

Can formal verification be as successful
for Embedded Control Systems
as it has been for Hardware and System Software ?

Rajeev Alur
University of Pennsylvania

CalTech Worksop on Verification & Validation, September 2009

Hardware Verification

Some Ingredients of Success

1. Impressive advances in tools/methods

- Symbolic model checking, SAT, counter-example guided abstraction refinement, automatic abstraction ...

2. Industrial need and interest

- Spurred by Intel's Pentium floating point bug
- Clearly documented success in finding bugs that escaped existing industrial practice (based on extensive simulation)

3. Pragmatic focus

- Scaled down goals with narrow, but well-defined, applicability (e.g. model checking to find bugs in cache coherence protocols)

4. Integration in design cycle

- Specification languages such as PSL used by simulation tools also
- Marketed by tool vendors, verification consulting companies..

System Software Verification

Some Ingredients of Success

1. Impressive advances in tools/methods

- Program analysis, abstract interpretation, predicate abstraction, SMT solvers, integrated toolkits...

2. Industrial need and interest

- Microsoft's need to reduce Windows crashes and security vulnerabilities
- How to enforce well-documented programming rules?

3. Pragmatic focus

- Specific error conditions: locking errors in device drivers, buffer overflows and null pointer dereferencing, Directed testing (DART)

4. Integration in design cycle

- Code must be certified by verifier before it can be checked in
- Integration of verification into type systems for new languages

Embedded Control Systems Verification

Do we have the right ingredients?

1. Advances in tools/methods ?

- Symbolic representations such as polyhedra for over-approximation of reachable states
- Integration of CS and control theory ideas (e.g. approximate simulation relations)
- Adoption of general techniques such as abstraction, compositionality, counter-example guided refinement ...
- Tools are not particularly scalable and robust (but neither are they in hardware/software verification!)
- **Need: Well-defined and accepted benchmarks / challenge problems to spur competition and measure progress**

Embedded Control Systems Verification

Do we have the right ingredients?

2. Industrial need / interest / economic driving factors ?

- With possible exception of Airbus, investment in formal verification by industry is low
- Positive developments: in-house verification groups in GM, Toyota; Some interest by FDA
- Challenge: Identify a problem domain for which applying FV is feasible and demonstrably beneficial in terms of development cost

Embedded Control Systems Verification

Do we have the right ingredients?

3. Pragmatic focus

- Proving safety of high-dimensional nonlinear systems is probably beyond the scope of current tools, so what can we do?
- From verification to improved simulation: Can symbolic techniques based on verification technology improve efficacy of simulation-based validation?

(promising results for symbolic simulation of Stateflow/
Simulink, reported in EMSOFT 2009)

Embedded Control Systems Verification

Do we have the right ingredients?

4. Integration in design cycle ?

- Opportunity: Model-based design is common for embedded software: analyzing models is an easier task than analyzing C code, so formal verification should fit better
- For tech transfer, buy-in from tool developers needed
- New tools such as Reactis
- Mathworks now supports hybrid automata, and has a tool called Simulink Design Verifier
- Challenge: Specifications are central to verification, we need a industry-accepted standard for inserting assertions and requirements in the designs