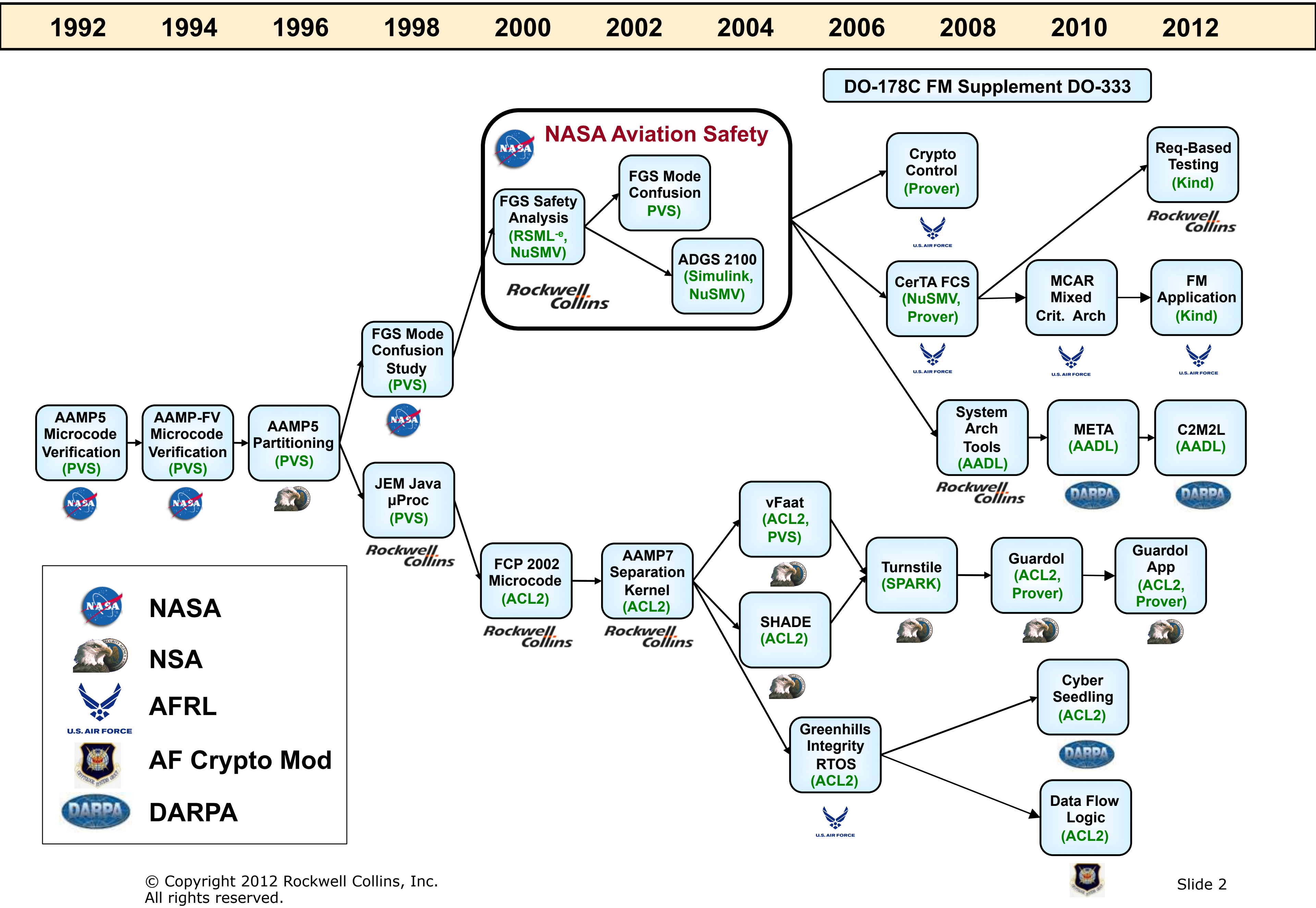
Lessons from Twenty Years of Industrial Formal Methods

Dr. Steven P. Miller
Advanced Technology Center
Rockwell Collins



1992 1994 1996 1998 2000 2002 2004 2006 2008 2010 2012

DO-178C FM Supplement DO-333

NASA Aviation Safety

AAMP5 Microcode Verification (PVS)

AAMP-FV Microcode Verification (PVS)

AAMP5 Partitioning (PVS)

FGS Mode Confusion Study (PVS)

JEM Java µProc (PVS)

FGS Safety Analysis (RSML-e, NuSMV)

FGS Mode Confusion PVS)

ADGS 2100 (Simulink, NuSMV)

Crypto Control (Prover)

CerTA FCS (NuSMV, Prover)

MCAR Mixed Crit. Arch

FM Application (Kind)

Req-Based Testing (Kind)

System Arch Tools (AADL)

META (AADL)

C2M2L (AADL)

FCP 2002 Microcode (ACL2)

AAMP7 Separation Kernel (ACL2)

vFaat (ACL2, PVS)

SHADE (ACL2)

Turnstile (SPARK)

Guardol (ACL2, Prover)

Guardol App (ACL2, Prover)

Greenhills Integrity RTOS (ACL2)

Cyber Seedling (ACL2)

Data Flow Logic (ACL2)

NASA

NSA

AFRL

AF Crypto Mod

DARPA

Presentation Overview

1992 AAMP5 Microcode Verification

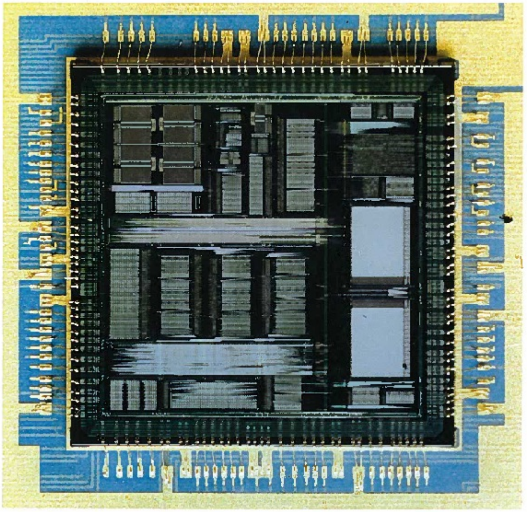
1994 AAMP-FV Microcode Verification

2003 ADGS-2100 Window Manager

2007 CerTA FCS UAV Adaptive Flight Control

2012 Conclusions

AAMP5 Microcode Verification (1993-94)



- Advanced Architecture Microprocessor
  - Family of Rockwell Collins microprocessors
  - Used in a variety of civil and military aircraft
  - Approximately 500,000 transistors
  - Very low power consumption
  - CISC architecture with 209 instructions
- Formal Verification of the Microcode
  - Sponsored by NASA Langley Research Center
  - Performed by Rockwell Collins and SRI International
  - Manually modeled the AAMP5 in PVS
  - Prove the correctness of the microcode
- Results
  - Completed verification of 11 representative instructions

AAMP5 Microcode Verification - Lessons

- Proof Could be Used to Find Design Errors
  - Found one actual error in the process of creating the model
  - Systematically found two seeded errors through proof

- Creating a Separate Verification Model Diminishes Benefit
  - Costly to create and review
  - Results are suspect since verification is not of the “real thing”

- Is Formal Verification Too Expensive?
  - 308 hours per instruction

Presentation Overview

1992 AAMP5 Microcode Verification

1994 AAMP-FV Microcode Verification

2003 ADGS-2100 Window Manager

2007 CerTA FCS UAV Adaptive Flight Control

2012 Conclusions

AAMP-FV Microcode Verification (1994-95)

- Safety Critical Member of the AAMP Family
  - Intended for use in ultra-critical applications
  - Designed but not fabricated
  - Approximately 100,000 transistors
  - Simpler design with 80 instructions
- Formal Verification of the Microcode
  - Sponsored by NASA Langley Research Center
  - Performed by Rockwell Collins and SRI International
  - Repeat of the AAMP5 Verification
- Results
  - Completed verification of 57 instructions
  - Cost per instruction dropped to 38 hours

AAMP-FV Microcode Verification - Lessons

- Your First Attempt is Probably Not a Good Measure of Cost
  - Cost dropped from 308 to 38 hours/instruction
- Pick the Right Problem
  - The AAMP-FV was intended for use in ultra-critical applications
- Amortize Costs Through Reuse
  - Model libraries, tool expertise, design expertise
- Formal Verification Can be Mastered by Real Engineers
  - Real issue is motivation, not ability
  - Perception of difficulty can be a very real barrier

Presentation Overview

1992 AAMP5 Microcode Verification

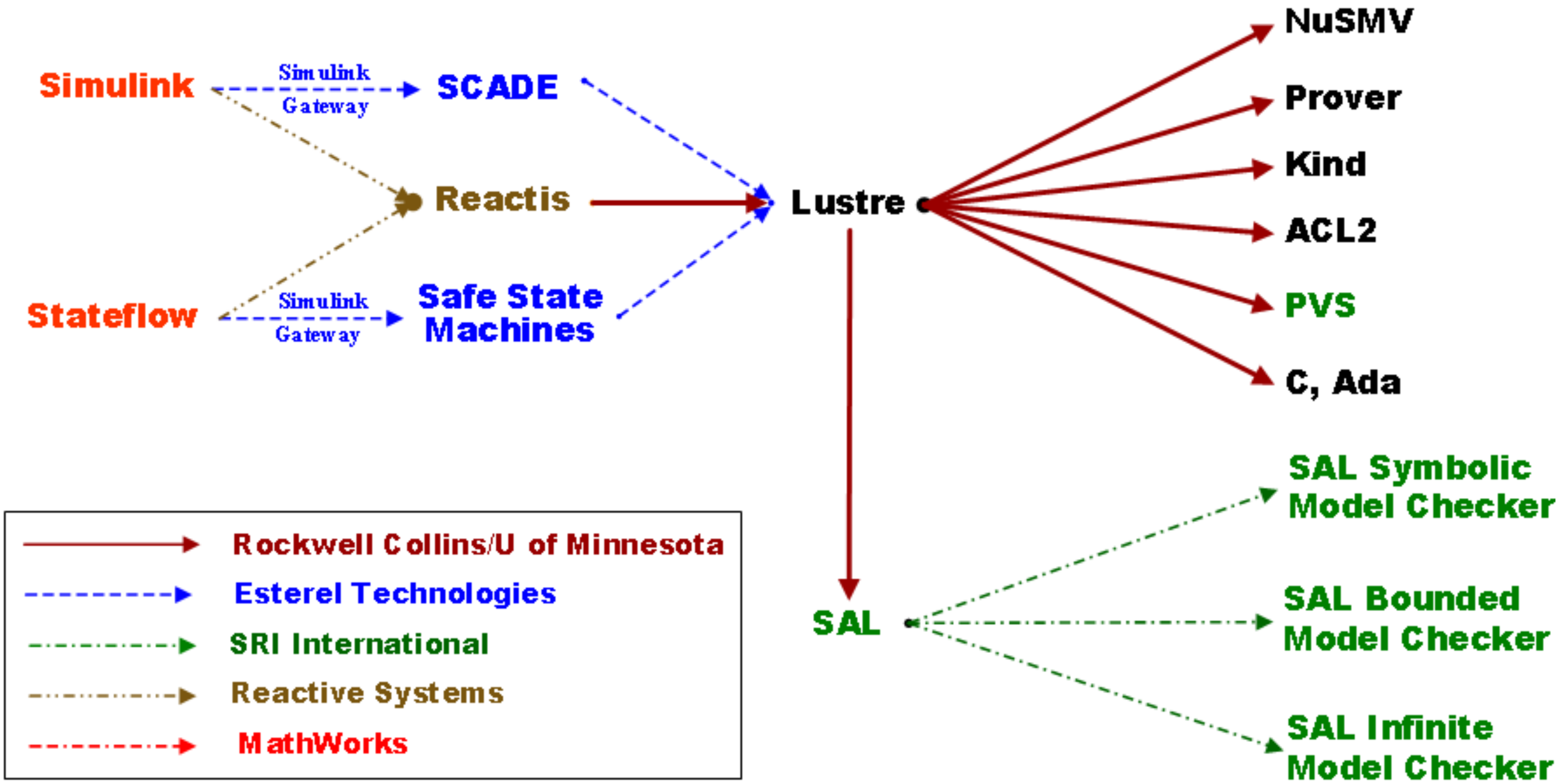
1994 AAMP-FV Microcode Verification

2003 ADGS-2100 Window Manager

2007 CerTA FCS UAV Adaptive Flight Control

2012 Conclusions

Rockwell Collins Formal Verification Framework



Simulink

Simulink Gateway

SCADE

Reactis

Stateflow

Simulink Gateway

Safe State Machines

Lustre

NuSMV

Prover

Kind

ACL2

PVS

C, Ada

SAL

SAL Symbolic Model Checker

SAL Bounded Model Checker

SAL Infinite Model Checker

Rockwell Collins/U of Minnesota

Esterel Technologies

SRI International

Reactive Systems

MathWorks

ADGS-2100 Window Manager

Highly Prone to Design Errors

Modeled in Simulink

Translated to NuSMV

4,295 Subsystems

16,117 Simulink Blocks

Over 10^{37} Reachable States

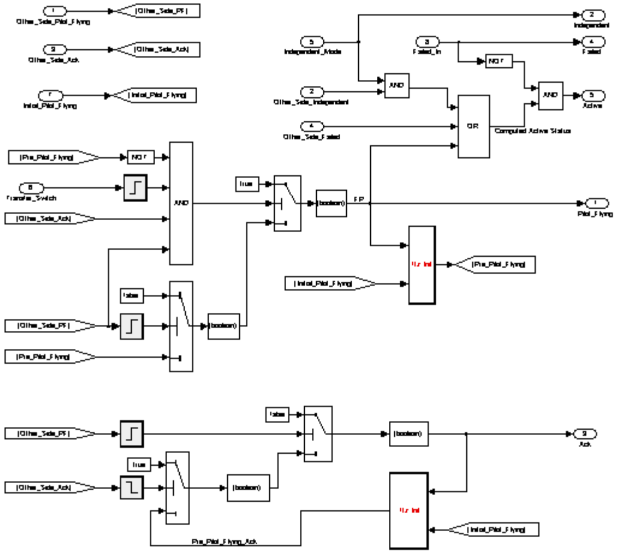
Example Requirement:

Drive the Maximum Number of Display Units
Given the Available Graphics Processors

Counterexample Found in 5 Seconds

Checked 573 Properties -
Found and Corrected 98 Errors in
Early Design Models

No Errors Discovered in the Field



ADGS 2100 Window Manager - Lessons

- There is Always Some Part That Can Be Formally Verified
  - Often the part of greatest concern to the developers
  - May require some modification for analysis
- Don't Let the Lack of a Formal Semantics Prevent Useful Work
  - Assign a formal semantics that matches the implementation
  - Use your tools to find bugs early in development
- Practicing Engineers Will Do Model Checking
  - Hide the formal methods behind the scene
  - Automate the translation to and from their domain
- At Some Point, You Have to Let Go

Presentation Overview

1992 AAMP5 Microcode Verification

1994 AAMP-FV Microcode Verification

2003 ADGS-2100 Window Manager

2007 CerTA FCS UAV Adaptive Flight Control

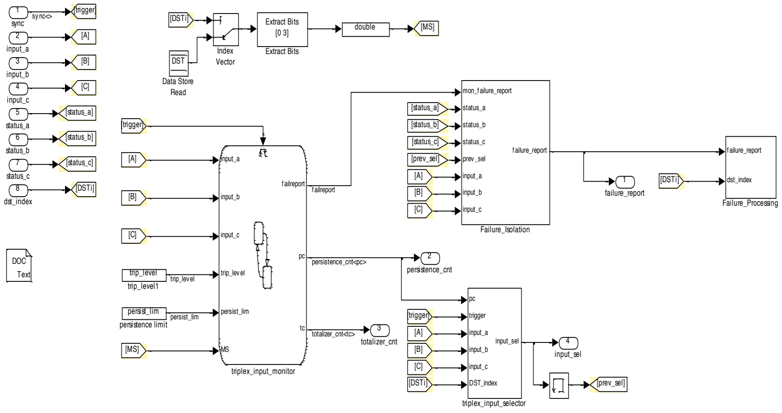
2012 Conclusions

CerTA FCS Phase I

- Sponsored by AFRL
  - Wright Patterson VA Directorate
- Compare FM & Testing
  - Testing team & FM team
- Lockheed Martin UAV
  - Adaptive Flight Control System
  - Redundancy Management Logic
  - Modeled in Simulink
  - Translated to NuSMV model checker

	Subsystem/ Blocks	Charts / Transitions / TT Cells	Reachable State Space	Properties
Triplex voter	10 / 96	3 / 35 / 198	6.0 * 10^{13}	48
Failure processing	7 / 42	0 / 0 / 0	2.1 * 10^{4}	6
Reset manager	6 / 31	2 / 26 / 0	1.32 * 10^{11}	8
Totals	23 / 169	5 / 61 / 198	N/A	62

… for each of ten control surfaces

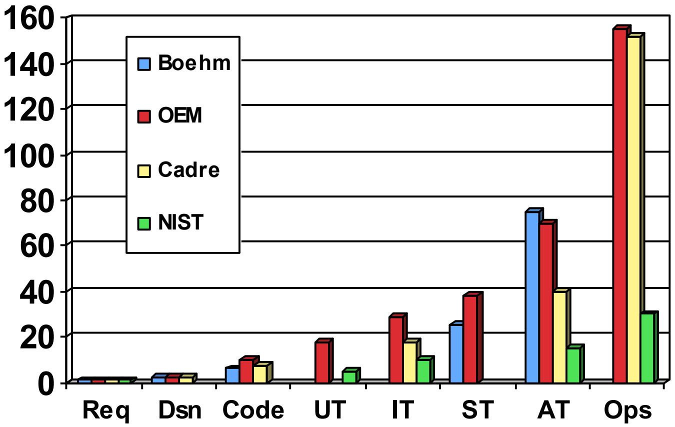


Phase I Results

	Effort (% total)	Errors Found
Testing	60%	0
Model-Checking	40%	12

CerTA FCS Phase I - Lessons

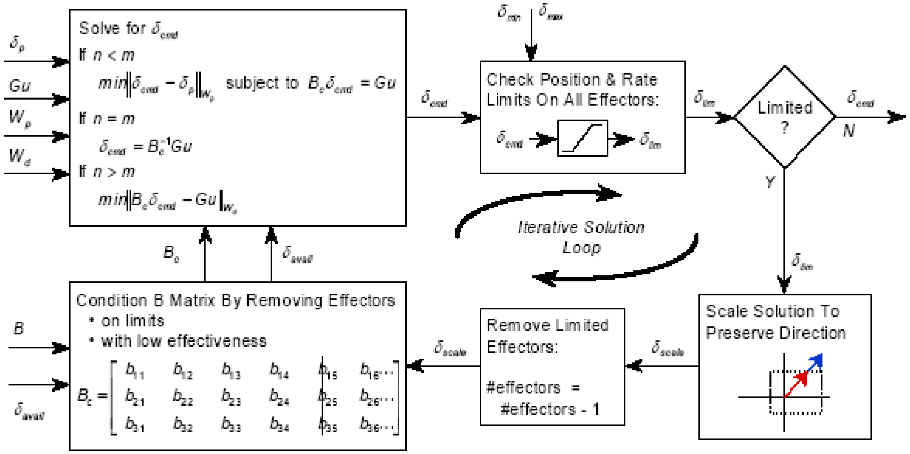
- Model Checking Can be Less Expensive than Testing
  - Better at finding intermittent, race, and rare sequence errors
  - Testing has always been expensive
- Errors Can be Found Early
  - Savings are amplified by avoiding rework late in development
- Complex Semantics are an Opportunity
  - Engineers quickly realize the value of formal verification tools



- Commercial Tools Like Simulink/Stateflow are an Advantage
  - Hard to train thousands of engineers in formal verification
  - Easy to design formal verification tools around a commercial tool

CerTA FCS Phase II

- Sponsored by the AFRL - Wright Patterson VA Directorate
- Can Model-Checking be Used on Large, Non-linear Systems?
  - Lockheed Martin Adaptive UAV Flight Control System
  - Extensive Use of matrix arithmetic
  - Inputs – 33 floating point inputs (including one 3 x 6 matrix)
  - Outputs –6 floating point values
  - 166 Simulink subsystems
  - 2000+ basic Simulink blocks
  - Translated to Prover model checker



Solve for δ_{cmd}

If $n < m$

$\min \|\delta_{cmd} - \delta_p\|_{W_p}$ subject to $B_c \delta_{cmd} = Gu$

If $n = m$

$\delta_{cmd} = B_c^{-1} Gu$

If $n > m$

$\min \|B_c \delta_{cmd} - Gu\|_{W_d}$

Check Position & Rate Limits On All Effectors:

Condition B Matrix By Removing Effectors
- on limits
- with low effectiveness

$$B_c = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16}\cdots \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26}\cdots \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} & b_{36}\cdots \end{bmatrix}$$

Remove Limited Effectors:

#effectors = #effectors - 1

Scale Solution To Preserve Direction

- Challenges
  - Verification of floating point matrix arithmetic
  - Verification of Stateflow *flowcharts* with cycles
  - Compositional Verification
- Final Results
  - Identified five previously unknown errors
  - Identified several implementation errors that were being masked by defensive programming

CerTA FCS Phase II - Lessons

- Numerically Intensive Systems are Still a Challenge
  - Floating point arithmetic
  - Non-linear arithmetic
- Need for Compositional Verification
  - More like theorem proving than model checking
  - Use model checking for leaf nodes
  - Theorem proving for composing the results
- Modeling and Analysis of Architectures
  - Assign assume/guarantee contracts to components
  - Use contracts rather than model for compositional verification

Presentation Overview

1992	AAMP5 Microcode Verification
1994	AAMP-FV Microcode Verification
2003	ADGS-2100 Window Manager
2007	CerTA FCS UAV Adaptive Flight Control
2012	Conclusions

Lessons

- Most Systems have Large Parts that can be Formally Verified
  - Often the parts causing the most problems
  - Tools are ready today if we choose wisely

- Finding Errors Early is One of the Most Important Benefits
  - Industry understands they need to find errors sooner
  - Formal methods provides a systematic way of doing this

- Formal Methods Will Find Errors that Traditional Methods Miss
  - Especially good at finding the errors hiding in the corners
  - There is still a role for testing an reviews

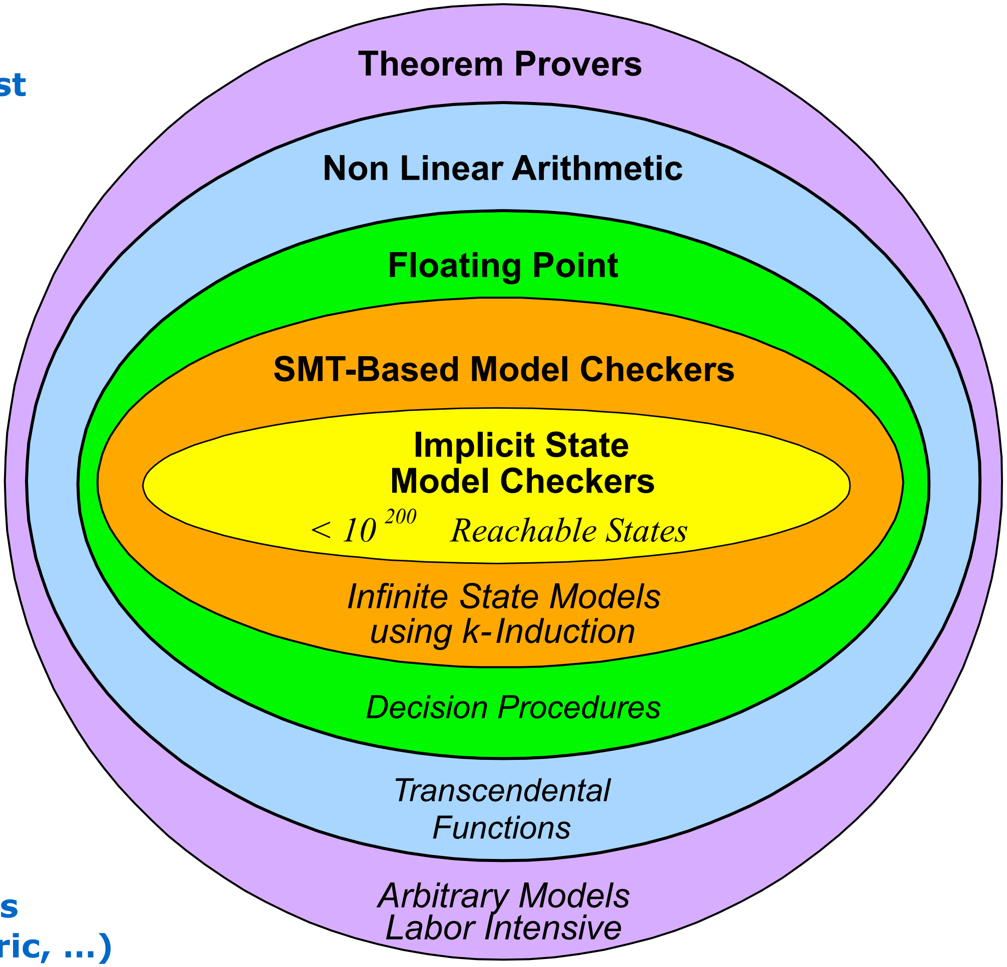
- Take Advantage of the Ease of Repeating Formal Verification
  - Greatest benefit comes when models are changing rapidly

Lessons

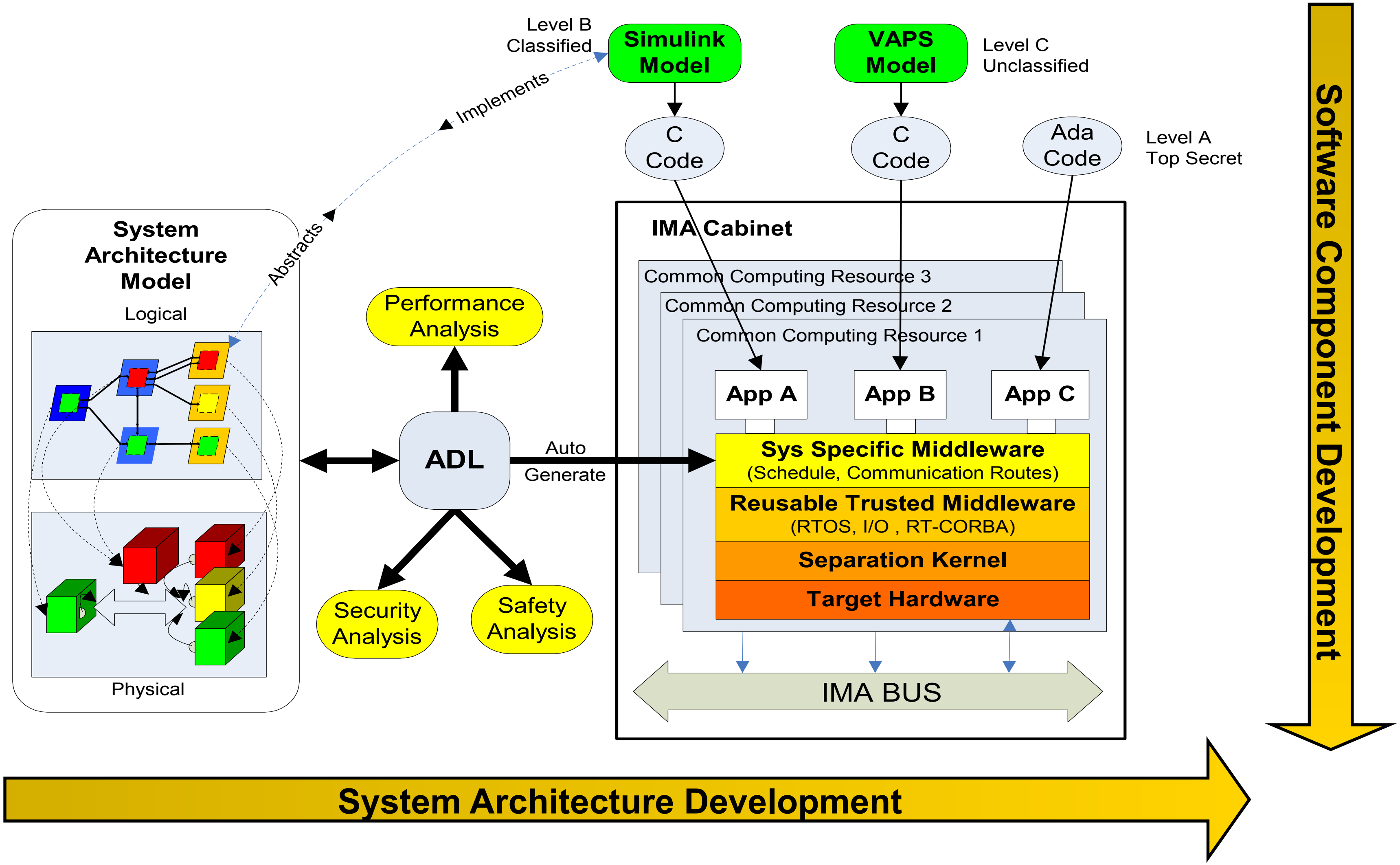
- Take Advantage of the Formalisms Already in Use in Industry
  - Easier to build new tools than to retrain thousands of engineers
  - Don’t let great be the enemy of good
- Verify the Models the Developers Care About
  - Automate the translation to your verification tools
- Expect Costs to Drop Rapidly with Experience
- Pick Your Problems Carefully
  - Use the right tools for the problem
  - Can the verification add real value?
  - Is it important?

Future Directions

- **Theorem Provers**
  - **Deal with arbitrary models**
  - **Concerns are ease of use and labor cost**
- **Large Finite Systems (<10^{200} States)**
  - **Implicit state (BDD) model checkers**
  - **Easy to use and very effective**
- **Infinite State Systems**
  - **SMT-Based model checkers**
  - **Large integers and reals**
  - **Limited to linear arithmetic**
  - **Ease of use is a concern**
- **Floating Point Arithmetic**
  - **Most modeling languages use IEEE 754 floating point numbers**
  - **Decision procedures**
- **Non-Linear Arithmetic**
  - **Multiplication/division of real variables**
  - **Transcendental functions (trigonometric, …)**


Theorem Provers
Non Linear Arithmetic
Floating Point
SMT-Based Model Checkers
Implicit State Model Checkers
< 10^200 Reachable States
Infinite State Models using k-Induction
Decision Procedures
Transcendental Functions
Arbitrary Models Labor Intensive

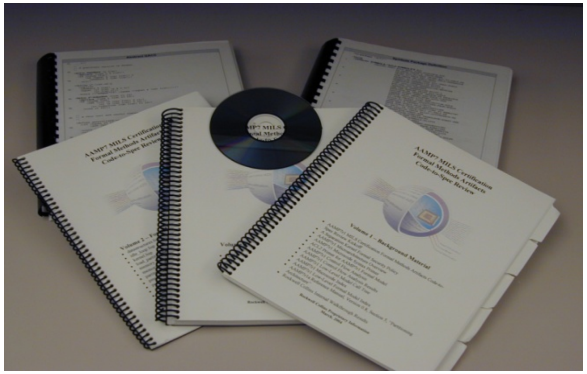
System Architectural Modeling & Analysis



Backup Slides

AAMP7G Microprocessor Intrinsic Partitioning

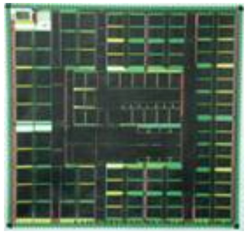
- Formal proof of the MILS security partitioning implemented in the AAMP7G microprocessor
- Example of the industrial use of theorem proving using ACL2
- Developed formal description of separation for uniprocessor, multipartition system (GWV)
- Modeled trusted AAMP7G microcode in ACL2
- Constructed machine-checked proof of separation of the AAMP7G model
- Model subject of intensive code-to-spec review with AAMP7G microcode
- Satisfied formal methods requirements for NSA AAMP7G certification awarded in May 2005
  - *"verified using Formal Methods techniques as specified by the EAL-7 level of the Common Criteria"*
  - *"capable of simultaneously processing unclassified through Top Secret Codeword Information"*

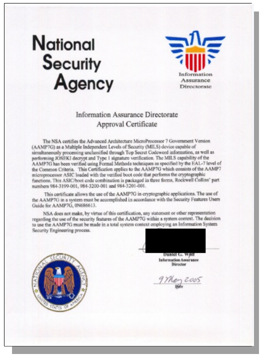


A COMPUTATIONAL LOGIC

ACL2

APPLICATIVE COMMON LISP





National Security Agency

Information Assurance Directorate

Information Assurance Directorate
Approval Certificate

The NSA certifies the Advanced Architecture MicroProcessor 7 Government Version (AAMP7G) as a Multiple Independent Levels of Security (MILS) device capable of simultaneously processing unclassified through Top Secret Codeword information, as well as performing JOSEKI decrypt and Type 1 signature verification. The MILS capability of the AAMP7G has been verified using Formal Methods techniques as specified by the EAL-7 level of the Common Criteria. This Certification applies to the AAMP7G which consists of the AAMP7 microprocessor ASIC loaded with the verified boot code that performs the cryptographic functions. This ASIC/boot code combination is packaged in three forms, Rockwell Collins' part numbers 984-3199-001, 984-3200-001 and 984-3201-001.

This certificate allows the use of the AAMP7G in cryptographic applications. The use of the AAMP7G in a system must be accomplished in accordance with the Security Features Users Guide for AAMP7G, 0N686613.

NSA does not make, by virtue of this certification, any statement or other representation regarding the use of the security features of the AAMP7G within a system context. The decision to use the AAMP7G must be made in a total system context employing an Information System Security Engineering process.

[redacted]

Information Assurance
Director

9 May 2005
Date

AAMP7G Intrinsic Partitioning - Lessons

- Problems for Which Interactive Theorem Proving is Valuable
  - Problems that are inherently important
  - Not suitable for model checking
  - Design artifacts need to be stable
  - Verification provides lasting value that can be replicated

- Better if the Verification Model is Automatically Translated
  - Links the verification to the “real thing”
  - Usually more reliable than human review
  - Lowers the cost of verification