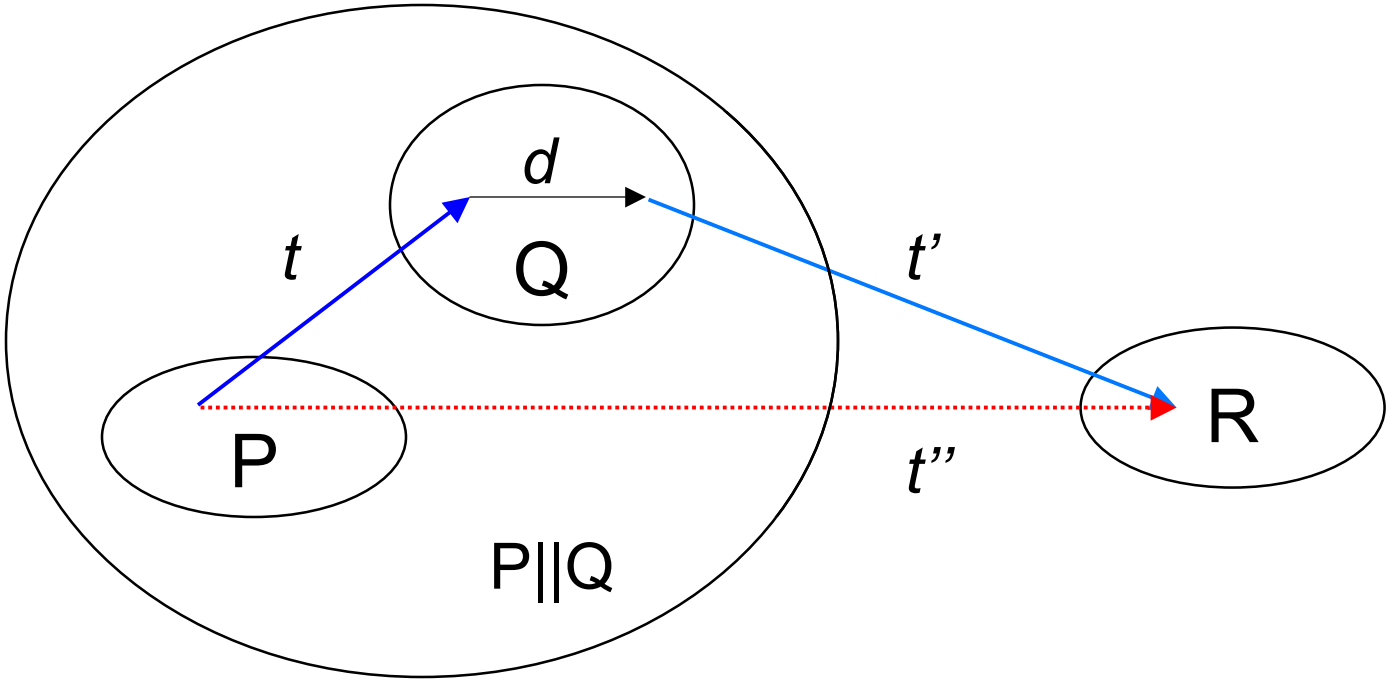


Concurrency : the compositionality principle



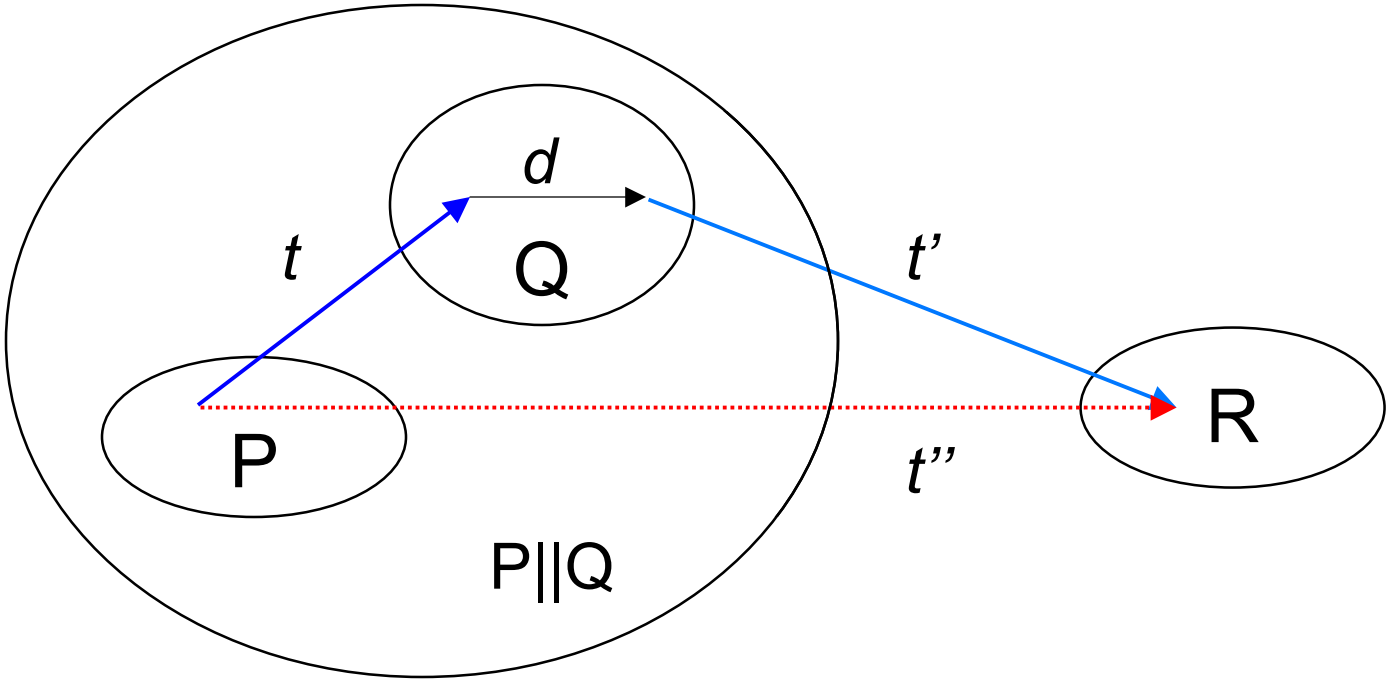


$$t'' = t + d + t'$$



$$t'' = t + d + t'$$

$$t'' \sim t \sim d \sim t'$$



$$t'' = t + d + t'$$

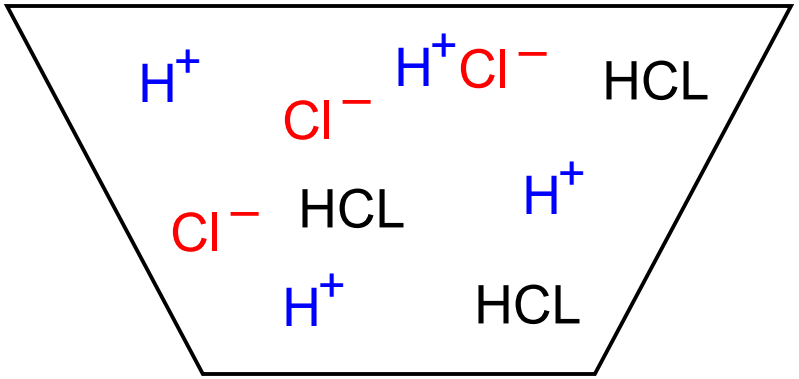
$$t'' \sim t \sim d \sim t'$$

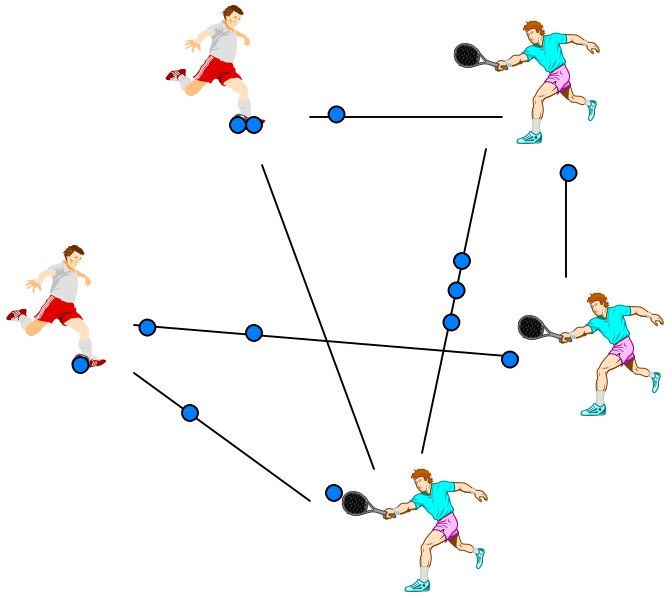
$$t \sim t + t$$

Only 3 solutions :

- *t* arbitrary asynchrony
- *t* = 0 synchrony
- t predictable vibration

Arbitrary Delay : Brownian Motion



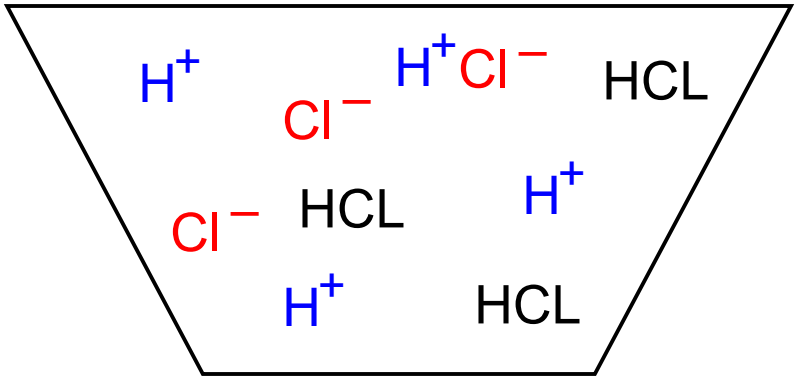


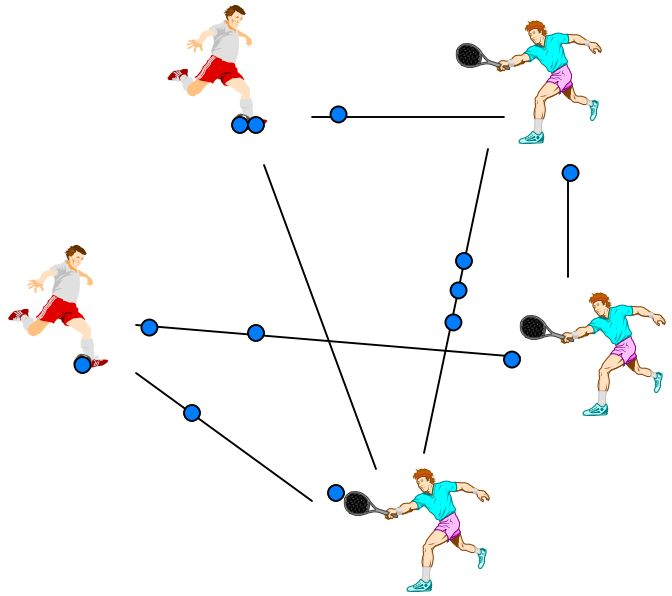
Chemical reaction

$H^+ + Cl^- \longleftrightarrow HCL$

Internet routing

Arbitrary Delay : Brownian Motion





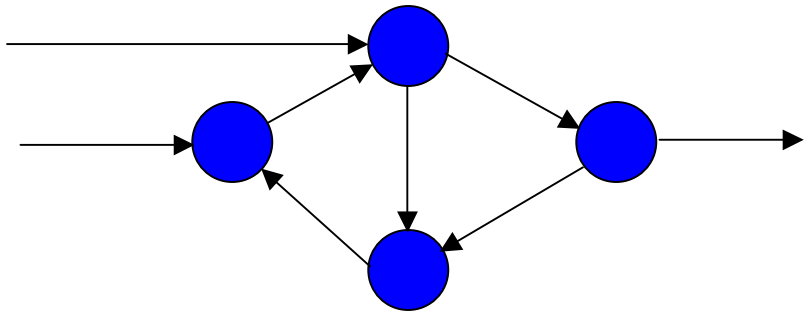
Chemical reaction

$H^+ + Cl^- \longleftrightarrow HCL$

Internet routing

Models : Kahn networks, CSP / ADA,..., π-calculus, CHAM, Join-Calculus, Ambients,

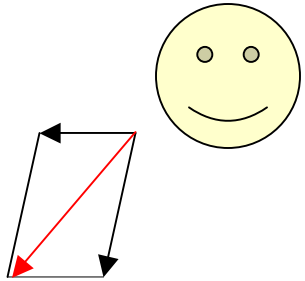
Kahn Networks



nodes = deterministic programs
arrows = infinite fifos

- result-deterministic (independent of computation order)
- easy semantics by flow equations
- heavily used in streaming applications (audio, TV)

Zero delay example: Newtonian Mechanics



Concurrency + Determinism
Calculations are feasible

The Esterel Runner

abort run Slowly when 100 Meter ;

The Esterel Runner

```
abort  run Slowly  when  100 Meter ;
abort
   every Step do
      run Jump || run Breathe
   end every
when 15 Second ;
```

The Esterel Runner

```
abort  run Slowly  when  100 Meter ;
abort
   every Step do
      run Jump || run Breathe
   end every
when 15 Second ;
run FullSpeed
```

The Esterel Runner

```
loop
   abort  run Slowly  when  100 Meter ;
   abort
      every Step do
         run Jump || run Breathe
      end every
   when 15 Second ;
   run FullSpeed
each  Lap
```

The Esterel Runner

```
abort
   loop
      abort  run Slowly  when  100 Meter ;
      abort
         every Step do
            run Jump || run Breathe
         end every
      when 15 Second ;
      run FullSpeed
    each  Lap
when  4 Lap
```

The Esterel Runner

```
every Morning do
   abort
      loop
         abort  run Slowly  when  100 Meter ;
         abort
            every Step do
               run Jump || run Breathe
            end every
         when 15 Second ;
         run FullSpeed
       each  Lap
   when  4 Lap
end every
```

The Esterel Runner

```
trap HeartAttack in
    every Morning do
        abort
            loop
                abort  run Slowly  when  100 Meter ;
                abort
                    every Step do
                        run Jump || run Breathe || run CheckHeart
                    end every
                when 15 Second ;                    exit HeartAttack
                run FullSpeed
            each  Lap
        when  4 Lap
    end every
handle HeartAttack fo
   run RushToHospital
end trap
```

predictable time = vibration

Nothing can illustrate vibration better than Bianca Castafiore, Hergé's famous prima donna. See [1] for details. The power of her voice forcibly shakes the microphone and the ears of the poor spectators.

[1] King's Ottokar Sceptre, Hergé, page 29, last drawing.

propagation of light, sound, electrons, program counter...

Full Abstraction

Bianca Castafiore singing for the King Muskar XII in Klow, Syldavia. King's Ottokar Sceptre, page 38, first drawing.

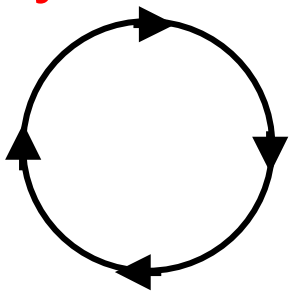
Although the speed of sounds is finite, it is fast enough to look infinite. Full abstraction!

If room is small enough,
predictable delay implements zero-delay

Specify with zero-delay
Implement with predictable delay
Control room size

Software Synchronous Systems

Cycle based



read inputs

compute reaction

produce outputs

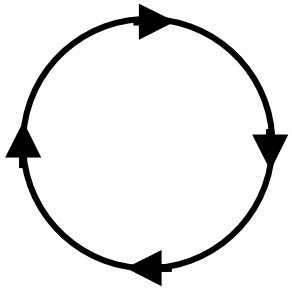
Synchronous = 0-delay = within the same cycle

propagate control

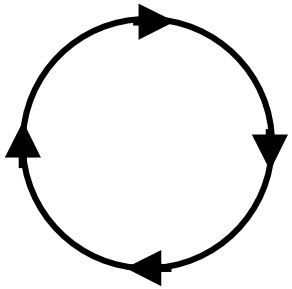
propagate signals

No interference between I/O and computation
Room size control = Worst Case Execution Time (AbsInt)

Concurrency = Cycle Fusion

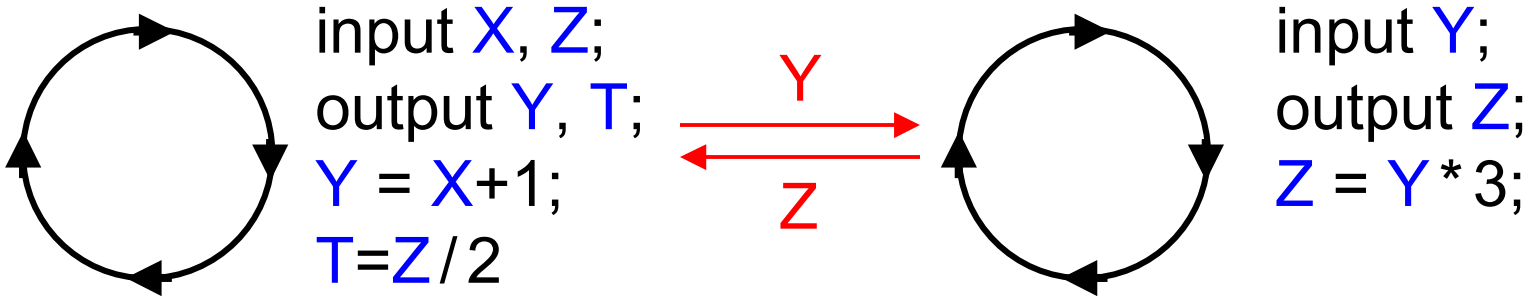


```
input X, Z;
output Y, T;
Y = X+1;
T=Z/2
```

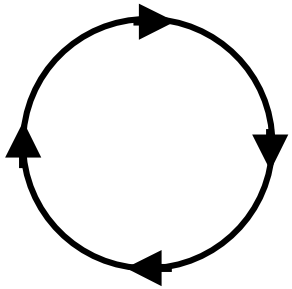


```
input Y;
output Z;
Z = Y*3;
```

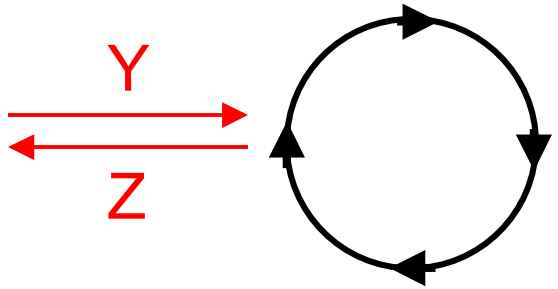
Concurrency = Cycle Fusion



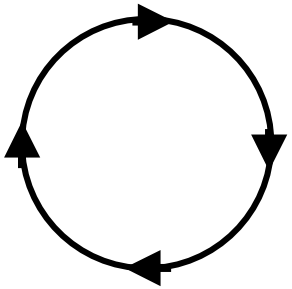
Concurrency = Cycle Fusion



```
input X, Z;
output Y, T;
Y = X+1;
T=Z/2
```

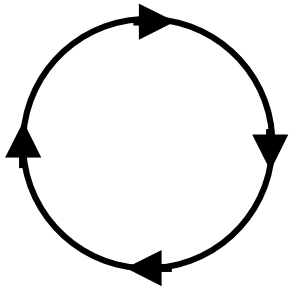



```
input Y;
output Z;
Z = Y*3;
```

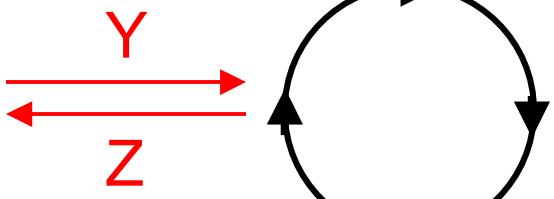


```
input X;
output T;
local Y, Z;
Y = X+1;
Z = Y*3;
T=Z/2;
```

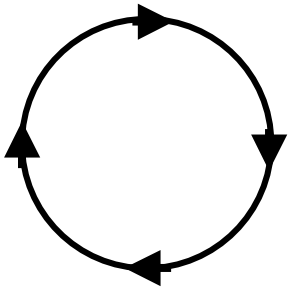
Concurrency = Cycle Fusion



```
input X, Z;
output Y, T;
Y = X+1;
T=Z/2
```




```
input Y;
output Z;
Z = Y*3;
```



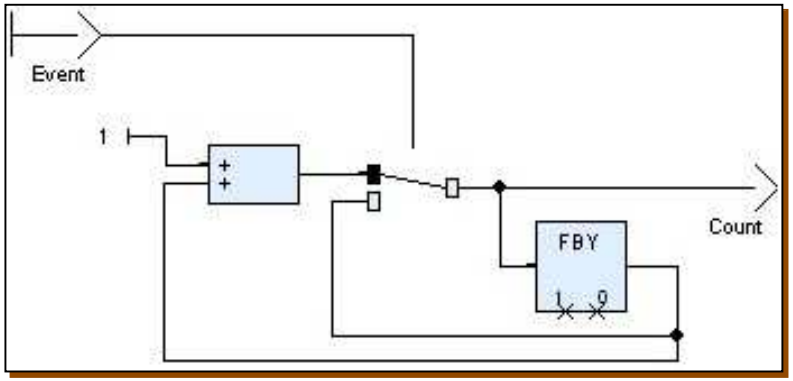
```
input X;
output T;
local Y, Z;
Y = X+1;
Z = Y*3;
T=Z/2;
```

Safe deterministic global variable sharing
No context-switching cost, makes WCET easier

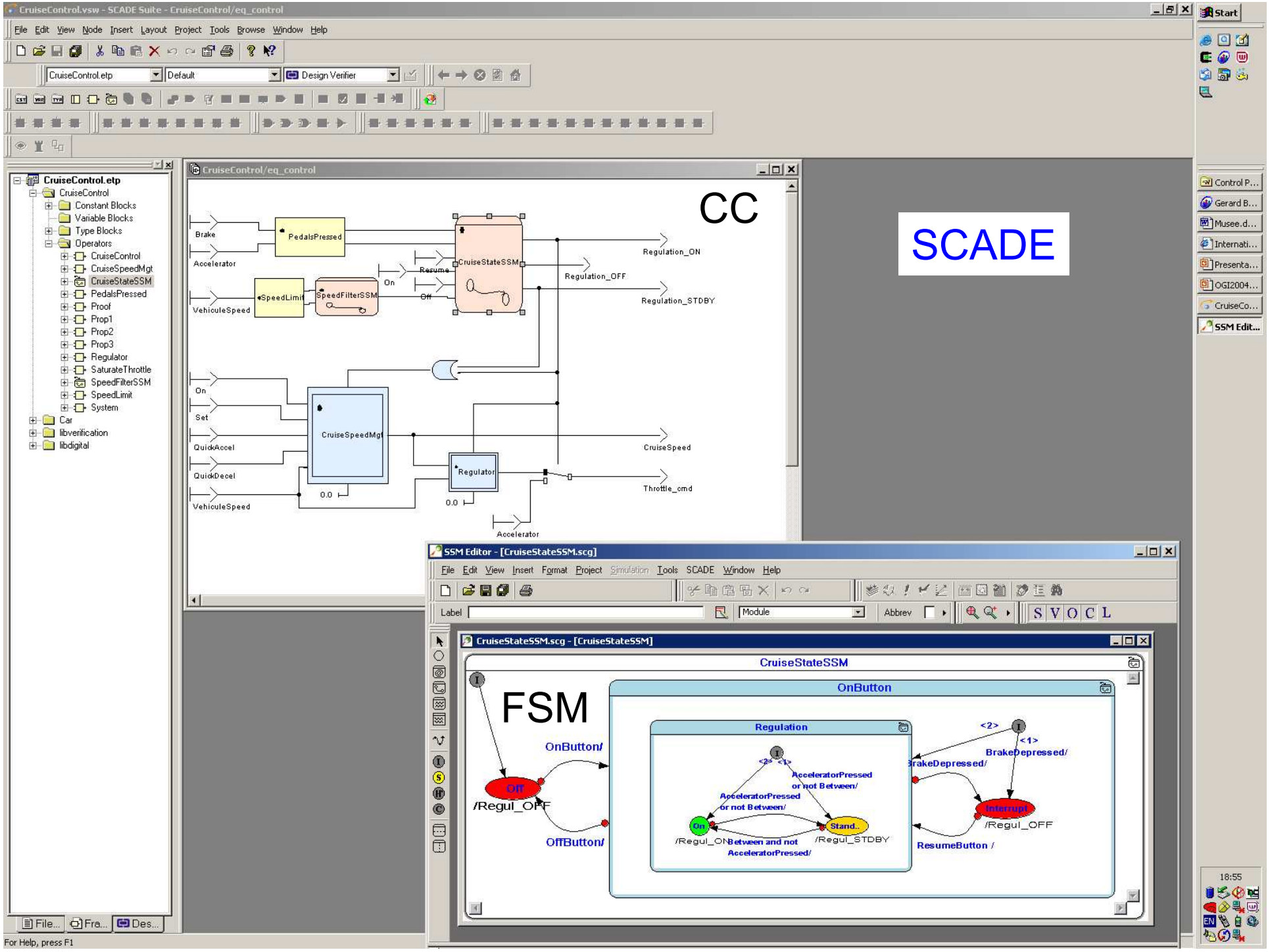
Lustre = Synchronous Kahn Networks

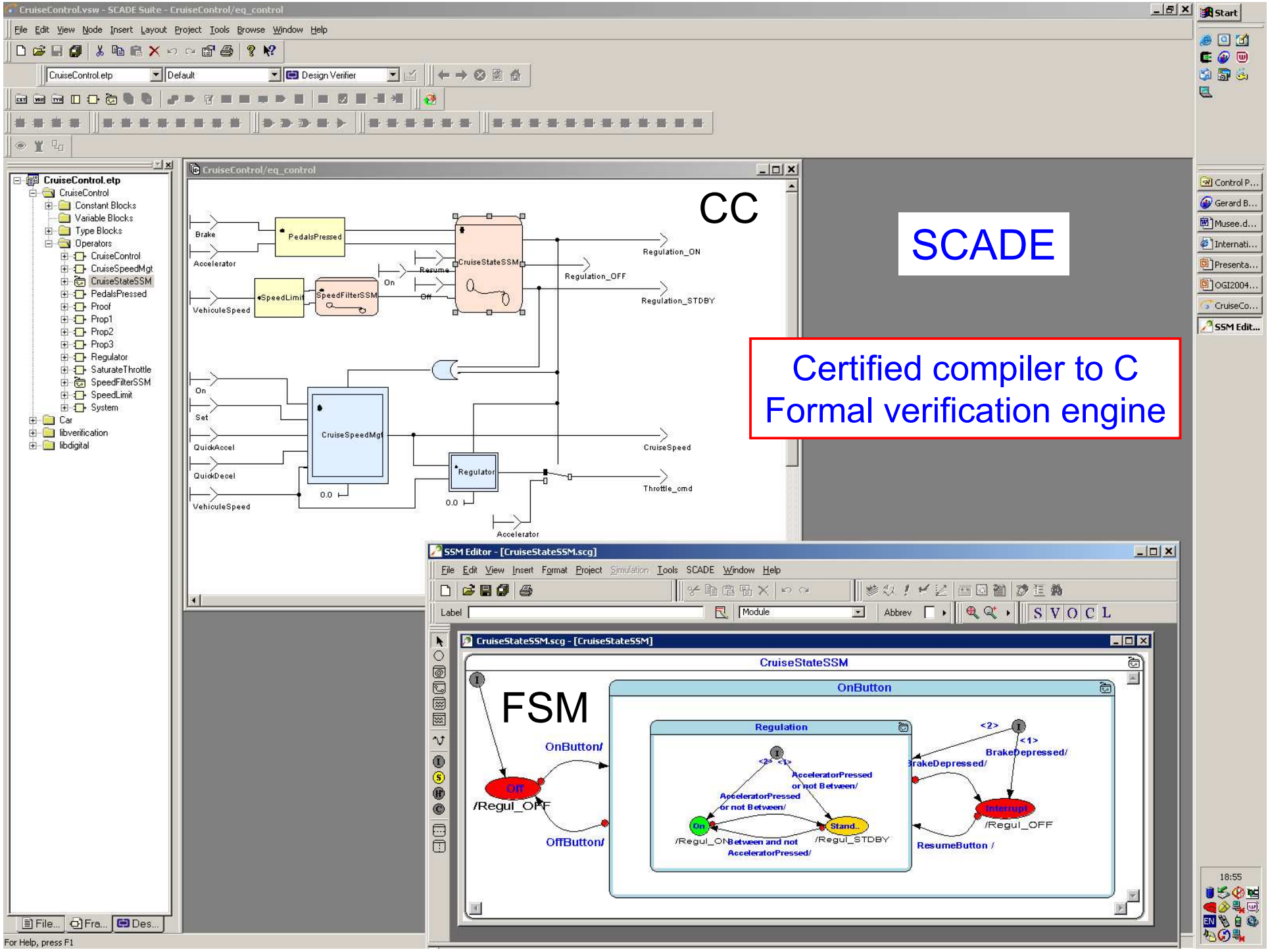
A simple counter

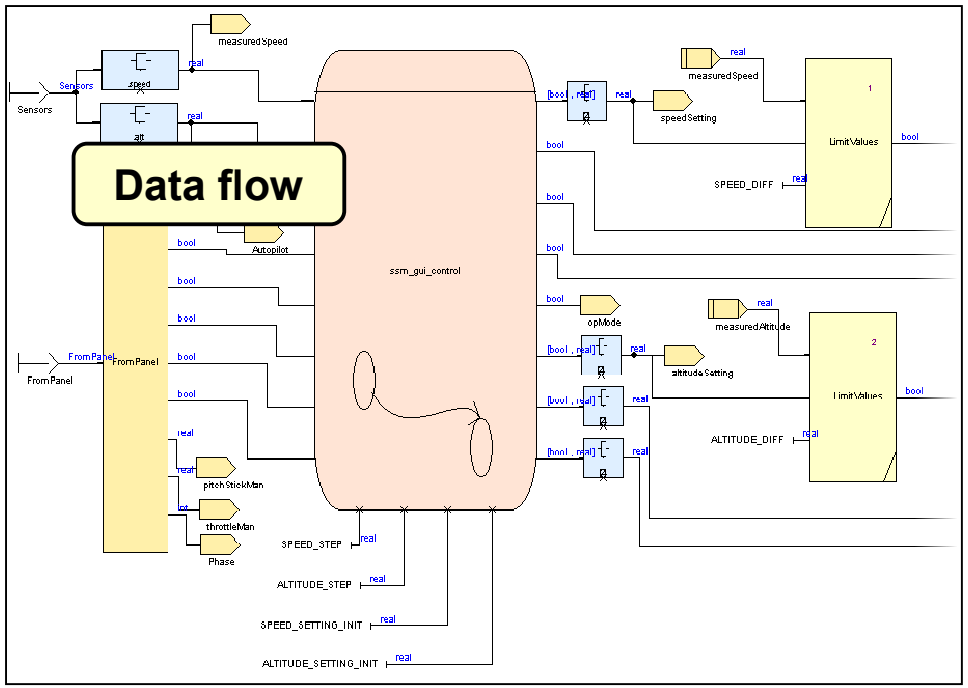
$$\begin{cases} Count(0) = 0 \\ \forall t > 0,\ Count(t) = \begin{cases} Count(t-1)+1, & if\ Event(t) = true \\ Count(t-1), & otherwise \end{cases} \end{cases}$$



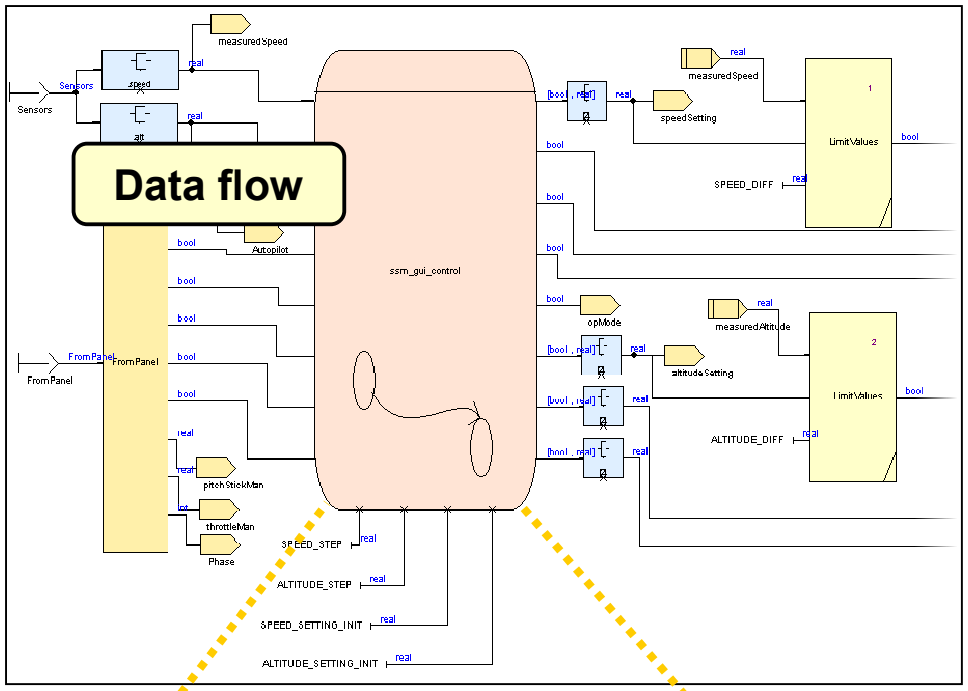

```
Count = 0 ->
    (if Event
     then pre(Count)+1
     else pre(Count))
```

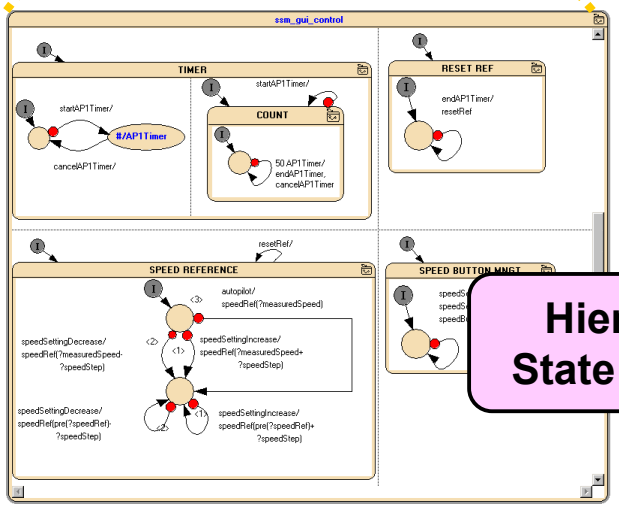

CruiseControl.vsw - SCADE Suite - CruiseControl/eq_control
File Edit View Node Insert Layout Project Tools Browse Window Help
CruiseControl.etp
Default
Design Verifier
CruiseControl.etp
CruiseControl
Constant Blocks
Variable Blocks
Type Blocks
Operators
CruiseControl
CruiseSpeedMgt
CruiseStateSSM
PedalsPressed
Proof
Prop1
Prop2
Prop3
Regulator
SaturateThrottle
SpeedFilterSSM
SpeedLimit
System
Car
libverification
libdigital
CruiseControl/eq_control
CC
Brake
Accelerator
PedalsPressed
CruiseStateSSM
Resume
On
Off
VehicleSpeed
SpeedLimit
SpeedFilterSSM
Regulation_ON
Regulation_OFF
Regulation_STDBY
On
Set
QuickAccel
QuickDecel
VehicleSpeed
CruiseSpeedMgt
0.0
Regulator
0.0
CruiseSpeed
Throttle_cmd
Accelerator
SCADE
SSM Editor - [CruiseStateSSM.scg]
File Edit View Insert Format Project Simulation Tools SCADE Window Help
Label
Module
Abbrev
S V O C L
CruiseStateSSM.scg - [CruiseStateSSM]
CruiseStateSSM
FSM
OnButton
Regulation
OnButton/
Off
/Regul_OFF
OffButton/
On
/Regul_ON
Stand..
/Regul_STDBY
AcceleratorPressed or not Between/
Between and not AcceleratorPressed/
<2>
<1>
BrakeDepressed/
Interrupt
/Regul_OFF
ResumeButton /
Start
Control P...
Gerard B...
Musee.d...
Internati...
Presenta...
OGI2004...
CruiseCo...
SSM Edit...
18:55
File...
Fra...
Des...
For Help, press F1


CruiseControl.vsw - SCADE Suite - CruiseControl/eq_control
CruiseControl/eq_control
CC
SCADE
Certified compiler to C
Formal verification engine
SSM Editor - [CruiseStateSSM.scg]
CruiseStateSSM.scg - [CruiseStateSSM]
FSM
CruiseStateSSM
OnButton
Regulation
Off
/Regul_OFF
OnButton/
OffButton/
On
/Regul_ON
Stand..
/Regul_STDBY
Interrupt
/Regul_OFF
BrakeDepressed/
ResumeButton /
AcceleratorPressed
or not Between/
Between and not
AcceleratorPressed/
Brake
Accelerator
VehiculeSpeed
PedalsPressed
SpeedLimit
SpeedFilterSSM
CruiseStateSSM
Resume
On
Off
Regulation_ON
Regulation_OFF
Regulation_STDBY
Set
QuickAccel
QuickDecel
CruiseSpeedMgt
Regulator
CruiseSpeed
Throttle_cmd
0.0



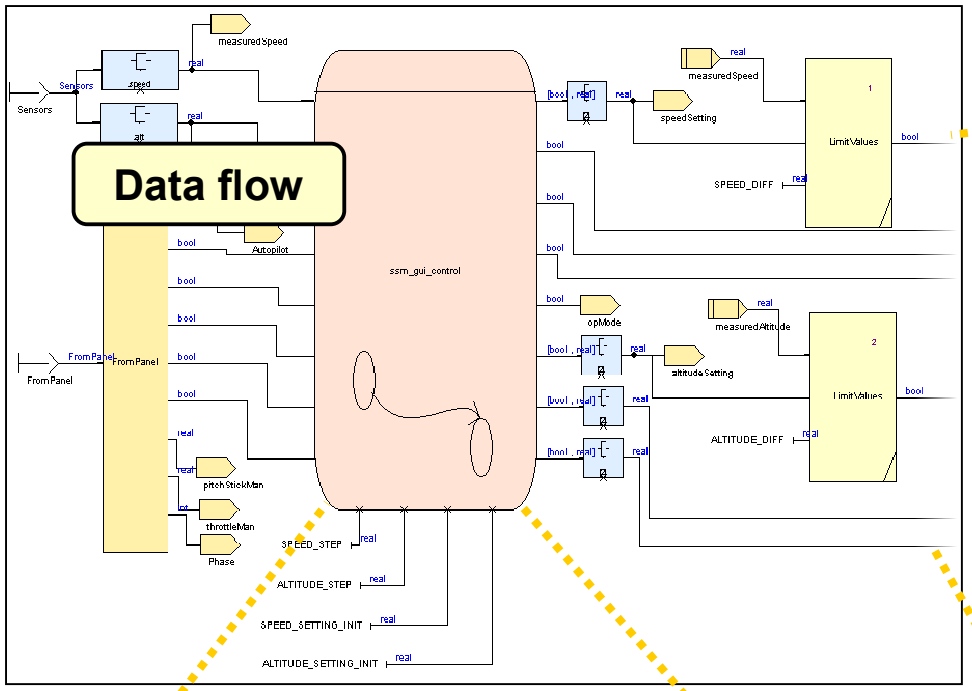
Existing Capabilities

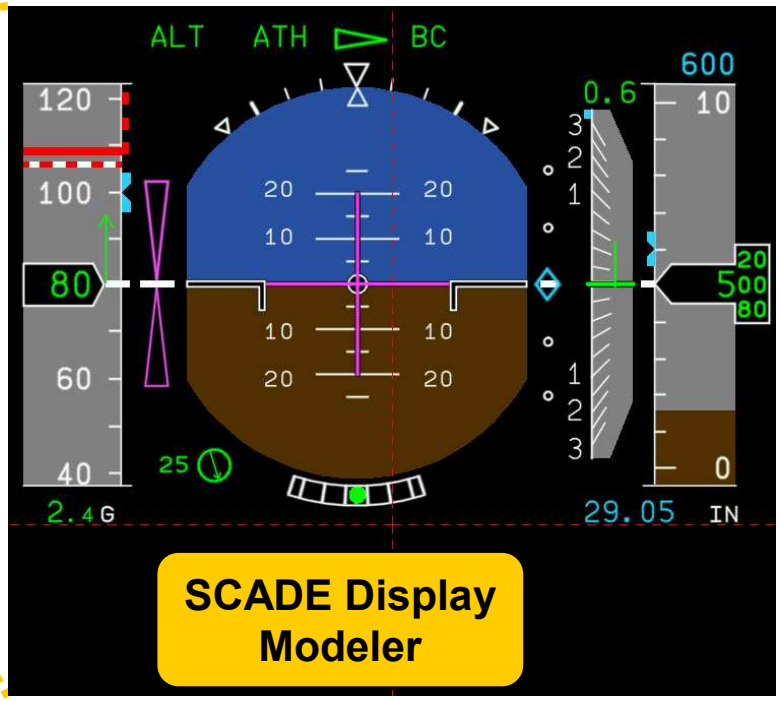


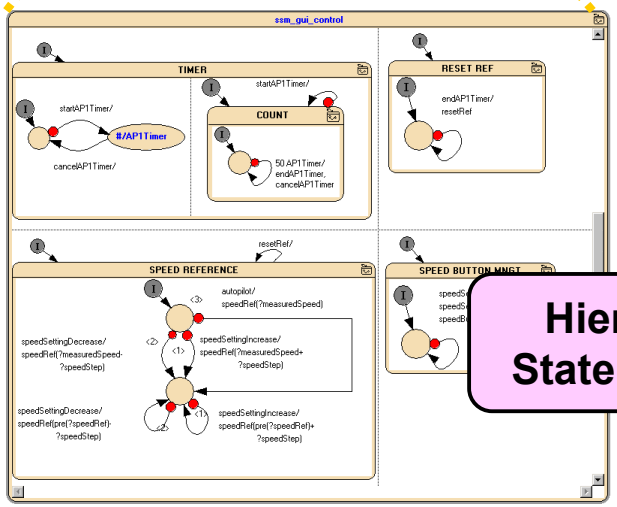


Hierarchical State Machines

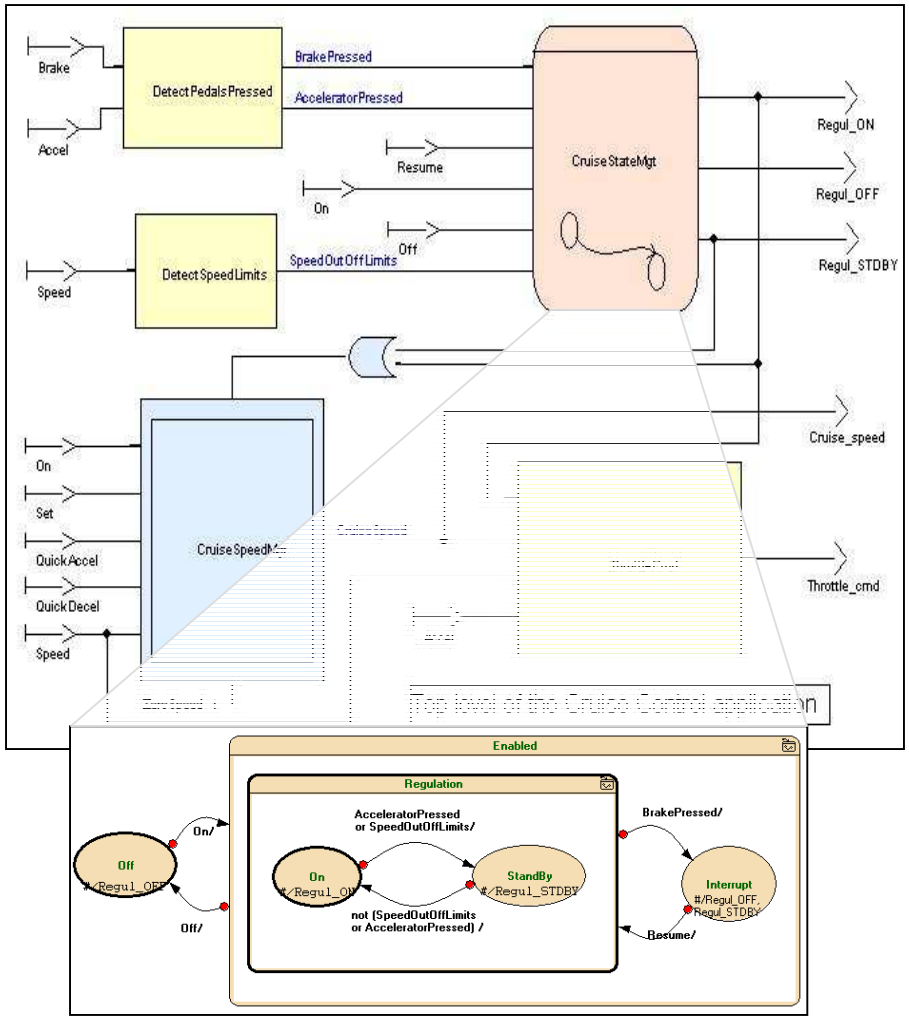
Existing Capabilities







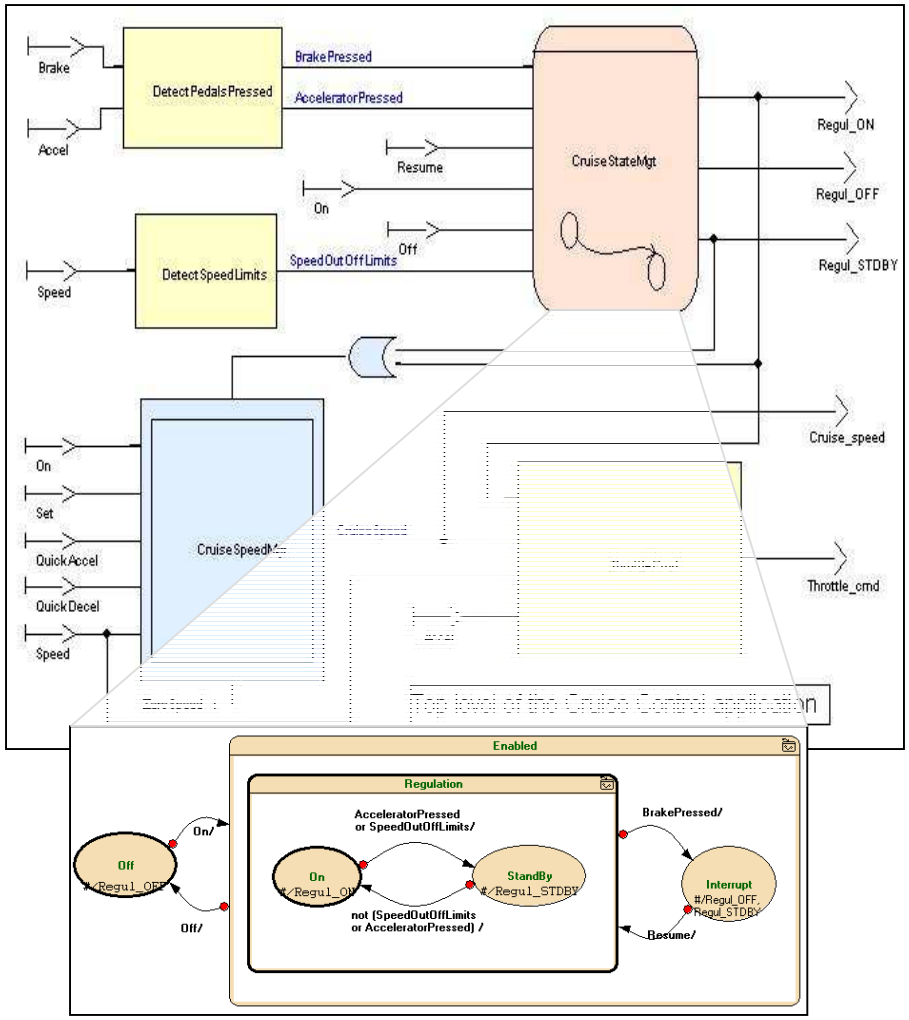
Scade 6 : Data - Control Flow Unification



SCADE 5

Pure Control Flow into
Pure Data Flow

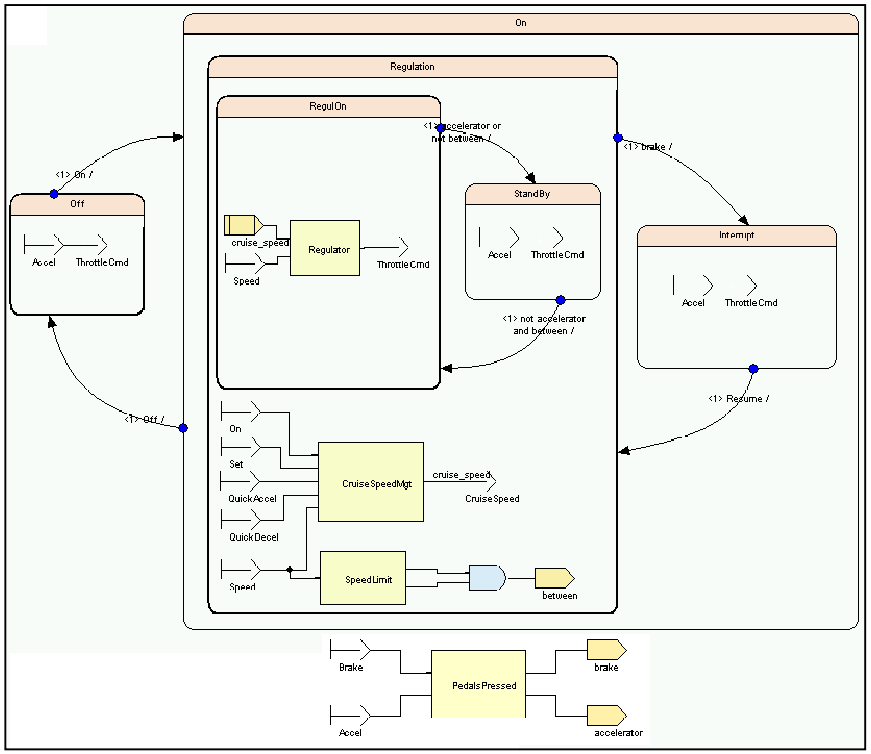
Scade 6 : Data - Control Flow Unification



SCADE 5

Pure Control Flow into Pure Data Flow

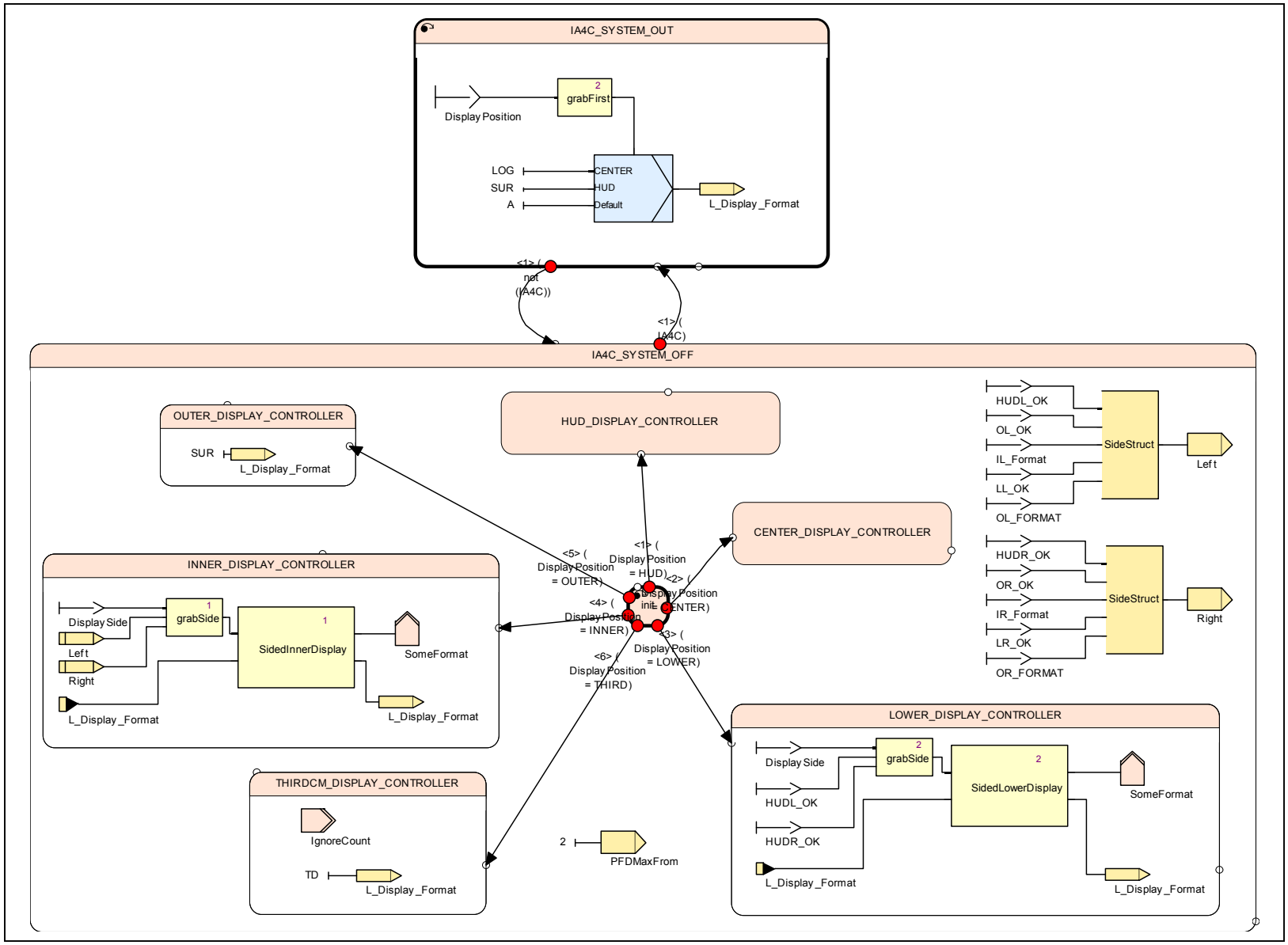
- Freely mixable in hierarchy



SCADE 6

Unified Data Flow & Control Flow

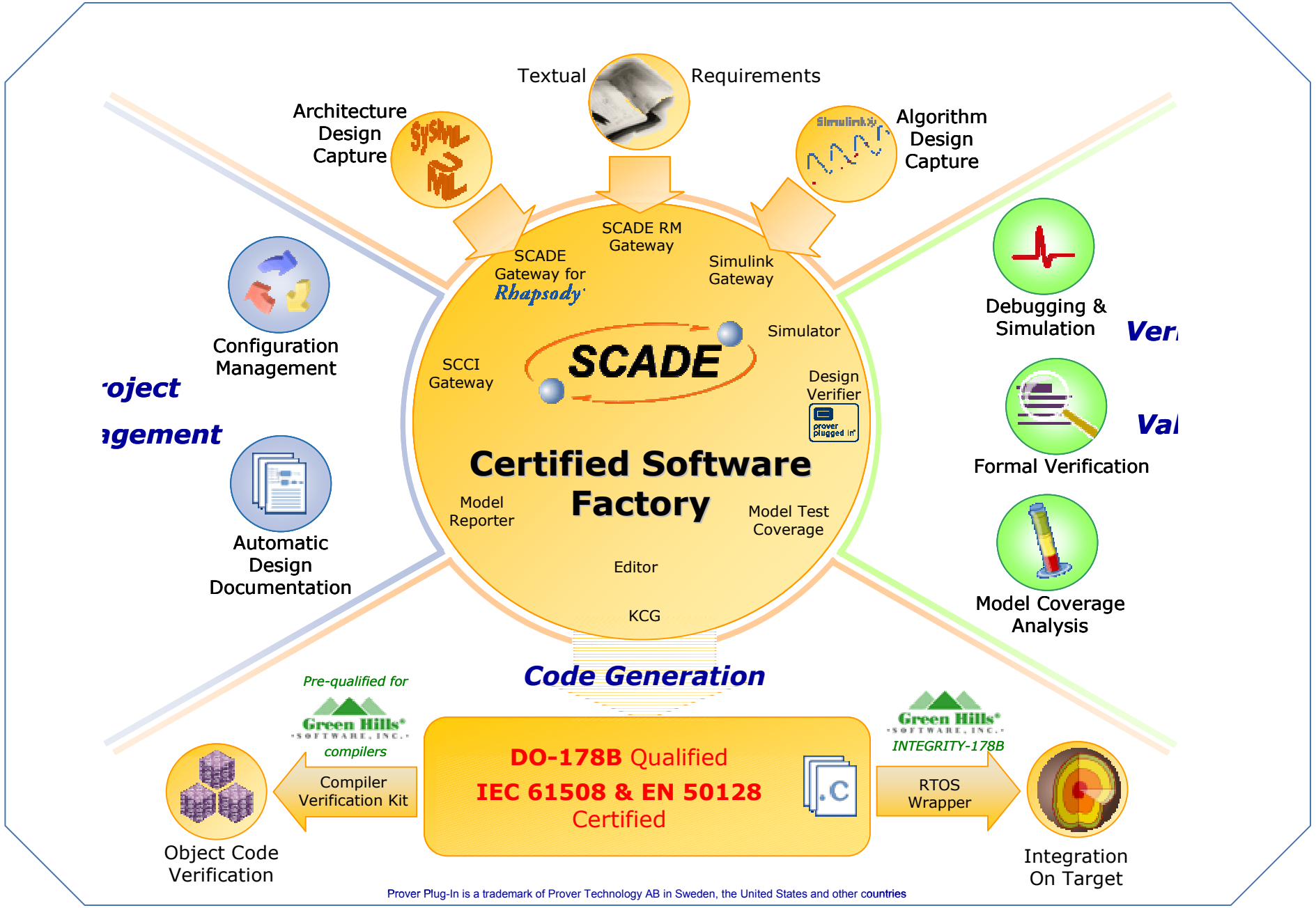
A Typical Application: Cockpit Display

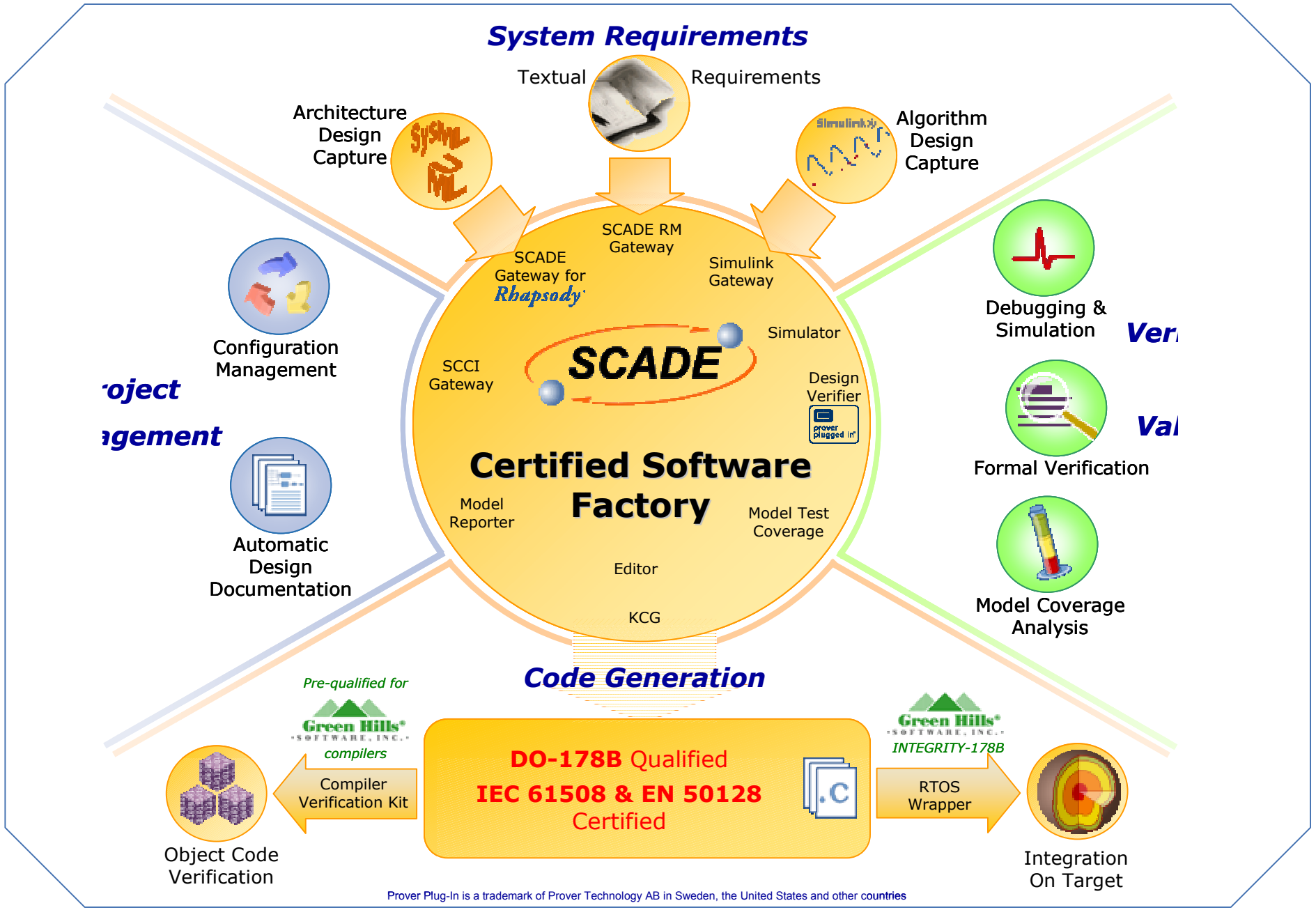


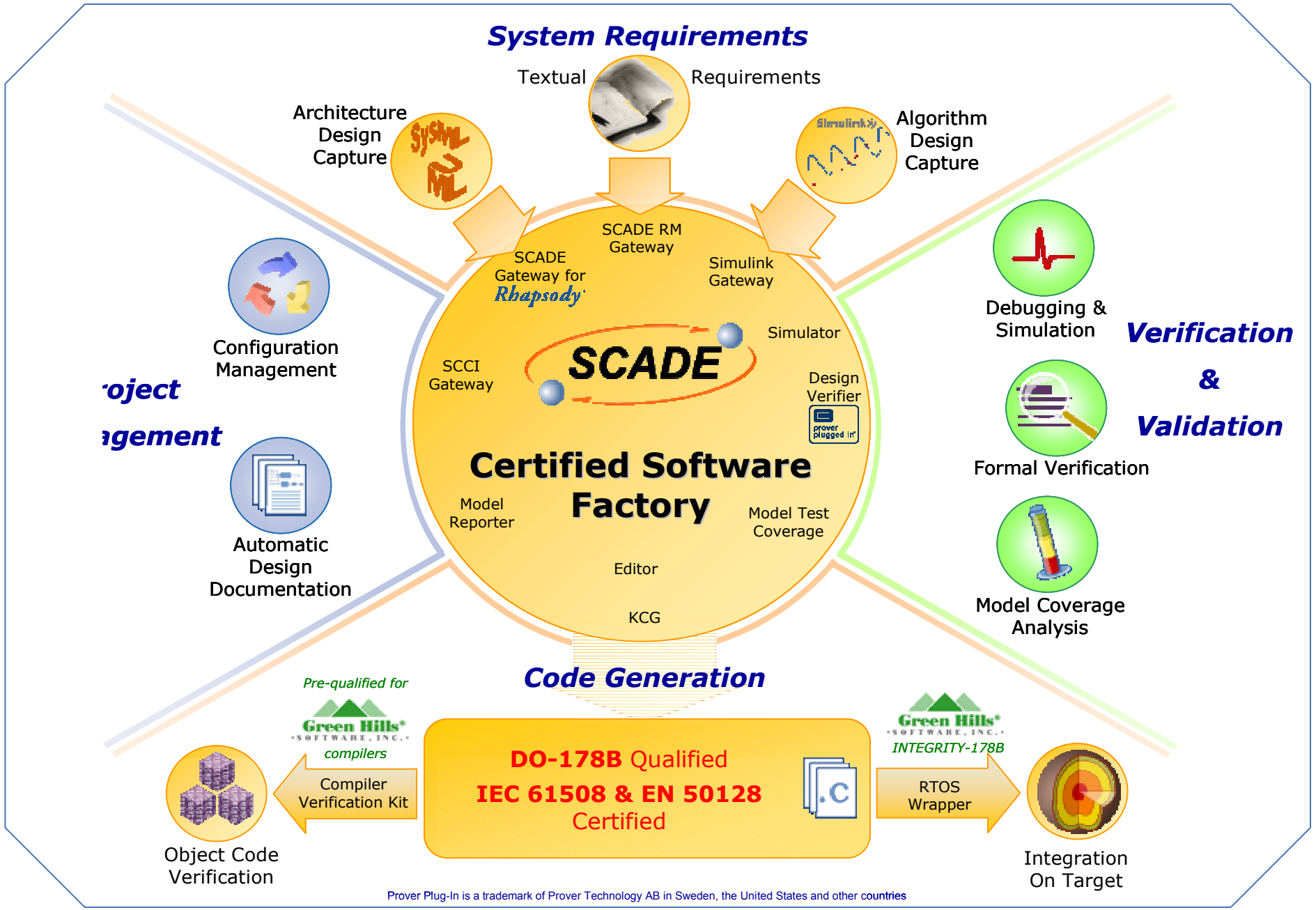
Synchronous Semantics

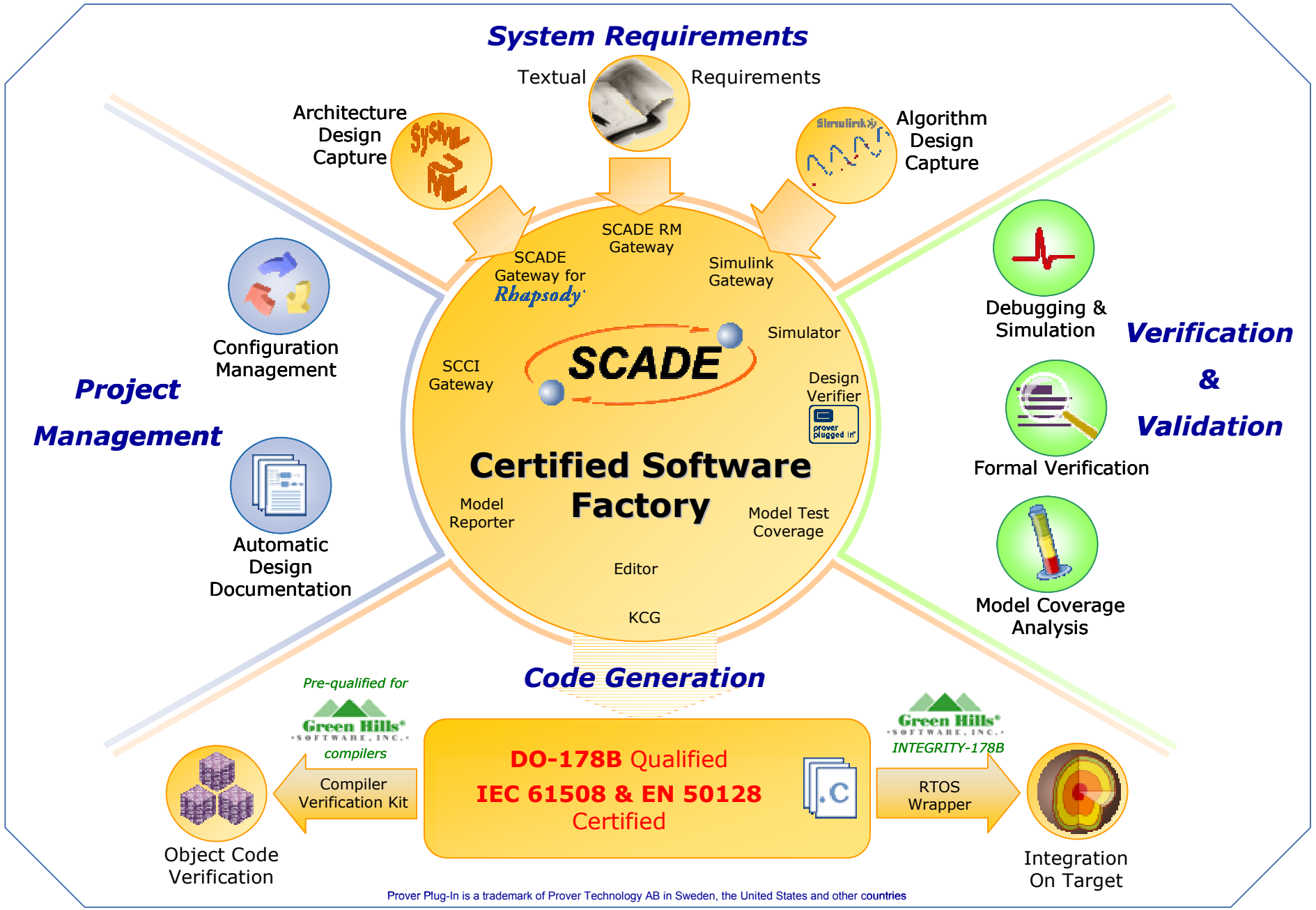
- Ensures every data is produced exactly once
- Additional checks
  - no access to undefined data
  - no race condition (combinational cycle)
    => deterministic scheduling-independent result
  - no recursion in node calls
    => static memory allocation, bounded stack

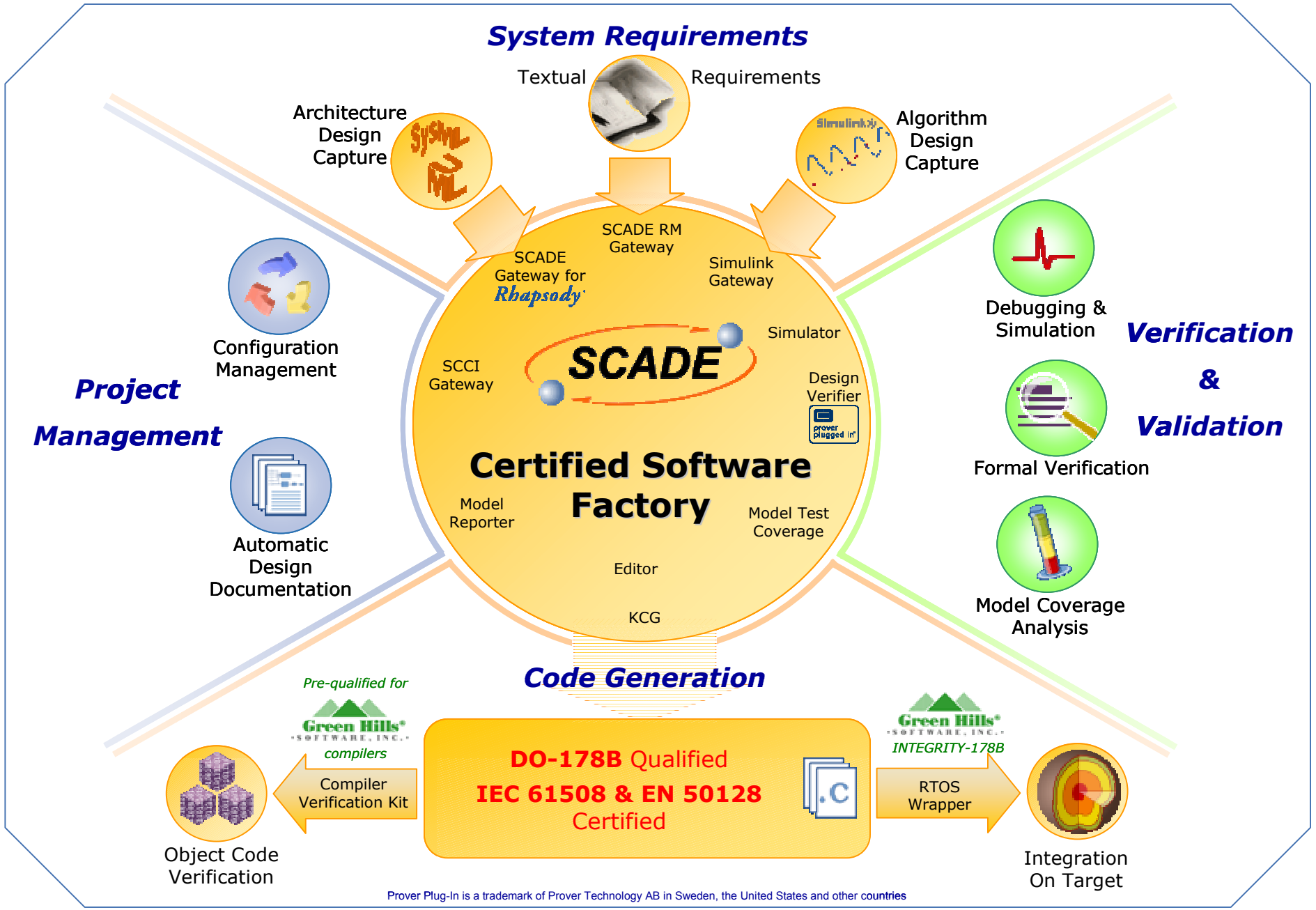
Checked by the qualified code generator as a prerequisite to code generation











Code Generation with KCG

- KCG is the qualifiable C code generator
  - developed with a DO-178B Level A process
  - certification authorities certified that “KCG can fly” and qualified it as a development tool

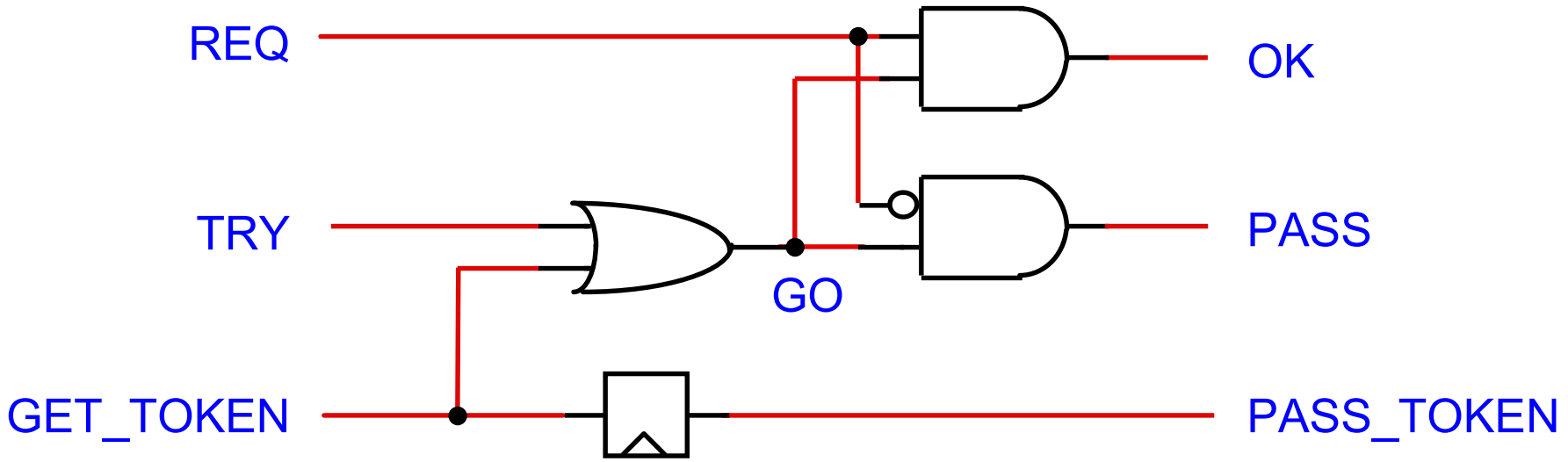
  => no need to unit-test the generated C code
- Evidences provided to users
  - qualification kit
  - verifiable, traceable, and safe code
  - C compiler verification kit

Generated Code Properties

- Small C subset
- Portable (compiler, target and OS independent)
- Structured (by function or by blocks)
- Readable, traceable (names/annotations propagation)
- Safe static memory allocation
- No pointer arithmetic
- No recursion, no loop
- Bounded execution time
- Size and / or speed optimizations

Easy static analysis (Astrée)

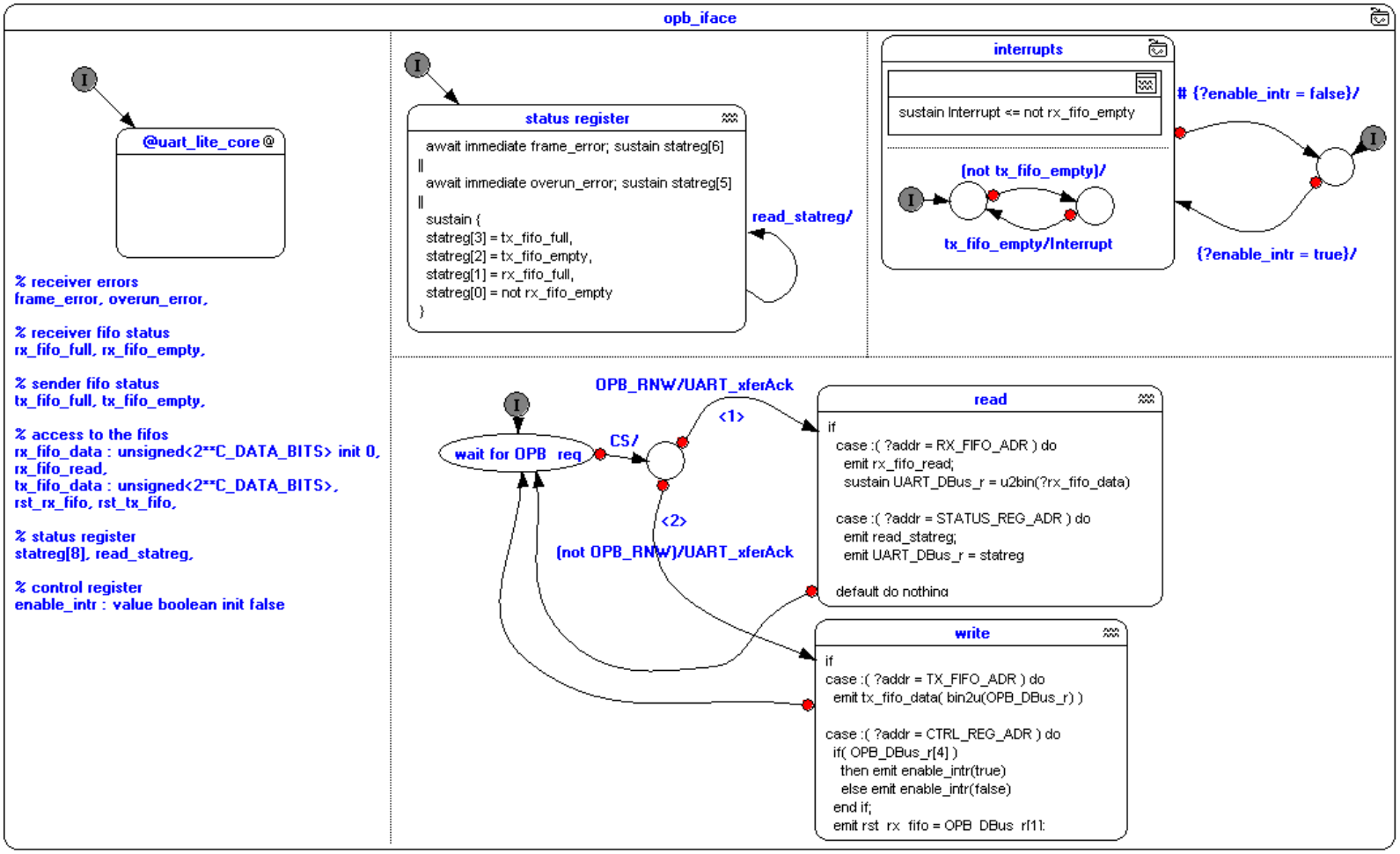
Hardware Synchrony: the RTL model



OK = REQ and GO
PASS = not REQ and GO
GO = TRY or GET_TOKEN
PASS_TOKEN = reg(GET_TOKEN)

Room size control = timing closure

Esterel v7 (Berry – Kishinevsky)

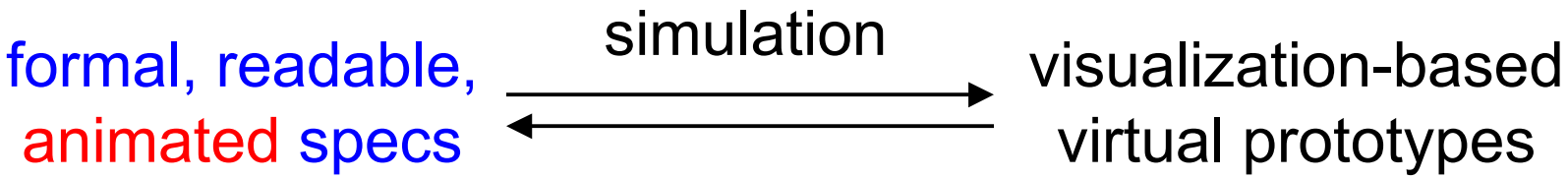


text + graphics, concurrency + sequencing
clear semantics

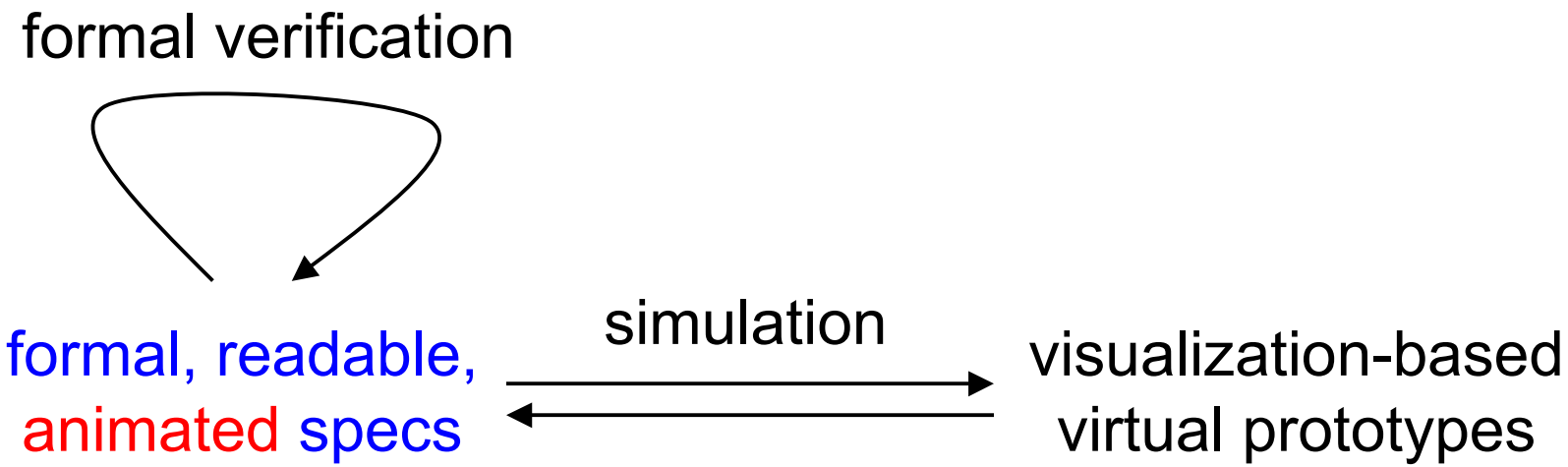
The Esterel Studio Usage Model

formal, readable,
animated specs

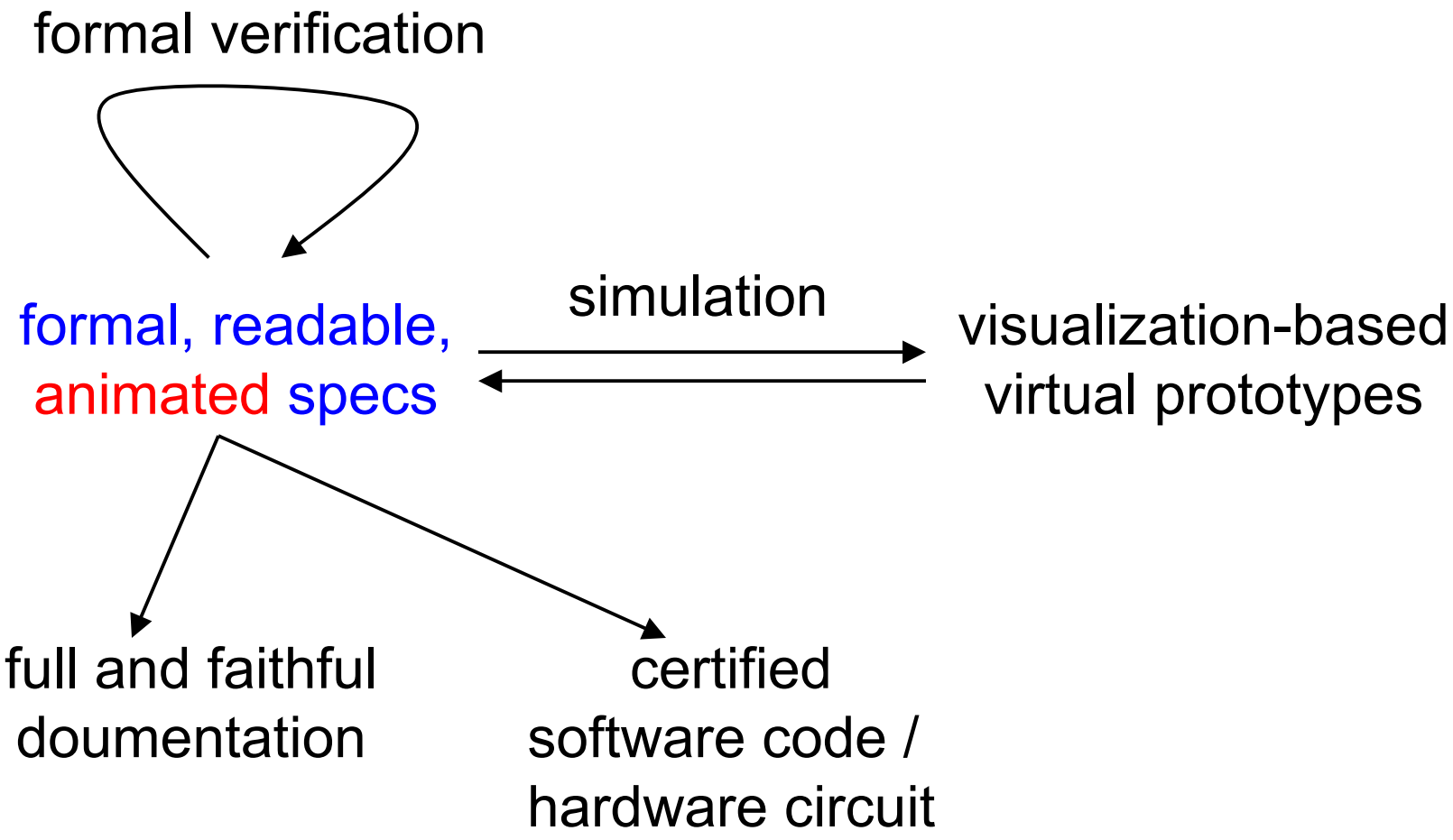
The Esterel Studio Usage Model



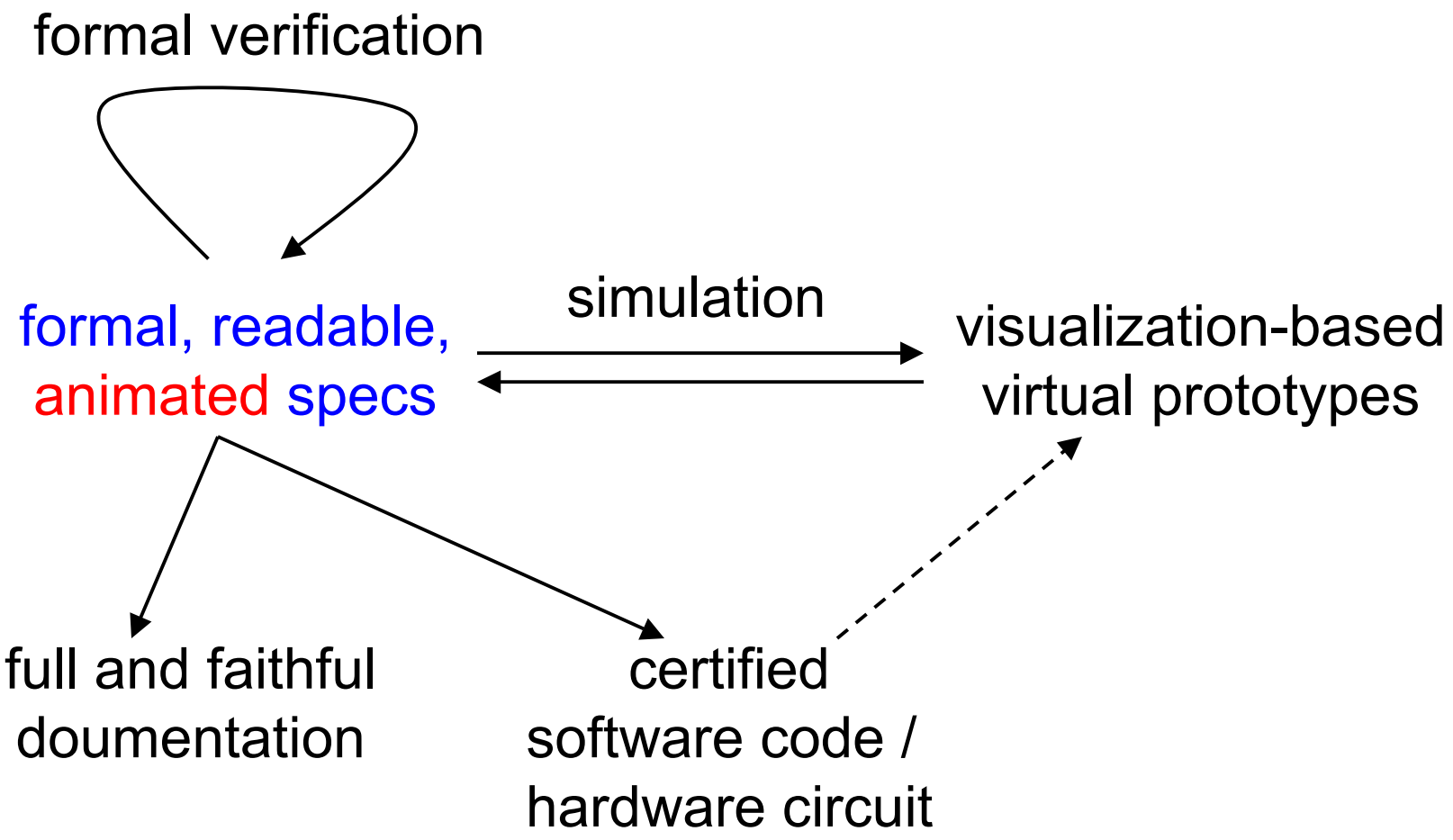
The Esterel Studio Usage Model



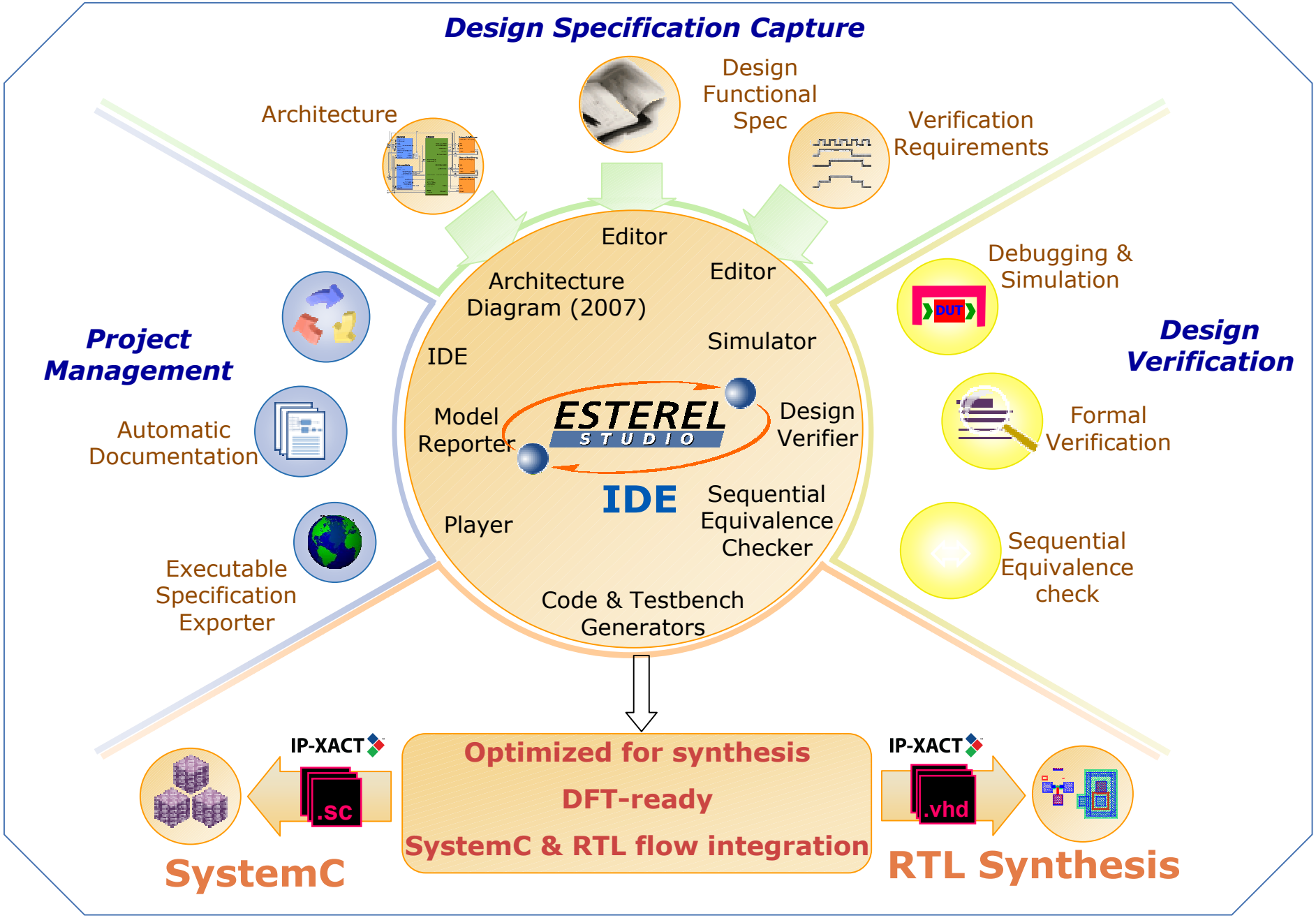
The Esterel Studio Usage Model

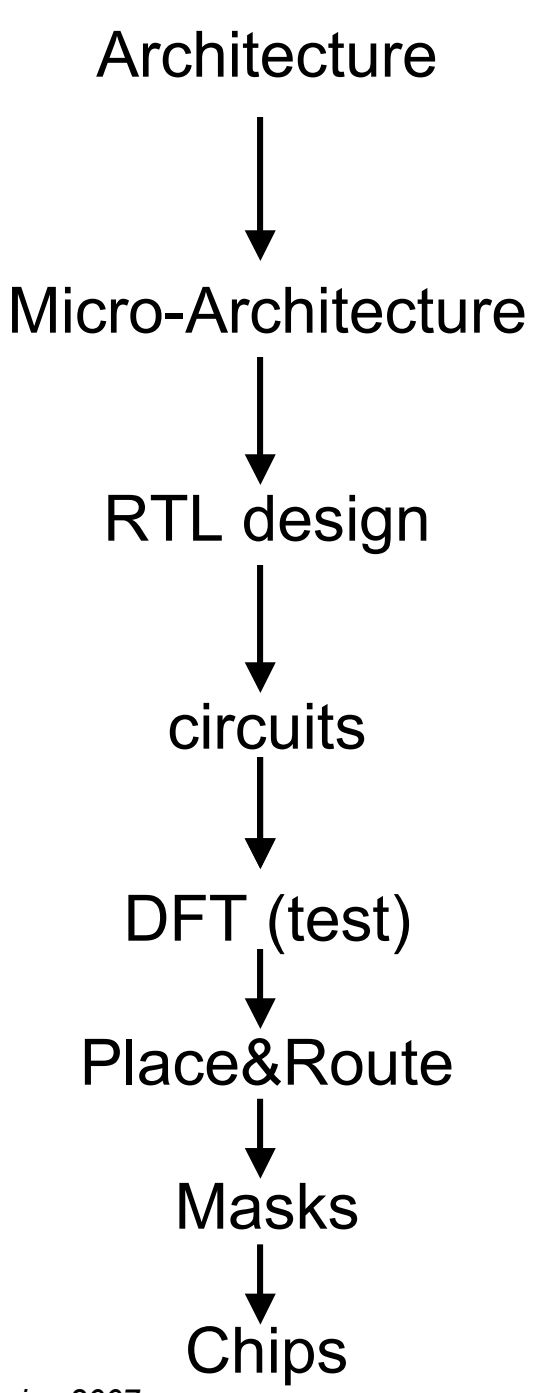


The Esterel Studio Usage Model



Esterel Studio at a glance





components
dimensioning
communication

concurrency
pipeline
resource sharing

gates, clocks
registers, RAMs
critical path

cells, clock trees
area, speed

testability
scan insertion

Word, Excel, Visio
System C

Word, Visio

VHDL, Verilog

Verilog + ...

Architecture
components
dimensioning
communication
Word, Excel, Visio
System C

↓

Micro-Architecture
concurrency
pipeline
resource sharing
Word, Visio

↓

RTL design
gates, clocks
registers, RAMs
critical path
VHDL, Verilog

↓

circuits
cells, clock trees
area, speed
Verilog + ...

↓

DFT (test)
testability
scan insertion

↓

Place&Route

↓

Masks

↓

Chips

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

DFT (test)

testability
scan insertion

Place&Route

Masks

Chips

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

DFT (test)

testability
scan insertion

Place&Route

Masks

Chips

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

DFT (test)

testability
scan insertion

Place&Route

Masks

Chips

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

DFT (test)

testability
scan insertion

Place&Route

$ 1,000,000 Masks

Chips

Esterel

$ 1,000,000

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

DFT (test)

testability
scan insertion

Place&Route

Masks

Chips

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Esterel

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

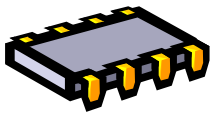
DFT (test)

testability
scan insertion

Place&Route

$ 1,000,000

Masks

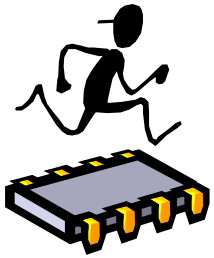


Chips

Esterel

Architecture	components dimensioning communication	Word, Excel, Visio System C
Micro-Architecture	concurrency pipeline resource sharing	Word, Visio
RTL design	gates, clocks registers, RAMs critical path	VHDL, Verilog
circuits	cells, clock trees area, speed	Verilog + ...
DFT (test)	testability scan insertion	
Place&Route		
Masks		
Chips		

$ 1,000,000



Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Esterel

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

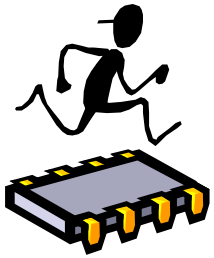
DFT (test)

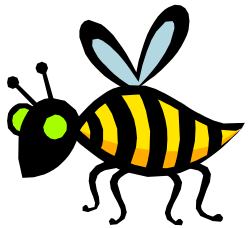
testability
scan insertion

Place&Route

$ 1,000,000

Masks





Chips

Esterel

$ 1,000,000

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

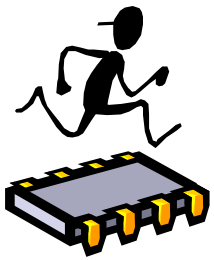
DFT (test)

testability
scan insertion

Place&Route

Masks

Chips



Esterel

$ 1,000,000

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

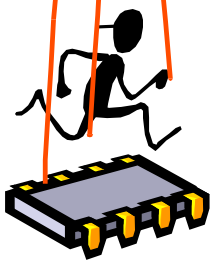
DFT (test)

testability
scan insertion

Place&Route

Masks

Chips



Esterel

$ 1,000,000

Architecture

components
dimensioning
communication

Word, Excel, Visio
System C

Micro-Architecture

concurrency
pipeline
resource sharing

Word, Visio

RTL design

gates, clocks
registers, RAMs
critical path

VHDL, Verilog

circuits

cells, clock trees
area, speed

Verilog + ...

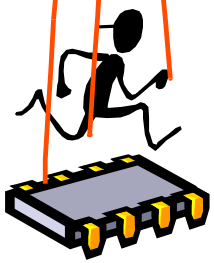
DFT (test)

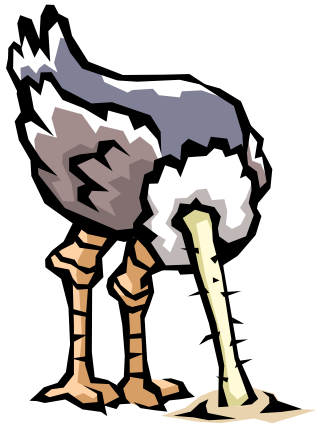
testability
scan insertion

Place&Route

Masks

Chips





Computer Science at Work

Computer Science at Work

1. Language design & mathematical semantics
   - Esterel: imperative, SOS semantics (residuals)
     - constructive logic, proof networks
   - Lustre/SCADE: declarative, functional, denotational semantics
     - clock calculus = static type-check of dynamics

Computer Science at Work

1. Language design & mathematical semantics
   - Esterel: imperative, SOS semantics (residuals)
     - constructive logic, proof networks
   - Lustre/SCADE: declarative, functional, denotational semantics
     - clock calculus = static type-check of dynamics
2. Compiling
   - Esterel: translation to circuits (hardware)
     - translation to concurrent flow graphs (software)
   - Lustre/SCADE: clock calculus to drive expression execution
   - All: static scheduling of elementary actions

Computer Science at Work

1. Language design & mathematical semantics
   - Esterel: imperative, SOS semantics (residuals)
     - constructive logic, proof networks
   - Lustre/SCADE: declarative, functional, denotational semantics
     - clock calculus = static type-check of dynamics
2. Compiling
   - Esterel: translation to circuits (hardware)
     - translation to concurrent flow graphs (software)
   - Lustre/SCADE: clock calculus to drive expression execution
   - All: static scheduling of elementary actions
3. Formal Verification: properties and equivalence
   - forward / backward reachable state space analysis (BDDs)
   - SAT + numerical solving
   - Abstract Interpretation (Astrée, Cousot)

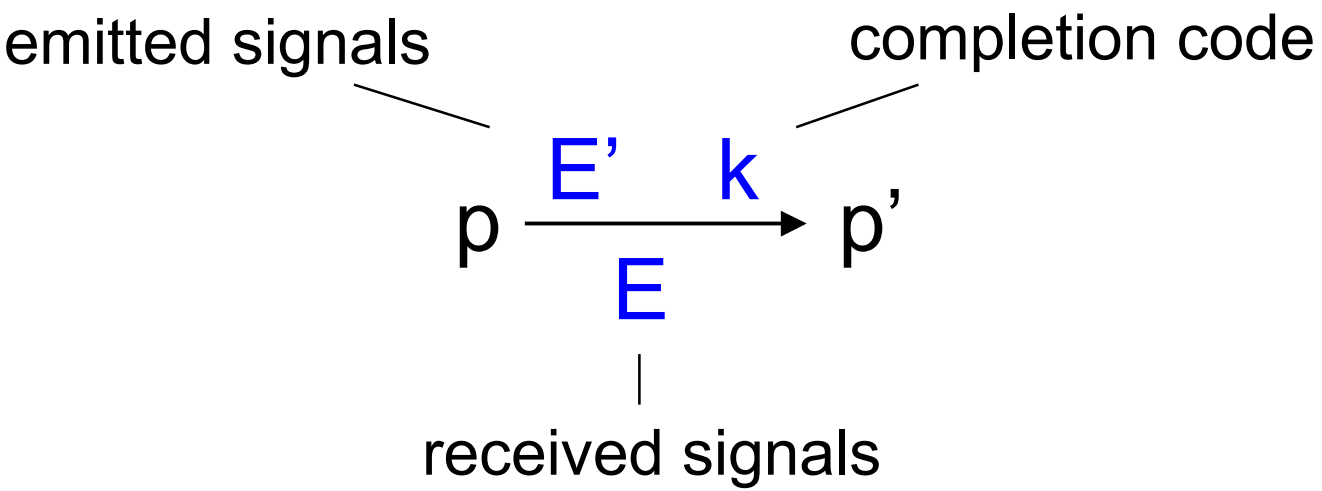
The Pure Esterel Kernel

nothing
pause
emit S
present S then p else q end
suspend p when S
p; q
loop p end
p || q
trap T in p end
exit T
signal S in p end

The Pure Esterel Kernel

nothing	0
pause	1
emit S	! s
present S then p else q end	s ? p, q
suspend p when S	s ⊃ p
p; q	p; q
loop p end	p*
p || q	p | q
trap T in p end	{p} ↑ p
exit T	k, k > 1
signal S in p end	p \ s

The Behavioral Semantics



Broadcasting : $E' \subset E$

k
- 0 : termination
- 1 : waiting
- 2 : exiting one trap level
- 3 : exiting two trap levels

$$\frac{p \xrightarrow[E]{E' \quad k} p' \qquad k \neq 0}{p\,;\,q \xrightarrow[E]{E' \quad k} p'\,;\,q}$$

$$\frac{p \xrightarrow[E]{E' \; k} p' \qquad k \neq 0}{p \; ; \; q \xrightarrow[E]{E' \; k} p' \; ; \; q}$$

$$\frac{p \xrightarrow[E]{E' \; 0} p' \qquad q \xrightarrow[E]{F' \; l} q'}{p \; ; \; q \xrightarrow[E]{E' \cup F' \; l} q'}$$

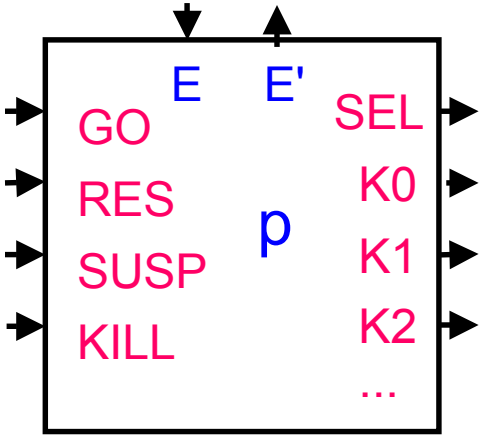
$$\frac{p \xrightarrow[E]{E' \quad k} p' \qquad k \neq 0}{p* \xrightarrow[E]{E' \quad k} p' ; p*}$$

$$\frac{p \xrightarrow[E]{E' \quad k} p' \qquad k \neq 0}{p* \xrightarrow[E]{E' \quad k} p' ; p*}$$

$$\frac{p \xrightarrow[E]{E' \quad k} p' \qquad q \xrightarrow[E]{F' \quad l} q'}{p \mid q \xrightarrow[E]{E' \cup F' \quad max(k,l)} p' \mid q'}$$

Basic syntax directed translation scheme

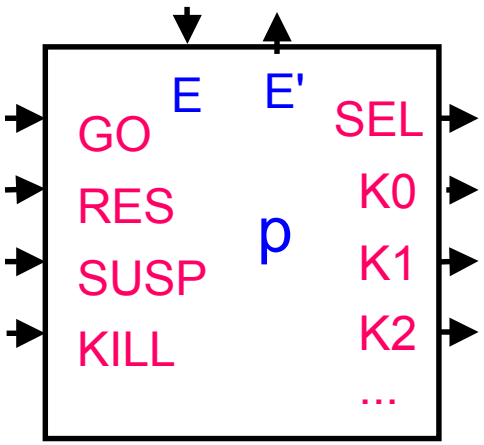
Each statement p corresponds to a box:



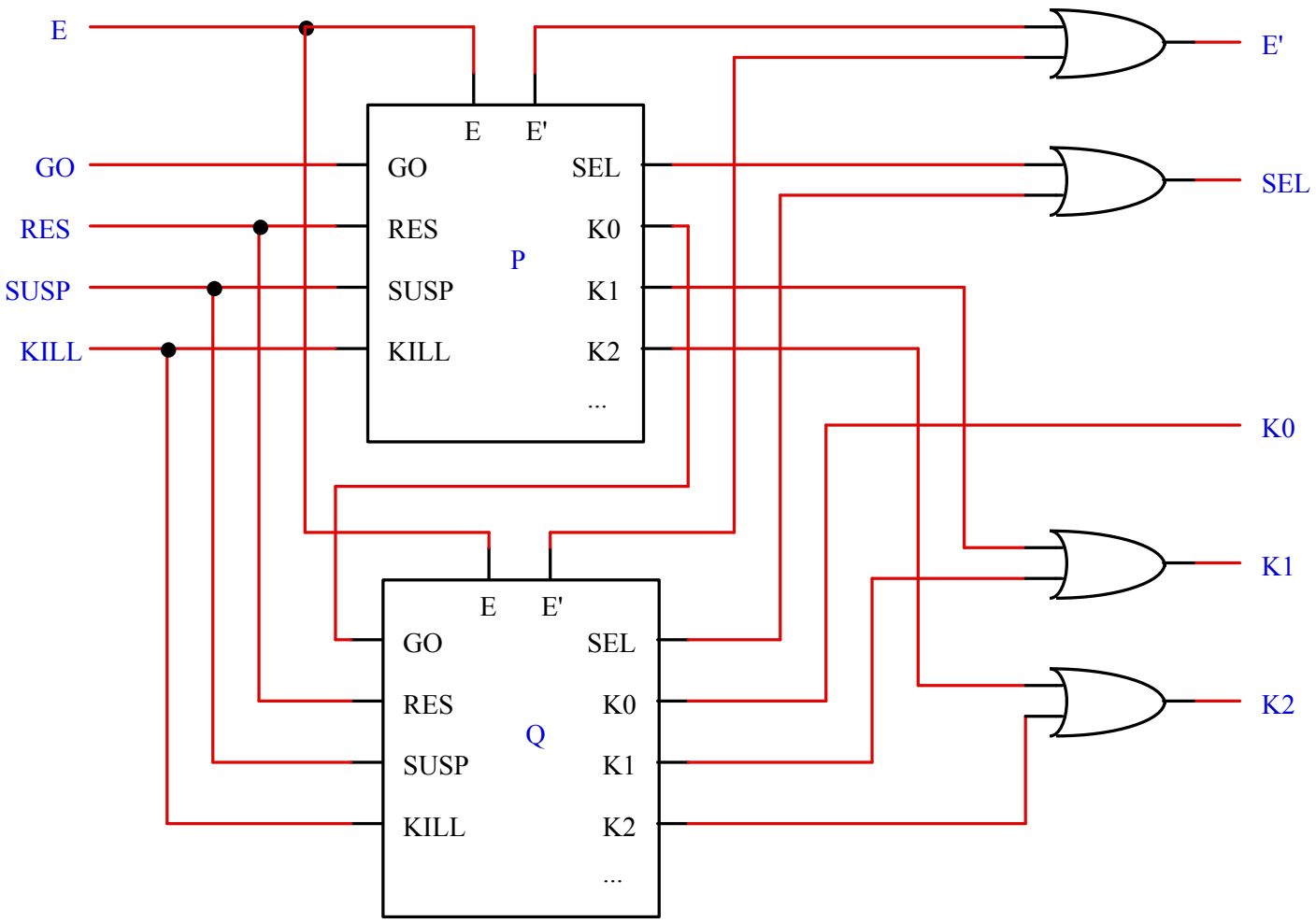
Exclusive relation:
GO # RES # SUSP

- E and E': signals received and emitted
- GO: start p (first cycle)
- RES: continue from the previous state
- SUSP: freeze for a cycle (keep registers)
- KILL : reset registers
- SEL: at least one register set = statement *alive*
- Ki : 1-hot encoded *completion code*
  - K0: normal terminate
  - K1: pause for a cycle
  - K2,K3,… - exit enclosing traps

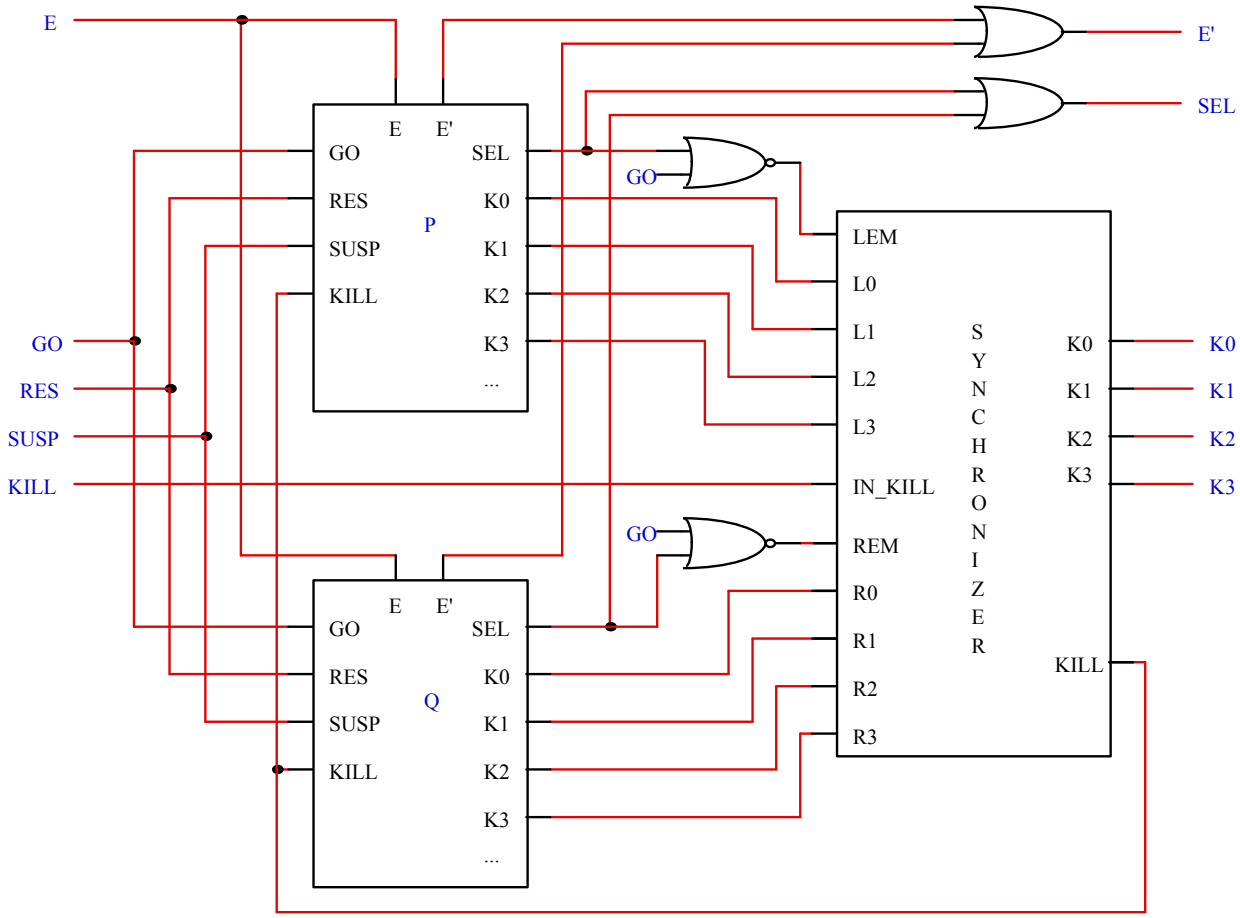
Basic syntax directed translation scheme



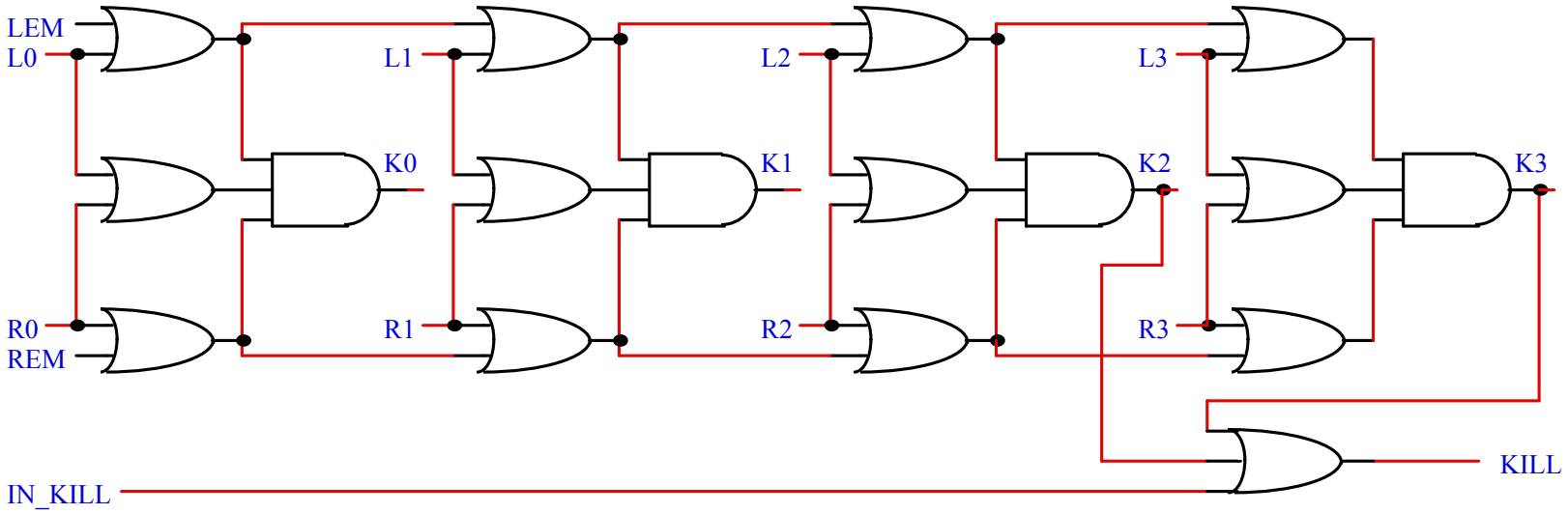
- E and E': signals received and emitted
- GO: start p (first cycle)
- RES: continue from the previous state
- SUSP: freeze for a cycle (keep registers)
- KILL : reset registers
- SEL: at least one register set = statement *alive*
- Ki : 1-hot encoded *completion code*
  - K0: terminate
  - K1: pause for a cycle
  - K2,K3,… - exit enclosing traps



Circuit for sequencing P; Q

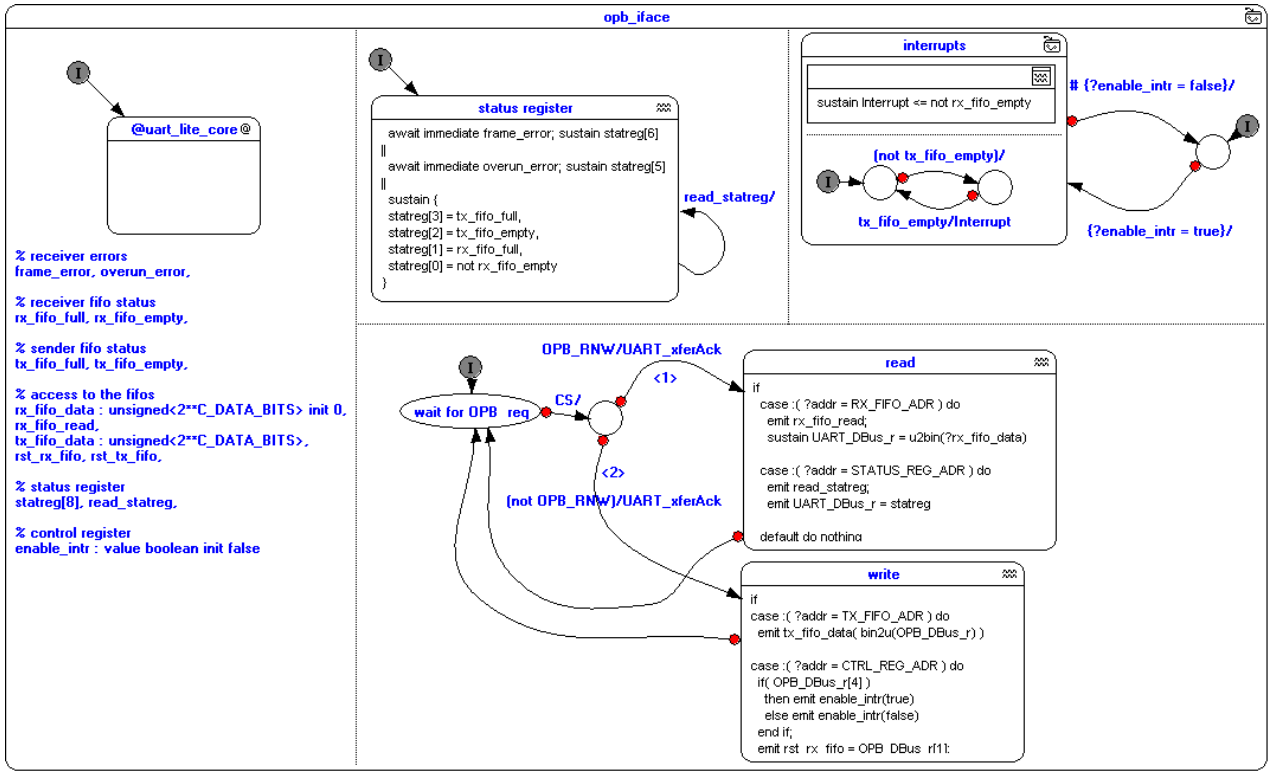


Circuit for P||Q



The parallel synchronizer (a 2-adic arithmetic operator)

Optimization: exploit hierarchy



1. Start with structural encoding implicit in design
2. carefully remove register redundancies (only if cheap)

⇒ good balance between logic and registers
(with Touati, Toma, Sentovich)

Conclusion

- Synchrony is much simpler than asynchrony
  - manageable large-scale concurrency + sequencing
  - equally good for software and hardware
- Synchronous formal methods are used in industry
  - formal languages
  - formal compilation schemes
  - formal verification
- Computing on formal programs is the future
  - automatic / semi-manual property verification
  - compiler correctness proof

Conclusion

- Synchrony is much simpler than asynchrony
  - manageable large-scale concurrency + sequencing
  - equally good for software and hardware
- Synchronous formal methods are used in industry
  - formal languages
  - formal compilation schemes
  - formal verification
- Computing on formal programs is the future
  - automatic / semi-manual property verification
  - compiler correctness proof

Get Esterel Studio and SCADE
they are free
for teaching and academic usage

Research Directions

Research Directions

- Scale verification techniques
  - improve SAT / numerical / Abstract interpretation engines
  - develop assume / guarantee verification
  - prove compilers correct: Schneider (HOL), Leroy (Coq)

Research Directions

- Scale verification techniques
  - improve SAT / numerical / Abstract interpretation engines
  - develop assume / guarantee verification
  - prove compilers correct: Schneider (HOL), Leroy (Coq)

- Extend model to distributed systems (castles instead of rooms)

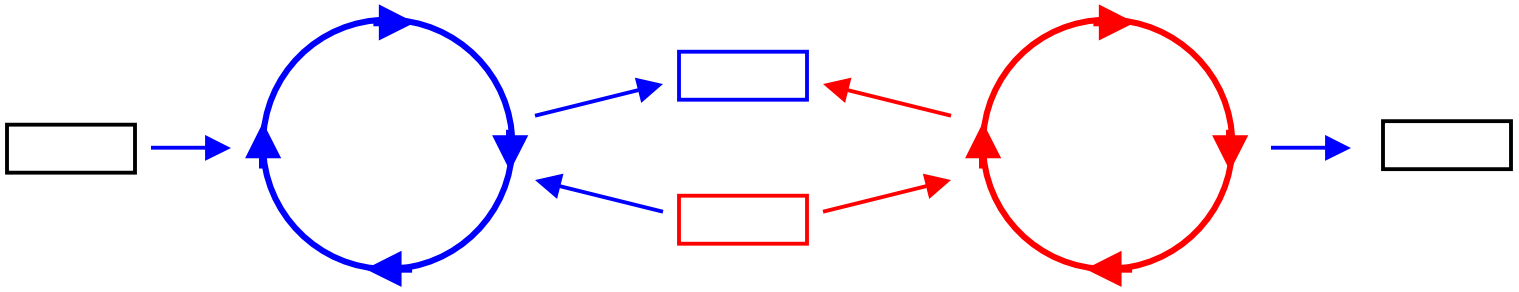
Research Directions

- Scale verification techniques
  - improve SAT / numerical / Abstract interpretation engines
  - develop assume / guarantee verification
  - prove compilers correct: Schneider (HOL), Leroy (Coq)

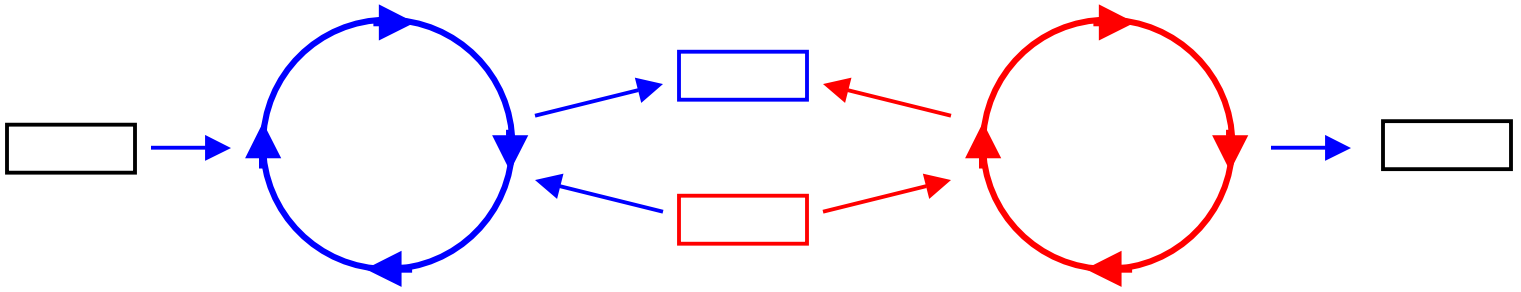
- Extend model to distributed systems (castles instead of rooms)

  =>Add a controllable amount of asynchrony
  - timed-triggered networks (Kopetz, TTP, FlexRay)
  - distributed sampling / Nyquist theorem (Caspi)
  - N-synchronous Kahn Networks (Cohen, Duranton, Pouzet, etc.)
  - elastic circuits (Cortadella & Kishinevsky)
  - ...

Distribution by Mutual Sampling



Distribution by Mutual Sampling



- Works because of control theory stability results, not because of computer science ones (Caspi & al.)
- Similar to multiclock hardware, but much simpler (no metastability issues)