Pervasive Formal Verification in Control Systems

FMCAD 2011

Darren Cofer

ddcofer@rockwellcollins.com

Rockwell Collins

Lee's questions

1. How can formal verification compliment current simulation and testing procedures? Drive all 3 from same models
2. What will control system design look like in 10 years? 20 years? Software elimination (compile to HW)
3. Can formal verification help build safer "intelligent" control systems? Yes, but SW is both problem & solution
4. Where can the greatest impact be made in improving control system quality and reducing design costs? Better hybrid system verification tools? Yes Better languages? No More compiler assurance? Don't care Easier timing analysis? Boring Automated power analysis? Don't care
5. Could more aggressive control systems (i.e., that save energy, reduce operational wear, reduce the need for redundancy) be pursued if better design assurance could be provided? Yes, see 3
6. What social and educational impediments are there to having control systems engineers use formal verification tools? Model checking integrated with MBD

Rockwell Collins

Why use formal methods with avionics SW? (A lesson in marketing)

Why use formal methods with avionics SW? (A lesson in marketing)

- Increase safety
  - Complete examination of models and requirements
  - "Our systems are already safe."

Why use formal methods with avionics SW? (A lesson in marketing)

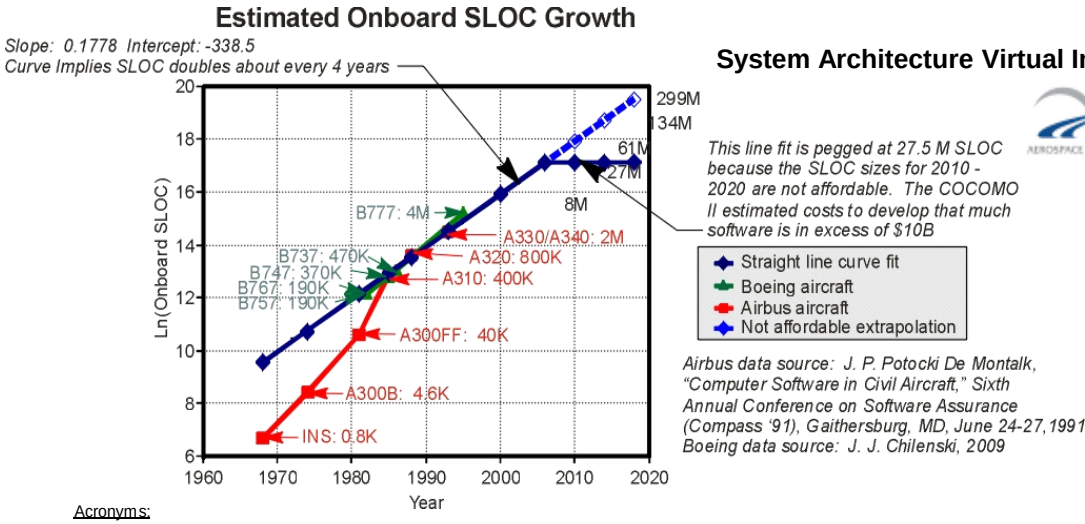
- Increase safety
  - Complete examination of models and requirements
  - "Our systems are already safe."
- Satisfy certification objectives
  - DO-178C allows certification credit for formal verification
  - Requirements/model verification is done by review (cheap), and formal source/object code verification is difficult (too expensive)

Why use formal methods with avionics SW? (A lesson in marketing)

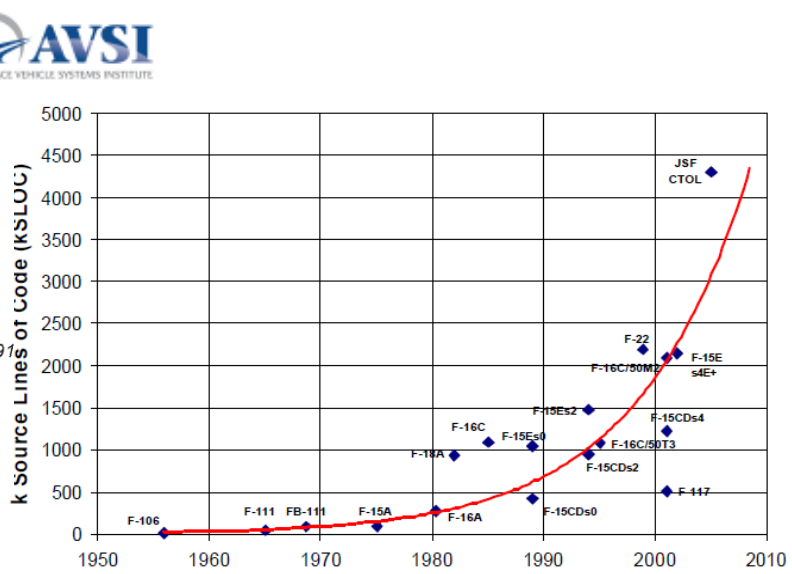
- Increase safety
  - Complete examination of models and requirements
  - "Our systems are already safe."
- Satisfy certification objectives
  - DO-178C allows certification credit for formal verification
  - Requirements/model verification is done by review (cheap), and formal source/object code verification is difficult (too expensive)
- Reduce cost
  - YES!
  - Early detection/elimination of defects
  - Focus on model checking and debugging, rather than theorem proving

Software as both solution and problem

- New functionality for advanced mission capabilities, decision aides, precision navigation, safety of flight greatly increases software load
- Presents an enormous verification challenge



Acronyms:
SLOC: source lines of code
COCOMO II: COnstructive COst MOdel *II*

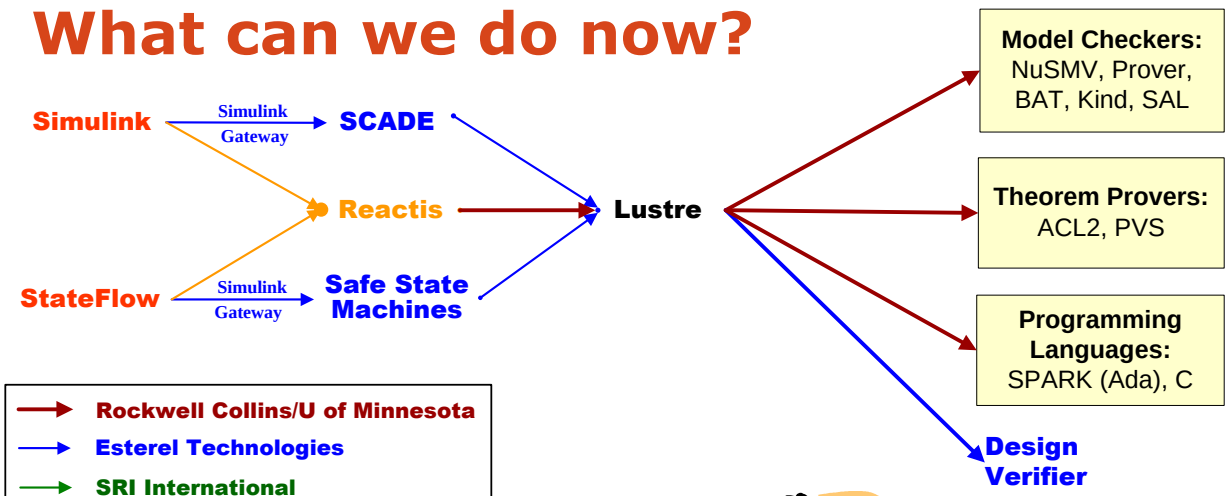


D. Gary Van Oss, "Avionics Acquisition, Production, and Sustainment: Lessons Learned – The Hard Way," NDIA Systems Engineering Conference, Oct 2002

Model Based Development

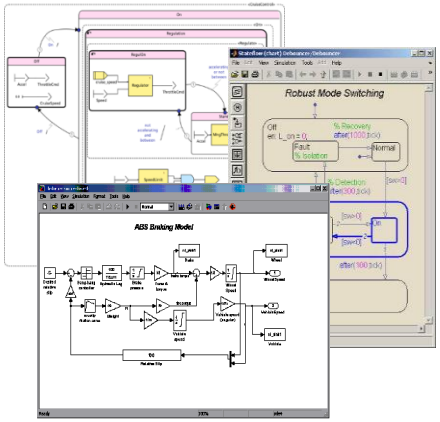
- "If formal methods are so great, why aren't they more widely used?"
- The main barriers in the past have been:
  1. **Cost**: building/maintaining separate analysis models
  2. **Fidelity**: models don't match real system
  3. **Usability**: unfamiliar notations/tools
  4. **Scale**: inadequacy of tools for industrial-sized problems
- ***MBD is eliminating the first three barriers***
  - Leverages existing modeling effort
  - Automated translations and analysis
  - Familiar notations for engineers (Simulink + Stateflow)
- ***Fourth barrier is also falling…***
  - Moore's Law = more power available on desktop
  - Exploit rapid advances in model checking (e.g., SMT)

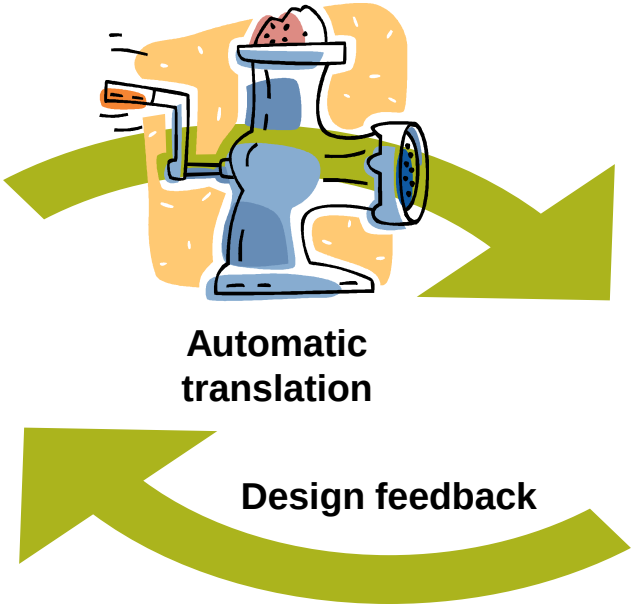
What can we do now?

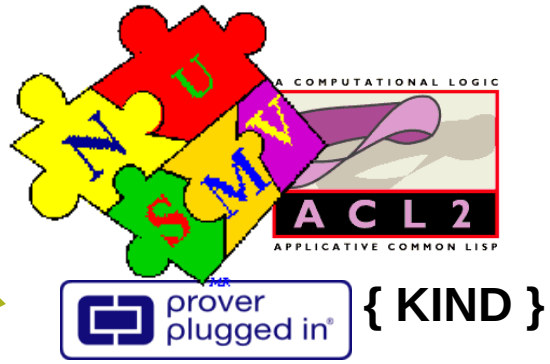


Gryphon translation famework

- **Supports a wide variety of back end tools and languages**
- **Straightforward to add new tools (e.g. Prover support added in 4 days)**
- **Apply "the right tool for the job"**







Model checking integrated with development

Application: Certification Techniques for Advanced Flight Critical Systems (CerTA FCS)

- AFRL program
  - Team: Lockheed Martin + Rockwell Collins
- Problem
  - The cost of software V&V for UAVs has been identified as *the* primary obstacle to their future development
  - These costs are expected to grow rapidly as sophisticated adaptive control systems are introduced (AAR, Sense & Avoid)
- Measure cost and quality improvements using model checking for verification of UAV software
  - Use RC model-checking tools to verify LM Aero advanced flight control models
  - Quantify the cost and quality achieved by formal verification vs. test-based verification

It's a contest!

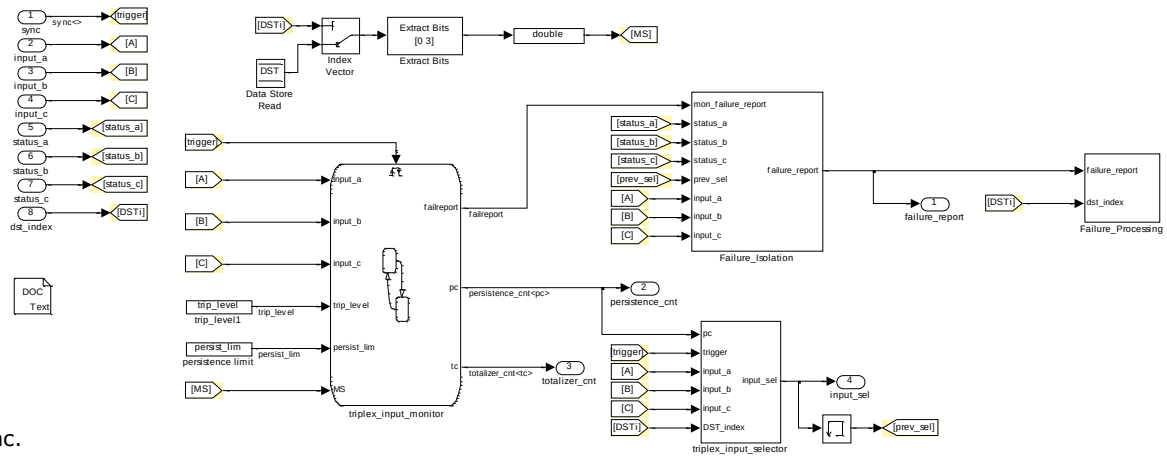
Testing vs. Model Checking

- LM and RC teams start with same set of requirements and software models
- Both teams spent comparable effort to add enhancements to their verification framework (support for new blocks, graphical test case viewer, XML test case generation)
- Measure effort to perform verification and diagnose results
- [FMICS 2007]

Verification effort

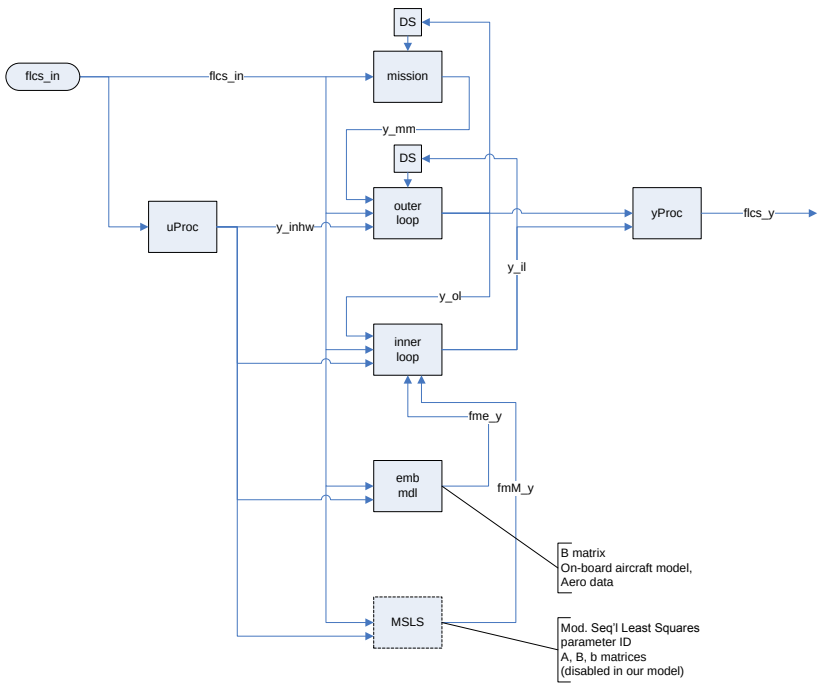
	Hours	Errors found
LM: Test	**196**	**0**
RC: MC	**133**	**12**

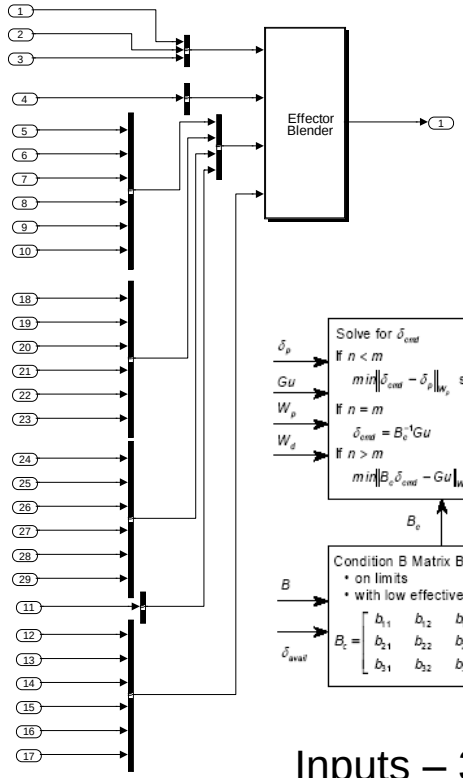
RC effort includes fixing the errors found!

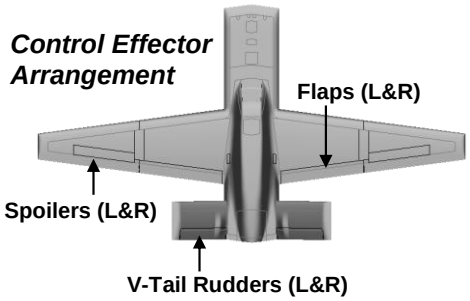


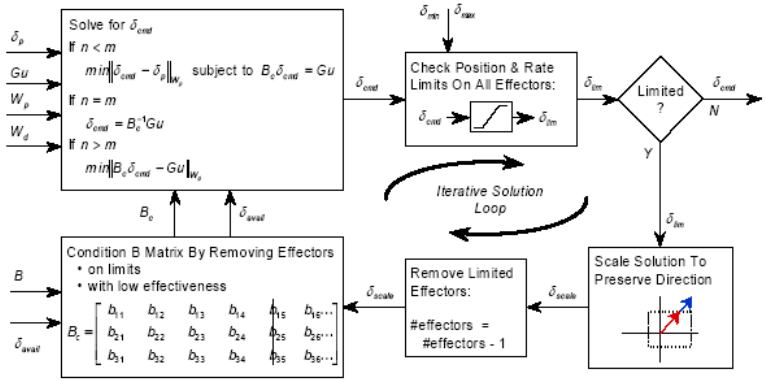
That was nice but…

- Here's what I really want to be able to verify:









Operation Flight Program
Including inner loop control
Including adaptive control algorithms
Including effector blender optimization
Including dynamic inversion

Inputs – 33 floating point inputs
Outputs –6 floating point values
Extensive use of matrix arithmetic
166 Simulink subsystems
2000+ Simulink blocks

Rockwell Collins

Research Challenges

- Floating point types
- Non-linear arithmetic, non-linear functions
- Complex requirements capture and formalization
- Combined methods and tools approaches
- Compositional verification
- Analysis of system architecture models
- And make it bigger

Summary

- Model-based development is key to adoption of formal methods
- Software is both solution and problem
- Need to expand scope of systems/models that can be analyzed
- It’s all about the money