How Fast Can You Run SLEC For Verifying Design Optimizations and Bug Fixes

Jin Hou
Francisco Chen
Siemens EDA

Wenli Liang, Lina Guo
Chunlin Wang
Nokia Shanghai Bell Co., Ltd

Agenda

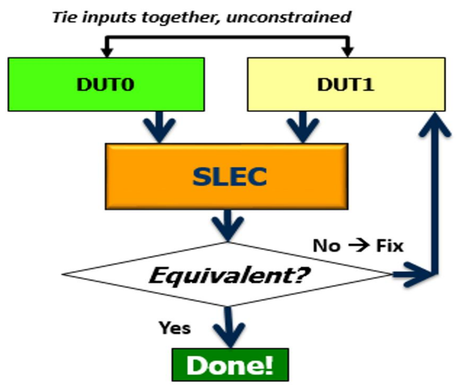
- How to verify design optimization using SLEC
- How to verify bug fixes and ECO using SLEC
- How to verify the functional correctness of bug fixes using SLEC
- When and where to apply SLEC

SLEC Flow For Design Optimization

Inputs:
DUT0 = original RTL
DUT1 = After DUT0 optimization

Desired output:
Does the output of the DUT1 match the original DUT0?

Benefit:
Can eliminate weeks of simulation regressions



Verify Synchronizer Insertion

Verify synchronizer insertion hasn't injected any bug for module "dp_phy_icb"

- Before synchronizer insertion:

```
wire us_kill      = (~icb_ena_traffic & auto_kill_us ) | kill_us;
wire ds_kill      = (~icb_ena_traffic & auto_kill_ds ) | kill_ds;
wire ext_kill     = (~icb_ena_traffic & auto_kill_ext) | kill_ext;
wire traffic_dead = us_kill | ds_kill | ext_kill;

wire  cap_mode          =  debug_con[4] ;
assign  debug_contrl[3:0] =  debug_con[3:0] ;
//---------------------------------------------------------------
// Combinatorial Assignments
//---------------------------------------------------------------
wire  frame_kick = icb_frm_kick_rt[1] & ~ icb_frm_kick_rt[2]; // edge detect
```

After synchronizer insertion:

```
wire us_kill_wire      = (~icb_ena_traffic & auto_kill_us ) | kill_us;
wire ds_kill      = (~icb_ena_traffic & auto_kill_ds ) | kill_ds;
wire ext_kill     = (~icb_ena_traffic & auto_kill_ext) | kill_ext;
wire traffic_dead = us_kill | ds_kill | ext_kill;

wire  cap_mode          =  debug_con[4] ;
assign  debug_contrl[3:0] =  debug_con[3:0] ;
//---------------------------------------------------------------
// Combinatorial Assignments
//---------------------------------------------------------------
wire  frame_kick = icb_frm_kick_rt[1] & ~ icb_frm_kick_rt[2]; // edge detect


always @(posedge clk_ds or negedge areset_ds_)
if (!areset_ds_)
    us_kill <= 1'b0;
else
    us_kill <= us_kill_wire;
```

Script To Run Questa SLEC

Makefile to run Questa SLEC:

```
run: compile_designs run_slec
compile_designs:
       vlib work_orig
       vlog -f filelist_orig -work work_orig
       vlib work_updated
       vlog -f filelist_updated -work work_updated
run_slec:
       qverify -c -od log -do " \
       slec configure -spec -d dut -work work_orig;\
       slec configure -impl -d dut -work work_updated;\
       slec map {spec.us_kill impl.us_kill} -target -impl_latency 1
       slec compile; \
       slec verify; \
       exit"
```

Compile original design

Compile optimized design

SLEC configuration:
Original design is spec
Optimized design is impl

Synchronizer added 1 cycle

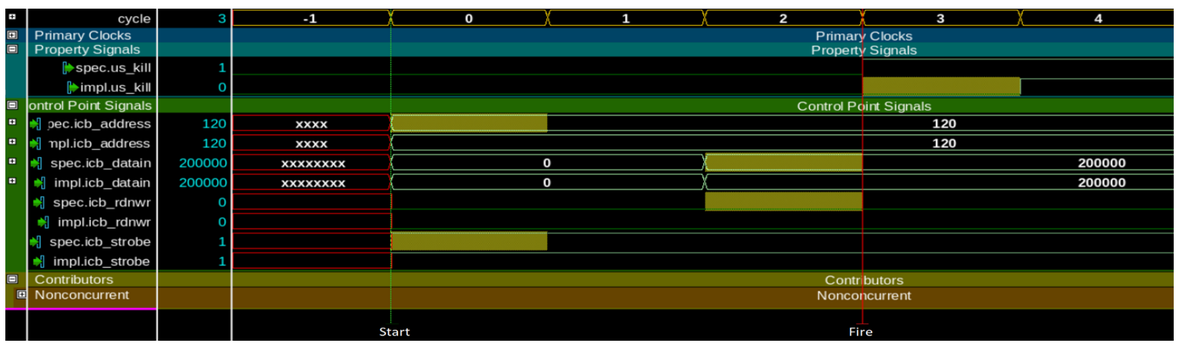
SLEC compile and run

Results

- Took only seconds to finish verifying 40 outputs.

	Name	Radius	Time	Spec Signal	Impl Signal
	SLEC_output_19	6	0s	spec.icb_dataout	impl.icb_dataout
	SLEC_output_20	6	0s	spec.icb_interrupt	impl.icb_interrupt
	SLEC_output_25	3	0s	spec.us_kill	impl.us_kill

- Waveforms of SLEC_output_25



- Defined the delay between spec.us_kill and impl.us_kill, then the tool proved it.

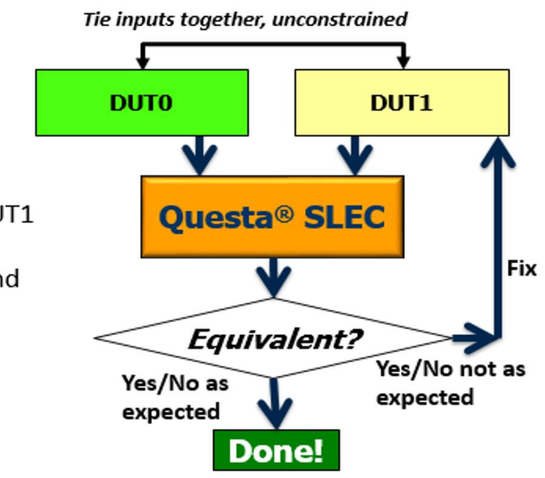
```
slec map {spec.us_kill impl.us_kill} -target -impl_latency 1
```

SLEC Flow For Bug Fixes And ECO

Inputs:
DUT0 = original RTL
DUT1 = DUT0 with ECO or bug fix

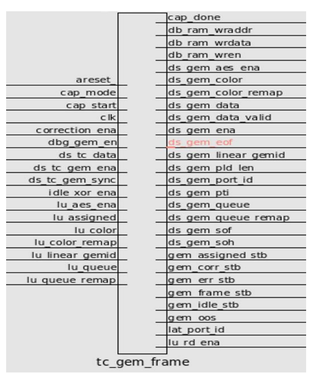
Desired output:
Match/Mismatch = is the behavior of DUT1 different only in the way I expect?
(i.e. did I fix only what had to be fixed and nothing else got broken?)

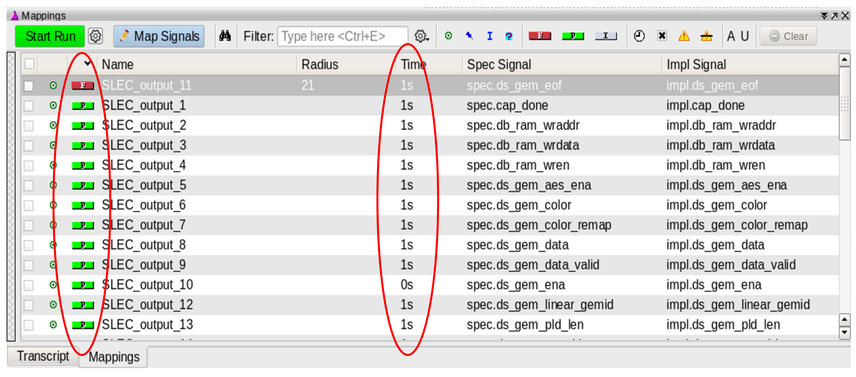
Benefit:
Can eliminate weeks of sim. regressions



Verify Bug Fixes For Module "tc_gem_frame"

- Bug: For a certain type of frames, the design could generate "sof", but didn't generate "eof". Only output "ds_gem_eof" was affected.
- Results: Only seconds to verify bug fix didn't damage good functions.





Verifying The Functional Correctness Of The Bug Fixess

- PropCheck and simulation are the traditional ways to verify functional correctness
- SLEC is to verify equivalence between signal pairs
  - Leverage it to verify functional correctness by providing SVA assumption as golden reference.
    - Cut the incorrect output "ds_gem_eof" in the version before bug fix.
    - Write SVA assumption to constrain free "ds_gem_eof" (golden reference)
    - SLEC to compare the golden reference with the one in the bug fix version.

Makefile

- Cut bad "spec_ds_gem_eof" and constrain it as golden reference
- Compare {spec.ds_gem_eof impl.ds_gem_eof}

```
verify_fix:
     qverify -c -od log_fix -do "\
     do compile_rtl.do;\
     slec configure -spec -d tc_gem_frame -work work_spec; \
     slec configure -impl -d tc_gem_frame -work work_impl; \
     netlist cutpoint spec.ds_gem_eof; \
     vlog -sv properties.sv -mfcu -cuname sva_bind; \
     slec compile -cuname sva_bind; \
     slec unmap -name SLEC_output_*; \
     slec map {spec.ds_gem_eof impl.ds_gem_eof} -target; \
     slec verify; \
     exit"
```

Cut spec.ds_gem_eof

Control spec.ds_gem_eof using SVA assume property

When And Where To Apply SLEC

- Run SLEC when designs are relatively stable
  - Use Lint or AutoCheck tools at early stage of RTL development
  - Use SLEC at late stage of RTL development
    - Design functions are stable
    - Functional verification has been applied since one design version is the reference
- Run SLEC at module level where modifications are made
  - Most efficient and run time is fast.
- Conclusions:
  - Results:

Module	Targets	Proven	Fired	Run time (second)
us_bwc_crc_aid	37	20	17	17
ds_phy_icb	40	37	3	2
tc_gem_frame	27	26	1	1

  - Questa SLEC is easy to setup, run and debug non-equivalence
    - Its automatic mapping and assume-guarantee features make setup and run fast.
  - Save lots of simulation time to rerun tests after design optimization and bug fixes.
    - Simulation time hours vs Questa SLEC seconds in our test cases.