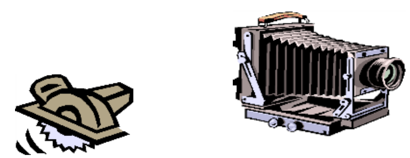
The Synchronous Languages 12 Years Later

Tawatchai Siripanya
Seminar in Programming Language(19666)
Advisor: Lilit Hakobyan
Supervisor: Prof. Dr. Elfriede Fehr
Institute of Computer Science
Freie Universität Berlin
Date: 26.02.13

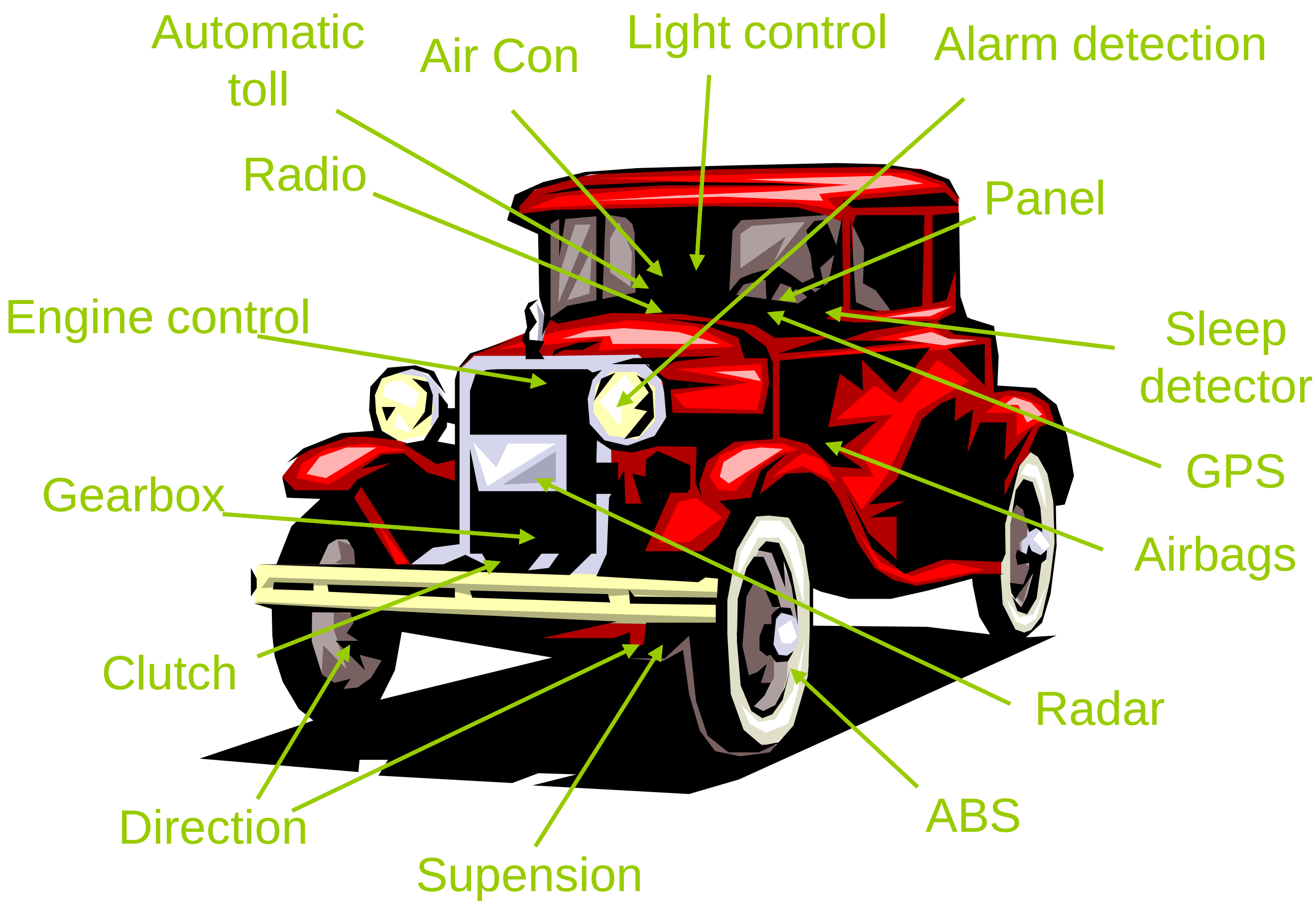
Outline

- Motivation
  - Goals of the talk
- History of the Languages
  - How the languages have been commercialized
- Distinguish of the Synchronous Languages
- Successes and Improvements
- Difficult  Open Issues
- Summary



Freie Universität Berlin

Motivation



Global Coordination

Source: http://www.esterel-technologies.com

Goals Of The Talk

To answer these questions:

1. Review the history of the synchronous programming languages
   - Esterel
   - Lustre
   - Signal
2. what have been achieved in the languages
3. What are difficulties in synchronous programming languages
4. What majors problems remain

Requirements of The Language

1. Concurrency
2. Simplicity
3. Synchrony

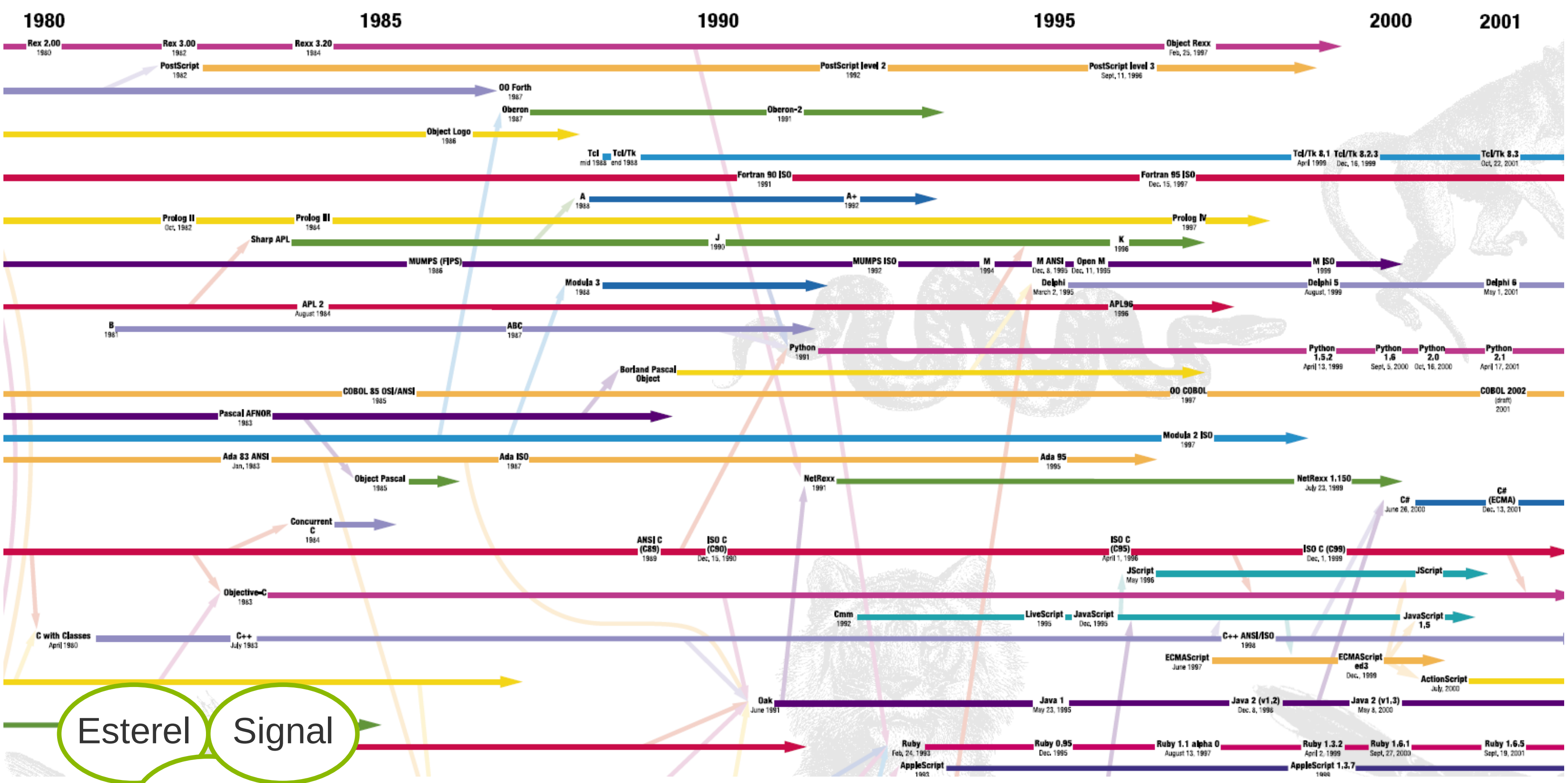
```
Initialize Memory
for each input event do
    Compute Outputs
    Update Memory
end
```

```
Initialize Memory
for each clock tick do
    Read Inputs
    Compute Outputs
    Update Memory
end
```

An implementation model (simple and frequently used)
left (event driven), right (sample driven)

Source: [1]

The history of the synchronous programming languages



Esterel

Signal

Lustre

When were the synchronous languages born?

TIME TO THE MARKETS

Time To The Markets

Esterel

Lustre

Signal

Time To The Markets: Esterel

Esterel

1980s:

A free Esterel compiler from Berry‘s group at INRIA/CMA

1998:

First martketed by Simulog

1999:

Founded Esterel Technologies

2001:

Esterel Technologies brought a tool based on Lustre called SCADE(Safety Critical Application Development Environment)

Time To The Markets: Lustre

Lustre

1980s:

Two big indrustrial Safety-Critical Software projects were born:

- The N4 Series Of Nuclear Power Plants
- The Airbus A320

But, no suitable tools available:

→ Build Own Tools

  → Airbus industries built SAO

  → Schneider Eclectrics built SAGA based on Lustre because cooperation with Lustre research group

Maintainace problems:

- Verilog undertook the problems
- SCADE Tool

Time To The Markets: Lustre

Lustre

2001:

Esterel Technologies brought SCADE

Time To The Markets: Signal

Signal

1990s:

- TNI owned license of Signal
- Cooperation between TNI and Snecma

1993:

- TNI developed Sildex Tool

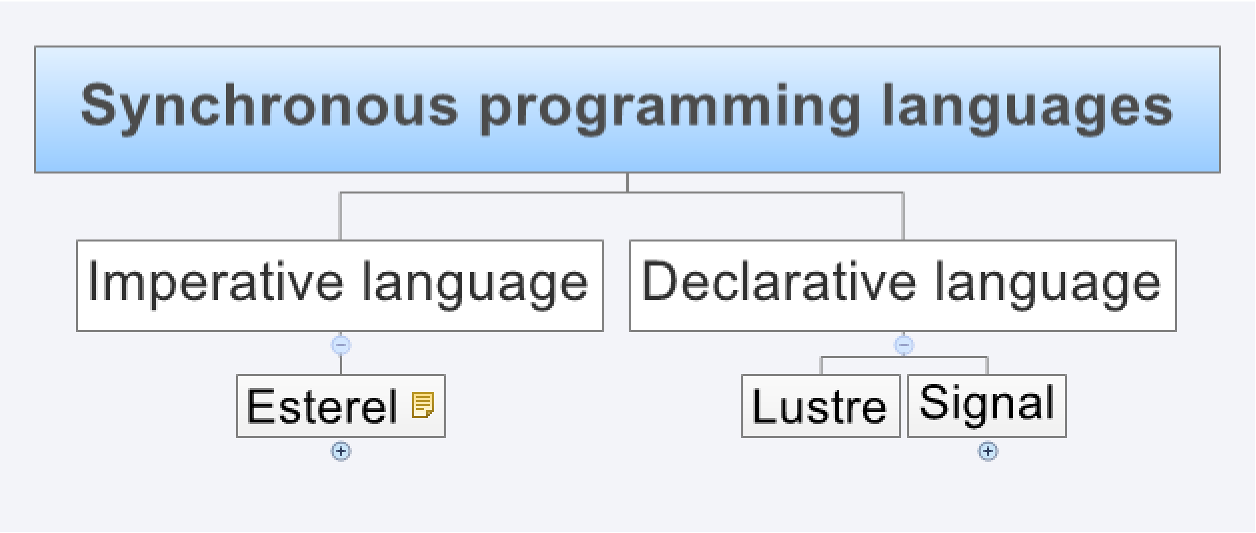
DISTINGUISH THE SYNCHRONOUS LANGUAGES

Distinguish Of The Synchronous Languages

We can distinguish the synchronous languages based on two assumptions:

- Programming Paradigm
- The Philosophy of the languages

Programming Paradigms



The philosophies of synchronous language

- Microsteps
  - Very High Speed Integrated Circuit Hardware (VHDL)
  - Verilog modeling languages
  - Harel's Statecharts,
  - Control System (to program programmable logic controllers)
- Acyclic
  - Lustre
- Unique fixpoint
  - Esterel
- Relation or Constraint
  - Signal

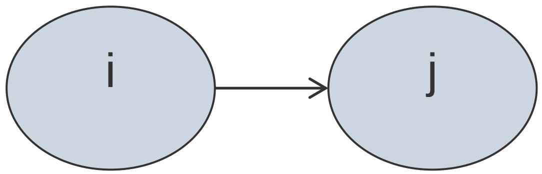
Microsteps

We define a reaction as microsteps to confirm it is operational.

Typically applied in:

- Very High Speed Integrated Circuit Hardware (VHDL)
- Verilog modeling languages
- Harel's Statecharts,
- Control System (to program programmable logic controllers)

Acyclic



$\rightarrow i < j$

No directed cycles

We can insist that a system behaves functionally, when the block diagrams of control systems contain no zero-delay loops

- Lustre

Unique fixpoint

Each reaction of a system is assumed to be the solution of a fixpoint equation.

- Esterel

Relation or Constraint

Each reaction of a system is assumed to be a constraint

- Signal

SUCCESSES AND IMPROVEMENTS

Successes and Improvements

Lustre

Esterel

Signal

Successes and Improvements :Lustre

- Airbus A320
  - Airbus Industries
- N4 series of nuclear power plants
  - Schneider Electric

Successes and Improvements: Esterel

- Dassault Aviation
  - Landing gear system and a fuel management system
- Simulog
- Texas Instrument

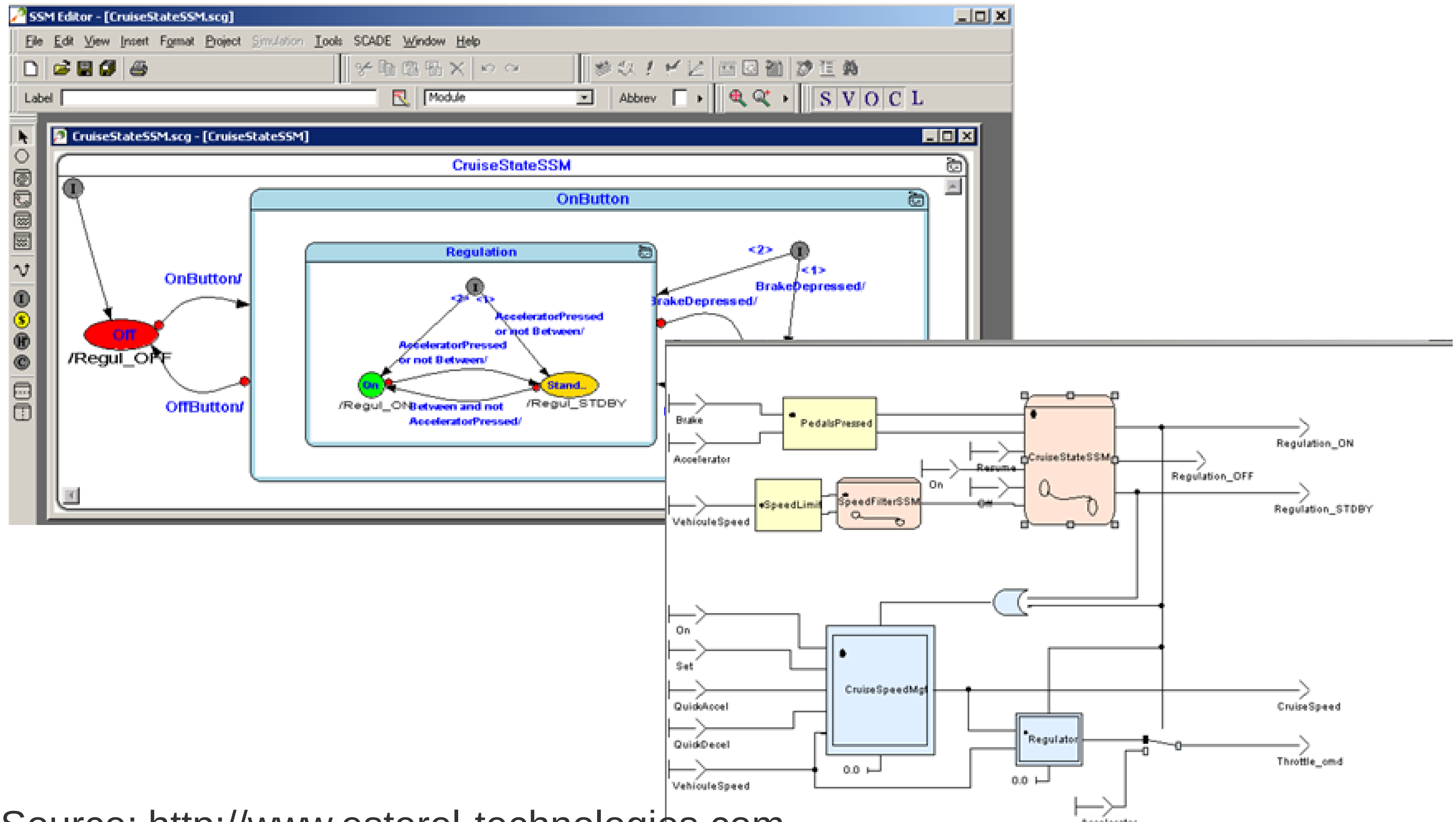
Successes and Improvements: Signal

- CNET
- INRIA
- TNI
- Snecma

Observers for Verification and Testing

- **With Observers, we can:**
  - Describe non-deterministic behaviors of programs using synchronous observers
    - We use the approach called “an automata-theoretic approach to automatic program verification” (see [11])
      - First, describe the unwanted traces of the program
      - Then, make sure that these traces are unaccepted by the automaton
  - Specify safety properties
  - Observe variables or signals of interest
- **Advantages of using observers:**
  - We can specify the safety properties within the program itself
  - Observers can be executed (good for testing)
  - It can be run during execution →We can perform auto test

Visual Notations: Examples



Source: http://www.esterel-technologies.com

Visual Notations: Benefits

- Easy to use
- Reusable
- Can be generated into codes
  - E.g., UML(Universal Modeling Language), Simulink/Stateow, and SyncCharts
- Potential features of the languages

DIFFICULTIES AND OPEN ISSUES

Difficulties and Open Issues

- Compilation
  - Esterel
- Handling of arrays

Compilation (Esterel) (1)

- V1,V2, and V3
  - Based on literal interpretation
  - Based on automata using Brzozowski's algorithm
  - Worked good for small a program
  - However, cannot compile concurrent programs that have longer than 1000 lines
- V4
  - Based on automata by translating Esterel into digital logic (can minimize the size of the executable programs)
  - However, incompatible with the prior versions e.g., V3
- V5
  - Slower than automata-based 100 times
- SAXO-RT (Weil et al. [12]) : "compiled-code discrete-event simulators"
  - The program is divided into segments. Each one becomes a separate C function and can be invoked by a centralized scheduler

Compilation (Esterel) (2)

- Certification constraints of the safety-critical software (DO178B)
  - E.g., traceability (mandatory)
- Tradeoff between **traceability** and **efficiency**
  - In [1] suggests choose one of them
  - Therefore Scade/Lustre compiler → traceability

Handling Of Arrays

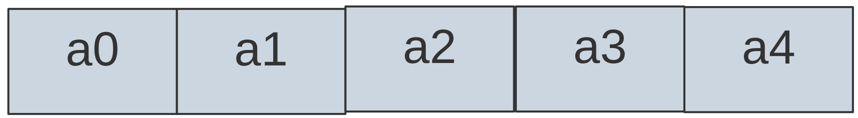
- Powerful to structure programs and to define parameterized regular networks
  - E.g., Apply one operator for the whole element of an array
- First introduced in Lustre to describe circuit (mandatory to manipulate bits)
- An Example Problem (Lustre V4) (See the next slide)
- We have a loop

```
c = 0;
for (i = 0; i <= n; i++) {
    s[i] = A1(a[i], b[i], c);
    c = A2(a[i], b[i], c);
}
```

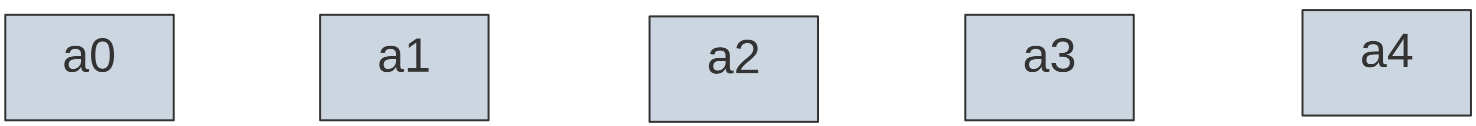
Source: [1]

An Example Problem (Lustre V4)

Array a



Compilation with expansion:



Array elements are expanded into independent variables

SUMMARY

Summary

- The synchronous languages used in the industries as a technology to model, specify, validate, and to implement real-time embedded applications
- We can answer the questions described in the first section
- We have reviewed the history of the synchronous programming languages
  - Esterel ,Lustre , Signal
- We have found that languages have been used in many projects:
  - Observers for Verification and Testing
  - Visual Notations
- We have found compilation of Esterel was difficult and still need to improove
- We have found the problem of handling with arrays is still an open issue
- We think the potential features of the languages might be their visual notations

References

[1] Benveniste, A., Caspi, P., Edwards, S. A., Halbwachs, N., Guernic, P. L., Simone, R. D., 2003. The synchronous languages twelve years later. In: Proceedings of the IEEE. pp. 64–83.

[2] Borgne, M., Marchand, H., Rutten, r., Samaan, M., 1996. Formal verification of signal programs: Application to a power transformer station controller. In: Wirsing, M., Nivat, M. (Eds.), Algebraic Methodology and Software Technology. Vol. 1101 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 271–285.

[3] Bouali, A., 1997. Xeve : an esterel verification environment : (version v1_3). Tech. rep.
URL `http://hal.inria.fr/inria-00069957`

[4] de Simone, R., Ressouche, A., 1994. Compositional semantics of esterel and verification by compositional reductions. In: CAV. pp. 441–454.

References

[5] Halbwachs, N., Caspi, P., Raymond, P., Pilaud, D., 1991. The synchronous dataflow programming language lustre. In: Proceedings of the IEEE. pp. 1305–1320.

[6] Halbwachs, N., Lagnier, F., Raymond, P., 1993. Synchronous observers and the verification of reactive systems. In: AMAST. pp. 83–96.

[7] Halbwachs, N., Raymond, P., 1999. Validation of synchronous reactive systems: From formal verification to automatic testing. In: ASIAN. pp. 1–12.

[8] Jeannet, B., 2003. Dynamic partitioning in linear relation analysis: Application to the verification of reactive systems. Formal Methods in System Design 23, 5–37.

[9] Le Guernic, P., Gautier, T., Le Borgne, M., Le Maire, C., 1991. Programming Real-Time Applications with Signal. Proceedings of the IEEE 79 (9), 1321–1336.
URL http://hal.inria.fr/inria-00540460

Freie Universität Berlin

References

[10] Marchand, H., Bournai, P., LeBorgne, M., Guernic, P. L., 2000. Synthesis of discrete-event controllers based on the signal environment. In: IN DISCRETE EVENT DYNAMIC SYSTEM: THEORY AND APPLICATIONS. pp. 325–346.

[11] Vardi, M. Y., Wolper, P., 1986. An automata-theoretic approach to automatic program verification (preliminary report). In: LICS. pp. 332–344.

[12] Weil, D., Bertin, V., Closse, E., Poize, M., Venier, P., Pulou, J., 2000. Efficient compilation of esterel for real-time embedded systems. In: Proceedings of the 2000 international conference on Compilers, architecture, and synthesis for embedded systems. CASES '00. ACM, New York, NY, USA, pp. 2–8.
URL http://doi.acm.org/10.1145/354880.354882

Thank You

Q & A