Secure, High-Assurance Development Environment (SHADE) Program

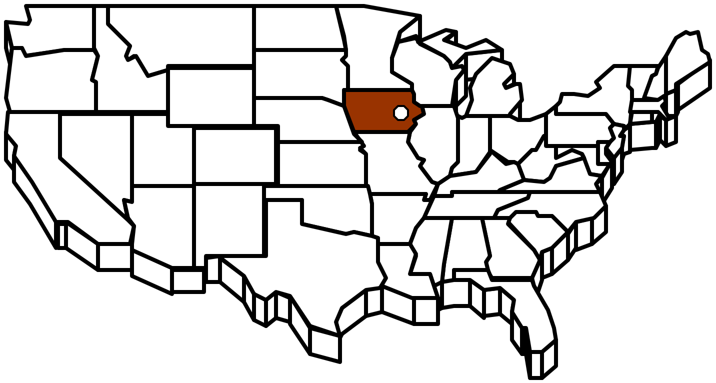
David Hardin
Tom Johnson
Advanced Technology Center
Rockwell Collins, Inc.

Bill Young
University of Texas at Austin

John Matthews
Mark Shields
Galois Connections, Inc.

Rockwell Collins

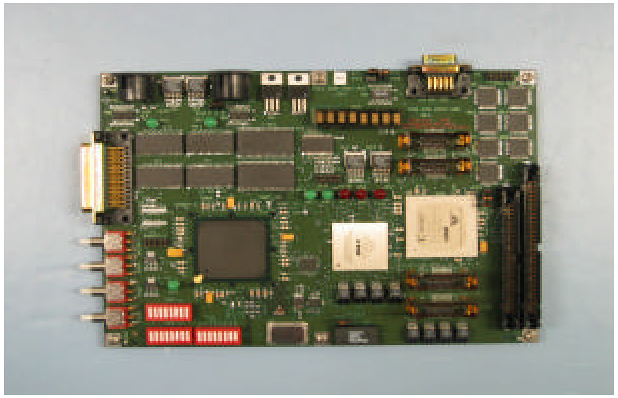
- **Provider of Advanced Communication and Aviation Equipment to Air Transport, Business and Regional, and Military Markets**
  - **$2.8 Billion in Sales**
  - **Headquartered in Cedar Rapids, IA**
  - **14,500 Employees Worldwide**

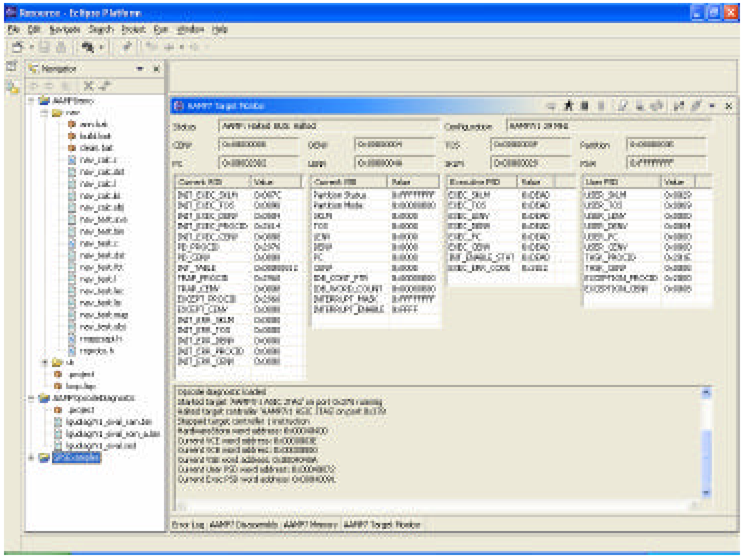


- **The** *Automated Analysis section* **of the RCI Advanced Technology Center applies advanced mathematical tools to the problem of producing high assurance systems**
  - Perform applied research in model-checking and theorem proving for safety-critical and secure systems
  - 6 full-time formal methods researchers
  - Particular expertise in processor modeling, separation kernels, avionics system requirements
  - **We're hiring!**

Secure High-Assurance Development Environment (SHADE)

- A “nuts-and-bolts” partitioned development environment that automates important aspects of secure system development
- A highly-assured, evaluatable method for implementing cryptographic algorithms written in the Cryptol language, including a verifying Cryptol-to-AAMP7 compiler
- Support for automatic machine-code proofs of AAMP7 code
- Tool support for the creation and analysis of secure multipartition cryptographic applications that exploit the AAMP7’s intrinsic partitioning capability
- Funded by NSA R2/I2 and Rockwell Collins

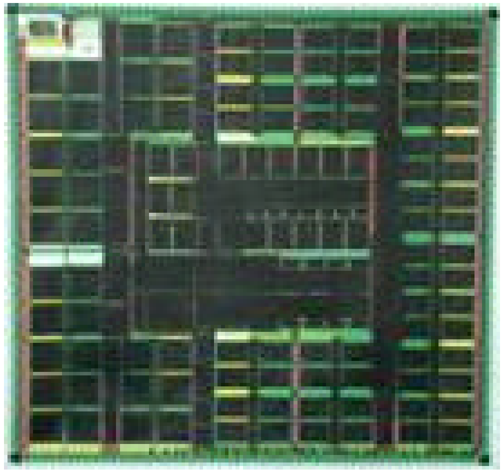




Why a verifying compiler for Cryptol?

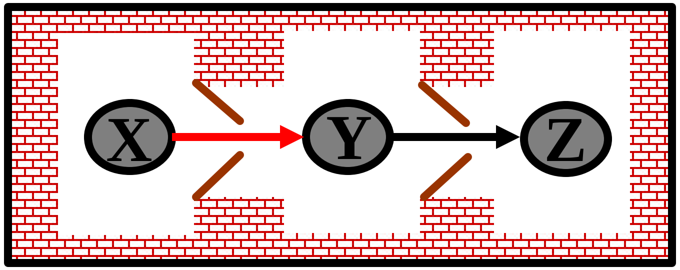
- Cryptographic systems need to be correct
  - NSA is a demanding customer
  - NSA suppliers realize that typical “commercial grade” engineering just won’t cut it
- Cryptographic systems are difficult, expensive to certify
  - A verifying compiler could markedly reduce code-to-spec review costs and reduce time-to-market for cryptographic devices
- Reference Cryptol specifications for common crypto algorithms are available
- A domain-specific language, such as Cryptol, seems to present lower risk than attempting a verifying compiler for a general-purpose programming language
- The AAMP7 is an “easy” code generation target (think JVM)
- Theorem prover technology has matured sufficiently to make this program feasible

Rockwell Collins AAMP7 CPU



Features
- Used in RCI GPS and Infosec products
- High Code Density
- Low Power Consumption (250 mW)
- 100 MHz operation
- Screened for full military temp range
- Implements *intrinsic partitioning*

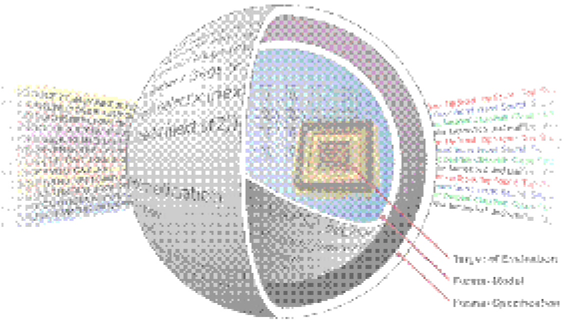
Intrinsic partitioning
- Computing Platform Enforces Data Isolation
- “Separation Kernel in Hardware”



AAMP7r1 Intrinsic Partitioning Formal Verification

- **Formal description of separation for uniprocessor, multipartition system**
  - **“GWV” separation theorem**

- **Detailed formal models of Trusted AAMP7r1 microcode operation, subjected to intensive NSA code-to-spec review against microcode listings.**



- **Machine-checked proof that separation holds of AAMP7r1 model -- “EAL7+”**

- **Artifacts accepted by NSA evaluators in March 2004. Official NSA MILS certification expected soon.**

A COMPUTATIONAL LOGIC

ACL2

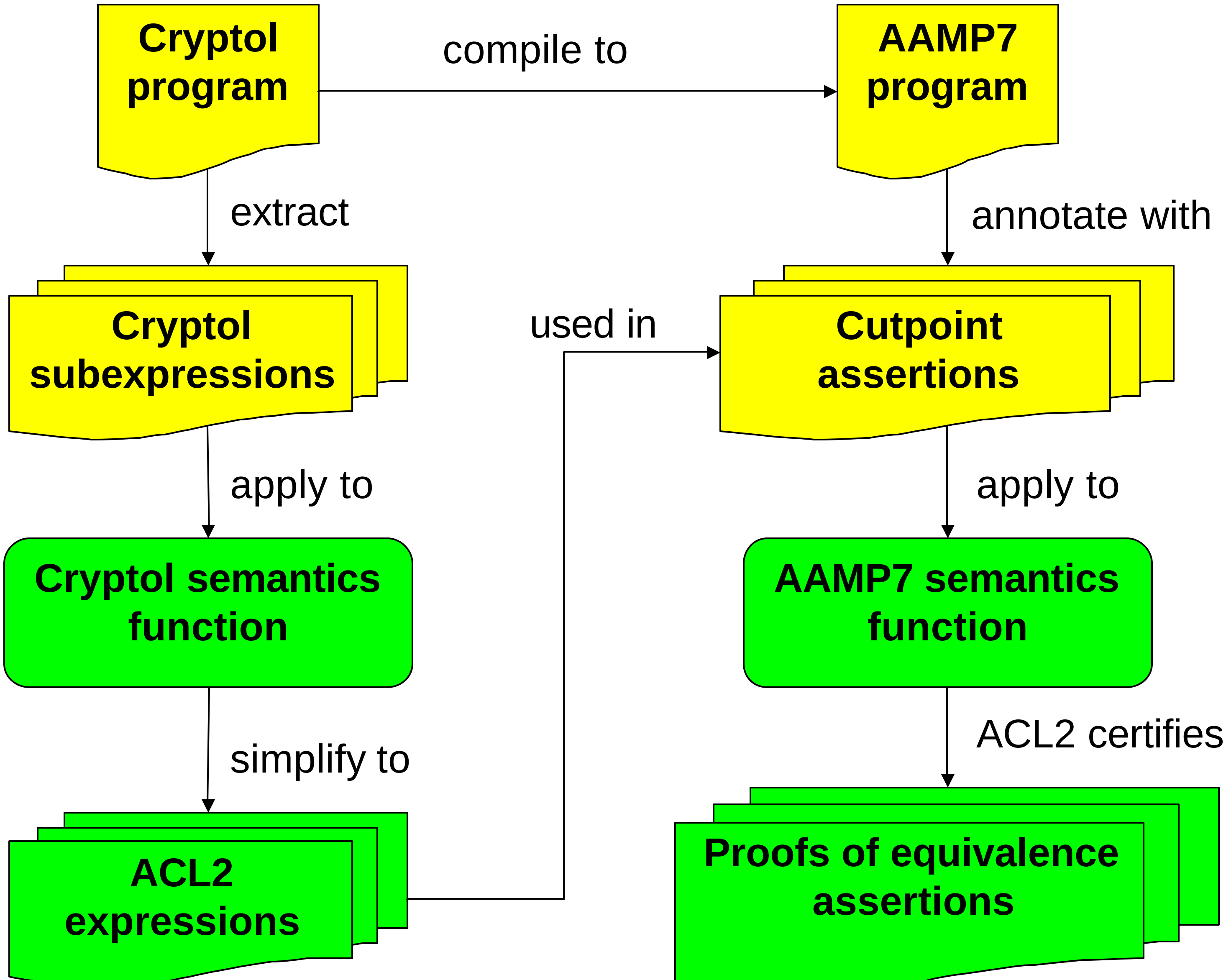
APPLICATIVE COMMON LISP

Cryptol

- Cryptol is a domain-specific language for cryptography, developed by Galois Connections, Inc.
- Cryptol specifications are compact and expressive – DES core is at right
- Cryptol specifications can be compiled to C, or to machine code

```
des : {a b} (a >= 7) => ([2**(a-1)],[b][48]) -> [64];
des (pt, keys) = permute (FP, swap (split last))
  where { pt’ = permute (IP, pt);
          iv = [| round (k, split lr)
               || k <- keys
               || lr <- [pt’] # iv
               |];
          last = iv @ (width keys - 1);
         };

round (k, [l r]) = r # (l ^ f (r, k));

f (r, k) = permute (PP, SBox (k ^ permute (EP, r)));

swap [a b] = b # a;

permute : {a b} (b >= 1) =>
                    ([a][b], [2**(b - 1)]) -> [a];
permute (p, m) = [| m @ (i - 1) || i <- p |];
```

Verifying Compiler Dataflow

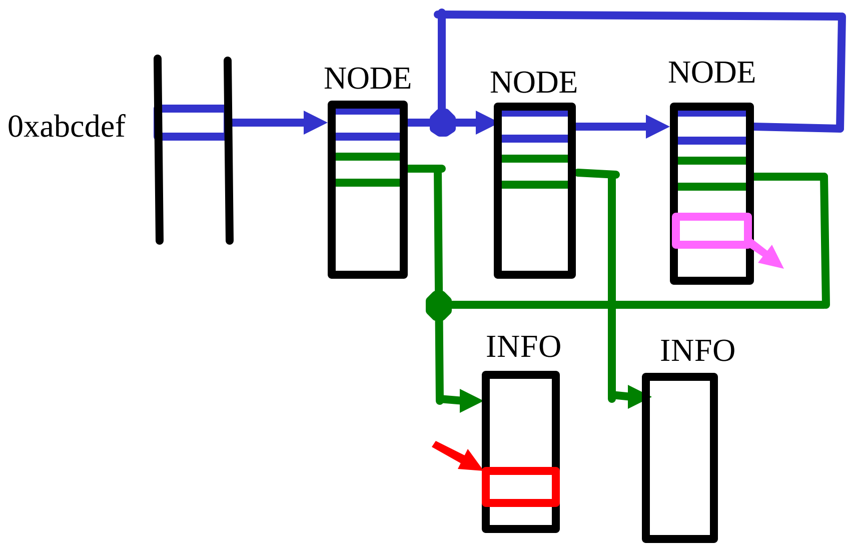


AAMP7 Semantics Function

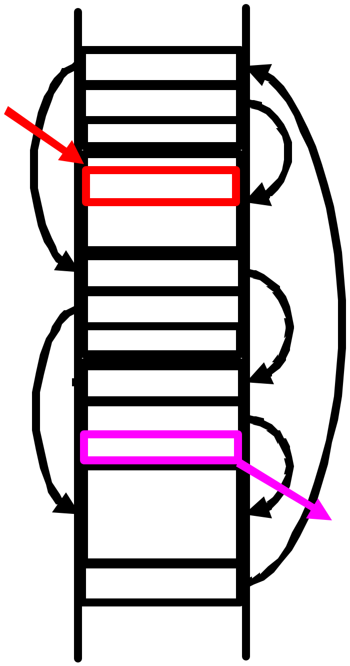
- Provides instruction-level simulator for the AAMP7
- Written in ACL2 (~50 KSLOC with all RCI support books)
- Can be used as a processor simulator, as well as a vehicle for proof
- GACC (Generalized Accessor) library now used to model memory, same as used in AAMP7 separation proofs
  - Underlying bags (multiset) library optimized to support large models

Data Structure Representation

Programmer's view --
"boxes and arrows"



Reality –
mapped into a single linear address space



We must "face reality" in order to verify a compilation

AAMP7 Model State

- **Processor state is modelled using an ACL2 Single-Threaded Object (stobj)**
  - **Stobj mechanism in ACL2 allows functional program objects to be updated in place, rather than updating copies**
- **AAMP7 state is composed of nearly 60 elements, including Program Counter, Top-of-Stack pointer, Partition Management Unit, RAM, etc., many of which are updated every instruction**
  - **Stobj’s are a huge win for the AAMP7 model!**

Status and Summary

- We are a work in progress -- SHADE program is scheduled to run through FY06
- SHADE is a significant engineering effort, encompassing contributions from 10 different developers in three locations
- The SHADE compiler can now generate AAMP7 binary code for canonical examples that execute on the AAMP7 ACL2 model, as well as on the real machine
- Currently investigating whether some of the “middle-end” passes of the compiler can actually be implemented as rewrite rules within the theorem prover