Injecting Various Faults for the Dependability Validation of Commercial Microcontrollers

D. S. SHYLU, M.TECH., SILJA PURUSHOTHAMAN, JYOTHI P. THOMAS
VLSI Design, ECE, Karunya University
Silukp01@gmail.com

Abstract- Faults will be a great challenge in modern VLSI circuits. Different faults are injected into the VHDL model of a commercial microcontroller and their effects have been compared. Bridging fault and stuck-at faults are injected. The methodology used is VHDL based fault injection technique, which allows an exhaustive analysis of the influence of different fault models and system parameters. Fault models are used at logic and RTL levels. From the simulation result, the occurrences of failures and have been found out. Fault coverage and test coverage are found out using DFT compiler and Tetramax tool.

I. INTRODUCTION
Faults will have a great impact in deep submicron technologies. Fault models used are stuck-at fault, bridging fault, delay fault. A stuck-at fault is a particular fault model used by fault simulators and Automatic test pattern generation (ATPG) tools to mimic a manufacturing defect within an integrated circuit. Individual signals and pins are assumed to be stuck at Logical '1', '0' and 'X. Two signals are connected together when they should not be. Depending on the logic circuitry employed, this may result in a wired-OR or wired-AND logic function. Total number of violations in the memory and register can be detected using DFT compiler and fault coverage as well as test coverage is detected by using tetramax tool, which allows a systematic and exhaustive analysis of the influence of different fault and system parameters. Tetramax is a high-speed, high-capacity automatic test pattern generation (ATPG) tool. It can generate test patterns that maximize test coverage, using a minimum number of test vectors, for a wide variety of design types and design flows. From the simulation result, the occurrences of failures have been logged.

II. FAULT MODELS.
Fault models used are stuck-at fault, bridging fault, delay fault. A stuck-at fault is a particular fault model used by fault simulators and Automatic test pattern generation (ATPG) tools to mimic a manufacturing defect within an integrated circuit. Individual signals and pins are assumed to be stuck at Logical '1', '0' and 'X. The fault can be at an input or output of a gate. The stuck at fault model assumes that only one input on one gate will be faulty at a time, assuming that if more are faulty, a test that can detect any single fault, should easily find multiple faults. To use this fault model, each input pin on each gate in turn, is assumed to be grounded, and a test vector is developed to indicate the circuit is faulty. Two signals are connected together when they should not be. Depending on the logic circuitry employed, this may result in a wired-OR or wired-AND logic function.

III. FAULT INJECTION EXPERIMENTS.
Fault injection experiments were carried out on the VHDL model of 8051 microcontroller. Stuck-at fault, bridging faults are injected into the RAM and PSW register of 8051 microcontroller. A stuck-at fault is a particular fault model used by fault simulators and Automatic test pattern generation (ATPG) tools to mimic a manufacturing defect within an integrated circuit. Individual signals and pins are assumed to be stuck at Logical '1', '0' and 'X. Two signals are connected together when they should not be. Depending on the logic circuitry employed, this may result in a wired-OR or wired-AND logic function.
IV. FAULT INJECTION TARGET.

RAM (128 bytes), registers, ALU, decoder, buses and ROM of 8051 microcontroller. Depending upon the number of places activated in each fault injection, faults can be classified as single (one cell or bus line) or multiple (various cells or bus lines).

Fig.1. Injection targets

V. RESULT

The Fig.1 shows the fault injection target were the faults are injected. Fig.2 shows the waveform of fault free RAM, in that various faults were injected and the effects of faults were obtained and compared. Fig.5 shows the waveform of 8051 microcontroller in which stuck-at-faults are injected.

Fig.2. Waveform of fault free 8051 RAM

Fig.3. Waveform of RAM with delay fault
### Uncollapsed Stuck Fault Summary Report generated by DFT compiler

<table>
<thead>
<tr>
<th>fault class</th>
<th>code</th>
<th>#faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected</td>
<td>DT</td>
<td>25792</td>
</tr>
<tr>
<td>Possibly detected</td>
<td>PT</td>
<td>0</td>
</tr>
<tr>
<td>Undetectable</td>
<td>UD</td>
<td>64</td>
</tr>
<tr>
<td>ATPG untestable</td>
<td>AU</td>
<td>0</td>
</tr>
<tr>
<td>Not detected</td>
<td>ND</td>
<td>0</td>
</tr>
</tbody>
</table>

---

**Total faults**: 25856  
**Test coverage**: 100.00%

**STIL file for stuck-fault in RAM generated by Tetramax tool.**

```stil
STIL 1.0 {
  Design P2002.13;
  CTL P2002.18;
  TRC P2002.3;
}
```

**Header {**

Title "TetraMAX (TM) B-2008.09-SP4-i090227_182204 STIL output";
Uncollapsed Bridging Fault Summary Report generated by DFT compiler

<table>
<thead>
<tr>
<th>fault class</th>
<th>code</th>
<th>#faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected</td>
<td>DT</td>
<td>21600</td>
</tr>
<tr>
<td>Possibly detected</td>
<td>PT</td>
<td>0</td>
</tr>
<tr>
<td>Undetectable</td>
<td>UD</td>
<td>38</td>
</tr>
<tr>
<td>ATPG untestable</td>
<td>AU</td>
<td>0</td>
</tr>
<tr>
<td>Not detected</td>
<td>ND</td>
<td>0</td>
</tr>
<tr>
<td>total faults</td>
<td></td>
<td>21638</td>
</tr>
<tr>
<td>test coverage</td>
<td></td>
<td>100.00%</td>
</tr>
</tbody>
</table>

STIL file for Bridging fault in RAM generated by tetramax tool.

STIL 1.0
Design P2002.13;
CTL P2002.18;
TRC P2002.3;

Header
Title "TetraMAX (TM) B-2008.09-SP4-i090227_182204 STIL output";
Date "Tue Jan 5 15:25:07 2010";
Source "Minimal STIL for design 'ram224'";
History
Ann {* Tue Jan 5 14:47:02 2010 *}
Ann {* DFT Compiler B-2008.09-SP4 *}

PatternInformation
Pattern "_pattern_"
FileReference "/reports/ram_224_bridge_pats.stil";
PatternCharacteristics
BreakPoint No;
ConditionalStatements No;
CoreUsageReady No;
NonCyclized No;
DeltaChangeVectorData Yes;
GoTo No;
IddqTestPoints No;
Macro Yes;
MultiBitData No;
NumberPatternUnits 389;
PatternVariables No;
ProcedureCalls WithParameters;
Shift Yes;

Uncollapsed Stuck Fault Summary Report generated by Tetramax tool

<table>
<thead>
<tr>
<th>fault class</th>
<th>code</th>
<th>#faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected</td>
<td>DT</td>
<td>25792</td>
</tr>
<tr>
<td>Possibly detected</td>
<td>PT</td>
<td>0</td>
</tr>
<tr>
<td>Undetectable</td>
<td>UD</td>
<td>64</td>
</tr>
<tr>
<td>ATPG untestable</td>
<td>AU</td>
<td>0</td>
</tr>
<tr>
<td>Not detected</td>
<td>ND</td>
<td>0</td>
</tr>
<tr>
<td>total faults</td>
<td></td>
<td>25856</td>
</tr>
<tr>
<td>test coverage</td>
<td></td>
<td>100.00%</td>
</tr>
</tbody>
</table>
Type Bridge;
FaultCount 21638;
FaultsDetected 21600;
FaultCoverage 100;
} //end Fault
} //end Pattern
} //end PatternInformation

Uncollapsed Bridging Fault Summary Report generated by Tetramax tool.

<table>
<thead>
<tr>
<th>fault class</th>
<th>code</th>
<th>#faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected</td>
<td>DT</td>
<td>21600</td>
</tr>
<tr>
<td>Possibly detected</td>
<td>PT</td>
<td>0</td>
</tr>
<tr>
<td>Undetectable</td>
<td>UD</td>
<td>38</td>
</tr>
<tr>
<td>ATPG untestable</td>
<td>AU</td>
<td>0</td>
</tr>
<tr>
<td>Not detected</td>
<td>ND</td>
<td>0</td>
</tr>
<tr>
<td>total faults</td>
<td></td>
<td>21638</td>
</tr>
<tr>
<td>test coverage</td>
<td></td>
<td>100.00%</td>
</tr>
</tbody>
</table>

VI. CONCLUSION.

In this project, the effects of bridging fault and stuck-at fault on the behavior of commercial microcontroller (8051) are presented. The methodology used lies in VHDL-based fault injection technique, which allows a systematic and exhaustive analysis of the influence of different fault parameters and factors. The results of various fault models are compared. Then fault coverage and test coverage are detected using DFT compiler and Tetramax tool.

VII. ACKNOWLEDGEMENT

I wish to thank the staffs of Karunya University, Coimbatore and Synopsys, for the support in injecting the fault into the circuit and for the comparison of the three models respectively.

VIII. REFERENCES.

[7]. N. Weste, D. Harris, “CMOS VLSI design”, Addison- Wesley, 2005.