Speeding-up Simulation-based Fault Injection for Highly Complex HDL Models

PhD. Student: Ilya Tuzov

Collaborators and Supervisors: Juan-Carlos Ruiz, David de Andrés and Pedro Gil

Doctoral Program in Computer Science

ITACA, {tuil, jcruizg, ddandres, pgil}@disca.upv.es

Abstract — Increasing integration scales and clock frequencies also increase the sensitivity of integrated circuits to different kinds of faults. Early design verification in presence of faults and dependability assessment is commonly accomplished by means of Simulation-based fault injection (SBFI) techniques, which can be applied at different levels of HDL description. The closer to implementation models are, the more representative are simulation results. However, injecting faults in highly complex and detailed models is a very resource-intensive process that usually requires prohibitive simulation times. This work proposes an approach to speed up this process, making feasible the dependability assessment of very detailed implementation-level HDL models.

1. Motivation and objectives

- **Hardware designs** must be verified in presence of faults, caused by internal and/or environmental factors, e.g. single-event effects (SEE). Sensitivity to faults should be estimated to ensure device's safe behavior.
- **Faults effects** can be representatively simulated by means of implementation-level HDL models, which accurately reflect combinational/timing behavior and the structure of resulting circuit.
- **Simulation of implementation-level models** is up to 4 orders of magnitude slower than of source RTL (behavioral) models, resulting in prohibitive SBFI experimentation time in practice.

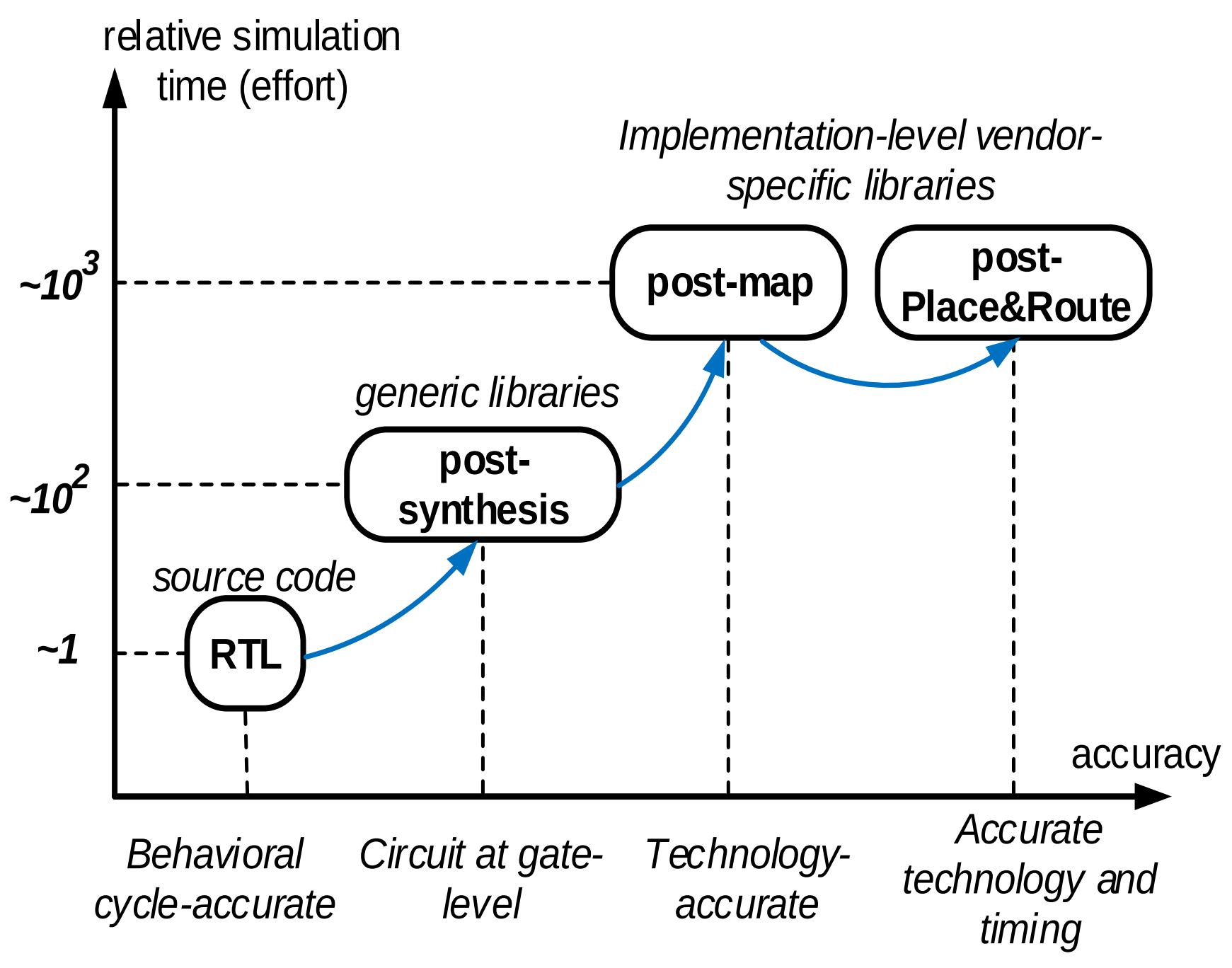


Figure 1. Simulation time growth drastically with the increase of model accuracy along the semi-custom design flow

Objectives:

- Analyze the factors affecting simulation complexity at implementation-level
- Optimize fault simulation procedures with the aim for speed-up to enable sensitivity analysis/dependability assessment for complex HDL designs.

2. Factors affecting complexity of faults simulation

SBFI complexity

- Micro-level (single experiment)
  - **Number of scheduled events per time unit**
    - Number of signals (simulation primitives) N
    - Switching activity α
  - **Simulated model time (workload complexity)**
    - Model initialization T_{init}
    - Experiment execution T_{exec} (incl. injection and observation)
  - **Tracing complexity**
    - Types and number of observed signals and variables
    - Sampling conditions and switching activity
- Macro-level (SBFI campaign)
  - **Faultload complexity**
    - Number of targeted nodes
    - Number of injections per fault model per node
  - **Complexity of sensitivity analysis**
    - Computed metrics (algorithmic complexity)
    - Size of observation traces (amount of samples)

3. Proposed optimizations and expected speed-up

1. Mixed-Level HDL assembly - reduces computational complexity of HDL model

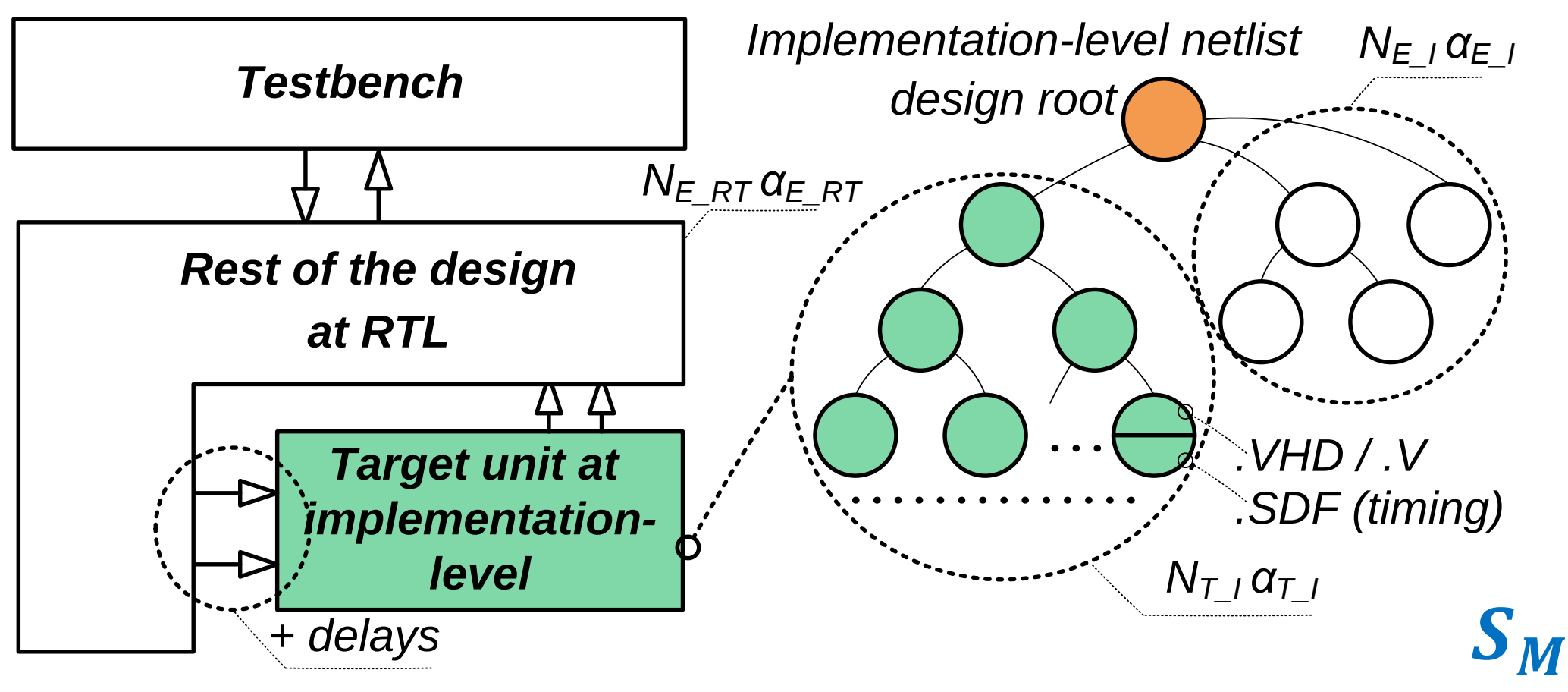
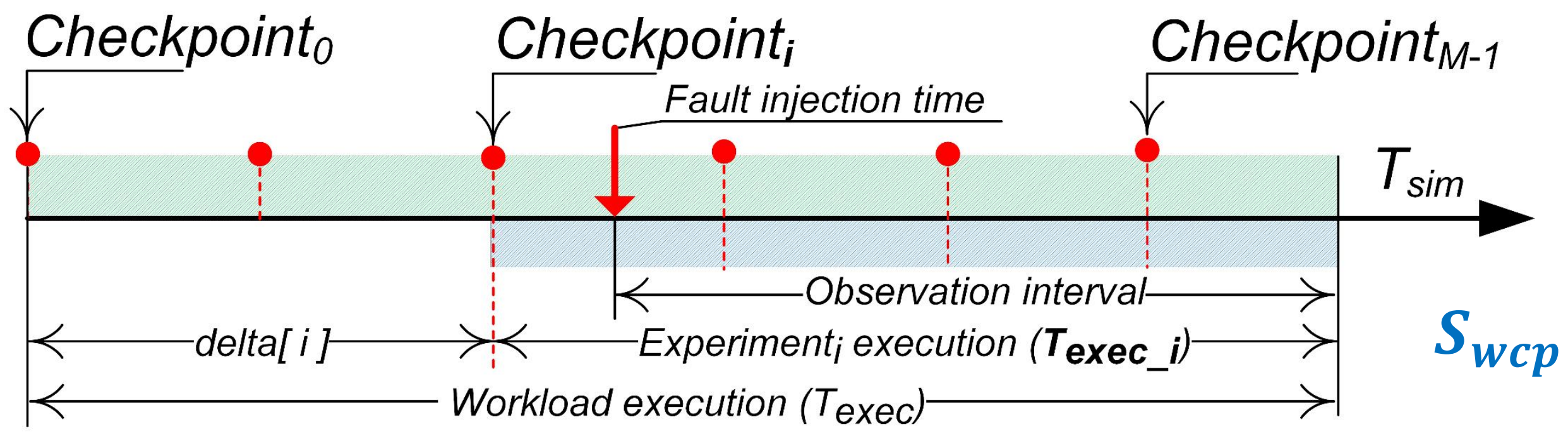

Testbench
Rest of the design at RTL
Target unit at implementation-level
+ delays
N_{E_RT} α_{E_RT}
Implementation-level netlist design root
N_{E_I} α_{E_I}
.VHD / .V
.SDF (timing)
N_{T_I} α_{T_I}

Figure 2. Target unit at implementation level interacts with high-level model of the rest of design

Expected speed-up factor to be verified experimentally

$$S_{MLA} = \frac{N_{T\_I} \times \alpha_{T\_I} + N_{E\_I} \times \alpha_{E\_I}}{N_{T\_I} \times \alpha_{T\_I} + N_{E\_RT} \times \alpha_{E\_RT}}$$

2. Checkpoints – save/restore pre-computed simulation state to bypass model initialization and reduce the workload execution in each of N experiments



$$S_{wcp} = \frac{N \times (T_{init} + T_{exec})}{T_{init} + \sum_{i=0}^{N-1} T_{exec\_i}}$$

Figure 3. Checkpoints reduce the simulation time to just fault injection and effects observation

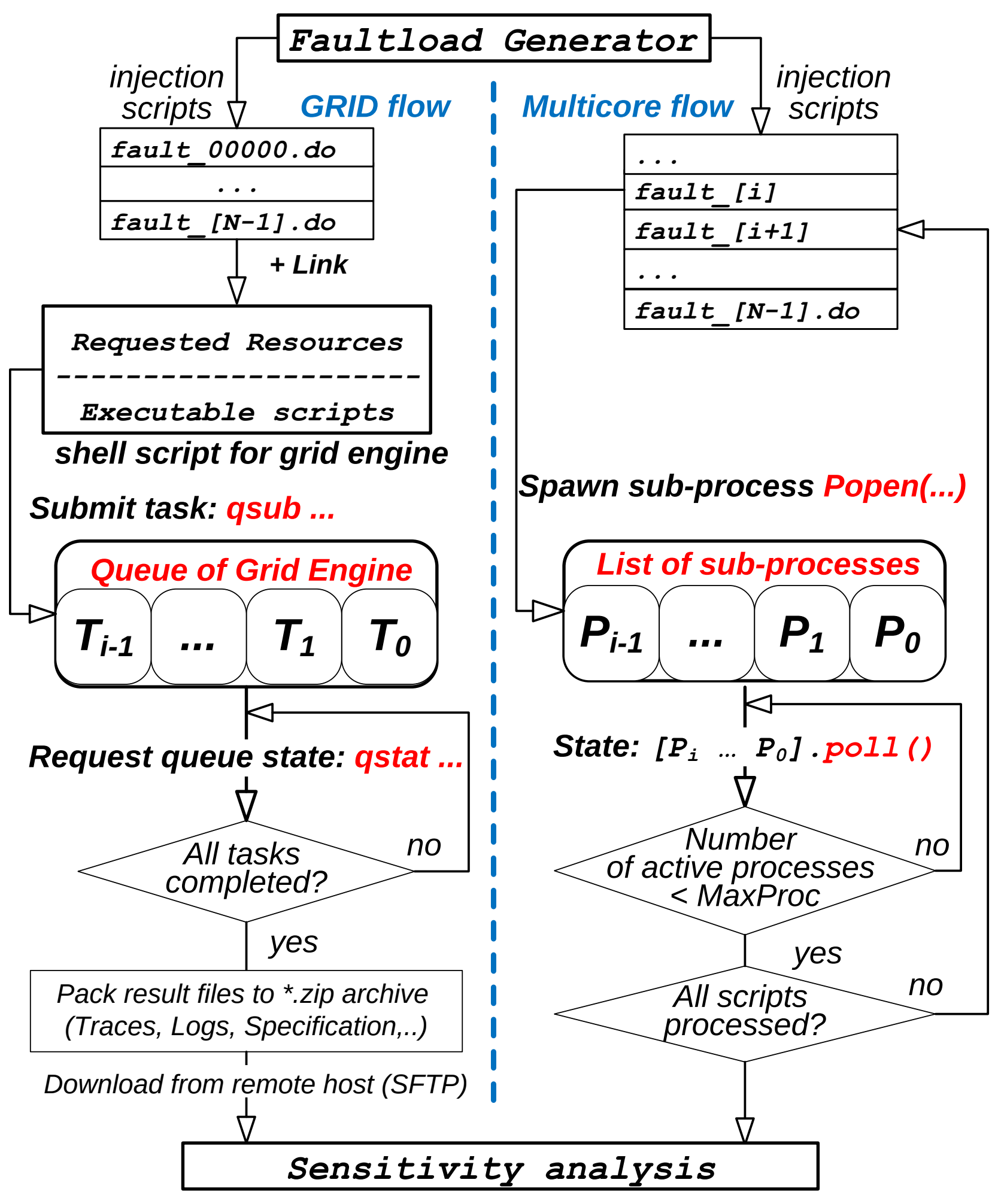


Figure 4. Grid- and multicore-based SBFI flow

3. Multiprocessing - execute N experiments in *Proc* processes in parallel on **Grid** or Mutlicore-based computing systems

$$S_{MP} = \frac{N \times (T_{init} + T_{exec})}{T_{init} + \frac{N}{Proc} \times T_{exec}}$$

4. Fine tuning of sampling procedure – observation traces are optimized for minimal size by adjusting sampling triggering conditions to particular workload. Accuracy of sensitivity analysis is maintained.

4. Experimental speed-up versus estimation

Target model – LEON3 soft-core processor synthesized for Virtex-6 FPGA.

Workload – integer matrix multiplication (*MiBench* automotive benchmark)

Faultload – single transient (bit-flip) and permanent (stuck-at-/0) faults.

Computing platforms:

a) Cluster 'Rigel' (UPV): 72 nodes Xeon E5-2450, CentOS 6, Sun Grid Engine;
b) Multicore PC: Intel Core i5-4670, CentOS 6.7.

Simulator: Mentor Graphics ModelSim 10.4 in both environments.

Table 1. Experimentation time measured with respect to enabled optimizations

	Optimizations					**Execution time**		
Config.	ICP	MLA	WCP	Multicore	Grid	Single experiment T_{avg} (seconds)	Standard deviation σ	Whole campaign T_{total} (hours)
C1	−	−	−	−	−	7327	11.0	15896#
C2	+	−	−	−	−	113	14.9	246.8*
C3	+	+	−	−	−	40	2.2	87.7*
C4	+	+	+	−	−	**27**	3.8	59.5*
C5	+	+	+	+	−	30	5.8	22.7
C6	+	−	+	+	−	79	15.7	59.1
C7	+	+	+	−	+	43	10.9	**3.0**
C8	+	−	+	−	+	114	28.0	5.7

# Estimation based on 10 injection experiments

* Estimation based on 500 injection experiments

CPI/CPW – initialization / workload checkpoints, MLA – mixed-level assembly

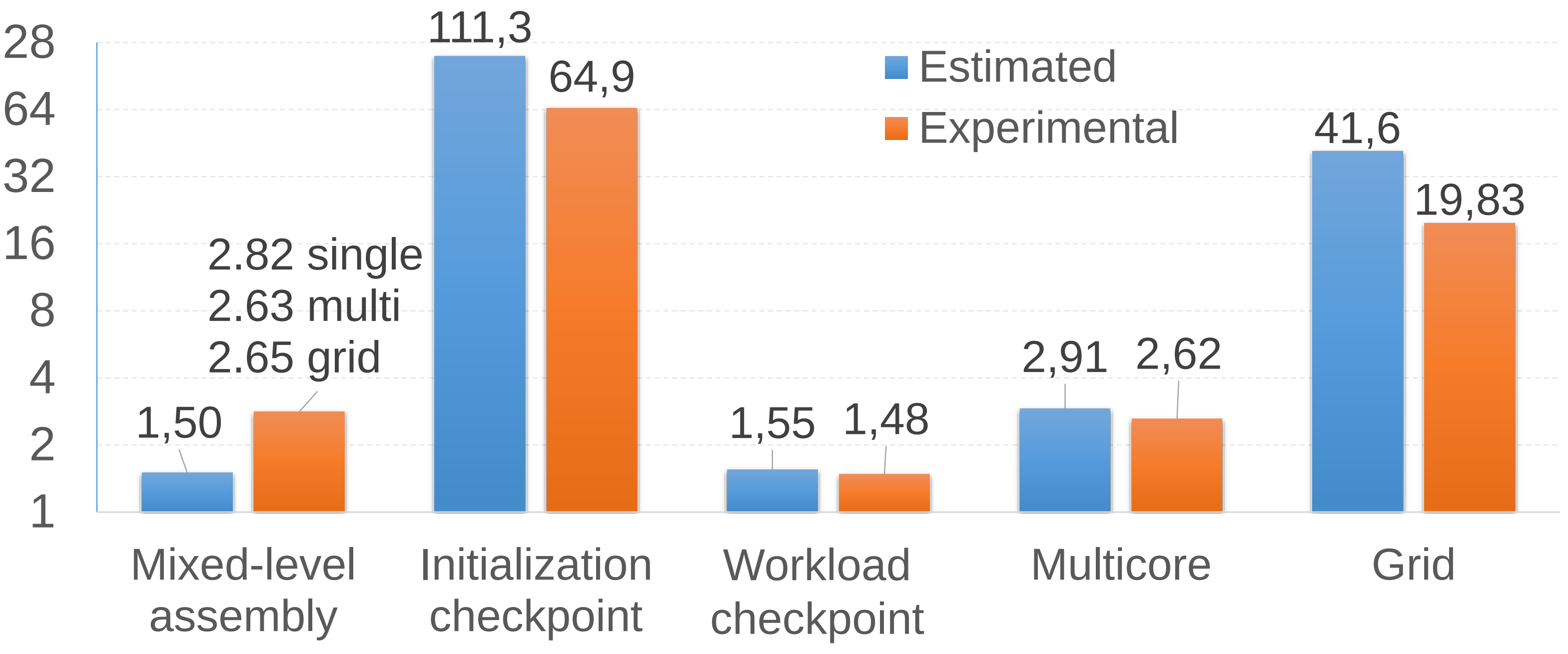


Figure 5. Expected and experimentally obtained speed-up factor (logarithmic scale)

- **Global speed-up factor: 5047 (**S_{MLA}×S_{ICP}×S_{WCP}×S_{MP})
- **Storage space reduction factor: 1387** + tracing speed-up of 1,36.
- No major discrepancies in SBFI results between implementation- and mixed-level models (less than 2% difference for all fault models and failure modes).

Conclusions: Proposed optimizations greatly accelerated the execution of fault simulation experiments and reduced the required storage space without any loss in accuracy of results, supporting the stated hypotheses and expressions for speed-up factors. Latter could be used to estimate the efficiency of each optimization depending on the properties of particular HDL model, workload and computing resources. This enabled extensive SBFI campaigns for complex implementation-level models, required (among other applications) for the ongoing study of design space exploration for HW design.

Acknowledgment: This work has been partially funded by the Ministerio de Economía, Industria y Competitividad de España under grant agreement no TIN2016-81075-R, and the "Programa de Ayudas de Investigación y Desarrollo" (PAID) de la Universitat Politècnica de València.