Early Analysis of Fault-Attack Effects for Cryptographic Hardware

Régis Leveugle*

TIMA Laboratory – Grenoble –FRANCE

* Partly supported by the DURACELL Project
Overview

- Context(s) of fault effect analysis
- Early analysis: methods and tools
- Differences between paradigms: impact on tools and fault models
- Conclusions and perspectives
Act I ...
Once upon a time ... in space ...

- Energetic particles (photons or charged particles) can affect the microelectronic devices and subsystems in several ways.
  - Two main classes of effects:
    - **Total Ionization Dose (TID):** long-term degradation of electronics due to the cumulative deposited charge.
    - **Single Event Effects (SEEs):** occur when a single charged particle strikes the material, ionizes it and provokes a current pulse.
      - SEEs:
        - Single Event Latchups (SELs) create shorts between ground and power, and cause permanent functional damages (hard errors).
        - Single Event Upsets (SEUs) occur when a transient pulse provokes a bit flip in a device memory cell (soft errors).

- Radiation effect problems in space applications can be solved by:
  - using radiation hardened devices, by technology or design,
  - qualifying commercial circuits by radiation ground testing ... and/or early analyses.
Consequences of CMOS evolutions

- CMOS shrinking
  - Reduced Vdd and noise margin
  - Reduced node capacitance
  - Increased frequency (increased probability of latching)

- Very deep sub-micron CMOS technologies are increasingly sensitive to the effects of alpha particles and atmospheric neutrons => SET / SEU / MBU.

- Studies focused on (but not limited to) SET/SEU-like faults
  - Can be extended to other faults (stuck-at, coupling, …), permanent or not
  - Partially covers noise problems (signal integrity)
And also …

- New security threats: fault attacks
  - Cryptography primitives: DES / RSA / AES …
  - Security locks (ratification counters, …)
  - …

- Various possibilities
  - Power glitch
  - Flash light
  - Laser
  - …

- Ultimately: logic fault(s)
There are "new" problem(s) with fault-based attacks …

… Are answers to "old" problems of some help ???

And up to what extent?
Basics of

Existing Dependability Analysis Environments
Current goals of analysis environments

- Dependability
- Increase
- Evaluation

- Working at various levels in the design flow (various design description levels),
- Automated,
- Compatible with classical up-to-date industrial design flows.
# Link between analysis levels

<table>
<thead>
<tr>
<th>Description level</th>
<th>Analysis</th>
<th>Qualitative info.</th>
<th>Quantitative info.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavioral/RTL</td>
<td>Behavioral simulation (emulation)</td>
<td>Error -&gt; failure (application point of view)</td>
<td>P(failure</td>
</tr>
<tr>
<td>Gate level (+ back annotation)</td>
<td>Gate level simulation (timed)</td>
<td>Glitch -&gt; Error (latched) + refinement previous analysis</td>
<td>P(error</td>
</tr>
<tr>
<td>Electrical/Physical</td>
<td>Electrical/Physical simulation</td>
<td>Particle or physical event -&gt; glitch or bit-flip</td>
<td>P(glitch</td>
</tr>
</tbody>
</table>

**Estimation principle of application failure (limitations to be considered at high levels):**

\[
P(\text{failure}) = P(\text{failure}|\text{error}) \times \left[ P(\text{bit-flip}|\text{particle}) + P(\text{error}|\text{glitch}) \times P(\text{glitch}|\text{particle}) \right] \times P(\text{particle})
\]

- **Critical logic paths**
- **Environment**
- **Critical nodes**
- **Sensitive nodes**
Summary of the "early evaluation" goals

- Develop injection methods and CAD environment to early analyze the functional impact of SEUs at the application level
  - Early: performed on RTL descriptions (VHDL)
  - Functional: technology independent (no detailed timing information – targets bit-flips, not transients in combinatorial logic network)
  - Based on commercial tools and standard design flows

- Early identification of
  - Functional failure modes (critical behaviors)
  - Error propagation paths (critical nodes)

- Early assessment of
  - Dependability level
  - Design hardening efficiency

Link with design hardening

functional model, including qualitative and quantitative information
Dependability analyses: alternative results

Cycle-by-cycle comparisons

Fault classification

Analysis of error propagation paths
Alternative approaches

- Gate-level simulation
- Gate-level simulation with back-annotation
- System-level simulation (e.g. SystemC)
- FPGA-based emulation
- RT-level simulation
- ISS-based simulation
- CEU* injection
- CEU: code-emulated upsets

Details (access to internal elements, propagation time)

Experiment time
Alternatives for fault injection campaigns

- VHDL specification
  - Initial VHDL specification
  - VHDL specification modified ("instrumented")

Saboteur insertion
- Simulation or emulation
- Mutant generation
- Simulator Commands/Database
- Run-Time Reconfiguration
- Simulation
- Emulation
Analysis flow: overview

Functional failure mode analysis of a digital integrated circuit

User Specification
- Hierarchical VHDL (synthesizable)
- Campaign definition
- Input vectors (workload)

Injection campaign
=> VHDL simulation
=> Hardware prototyping (FPGA)

Data analysis
=> Reached states
=> Transitions between states
=> Probability / latency
(no pre-defined model)

Fault/error model + target
=> saboteurs, mutants

Error 1
Error 2
Error 3
Failure
Detection

Correct

Latent
Controlled generation of mutants

- "Controlled generation" of mutants implies:
  - Generation from high-level (RT-level) descriptions (available early in the design process)
  - Significant faulty behaviors (related to actual fault effects observable in the field => SEUs)
  - Optimization for synthesis (compatibility with simulation and emulation)
  - Taking into account the limitations of hardware emulation systems

- Criteria for quality evaluation:
  - Number of additional I/Os (number of sub-campaigns)
  - Number of gates after synthesis (emulation hardware complexity)
  - Maximum frequency (time required for the injection campaign)
Levels of fault/error injection for SEUs

- Physical level: a single charged particle incident on the IC generates a dense track of electron hole pairs and this ionization causes a transient.

- High-level injection (RT-level control flowcharts, or FSMs – state registers), with or without knowledge on the state assignment (can be easily refined when the actual state codes are known):

  - Combinatorial Logic
  - State Register

  SEU: bit-flip

  Erroneous transition (between existing states)
Targeted faults

Modeling levels of a SEU

- Physical
  - (electron/hole pairs)

- Electrical
  - (current or voltage pulse)

- Logic
  - (bit-flip, signal glitch)

- Behavioral
  - (e.g. erroneous transition)
Act III ...

Adequate for Security-related Fault Injections?
Which aspect?

- Circuit/application modeling?
  - Similar ...
  - Main difference between security and safety assessment: protections sized according to the potential losses (and attack investments)

- Definition of failure types?
  - Up to the user! (conditions on signals)

- Type of faults to be injected during the experiments?
  - ... Here is the gap!
Fault modeling: paradigms

- **(Off-line) Test paradigm**
  - Defects: manufacturing, aging
  - Permanent / intermittent faults

- **(On-line) Test paradigm**
  - Faults induced by the environment (or signal integrity)
  - Transient (or intermittent) faults
  - Low occurrence probability
  - High locality (example: particle)

- **Security paradigm (attacks)**
  - Faults induced intentionally (hackers)
  - Transient (or intermittent) faults
  - High occurrence probability (induced intentionally)
  - Variable locality (example: flash light vs. focused laser beam)
Selection of fault models

- Gate-level (or upper) modeling, non-intrusive (or semi-intrusive) attacks (no circuit modification)

- Four basic models
  - Stuck-at (single / multiple - transient)
  - Delay faults
  - SET (transient inversion of signals)
  - SEU / MBU => memory elements

- Delay faults, SETs: require gate-level knowledge (propagation time)
- Stuck-ats: can be applied at RT-level on selected targets
Comparison of models (1)

- **Stuck-at**
  - Polarity to be defined: zero or one
  - Transient in the security paradigm
  - Can be applied at gate level, or at RT level (on selected nodes)

- **Delay faults (positive or negative)**
  - Can be applied only at gate level, mainly after P&R
  - Can be modeled as stuck-ats with the required polarity and a duration equal to the delay, occurring or disappearing when the event occurs on the target signal

![Diagram showing Stuck at 1 and Stuck at 0]
Comparison of models (2)

- **SET**
  - Can be applied only at gate level, mainly after P&R
  - Several definitions … Usually, forced inversion on a node (without taking into account events that should occur during the fault duration)
    => equivalent to a transient stuck-at … on a given polarity
  - Duration generally assumed inferior to the clock period

- **SEU/MBU**
  - Direct bit-flip in a memory element (direct error, without activation and propagation of a fault)
  - Can be applied at gate or RT level
  - Few common points with the other models
Comparison of models (3)

- Conclusion: 2 models can be sufficient

- SEU/MBU

- Transient stuck-ats, with duration D
  - D being potentially superior to the clock period (generalization of SETs => multi-cycle faults)
  - Analysis including all possible occurrence instants at gate level (to include all delay faults)

  - At RT-level, duration defined by a number of clock cycles (functional analysis) + selection of significant targets

[DURACELL project]
Attributes

- High number of possible attributes (or parameters)
  - Specify the characteristics and the selection of faults and targets for a given model
  - General framework of the study: logic level, transient faults, … => limitation of the list of attributes

- Main attributes in the studied context:
  - Duration of faults (if stuck-at)
  - Spatial and temporal multiplicity
  - Correlation of multiple faults (spatial or temporal)
  - Target and injection time selection (exhaustive/deterministic/random)
  - Type of random distributions (uniform, gaussian, …)

- Intervals of values: depend on context/technology
Definition of attributes: example

- Spatial multiplicity (MBU) – laser attack
  - Depends on
    - Laser focus
    - Placement/routing
    - Cell sensitivity
    - ...

- High-level analysis: no information on P&R
  => assumptions / limitations (e.g. limited to the elements in a given register), but gives constraints on P&R for coherence

P&R-1, focus 1
=> mult. up to 2 per element, 4 elements

P&R-2, focus 1
=> mult. up to 1 per element, 4 elements

P&R-2, focus 2
=> mult. up to 2 per element, 4 elements
Conclusions

- Suitable analyses for fault-based attacks are not so different from previous concerns ...

- … and existing analysis environments and methodologies can be used in this (new) context ...

- BUT fault models must be revisited ...

- … and tools must be extended (e.g. generation of new types of mutants).
Perspective: future view

Dependability analysis/Characterization

- Functional analysis
- Multi-level fault injection
- Behavioral model generation
- Anticipation of radiation ground testing

Fault tolerance / On-line testing

- Source-to-source transformations of synthetizable RTL descriptions
- Architectures with limited redundancy

Hardening

- Efficient implementation/perenniaility
- Earlier validation of the dependability properties (circuit and system level)

Campaign:
simulation/emulation + data analysis

Injection: saboteurs/mutants

+ configurable IPs

Sequencing error detection

Validation

R. Leveugle

Workshop on Fault Detection and Tolerance in Cryptography - Florence, Italy, June 30, 2004