Quantifying Robustness by Symbolic Model Checking

S. Baarir       C. Braunstein       E Encrenaz       J-M. Ilié       T. Li
I. Mounier       D. Poitrenaud       S. Younes

HWVV 2010, July 15, 2010
Outline

1. Motivation
2. Preliminaries
3. Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification
4. Experiments
5. Conclusion and ongoing work
Motivation

1. Motivation

2. Preliminaries

3. Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4. Experiments

5. Conclusion and ongoing work
Dependability Analysis

**Dependable circuit to transient faults**

Soft error (SET or SEU) is and will be even more a major concern of embedded hardware designers.

- Critical applications (space mission ...) submitted to particle strikes or electromagnetic interferences
- Many other applications (video stream, phones ...) submitted to crosstalk coupling and/or high temperature

**Early analyses to evaluate the impact of faults**

- Improve the confidence of a design
- Early identification $\Rightarrow$ less $ or € for modifications
  - Identify the precise locations to be protected
  - Choose between different architectures of a design
Robustness evaluation

<table>
<thead>
<tr>
<th>Analysing robustness with respect to soft errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge state-space exploration</td>
</tr>
<tr>
<td>• soft error may come for bit-flip or erroneous latched signals</td>
</tr>
<tr>
<td>• bit-flip may occurred different location and time</td>
</tr>
<tr>
<td>• circuits have hundred of thousands flip-flops</td>
</tr>
<tr>
<td>Fault occurrences may cause tons of possible error configurations</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Working at RTL level</td>
</tr>
<tr>
<td>• Handling time and space multiple faults simultaneously (vs. simulation/injection)</td>
</tr>
<tr>
<td>• Relaxing the strict equivalence to a golden model or a specification</td>
</tr>
</tbody>
</table>
Self-stabilization evaluation

After a period of particles strikes, how to insure that the circuit returns to a safe configuration?

Analysing the self-healing capabilities of circuits

Concerns of our measures:

1. Rates of reparation ability
   - Number of *potentially* and *eventually* repairable states

2. Reparation velocity
   - Bounds of the reparations sequences

This allows designers to

- Choose part of design to be hardened
- Choose between implementations of the same design
Preliminaries

1. Motivation

2. Preliminaries

3. Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4. Experiments

5. Conclusion and ongoing work
Reachable States and Sequences

- \( r \in 2^R \): a state of \( C \)
- \( R_0 \): the set of initial state:
- \( i_1.i_2 \ldots i_{n-1} \): an input sequence
- \( f(i_1.i_2 \ldots i_{n-1}, r) \): a state sequence
- \( g(r, i_1.i_2 \ldots i_{n-1}) \): an output sequence
- \( \text{reach}(C) \): the set of reachable states of \( C \) from \( R_0 \)
Our robustness proposition

1. Motivation

2. Preliminaries

3. Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4. Experiments

5. Conclusion and ongoing work
Fault Model

1 Motivation

2 Preliminaries

3 Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4 Experiments

5 Conclusion and ongoing work
Fault Model

Type of faults
- Errors appear as *bit-flips* on register elements.
- There exists a set of protected register elements $P \subseteq R$ (this set may be empty).

Fault occurrences
- Occurrence of Multiple Faults – Multiple Units, except in protected registers.
- Several faults may occur at different time instants.
Circuit functioning with fault occurrences

\begin{center}
\begin{tabular}{ccc}
reg_0 & reg_1 & reg_2 & reg_3 & reg_4 \\
0 & 1 & 0 & 1 & 1 \\
\end{tabular}
\end{center}

Reachability set with fault occurrences

\textit{Error}(C, P), is the smallest subset of }2^R\text{ satisfying:

- }R_0 \subseteq \textit{Error}(C, P)

- }r \in \textit{Error}(C, P) \implies \{r' \in 2^R | \forall p \in P, r'[p] = r[p]\} \subseteq \textit{Error}(C, P)
Circuit functioning with fault occurrences

Reachability set with fault occurrences

\(\text{Error}(C, P)\), is the smallest subset of \(2^R\) satisfying:

- \(R_0 \subseteq \text{Error}(C, P)\)
- \(r \in \text{Error}(C, P) \Rightarrow \{ r' \in 2^R \mid \forall p \in P, r'[p] = r[p] \} \subseteq \text{Error}(C, P)\)
- \(r \in \text{Error}(C, P) \Rightarrow \{ r' \in 2^R \mid \exists i \in 2^I, r' = f(i, r) \} \subseteq \text{Error}(C, P)\)
Circuit functioning with fault occurrences

Reachability set with fault occurrences

\[ \text{Error}(C, P) \], is the smallest subset of \( 2^R \) satisfying:

- \( R_0 \subseteq \text{Error}(C, P) \)
- \( r \in \text{Error}(C, P) \Rightarrow \{ r' \in 2^R \mid \forall p \in P, r'[p] = r[p] \} \subseteq \text{Error}(C, P) \)
- \( r \in \text{Error}(C, P) \Rightarrow \{ r' \in 2^R \mid \exists i \in 2^I, r' = f(i, r) \} \subseteq \text{Error}(C, P) \)

Each state in \( \text{Error}(C, P) \) is called an error state.
Repairing model

1 Motivation

2 Preliminaries

3 Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4 Experiments

5 Conclusion and ongoing work
## Repairing sequences

### Introduction

### Requirements

When faults do not occur anymore, we want to characterize the set of error state that are "repairable":

- Reach a state considered as "correct"
- The path between the error state and the correct state is "constrained"

### Definition (Repairing sequence)

A repairing sequence is a sequence from an error state up to a *correct state*

- when faults do not occur anymore,
- when the sequence respects a *repairing pattern*. 
Repairing Sequences
Repairing Pattern

Repairing path

The way to go from an error state to a "correct" configuration (safe) may be constrained.

- Some configuration may be avoided (forbidden)
- Some configuration may be mandatory (required)

Repairing automaton

- Usual way to express constraints on paths: an automaton.
- A Repairing automaton for C is defined by \( \langle S, T, S_0, F \rangle \) where:
  - \( S \) a finite set of states.
  - \( T \subseteq S \times 2^R \times S \) a finite set of labeled transitions.
  - \( S_0 \) a finite set of initial states.
  - \( F \) a finite set of accepting states.
Repairing automaton example 1/2

\[
\begin{align*}
\neg \text{required} \land \neg \text{forbidden} \\
\text{required} \land \neg \text{forbidden} \land \neg \text{safe} \\
\text{required} \land \neg \text{forbidden} \land \text{safe} \\
\neg \text{forbidden} \land \neg \text{safe} \\
\neg \text{forbidden} \land \text{safe}
\end{align*}
\]
safe($C$), required($C$), forbidden($C$) . . . can be easily characterized as CTL properties:

- $\phi = \text{reach}(C)$: the whole set of reachable states.
- $\phi = AG(AFR_0)$: set of states returning unavoidably into the initial state.
- $\phi = \neg(r_1 \lor r_2)$: a given configuration of registers.
Quantification

1 Motivation

2 Preliminaries

3 Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4 Experiments

5 Conclusion and ongoing work
To quantify the circuit’s robustness, we compute:

- The number of Error states.
- **Potentiality**: The number of Error states from which at least one infinite fair sequence is a repairing sequence.
- **Eventuality**: The number of Error states from which all infinite fair sequences are repairing sequences.
Computing potentially and eventually reparable states

Set of repaired configuration:
\[ \text{Repaired} = \left\{ (r_C, r_{AC}) \in 2^R_C \times 2^R_{AC} \mid g_{AC}(r_{AC}) = 1 \right\} \]

\[ \nu_{pot} = \frac{|EF_{fair} \text{ Repaired} \cap R_0|}{|R_0|} \]

\[ \nu_{ev} = \frac{|AF_{fair} \text{ Repaired} \cap R_0|}{|R_0|} \]
Robustness
Sequence-based quantification

The velocity of the circuits is characterized by:
- Minimal and maximal length of repairing sequences
- The number of repairing sequences for each length between the bounds

Hypothesis
- We focus on the first repairing state along a repairing sequence.
- The environment reacts as soon as possible.
Robustness
Sequence-based quantification

The velocity of the circuits is characterized by:
- Minimal and maximal length of repairing sequences
- The number of repairing sequences for each length between the bounds

Hypothesis
- We focus on the first repairing state along a repairing sequence.
- The environment reacts as soon as possible.
Computing length

Input C: an instrumented circuit;
Output t: array of Integer;
    k=0;
While SAT(WithoutLoop(C, k)){
    t[k] = #SAT(ElementaryRep(C, k));
    k=k+1;
}
Return(t);

Computation

We compute the elementary repairing sequences:

\[ \text{WithoutLoop}(C, k) \land [r_k \in \text{Repaired}] \land \left(\bigwedge_{0 \leq j < k} r_j \notin \text{Repaired}\right) \]

- Bounds are computed by applying SAT solver iteratively.
- Number of sequences is translated in a #SAT problem.
Experiments

1 Motivation

2 Preliminaries

3 Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4 Experiments

5 Conclusion and ongoing work
Tool: extension of VIS

- What we have, the VIS model checker:
  - RTL inputs: Verilog
  - Symbolic structure: BDD
  - Temporal logics: CTL, LTL
  - Sat techniques.

- What we need:
  - Counting Error states,
  - Counting Reparable states (Error states satisfying CTL formulae)
  - Counting Elementary repairing sequences (sequences satisfying LTL formulae) ⇒ \#Sat problem.
Case study: different versions of a \textit{gcd} circuit

- State-based quantification:

<table>
<thead>
<tr>
<th></th>
<th>\textit{reach}(C) \mid</th>
<th>Error(C, P) \mid</th>
<th>\nu_{pot}</th>
<th>\nu_{ev}</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{gcd}</td>
<td>137929</td>
<td>2097152</td>
<td>100%</td>
<td>21%</td>
<td>0.36</td>
</tr>
<tr>
<td>\textit{gcd}_{\textit{fair}}</td>
<td>2097152</td>
<td></td>
<td>100%</td>
<td>100%</td>
<td>2</td>
</tr>
<tr>
<td>\textit{gcd-\textit{v1}}_{\textit{fair}}</td>
<td>304528</td>
<td>5.368709e^{08}</td>
<td>100%</td>
<td>100%</td>
<td>0.40</td>
</tr>
</tbody>
</table>

- Sequence-based quantification:

<table>
<thead>
<tr>
<th>C</th>
<th>Time</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0-2</td>
</tr>
<tr>
<td>\textit{gcd}</td>
<td>211</td>
<td>5e^{-10}</td>
</tr>
<tr>
<td>\textit{gcd-v2}</td>
<td>1595</td>
<td>3,93e^{-15}</td>
</tr>
</tbody>
</table>
Conclusion and ongoing work

1 Motivation

2 Preliminaries

3 Our robustness proposition
   - Fault Model
   - Repairing model
   - Quantification

4 Experiments

5 Conclusion and ongoing work
Conclusion

A new Framework

- Multiple transient faults by symbolic management
- Early in a design flow
- First implementation within a classical model checker (VIS)

New metrics

- Self-healing capabilities criteria
- Metrics to help choosing more robust design
- Metrics to determine the minimal set of protected register
Ongoing work

More elaborate fault model

Spatio-temporal windows
- Limit the number of fault occurrences
- Bounded the time of fault occurrences

More elaborate reparation

- Environmental context
- Circuit execution
- Time constraints