A Meta-Modeling-Based Approach for Automatic Generation of Fault-Injection Processes

B.-A. Tabacaru,

M. Chaari, W. Ecker, T. Kruse

Infineon Technologies AG

accellera SYSTEMS INITIATIVE™

2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

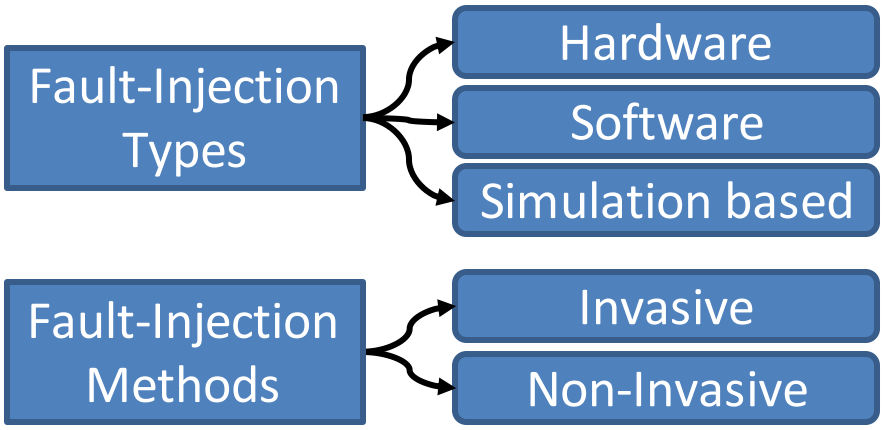
Outline

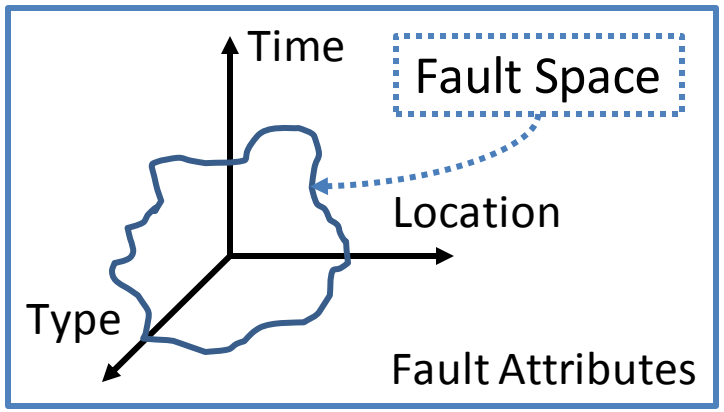
- Motivation
- Fault-Injection Theory
- Fault-Injection Requirements
- SCFIT (SystemC Fault-Injection Tool)
- SCFIT Gen
- Results
- Discussion
- Next Steps

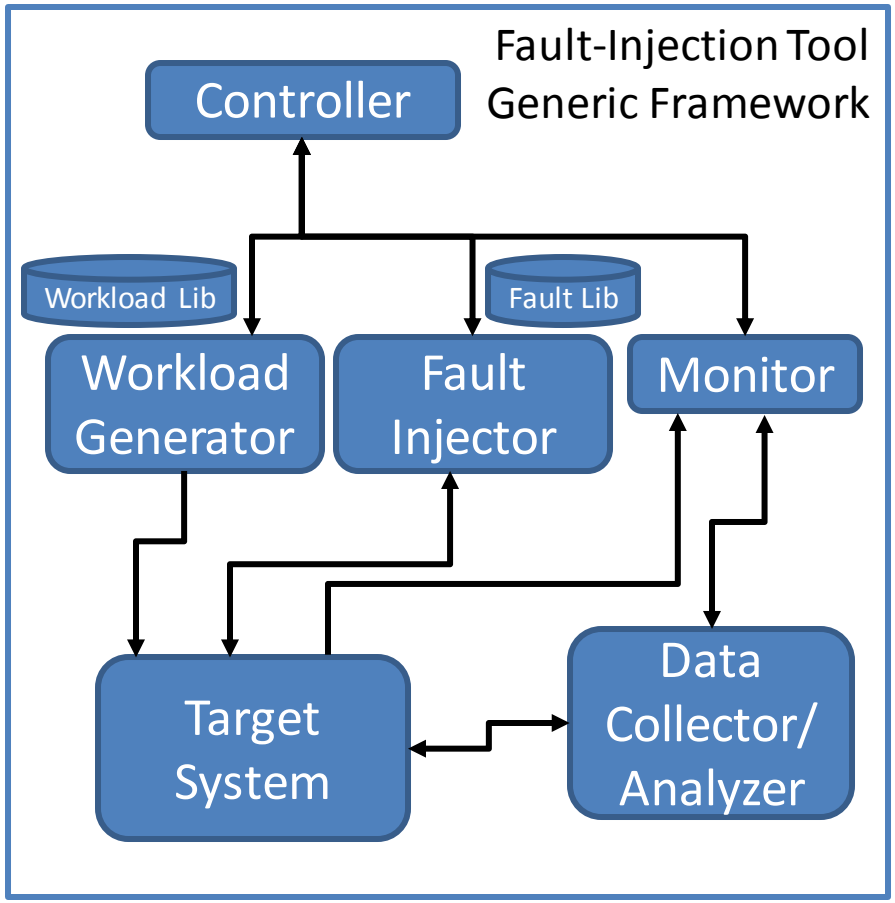
Motivation

- Safety-critical applications
  - Dependability analysis
  - ISO-26262 based safety analysis
- System reaction to fault injection
  - Virtual prototyping for early fault analysis
  - Get results before hardware is ready
  - Faster simulation time
- System exploration of fault-tolerant designs
  - Chip area
  - Design complexity/cost

Fault-Injection Theory



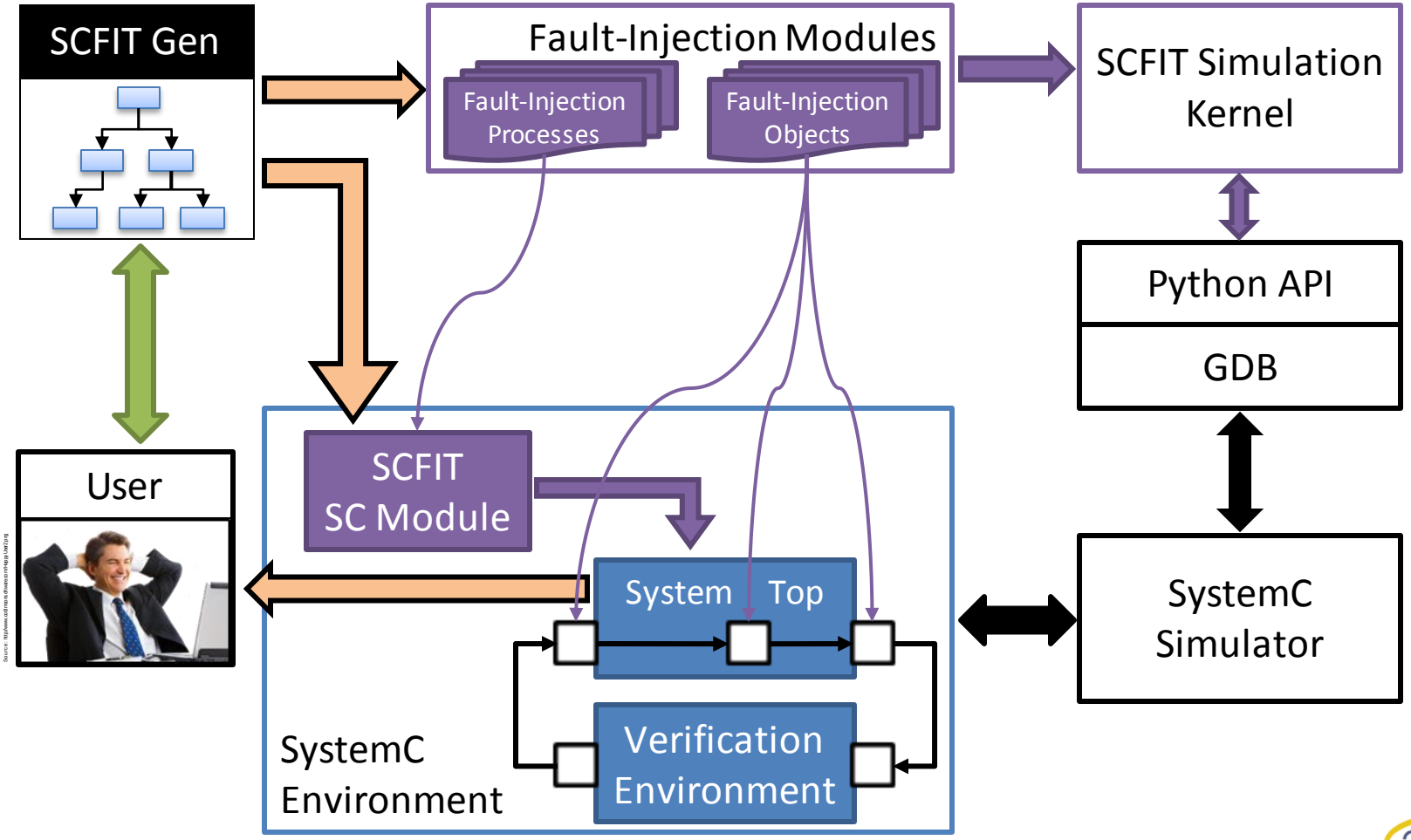




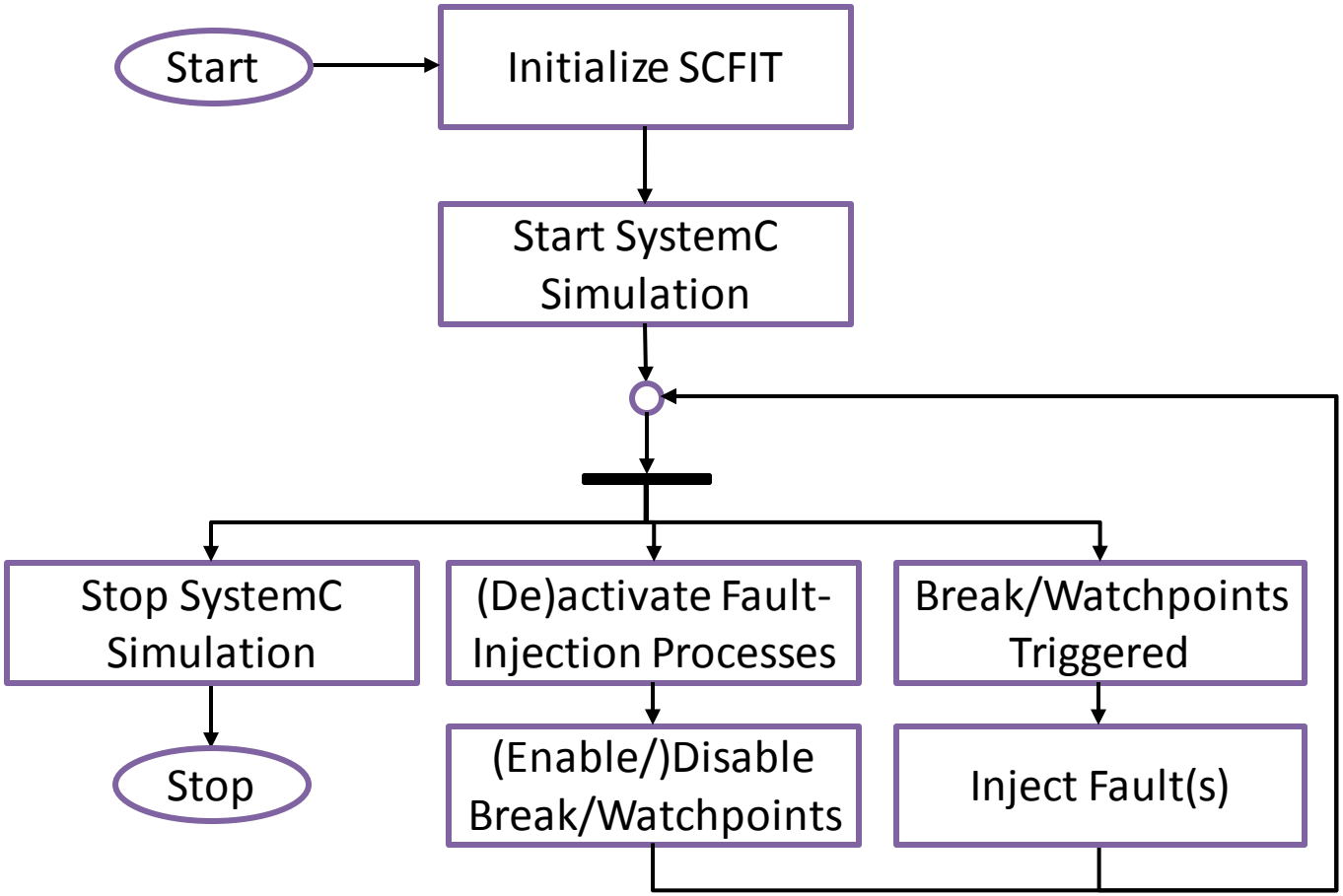
Fault-Injection Requirements

- SystemC models (e.g., TLM)
  - Separate solutions for FW and RTL already exist
  - Unique solution needed
- Non-intrusive simulation based fault injection
- Access private/protected data members
  - Newly introduced by the C++ coding style
- Separate fault behavior description from original behavior in models
  - Commercial simulators solved this problem for RTL code

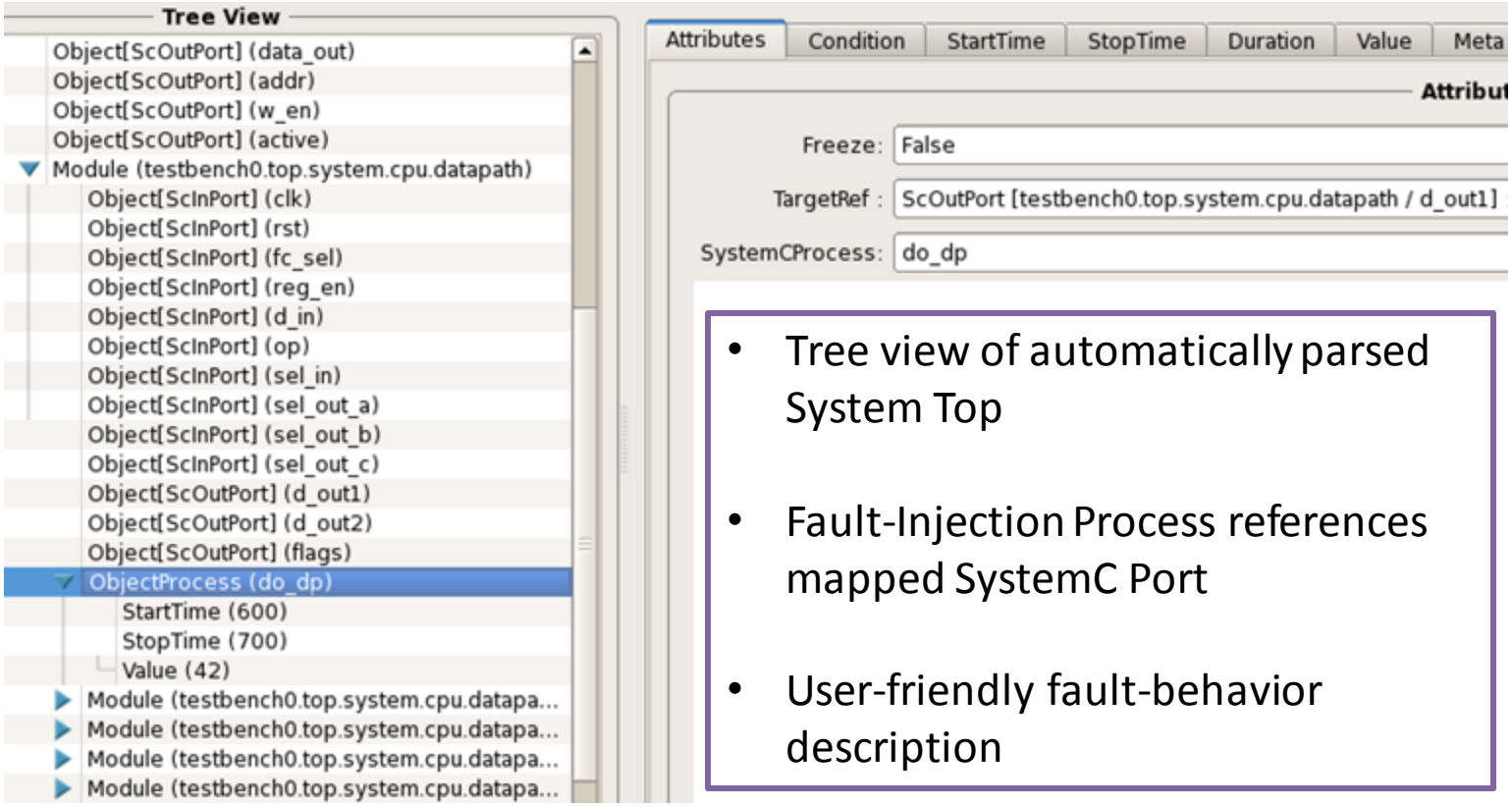
SCFIT Architecture



SCFIT Flow Diagram

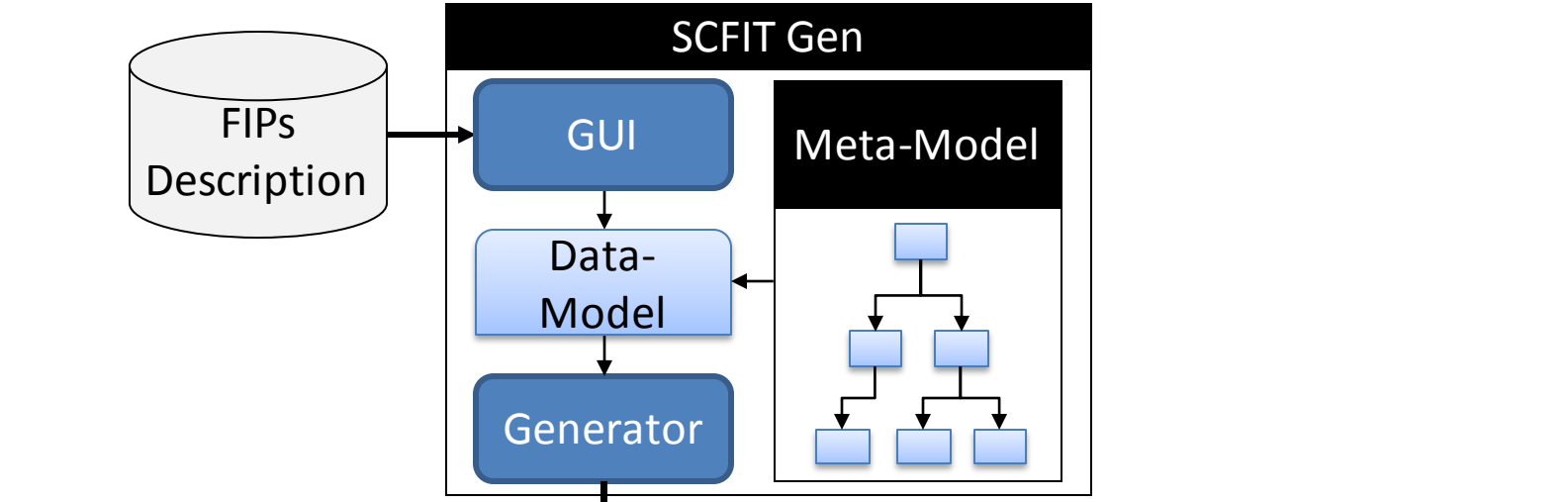


SCFIT Gen GUI



- Tree view of automatically parsed System Top
- Fault-Injection Process references mapped SystemC Port
- User-friendly fault-behavior description

SCFIT Gen Flow Diagram



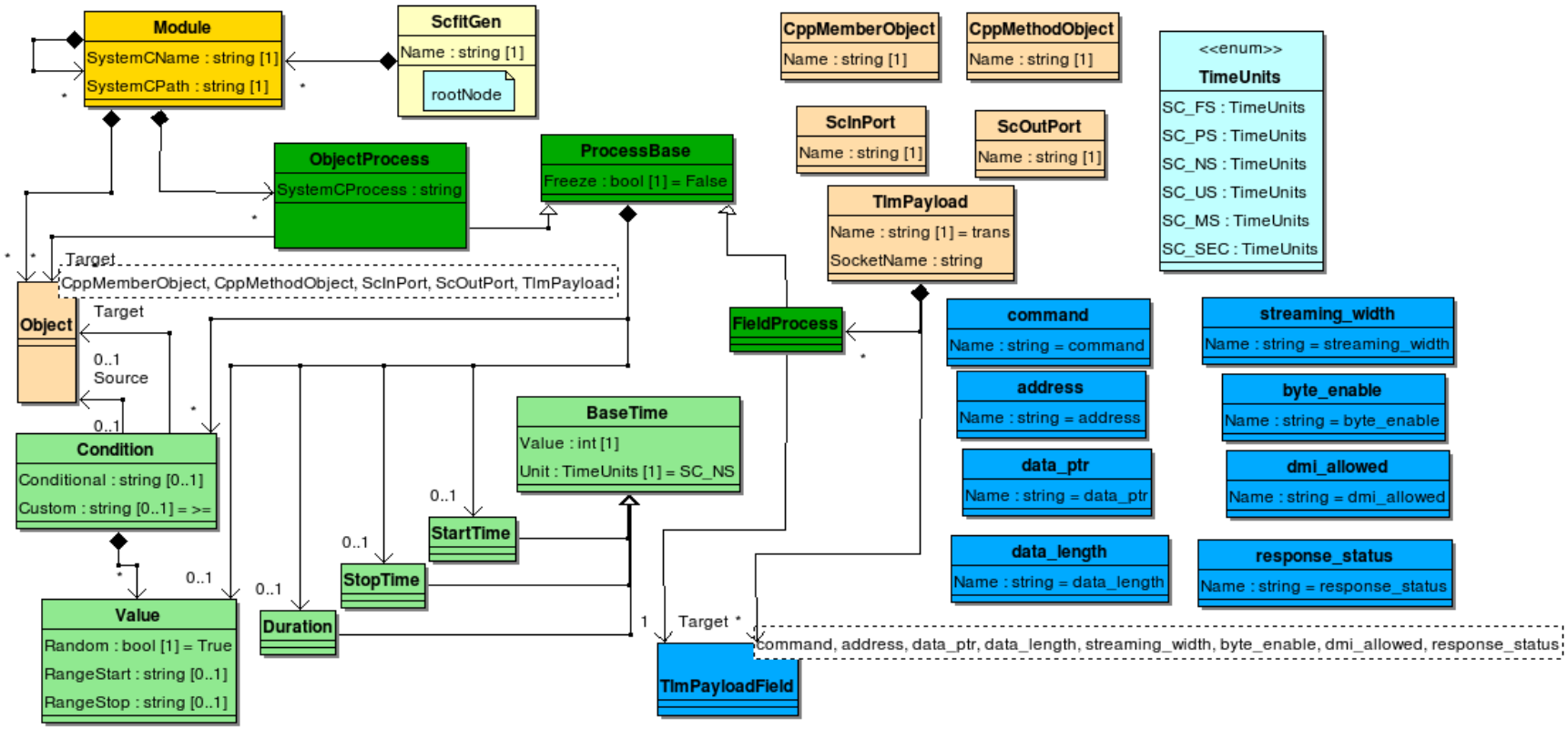
SCFIT Fault-Injection Report

Fault-Injection Documentation for SystemC test-case MaxCPU_base_test

Manipulated Variables:	
	testbench0.top.system.cpu.datapath.d_out1
	testbench0.top.system.cpu.controller.fsm1.d_out_mux

StartTime	StopTime	Object	Type	Conditions	Action	Value
600 SC_NS	700 SC_NS	testbench0.top.system.cpu.datapath.d_out1	ScOutPort		DRIVE	42
400 SC_NS	900 SC_NS	testbench0.top.system.cpu.controller.fsm1.d_out_mux	ScOutPort	d_in_mux == 3	FREEZE	range 0 to 1

SCFIT Gen Meta-Model



Results

- Simplified CPU architecture
- 12 bit wide registers
- Set of 32 instructions
- RTL model
- TLM reference model

SCFIT	# of Faults	Simulation Time (s)	Slowdown (%)
no	-	3.71	-
yes	0	3.77	1.61
yes	1	4.10	10.51
yes	2	4.50	21.29

Discussion I

- User-friendly fault-behavior description
- Measured code reduction by a factor of at least forty
  - User-written code
- Inject faults only in system specific variables
  - e.g., cannot change sc_time variable
- Automatic documentation of fault-injection scenarios per test case

Discussion II

Advantages

- Injection of any number of faults per simulation
- Fast simulation speed
  - For one fault per simulation
- Usable with O3 optimization
- User-friendly fault-behavior description
- Usable in regression runs

Disadvantages

- Dramatic slowdown for 2+ faults per simulation
  - Max 4 hardware supported watchpoints
- Compiler debug-mode enabled in SystemC models
  - Not portable to customers

Next Steps

- Automatic fault-detection and propagation analysis
  - Fault → Error → Failure
  - Simulation report generation
- Windows portability
  - GDB only available under Unix machines
- SCFIT's Simulation Speed Optimization
  - Migrate interpreted Python code to compiled C++ code
- Integrate SCFIT in project flow
  - Test its performance on bigger SoC designs

Questions

Thank you for your attention!

accellera SYSTEMS INITIATIVE™

2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE