Single Event Upsets Perl Tool (SPT)

Daniel González Gutiérrez TOS-EDM
Single Event Upsets Perl Tool

Objective

To Develop a tool that is able to emulate SEUs, easily and in a useful controlled manner, while still in the simulation (HDL) stages of the IC design flow:

– With independence of the particular design.

– Non intrusive (no vhdl code modification)
How can it be achieved? (our approach)

• Using a language good for file processing, parsing and pattern matching, that will allow us to locate the bits to be ‘Upset’: Perl.

• Using the ‘force’ simulator command included in Modelsim
Functional Overview

• The tool consists of a set of perl and tcl scripts used to prepare the environment to be able to upset (bit flip) in a controlled and effective manner, any register or internal signal of the design under test, while a simulation is running.
Running the tool

- Load the Test Bench of the DUT in the simulator
- Gather information about the design (SPT_startup_macro.do)
- Select the wires that are going to be upset and when (SPT_check_and_upset.pl)
- Run a simulation introducing the upsets (spt.do)
Block diagram (II)

SPT2

SPT_startup_macro.do
- Wire files
- all_instances.dat

SPT_check_and_upset.pl
- Spt.do

Command line interface options

Spt.do
- Upset.pl

MASK

Single Event Upsets Perl Tool

Daniel González Gutiérrez
Microelectronics Presentation Days
ESTEC 4-5 February 2004  slide: 7
SPT_startup_macro.do

- It is used to gather information about the design and should be run only once.
- It sets up the environment for the SPT_check_and_upset.pl script.
- It calls SPT_list_instances.pl and SPT_wires_parser.pl
This perl script is the core of the SEUs Perl Tool since the selection of wires to be upset is done here.

- How many wires do we want to upset?
- Is there a particular location on our design that we want to test?
- When do we want to upset the wires?
- Is there a particular pattern we can use to filter the search?
SPT_check_and_upset.pl

- The selection of wires is done via some command line interface options and switches
SPT_check_and_upset options and switches (-m)

Manual option.

- Both \textit{all_instances.dat} and the wire files have to be edited manually by the user of the script, in order to select the desired wires.
- This option excludes the rest of the command line interface options.
SPT_check_and_upset options and switches (-i)

**Instances option.** This option is used to specify in what instances, from the ones that can be found in `all_instances.dat`, we want to induce the upsets. It has three switches:

- `r` (read) -> The user has selected the instances by manually editing `all_instances.dat`.
- `f` (filter) -> The selection of instances will be done by filtering their names using patterns introduced via the command line interface.
- A number -> that specifies the amount of instances to be selected.

C:\test>SPT_check_and_upset -i filter*_fsm
**SPT_check_and_upset options and switches (-n)**

**Number option.** - This option is used to set which wires will be upset. It has 3 switches:

- **n(ot_fixed)** - The number of wires is calculated.
- **f(ilter)** - The selection of wires will be done by filtering their names using patterns introduced via the command line interface.
- **A number** - that specifies the amount of wires to be selected.

```bash
C:\test >SPT_check_and_upset -i read -n filter 1 _reg
```
**SPT_dump_setup options and switches (-t)**

**Time option** - This option is used to determine in which time window (within the test bench simulation limits) the upsets will be a time window, the script will randomly set a time value between the limits specified, for each signal to be upset.

```
C:\test>SPT_check_and_upset -i read -n filter 1 _reg -t 200us 20ns
```
SPT_dump_setup options and switches (-h)

- **Help option**. This option displays a help message with information about the use of the script and some examples.
This script is written in tcl and consists of a set of simulator commands and a perl function call, which will automatically run a simulation introducing the upsets. The script `SPT_check_and_upset.pl` generates it automatically.
The simulation is run up to the first scheduled upset.

The value of the wire is checked.

If that value is not undefined, it is upset using the perl script `upset.pl` and the `Modelsim` command `force –deposit`.

The simulation is run up to when the next upset is scheduled.

The value of the wire is checked and upset…

When the last upset is done, wait until the simulation ends.
**Single Event Upsets Perl Tool**

Load your design in Modelsim

Run SPT startup macro

Run SPT_check_and_upset perl script in a command prompt window

Run spt.do macro

```
Modelsim> vcom -93 -work my_lib my_design.vhd
Modelsim> vsim my_lib.test_bench
Modelsim> do SPT_startup_macro.do
C:\test>SPT_check_and_upset –iread –nfilter _reg –t20us 2us
Modelsim> do spt.do
```
We have a simulation with SEUS induced, but we still need to have a VHDL test bench to check if those SEUs were translated into actual errors.
Test example

SPT_startup_macro.do ➔ Selecting outputs

→

Selecting one output per run

Filtering _reg instances and q0 outputs.

Run spt.do macro

test_macro.do
Next Steps?

- Support other simulators
- ‘Third party’ testing (independent verification of the tool)
- Estimate the speed of the tool with more examples. Make stats.
- Inputs are welcomed.