SAND2011-0481C

Image: digitalart / FreeDigitalPhotos.net

# Maintaining Soundness in Hybrid Verification Approaches for Stateful Models: A Case Study


Kevin Hulin
The University of Texas at Dallas
kjhulin@sandia.gov

Yalin Hu
Sandia National Laboratories
yhu@sandia.gov


## Abstract


Formal verification techniques such as model checking (MC) and theorem proving (TP) have found increasingly widespread use in the design of critical digital systems as a means to ensure their functional correctness [4][5]. However, both MC and TP have their limitations. TP generally requires non-trivial human intervention, and MC is limited by the state explosion problem [3]. The situation for MC is especially daunting for systems with large data components. In these cases, it is common to rely on a model abstraction to make MC feasible. Unfortunately, this may also add inaccuracies to the verification process, rendering it invalid.

We build upon current research and propose a hybrid verification approach that leverages the automation of MC and the logical soundness and flexibility of TP to build a highly automated, high confidence verification system. We perform a preliminary investigation of the effectiveness of this method by verifying a random access memory (RAM) model. We also discuss how the lessons learned in this case study can be extended and implemented in a robust verification system to verify other similar systems.


## Problem

- Formal Verification research aims at broadening the class of systems that can be verified through increased automation and reduced complexity
- Most common technique for reducing complexity is abstraction
- Automatic discovery of such abstractions is an open problem
- Manual creation of abstractions requires a deep understanding of how the system works and some expertise in formal verification
- Manual abstractions + Human error = Potentially disastrous results
- An error in a model abstraction could mask errors in the original model, yielding inaccuracies in the overall verification.
- Goal: To reduce the probability of human error playing a role in the manual creation of abstractions
- Solution:
  (1) Using a Theorem Prover, we prove the correctness of our abstractions through type predicates and theorems.
  (2) Utilizing the proven theorems, we build a simpler abstract model to be verified with a Model Checker.

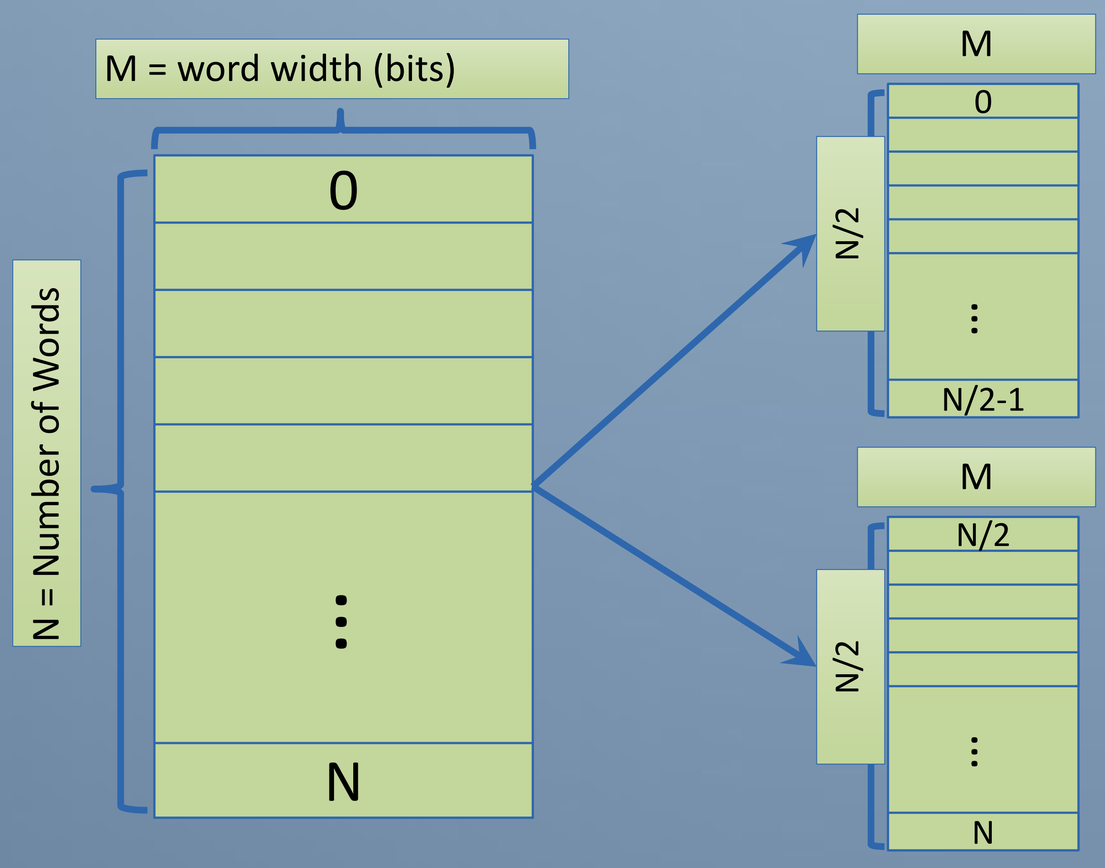


Fig. 1: Illustration of RAM Decomposition Abstraction. Applied iteratively, the size of the RAM model that is actually checked becomes trivially small. The problem is thus reduced from one of complexity $O(2^{NM})$ to $N$ problems of size $O(2^{M})$.

## Case Study: RAM

We demonstrate our sound verification system as applied to a simple RAM model. The RAM model (formally described below) is especially applicable to formal verification and abstraction because of its widespread use and notoriously stateful model. We use the NuSMV symbolic model checker [1] in conjunction with the ACL2 theorem prover [2] as our formal verifiers.

## Formal Definitions

We formally describe the RAM model using a Kripke structure with state space $S$, transition relation $R$, and edge label $L$ as follows:

### States

We define a state $S$ as a 5-tuple $(I, A, T, O, Y)$ where:
$I$ represents the input value, $I \in \mathbb{Z}_{2^M}$
$A$ represents the input address, $A \in \mathbb{Z}$
$T$ represents the Read/Write control bit, $T \in \{READ, WRITE\}$
$O$ represents the output value, $O \in \mathbb{Z}_{2^M}$
$Y$ represents the stored data as an ordered $N$-length list of $M$-bit values and $Y_i$ represents the $i$th value in this list. $Y \in \mathbb{Z}_{2^M}^N$
Initial state $S_0 = (0,0,\perp 0,0^N)$

### Transitions

The transition relation $R \subseteq S \times S$ is based off two predicates $read$ and $write$ as well as the control bit $T$:

$$R = \{(S, S') \mid T = READ \ \wedge read(S, S') \vee T = WRITE \wedge write(S, S')\}$$

We refer the reader to the paper for a detailed description.

## Decomposition Abstraction

The decomposition we use (illustrated in Fig. 1) takes advantage of the redundant nature of the RAM model. We conjecture that by using an iterative decomposition approach of splitting larger ram models into smaller ones, we can drastically reduce the total runtime for model checking and make verification feasible on commodity hardware.

## Proof

The critical component to abstraction in formal verification is verifying that the abstraction is actually representative of the target model. In our case study, we use the ACL2 theorem prover to prove that the abstraction is correct. An excerpt from our theorems is given in Fig. 2.

```
(defthm decompose-is-mem
 (implies
  (and
   (memoryp m)
   (posp n)
   (< n (mem-size m)))
  (let
   ((r (decompose-memory m n)))
   (and
    (memoryp (mv-nth 0 r))
    (memoryp (mv-nth 1 r))))))
```

```
(defthm dec-comp-equal
 (implies
  (memoryp mem)
  (let
   ((r (decompose-memory mem n)))
   (equal
    (compose-memory
     (mv-nth 0 r)
     (mv-nth 1 r))
    mem))))
```

Fig. 2: Theorems in ACL2 used to prove soundness of decomposition abstraction

## Results and Conclusion

The resulting runtime advantages from the decomposition abstraction can be seen in Fig. 3 and 4 below. Abstractions like the RAM decomposition here make formal verification feasible in larger stateful models. However, it is critical that these abstractions are correct in order for soundness to be maintained. Using the ACL2 theorem prover in conjunction with the NuSMV model checker, we show how we can achieve the soundness necessary for verifying high consequence systems. In the future, we hope to increase the automation of our system to promote the use of formal verification in all safety critical digital systems.

**Naïve Policy Run-times for M = 8, 12, 16**

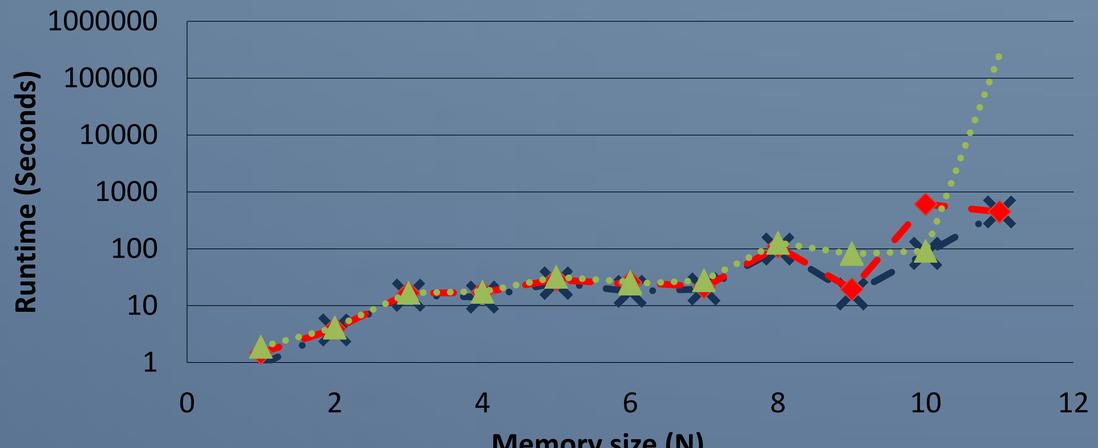


Fig. 3 : Runtime explosion for naïve verification

**Comparison of Runtimes for RAM (M=16)**

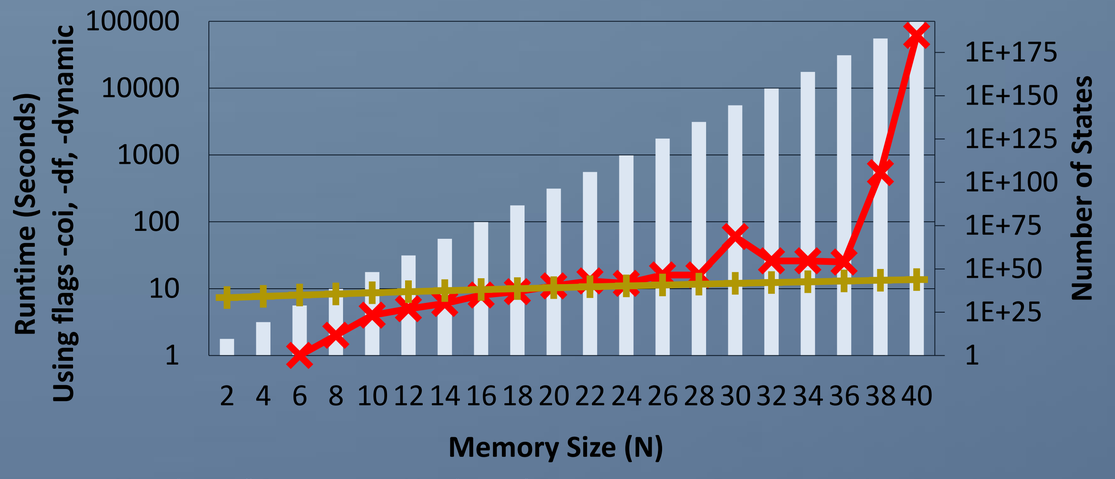


Fig. 4: Comparison of runtimes for decomposed and unaltered verification


References
[1] NuSMV Model Checking Tool (http://nusmv.fbk.eu/)
[2] ACL2 A Computational Logic for Applicative Common Lisp (http://www.cs.utexas.edu/~moore/acl2/)
[3] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model checking*, The MIT Press, 1999.
[4] Intel Corporation, Statistical Analysis of Floating Point Flaw, FDIV Replacement Program, November 1994.
[5] J. L. Lions, Report by the Inquiry Board, Ariane 5 Flight 501 Failure, July 1996.



Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly subsidiary of Lockheed Martin Corporation for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000

SAND# 2012-XXXXX