SAFELOGIC CUSTOMER SUCCESS STORY

TRANSWITCH

By utilizing formal verification for tracking bugs in an RTL design, semiconductor company TranSwitch in France minimized its respin costs and improved the design quality. The formal verification tool Safelogic Verifier® not only identified critical bugs, but also verified that the changes made complied with the design's established functional requirements.

Verification challenge

TranSwitch, a leader in semiconductor design, produces an EtherMap suite that is well adapted for effective development of services on SONET/SDH infrastructures. The EtherMap suite consists of three mappers, EtherMap-3, EtherMap-12 and EtherMap-48, which together with the in-house developed framer PHAST, provide a platform for TDM and EoS applications.

EtherMap-3 is a mapper developed in VHDL, which contains more than 7 million gates and consists of a large number of blocks. TranSwitch began verification of the EtherMap-3 mapper over a year ago and the total simulation time corresponded to over 1,000 hours on a 2.5 GHz processor. During the year, several thousands of man-hours were put in to prepare test-benches and code for simulation with ModelSim and Specman. Altogether, 12 parallel licenses were used by each tool during the simulation. The simulation was limited to a number of block tests and only a few stress tests, which entailed that large portions of the block relations were not tested at all. The substantial time spent on preparing test-benches and simulating, and the already tight schedule for the project, meant that verification tasks had to be concentrated to critical functionalities. The problems revealed during verification were corrected and EtherMap-3 was sent on to synthesis and implementation on silicon.

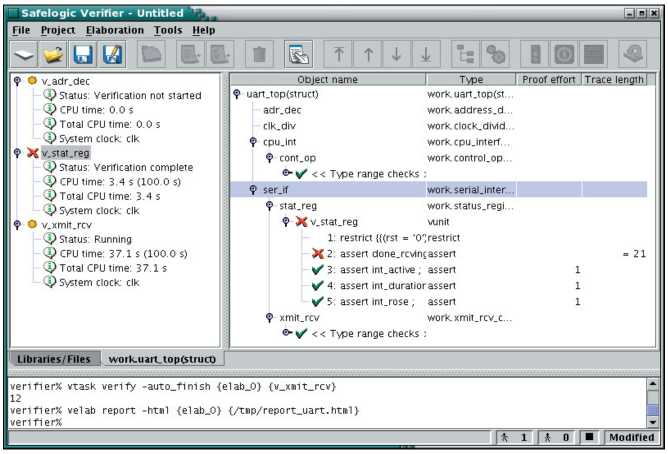
Case highlights - TranSwitch

- VHDL design of an Ethernet Mapper, 7 million gates
- Uncovered 4 critical bugs in silicon prototype
- Wrote properties in PSL and introduced formal property checking
- Identified and fixed bugs in 1 day, saved several man-months of work

Introduction of formal verification – Safelogic Verifier

When the completed silicon prototype was subsequently system-tested in the lab, four critical bugs were discovered. These bugs were never encountered during the traditional simulation process and thus went undetected to silicon. According to TranSwitch, it would have taken the efforts of four persons for one month to localize and correct the problems discovered on the silicon prototype in a traditional verification environment. It would also have required considerable resources in the form of processor power and license utilization. It is not uncommon that design engineers must rewrite large portions of the simulation environment to be able to recreate (if possible) the bug and the corresponding regression environment.

In conjunction with the discovery of the remaining bugs, Safelogic Verifier® was introduced – a formal static property checker – into the verification process. Safelogic Verifier is a tool that focuses on verifying design blocks expressed in VHDL or Verilog, and properties written in the standardized property language PSL. Safelogic Verifier ensures that the RTL design complies with the expressed properties, alternatively provides concrete counterexamples when the design conflicts with these properties. A counterexample consists of the shortest path from the verification's start state to the state where the bug appears. As Safelogic Verifier can generate a test-bench for triggering the bug at a later time, the user can simulate and use other debugging tools to more easily understand the bug in the design.



```
v_adr_dec
    Status: Verification not started
    CPU time: 0.0 s
    Total CPU time: 0.0 s
    System clock: clk
v_stat_reg
    Status: Verification complete
    CPU time: 3.4 s (100.0 s)
    Total CPU time: 3.4 s
    System clock: clk
v_xmit_rcv
    Status: Running
    CPU time: 37.1 s (100.0 s)
    Total CPU time: 37.1 s
    System clock: clk
```

Object name	Type	Proof effort	Trace length
uart_top(struct)	work.uart_top(st...		
adr_dec	work.address_d...		
clk_div	work.clock_divid...		
cpu_int	work.cpu_interf...		
cont_op	work.control_op...		
✓ << Type range checks :			
ser_if	work.serial_inter...		
stat_reg	work.status_regi...		
✗ v_stat_reg	vunit		
1: restrict {((rst = '0';	restrict		
✗ 2: assert done_rcvinç	assert		= 21
✓ 3: assert int_active ;	assert	1	
✓ 4: assert int_duratior	assert	1	
✓ 5: assert int_rose ;	assert	1	
xmit_rcv	work.xmit_rcv_c...		
✓ << Type range checks :			

```
verifier% vtask verify -auto_finish {elab_0} {v_xmit_rcv}
12
verifier% velab report -html {elab_0} {/tmp/report_uart.html}
verifier%
```

Graphical user interface of formal property checker Safelogic Verifier. For batch mode regressions, a TCL command interface is available.

Verification results

The verification team at TranSwitch was able to quickly set up formal properties in PSL for the design functionality that were related to the areas from which the bugs were believed to originate. An initial running of the RTL design together with the properties in Safelogic Verifier later showed that several of the properties were fulfilled and proofs for those properties were provided immediately. Safelogic Verifier was able to formally verify that the properties were fulfilled in all applicable design states. With this knowledge, continued verification could be focused on other properties and no additional costly verification time needed to be invested in verifying what was already proven.

For the four known bugs, Verifier could provide counterexamples that demonstrated how the design conflicted with the established properties. The counterexamples consisted of waveform files, similar to those generated in simulation, and showed the shortest path for recreating the bug from the start state. The waveform files also contained detailed information on relevant signal values, and with their help, it was easy for the engineers to localize critical blocks and recreate the bugs. In less than a day, the team was able to isolate and resolve the problems, and in less than a minute, verify that the bugs were corrected. TranSwitch assesses that formal verification saved several man-months compared with localizing and correcting the bugs in a traditional simulation environment. Seen from a broader verification perspective – during a larger project, for example – this means dramatic savings in verification time.

Gains in time and quality

The experiences from the EtherMap-3 project demonstrated how simple a traditional development and verification environment can be strengthened with the help of PSL properties and formal verification. It also demonstrated the substantial gains in time and expense that could be attained from formalizing design properties. TranSwitch could easily adapt the new technology, and time and quality gains were apparent even for the type of selective measure that in this case was conducted after completion of the simulation process. If the property writing is initiated earlier in the process, preferably before VHDL development has started, there are obviously additional time and quality gains to be made. In the case of TranSwitch, one silicon prototype might have been possible to avoid, and thereby it would have been possible to keep down the NRE costs. Verifying properties early at the block level not only saves major resources in the form of simulation time and costly physical implementation, but also permits interaction between those who implement and specify the design at an early stage.

TranSwitch has now introduced property-based, formal verification throughout the main process, from system properties at a high level to properties on the block level, and is applying the technology in bug detection and bug correction. New versions of EtherMap-3, EtherMap-12 and EtherMap-48 are now fully based on formal verification, resulting not only in improved quality of shipped products but also in quicker response time to market demands.

```
vunit RAM_RESET_FULLFLUSH(ETRC(rtl)) {
    default clock is SYSCLK = ´1´ and SYSCLK’event;
    restrict {not RST_SYSCLK; RST_SYSCLK[*]};

    -- Property to assure that RAM reset is respected
    property RAMRESETMGT is
          always {SHTC_RAM_RST(0) = ´1´; SHTC_RAM_RST(0) = ´0´}
                 |=> {[*1]; ETRC_PkDTWrPt(0) = 0 and ETRC_L2TWrPt(0) = 0};

    assume always cTRSTRAM = ´1´;
    assert RAMRESETMGT;
}
```

A PSL verification unit written by TranSwitch for the EtherMap-3 design. The unit contains a verification environment restriction, a signal assumption, and a property that defines a RAM reset sequence that needs to be respected by the design.

SAFELOGIC

Property based verification

Safelogic is an EDA company developing ground-breaking tools for improved simulation, analysis and verification of RTL designs. Products include Safelogic Monitor® - plug-in for property simulation, Safelogic Verifier® - formal property checker, and Safelogic ASG® – automatic stimuli generator. For more information visit www.safelogic.se.