Universidad
Antonio de Nebrija

Single Event Upsets Simulation Tool (SST) User Manual

Computer Architecture and Technology Group
Universidad Antonio de Nebrija

Ref.: TR-2006Nov-001

Madrid, November 3, 2006

Universidad
Antonio de Nebrija

Table of Contents

Universidad
Antonio de Nebrija

1. SCOPE

The purpose of this document is to describe how the new release of the Single Event Upsets Simulation Tool (SST) shall be used with the most recent updates and upgrade works.

2. TERMS AND ACRONYMS

DUT	Design Under Test
GUI	Graphical User Interface
SEU	Single Event Upset
SET	Single Event Transient
SST	Single Event Upset Simulation Tool

3. OVERVIEW

The SEUs Simulation Tool consists of a set of Perl and Tcl scripts that used in conjunction with a Design Under Test and a Test Bench, allows the user to upset (bit flip) any register or internal signal in a controlled and effective way. The scripts can be invoked using the Graphical User Interface provided with the tool.
The current version of the SST works with Modelsim 6.1b. Any simulator supporting the 'force' command could be easily added in the future.

For any comment regarding this document (question, suggestion, mistake detection …), please contact Oscar Ruano at oruano@nebrija.es

4. INSTALLATION

Prior to copying the SST files into your computer, make sure that your system has Perl installed. It can be downloaded for free from: http://www.perl.com/download.csp.

The following two actions have to be performed before running the tool:

- Extract all the SST files into the same directory installation, for example: "*C:/SST61*"

- The folder in which the scripts are finally placed, should be added both to your path environment variable and to the DOPATH environment variable (Figures 1, 2).

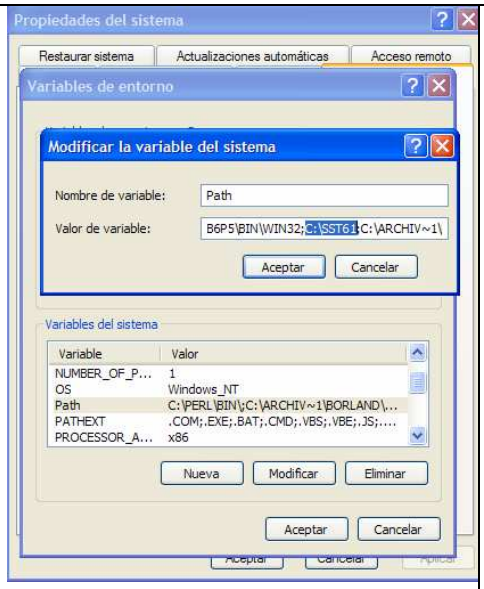


Figure 1

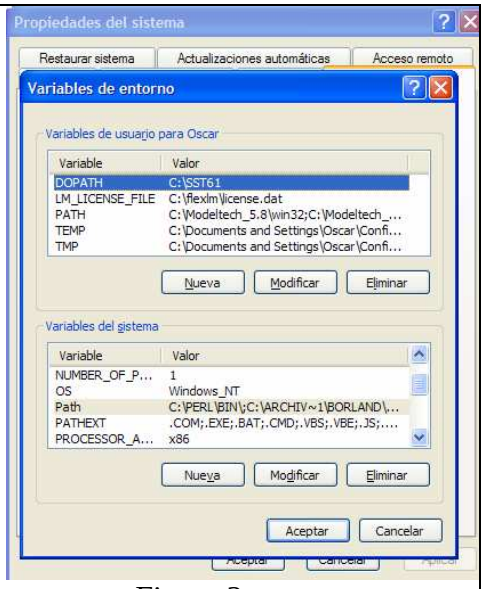


Figure 2

5. DIRETORY STRUCTURE

a. Files supplied by the user

./HDL testbench_files

b. Files generated by the tool

./SST/control_files

all_instances.dat, all_wires_parser.log, hierarchy.dat, sst.do, SST_perl_package.pm

./SST/wire_files

In this folder we can find all the wire files.

6. RUNNING THE TOOL

To invoke the GUI, type: ***"do SST_gui.tcL"*** in the Modelsim command line interface. The following window appears:

Figure 3: Graphical User Interface main window

6.1. Choose Version

The main idea of this option consists in integrating both versions into the same package:

Compatible with ModelSim 6.1
Compatible Modelsim 5.8

This allows the user changing the version depending on the simulator release.
In this document, we will only refer to the last update performed around version 6.1. For the version compatible with Modelsim 5.8, please consult the related documents.

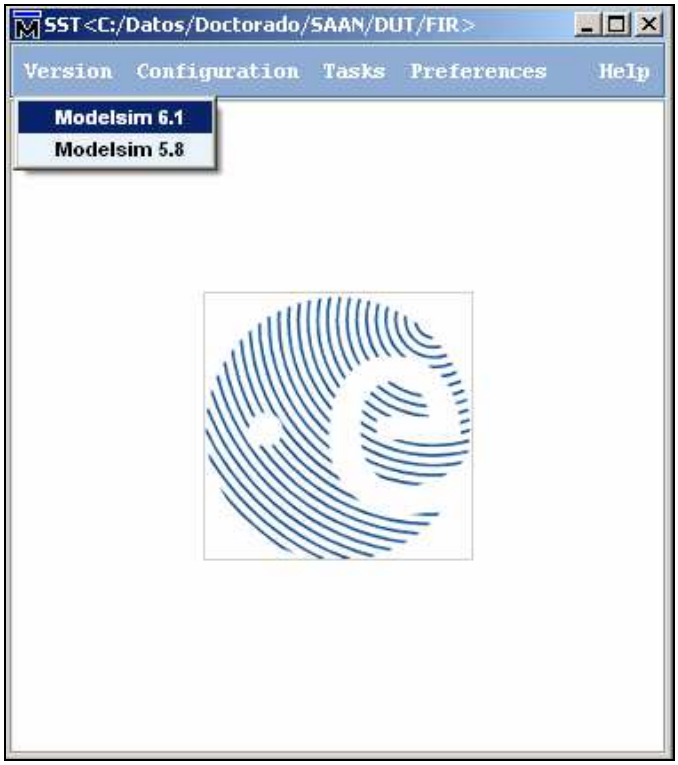

SST<C:/Datos/Doctorado/SAAN/DUT/FIR>
Version
Configuration
Tasks
Preferences
Help
Modelsim 6.1
Modelsim 5.8

Figure 4

6.2. Setup Some Parameters

This window (Figure 5) is used to configure some parameters via GUI that they are needed in order to setup the tool. For instance, you can modify the type of signals filtered by the SST with the "mask" parameter. Another example of this is the smallest time distance between two SEUs or SETs filling the "reference step".

Figure 5

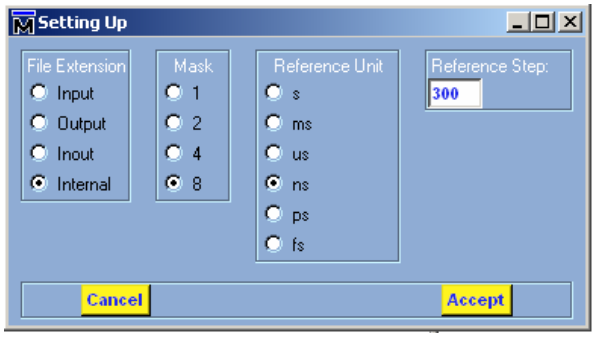


Figure 6

For more information about these parameters or switches use the following option (Figure 7):

Figure 7

and it will show a brief description for these parameters on the Modelsim Transcript screen (Figure 8):

TRANSCRIPT WINDOW:

```
# HELP OPTION
# ----------------------------SETTING UP--------------------------------------------
# wire_files_ext
# This variable holds the name of the wire files extension. The type of wire
# is selected using wire_mask. Use a meaningfull extension for each type
# of selected wire (ie .out if the outputs were selected)
#-------------------------------------------------------------------------
# wire_mask
# This variable holds the decimal value of the mask used to select the type
# of wire we are interested in.
# Please update the following binary layout (used to obtain the decimal number
# you need) every time you change the value of the mask.
#
# Mask
#  MSB                                LSB
# <internal          inouts    outputs   inputs>
#  0                 0         1         0
#-------------------------------------------------------------------------
#-------------------------------------------------------------------------
# reference_unit
# This variable holds the unit used as a reference for the creation of upsets.
# This reference value will be used when one of the input units of the -t
# switch of the SST_upset_generator script is missing.
# The value of this variable should be the same as the VHDL simulation step
# (resolution of our simulation).
# Forces with a smaller unit will not be allowed.
...
…
…
```

Figure 8

6.3. Load the Test Bench of the DUT in the simulator

Prior to start executing the SST, it is necessary to load the test bench that checks the correct functionality of the DUT into Modelsim 6.1.

6.4. Gather the information about the design

In order to let the SST know about the design to be tested (basically, its hierarchy and the number and type of signals found in each module), the user will have to click in the 'Tasks/Load Design' menu button of the GUI (Figure 9).

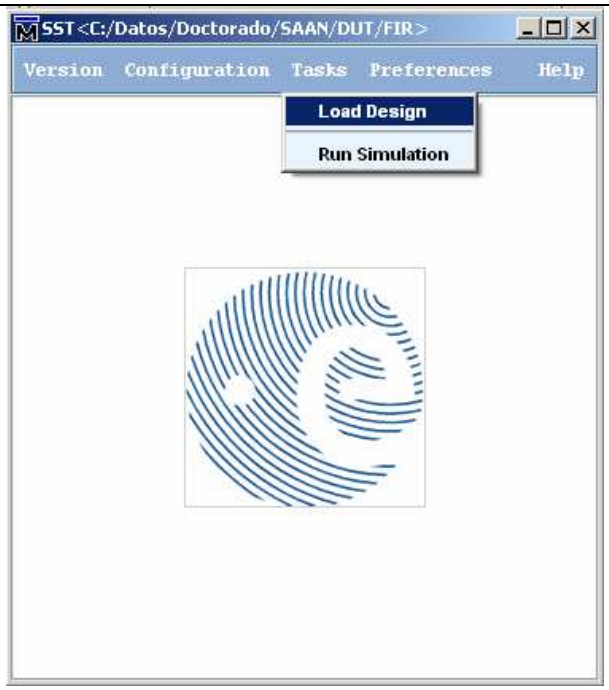


Figure 9

It is in this moment when the structure mentioned before ("Files generated by the tool") is created:

SST/wire_files
SST/control_files

6.5. Select the wires that will be upset

Once the information about the design has been collected, the selection of the wires that are going to be upset has to be done by clicking in the 'Configuration/SEU Configuration' GUI menu button, and filling the entry boxes and check buttons of the interface window. Figure 10 shows the 'SEU Configuration' window.

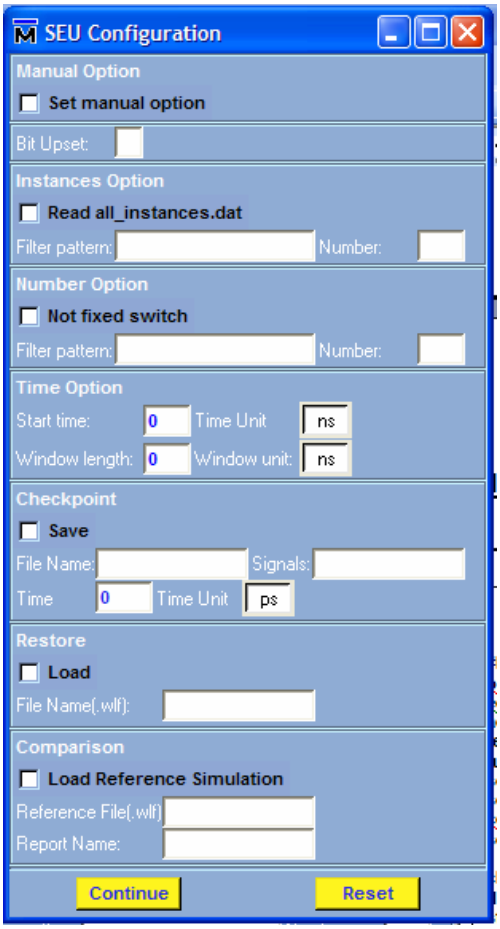


Figure 10

Bit Upset. This option indicates the bit number into a bus signal. If you do not fill it, the upset will be inserted randomly.

Manual option. The script just reads '*all_instances.dat*' and the Input files (which have been edited by the user) and prepares the evaluation of the selected signals at the specified time values. This option excludes the rest of fault injection: '*Instances Option*' and '*Number Options*'.

Instances option. This option is used to set the amount of instances to be selected in *all_instances.dat*. It has 3 switches:

- Read: The user has selected the instances by manually editing the file *all_instances.dat*. The script will read the file as an input
- Filter: The selection of instances will be done by filtering their names using patterns introduced via the command line interface. The patterns should be "perl-like" regular expressions.
- Number: It is the number of instances to be selected.

Number option. These options are used to set the amount of signals that will be evaluated. It has 3 switches:

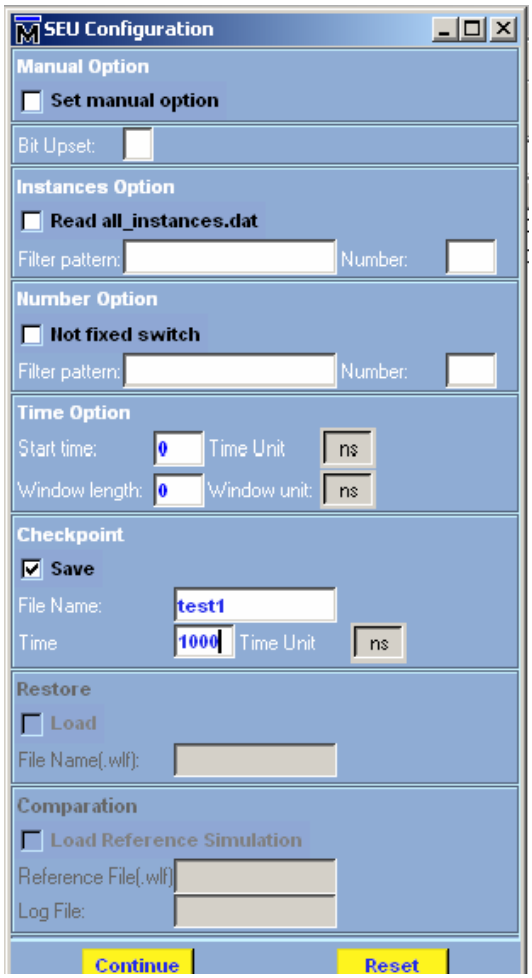
- Not_fixed: The number of inputs and signals is calculated using a parameter defined in SST_config.tcl
- Filter: The selection of signals will be done by filtering their names using patterns introduced via the command line interface. The patterns should be "perl-like" regular expressions.
- Number: It is the number of signals to be evaluated.

Time option. This option is used to schedule the evaluation process. Given a starting time value (zero if omitted) and a simulation window, the script will randomly set a time value between the specified limits, for each forced signal.

Respect to the other tree options (Checkpoint, Restore, Comparison), they are functionalities that have been upgraded to add the simulation options from the SST. The multiple combinations of these options with the injection campaigns show a widely range of utilities for the designer.

In the next pages, we will show some examples of these operations.

Save a simulation without SEUs or SETs: GOLDEN SIMULATION

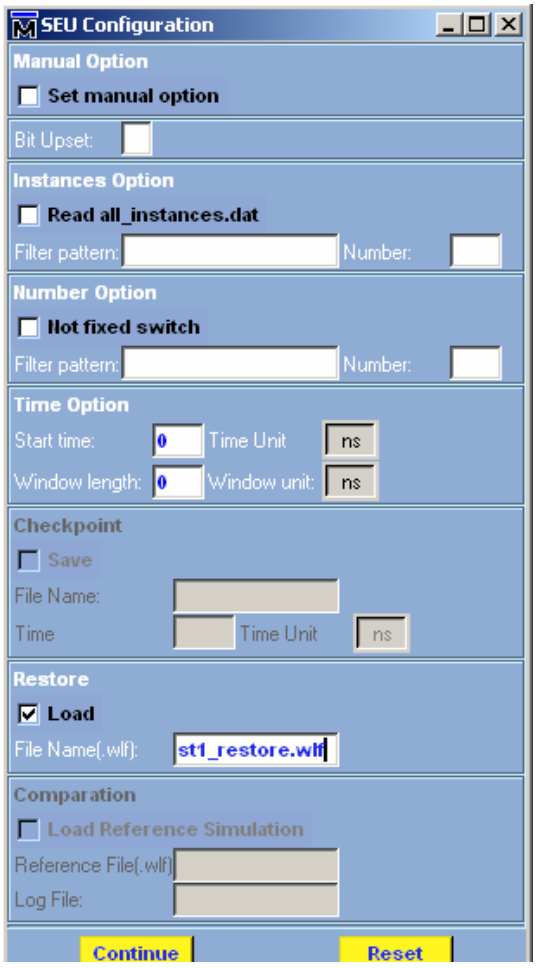


SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Starting GOLDEN simulation...}
add wave sim:/firtb/dut/*
restart -f
run @1000ns
set var [list compare.wlf restore.wlf]
set file_name_compare [join "test1 compare.wlf" _ ]
set file_name_restore [join "test1 restore.wlf" _ ]
checkpoint $file_name_restore
dataset save sim $file_name_compare
status
```

This golden simulation record 1000 ns for example to make a comparison later

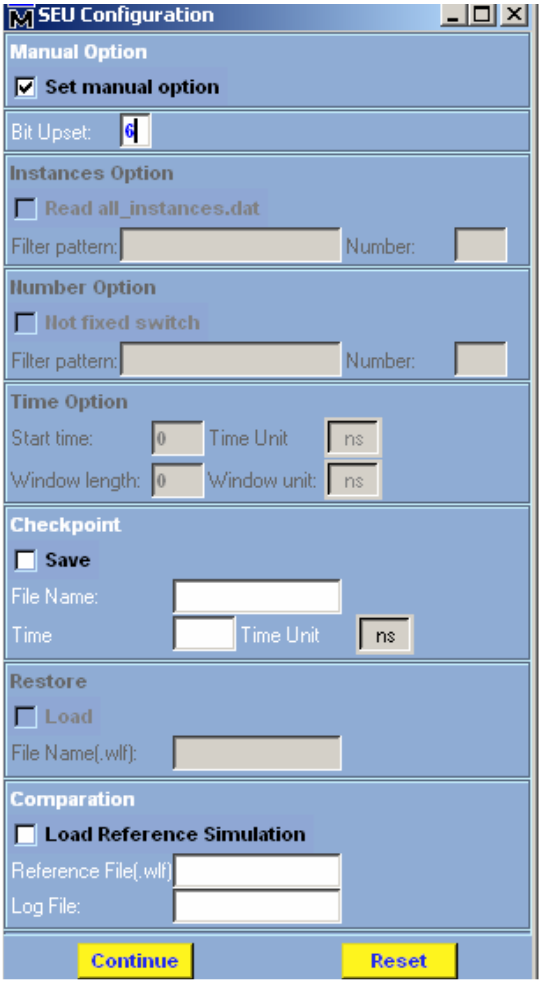
Restore a previous saved simulation, ready to inject SEUs or simply run more cycles



SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Reloaded a Simulation from Initial State...}
vsim -restore $file_name_restore
status
```

Inject SEUs from a previous RESTORED simulation: MANUAL



Times for SEUs are relative to the initial state.
For instance, 1000 reg6 and 2000 reg4 ns respect to the simulation restored, represent a SEU in 2000ns for reg6 and 3000 ns for reg4 in absolute times

Correct the Transcript message:
/firtb/dut/reg(6)
 # 01110001
 # 01110101
 # Forcing /firtb/dut/reg(6) to 01110101 @ 1000ns → 2000ns
 # /firtb/dut/reg(4)
 # 11110001
 # 11110101
 # Forcing /firtb/dut/reg(4) to 11110101 @ 2000ns →3000ns

¡¡¡¡Modified: all times are absolutes respect the current state !!!!

all_instances.dat

Force File name # wires
Instance full path

No firtb 4
/firtb
Yes dut 18
/firtb/dut

dut.internal

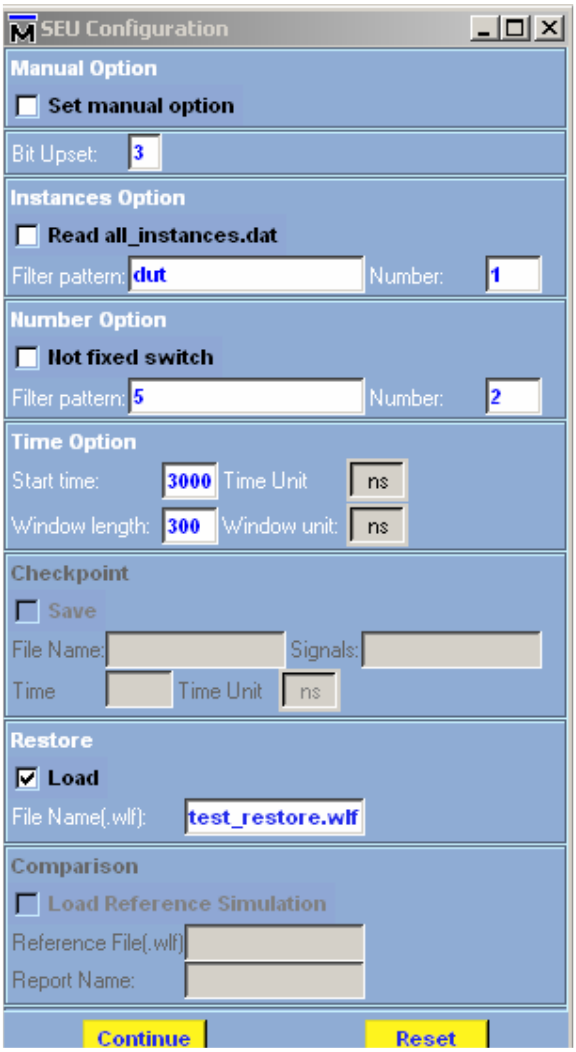
Force	Name	@Time
No	reg_salida	0ns
No	reg_salida_next	0ns
No	reg_next(1)	0ns
No	reg_next(2)	0ns
No	reg_next(3)	0ns
No	reg_next(4)	0ns
No	reg_next(5)	0ns
No	reg_next(6)	0ns
No	reg_next(7)	0ns
No	reg_next(8)	0ns
No	reg(1)	0ns
No	reg(2)	0ns
No	reg(3)	0ns
Yes	reg(4)	2000ns
No	reg(5)	0ns
Yes	reg(6)	1000ns
No	reg(7)	0ns
No	reg(8)	0ns

SST.DO

```
#Macro generated by SST_upset_generator.pl
#2 wires forced
onbreak {abort 1}
echo {Starting SEU simulation...}

run 1000 ns
# define signal path
set wire /firtb/dut/reg(6)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 1000ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
run 1000 ns
# define signal path
set wire /firtb/dut/reg(4)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2000ns
```

	force -deposit $wire $wire_upset_val **} else {** **echo Unable to upset $wire. Undef value.** **}** **run -all**

Inject SEUs from a RESTORED simulation: RANDOM

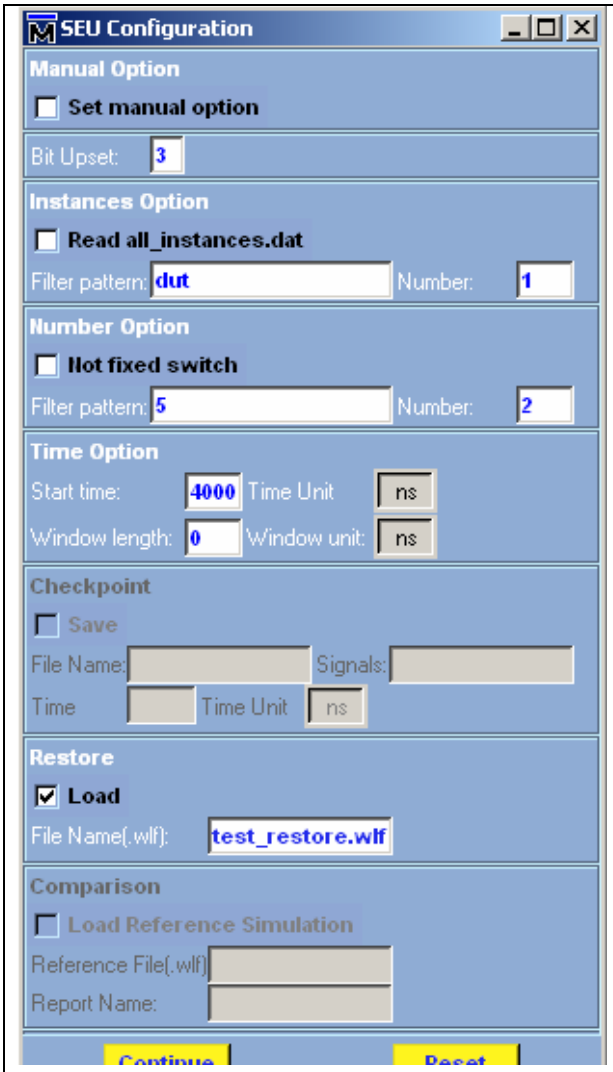


SEUs at different time slots

Force	Name	@Time
No	reg(1)	0ns
No	reg(2)	0ns
No	reg(3)	0ns
No	reg(4)	0ns
Yes	reg(5)	3.125us
No	reg(6)	0ns
No	reg(7)	0ns
No	reg(8)	0ns
No	reg_next(1)	0ns
No	reg_next(2)	0ns
No	reg_next(3)	0ns
No	reg_next(4)	0ns
Yes	reg_next(5)	3.03us
No	reg_next(6)	0ns
No	reg_next(7)	0ns
No	reg_next(8)	0ns
No	reg_salida	0ns
No	reg_salida_next	0ns

SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Reloaded a Simulation from Initial State...}
noview wave
vsim -restore $file_name_restore
#2 wires forced
onbreak {abort 1}
echo {Starting SEU simulation...}
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
if {$sample > 3030} {
echo {ERROR: SEU is inserted in a past time: 3030
}
} else {
# 3030 $sample
set run [expr 3030 - $sample]
run $run ns
# define signal path
set wire /firtb/dut/reg_next(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 3030ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
            }
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
if {$sample > 3125} {
echo {ERROR: SEU is inserted in a past time: 3125
}
} else {
# 3125 $sample
set run [expr 3125 - $sample]
run $run ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
```



SEUs in the same slots of time

Yes reg_next(5) 4us 4us

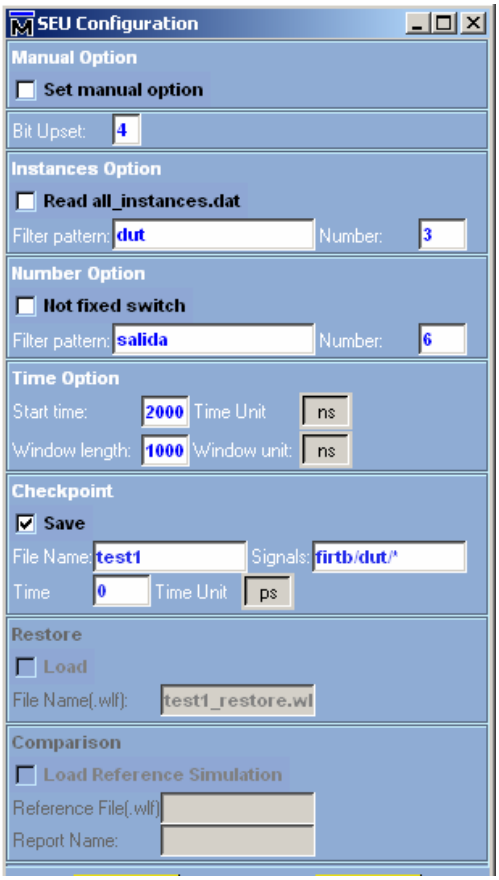
Both SEUs are considered like one

```
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 3125ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
```

SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Reloaded a Simulation from Initial State...}
noview wave
vsim -restore $file_name_restore
#2 wires forced

onbreak {abort 1}
echo {Starting SEU simulation...}
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
if {$sample > 4000} {
echo {ERROR: SEU is inserted in a past time: 4000
}
} else {
# 4000 $sample
set run [expr 4000 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg_next(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 4000ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
# define signal path
set wire /firtb/dut/reg_next(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 4000ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
            }
run -all
status
```

Inject SEUs from an initial state and save the simulation

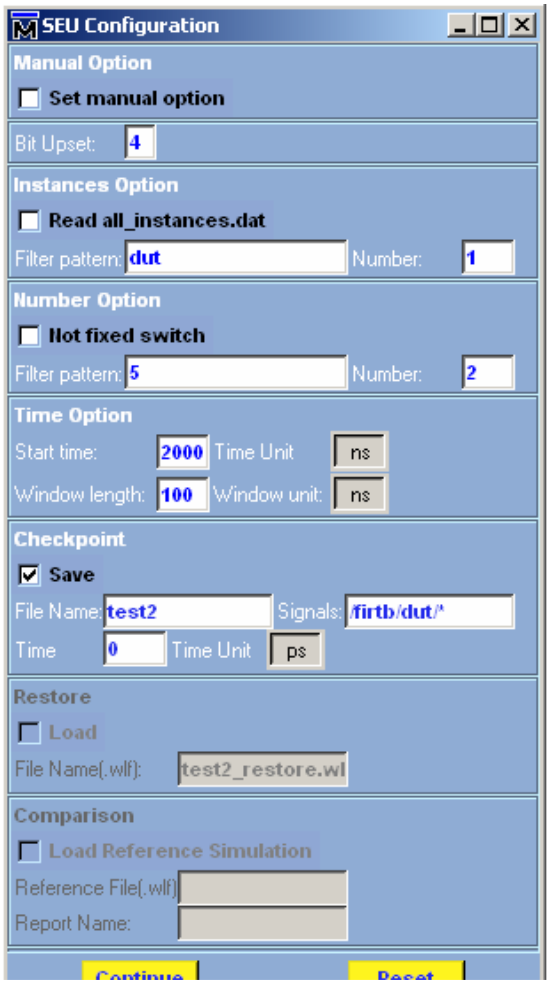


SST.DO

```
#Macro generated by SST_upset_generator.pl
#6 wires forced

onbreak {abort 1}
echo {Starting SEU simulation plus checkpoint...}
noview wave
add wave sim:$signals

run 1120 ns
# define signal path
set wire /firtb/dut/reg_salida
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2180ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 105 ns
# define signal path
set wire /firtb/dut/reg_salida
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2285ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 295 ns
# define signal path
set wire /firtb/dut/reg_salida
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2580ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 15 ns
# define signal path
set wire /firtb/dut/reg_salida
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2595ns
```

```
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 75 ns
# define signal path
set wire /firtb/dut/reg_salida_next
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2670ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 300 ns
# define signal path
set wire /firtb/dut/reg_salida_next
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire to $wire_upset_val @ 2970ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
if {$time_checkpoint == 0} {
set var [list compare.wlf restore.wlf]
set file_name_compare [join "$file_name_checkpoint
compare.wlf" _ ]
set file_name_restore [join "$file_name_checkpoint
        restore.wlf" _ ]
checkpoint $file_name_restore
dataset save sim $file_name_compare
status
} else {run @$time_checkpoint$t_u
set var [list compare.wlf restore.wlf]
set file_name_compare [join "$file_name_checkpoint
compare.wlf" _ ]
set file_name_restore [join "$file_name_checkpoint]
        restore.wlf" _ ]
checkpoint $file_name_restore
dataset save sim $file_name_compare
status}
```

Fault-Injection Campaign at unreachable past time

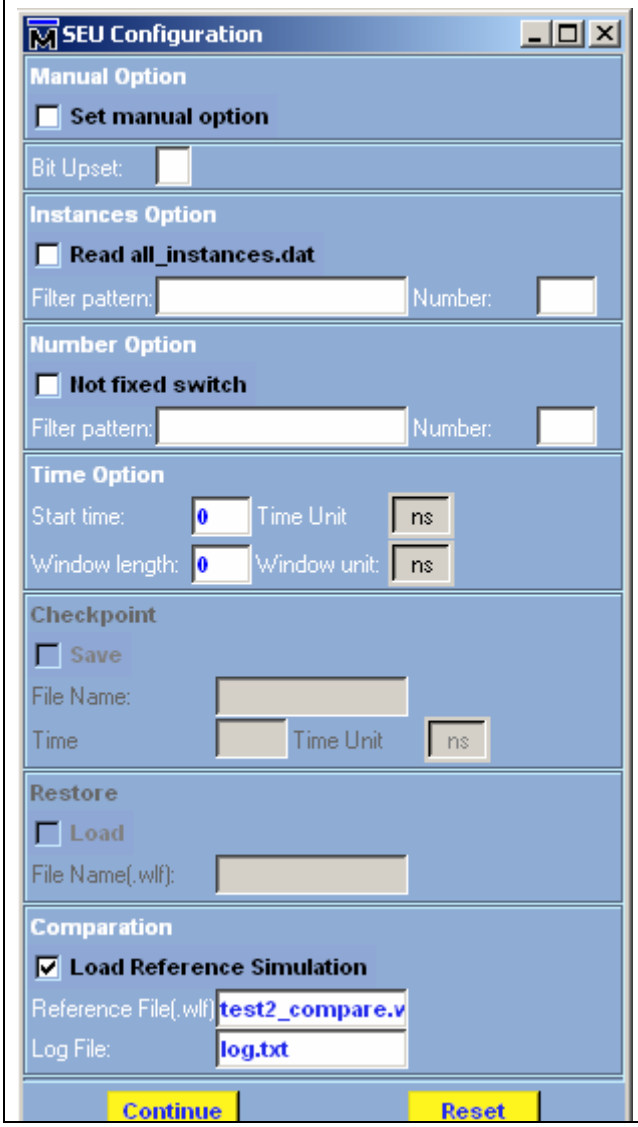


Starting simulation...
 # Starting SEU simulation plus checkpoint...
 # ERROR: SEU is inserted in a past time

SST.DO

```
#Macro generated by SST_upset_generator.pl
#2 wires forced

onbreak {abort 1}
echo {Starting SEU simulation plus checkpoint...}
noview wave
add wave sim:$signals
echo {ERROR: SEU is inserted in a past time}
```

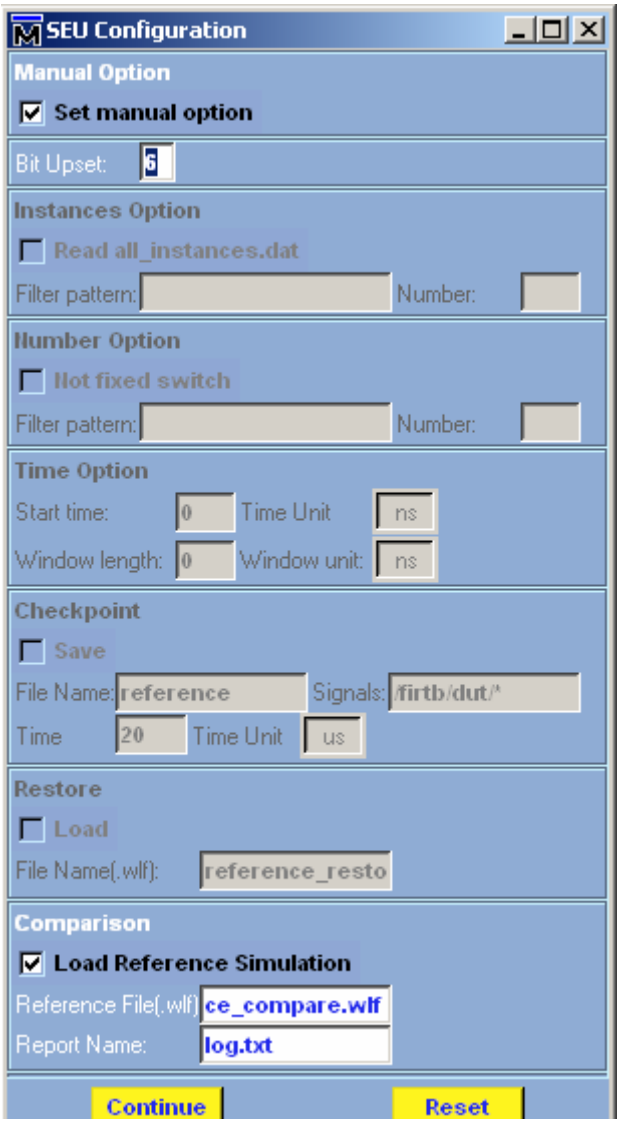
Comparison



SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Comparing...}
dataset open oscar_compare.wlf
regsub ".wlf" oscar_compare.wlf "" sample
compare start -hide $sample sim
compare add -r $sample:/firtb/dut/*
compare run
compare info -all -primaryonly -signals -
secondaryonly -summary -write xxx 20 50

status
```

Fault-Injection Campaign + Compare



Info Reported: log.txt

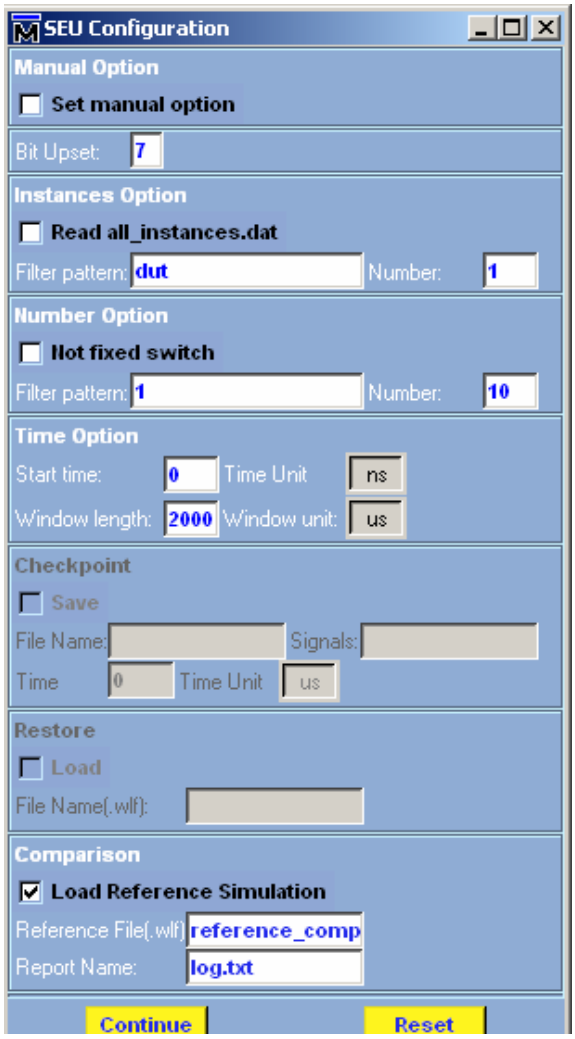
SST.DO

```
#Macro generated by SST_upset_generator.pl
#2 wires forced

onbreak {abort 1}
echo {Starting SEU simulation...}

run 4015 ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 4015ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 70 ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 4085ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
echo {Comparing...}
dataset open reference_compare.wlf
regsub ".wlf" reference_compare.wlf "" sample
compare start -hide $sample sim
compare add -r $sample:/firtb/dut/*
compare run
compare info -write {log.txt}
status
```

Sample Report: LOG.TXT

```
Total signals compared = 8
Total primary differences = 10
Total secondary differences = 12
Number of primary signals with differences = 9
Diff number 1, From time 4015 ns delta 2 to time 4050 ns delta 2.
reference_compare:/firtb/dut/reg = {01010010 01100000 01110100 10000011 11110001 00010110 01001000
10101111}
sim:/firtb/dut/reg = {01010010 01100000 01110100 10000011 11010001 00010110 01001000 10101111}
Diff number 2, From time 4015 ns delta 2 to time 4050 ns delta 2.
reference_compare:/firtb/dut/reg(5) = 11110001
sim:/firtb/dut/reg(5) = 11010001
Diff number 3, From time 4015 ns delta 2 to time 4050 ns delta 2.
reference_compare:/firtb/dut/reg(5)(5) = 1
sim:/firtb/dut/reg(5)(5) = 0
Diff number 4, From time 4015 ns delta 2 to time 4050 ns delta 2.
```

```
reference_compare:/firtb/dut/reg_salida_next = 00010100111
sim:/firtb/dut/reg_salida_next = 00010000111
Diff number 5, From time 4015 ns delta 2 to time 4050 ns delta 2.
reference_compare:/firtb/dut/reg_salida_next(5) = 1
sim:/firtb/dut/reg_salida_next(5) = 0
Diff number 6, From time 4015 ns delta 2 to time 4050 ns delta 2.
reference_compare:/firtb/dut/reg_next = {10010011 01010010 01100000 01110100 10000011 11110001 00010110
```




```
Comparing...
# reference_compare.wlf opened as dataset
"reference_compare"
# 1
# Created 8 comparisons.
# Computing waveform differences from time 0 ns to
1919735 ns
# Found 684 differences.
```

LOG.TXT

```
Total signals compared = 8
Total primary differences = 156
Total secondary differences = 150
Number of primary signals with differences = 32
Diff number 1, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg = {10111010 11100011
11101110 00000011 10010000 00101000 11100110 10011110}
```

SST.DO

```
#Macro generated by SST_upset_generator.pl
#10 wires forced

onbreak {abort 1}
echo {Starting SEU simulation...}

run 47485 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 47485ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 376645 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 424130ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 131960 ns
# define signal path
set wire /firtb/dut/reg_next(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 556090ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 592895 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
```

```
sim:/firtb/dut/reg = {10111000 11100011 11101110 00000011
10010000 00101000 11100110 10011110}
Diff number 2, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg(1) = 10111010
sim:/firtb/dut/reg(1) = 10111000
Diff number 3, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg(1)(1) = 1
sim:/firtb/dut/reg(1)(1) = 0
Diff number 4, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next =
11011001010
sim:/firtb/dut/reg_salida_next = 11011001000
Diff number 5, From time 89965 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(9) = 1
sim:/firtb/dut/reg_salida_next(9) = 0
Diff number 6, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg_next = {01110010
10111010 11100011 11101110 00000011 10010000 00101000
11100110}
sim:/firtb/dut/reg_next = {01110010 10111000 11100011
11101110 00000011 10010000 00101000 11100110}
Diff number 7, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg_next(2) = 10111010
sim:/firtb/dut/reg_next(2) = 10111000
Diff number 8, From time 89965 ns delta 2 to time 90050 ns
delta 2.
reference_compare:/firtb/dut/reg_next(2)(1) = 1
sim:/firtb/dut/reg_next(2)(1) = 0
Diff number 9, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg_salida = 11011001010
sim:/firtb/dut/reg_salida = 11011001000
Diff number 10, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg = {01110010 10111010
11100011 11101110 00000011 10010000 00101000 11100110}
sim:/firtb/dut/reg = {01110010 10111000 11100011 11101110
00000011 10010000 00101000 11100110}
Diff number 11, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg(2) = 10111010
sim:/firtb/dut/reg(2) = 10111000
Diff number 12, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg(2)(1) = 1
sim:/firtb/dut/reg(2)(1) = 0
Diff number 13, From time 90050 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida(9) = 1
sim:/firtb/dut/reg_salida(9) = 0
Diff number 14, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg_next = {10001000
01110010 10111010 11100011 11101110 00000011 10010000
00101000}
sim:/firtb/dut/reg_next = {10001000 01110010 10111000
11100011 11101110 00000011 10010000 00101000}
Diff number 15, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg_next(3) = 10111010
sim:/firtb/dut/reg_next(3) = 10111000
Diff number 16, From time 90050 ns delta 2 to time 90150 ns
delta 2.
```

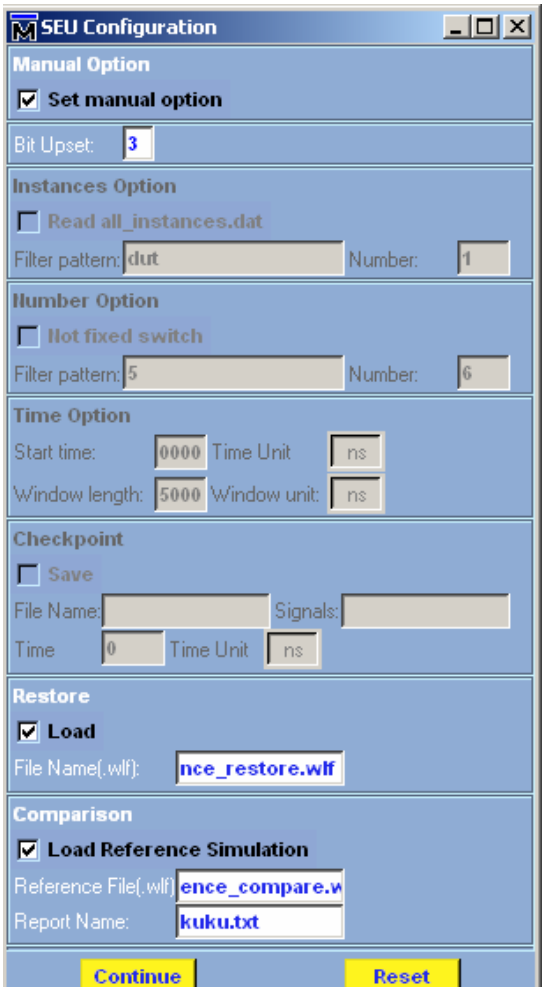
```
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1148985ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 2195 ns
# define signal path
set wire /firtb/dut/reg_next(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1151180ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 29115 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1180295ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 20750 ns
# define signal path
set wire /firtb/dut/reg_next(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1201045ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 407105 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1608150ns
  force -deposit $wire $wire_upset_val
```

```
reference_compare:/firtb/dut/reg_salida_next =
11110011110
sim:/firtb/dut/reg_salida_next = 11110011100
Diff number 17, From time 90050 ns delta 2 to time 90150 ns
delta 2.
reference_compare:/firtb/dut/reg_next(3)(1) = 1
sim:/firtb/dut/reg_next(3)(1) = 0
Diff number 18, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida = 11110011110
sim:/firtb/dut/reg_salida = 11110011100
Diff number 19, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg = {10001000 01110010
10111010 11100011 11101110 00000011 10010000 00101000}
sim:/firtb/dut/reg = {10001000 01110010 10111000 11100011
11101110 00000011 10010000 00101000}
Diff number 20, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg(3) = 10111010
sim:/firtb/dut/reg(3) = 10111000
Diff number 21, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg(3)(1) = 1
sim:/firtb/dut/reg(3)(1) = 0
Diff number 22, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_next = {11000011
10001000 01110010 10111010 11100011 11101110 00000011
10010000}
sim:/firtb/dut/reg_next = {11000011 10001000 01110010
10111000 11100011 11101110 00000011 10010000}
Diff number 23, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_next(4) = 10111010
sim:/firtb/dut/reg_next(4) = 10111000
Diff number 24, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next =
11101000000
sim:/firtb/dut/reg_salida_next = 11100111110
Diff number 25, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(4) = 1
sim:/firtb/dut/reg_salida_next(4) = 0
Diff number 26, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(6) = 0
sim:/firtb/dut/reg_salida_next(6) = 1
Diff number 27, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(7) = 0
sim:/firtb/dut/reg_salida_next(7) = 1
Diff number 28, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(8) = 0
sim:/firtb/dut/reg_salida_next(8) = 1
Diff number 29, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(9) = 0
sim:/firtb/dut/reg_salida_next(9) = 1
Diff number 30, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_next(4)(1) = 1
sim:/firtb/dut/reg_next(4)(1) = 0
Diff number 31, From time 90150 ns delta 2 to time 90250 ns
delta 2.
reference_compare:/firtb/dut/reg_salida_next(5) = 0
sim:/firtb/dut/reg_salida_next(5) = 1
```

```
} else {
  echo Unable to upset $wire. Undef value.
}

run 84230 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1692380ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}

run 227355 ns
# define signal path
set wire /firtb/dut/reg(1)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val  to
$wire_upset_val @ 1919735ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
echo {Comparing...}
dataset open $file_name_gold
regsub ".wlf" $file_name_gold "" sample
compare start -hide $sample sim
compare add -r $sample:/firtb/dut/*
compare run
compare info -write {$file_name_summary}
status
```

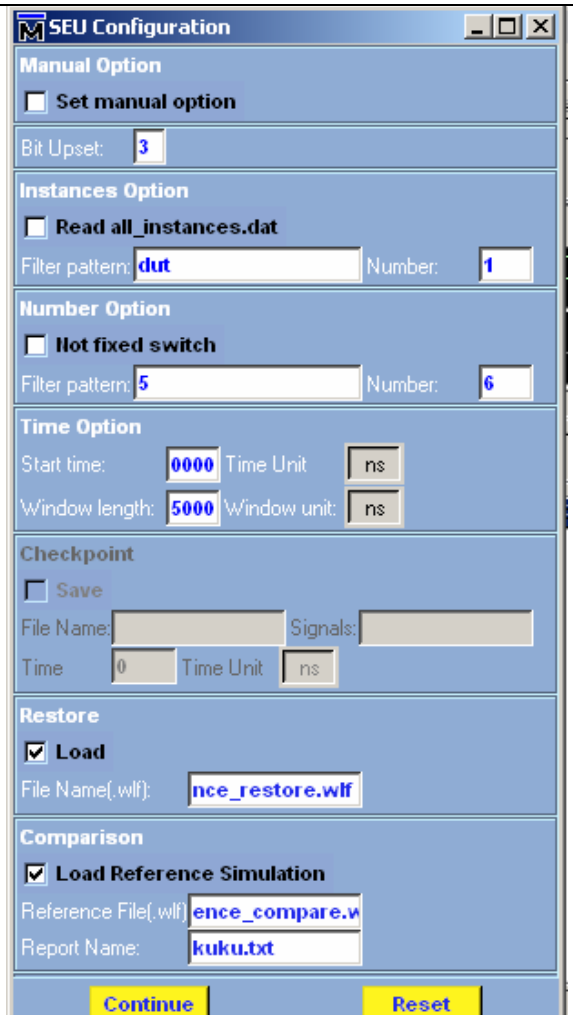
Diff number 32, From time 90250 ns delta 2 to time 90350 ns delta 2. **reference_compare:/firtb/dut/reg_salida = 11101000000** **sim:/firtb/dut/reg_salida = 11100111110** **Diff number 33, From time 90250 ns delta 2 to time 90350 ns delta 2.** **reference_compare:/firtb/dut/reg_salida(5) = 0** **sim:/firtb/dut/reg_salida(5) = 1** **Diff number 34, From time 90250 ns delta 2 to time 90350 ns delta 2.** **reference_compare:/firtb/dut/reg_salida(9) = 0** **sim:/firtb/dut/reg_salida(9) = 1** **Diff number 35, From time 90250 ns delta 2 to time 90350 ns delta 2.** **reference_compare:/firtb/dut/reg = {11000011 10001000 01110010 10111010 11100011 11101110 00000011 10010000}** **sim:/firtb/dut/reg = {11000011 10001000 01110010 10111000 11100011 11101110 00000011 10010000}** **Diff number 36, From time 90250 ns delta 2 to time 90350 ns delta 2.** **reference_compare:/firtb/dut/reg(4) = 10111010** **sim:/firtb/dut/reg(4) = 10111000** **Diff number 37, From time 90250 ns delta 2 to time 90350 ns delta 2……..**	

Restore + Fault-Injection Campaign + Compare



SST.DO

```
#Macro generated by SST_upset_generator.pl
onbreak {abort 1}
echo {Reloaded a Simulation from Initial State...}
noview wave
vsim -restore $file_name_restore
#6 wires forced

onbreak {abort 1}
echo {Starting SEU simulation...}
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2000280} {
puts "ERROR: SEU is being inserted at past
time:
                    SEU 2000280 vs NOW
$runningtime"
} else {
# 2000280 $sample
set run [expr 2000280 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2000280ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
          }
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2001230} {
puts "ERROR: SEU is being inserted at past
time:
                    SEU 2001230 vs NOW
$runningtime"
} else {
# 2001230 $sample
set run [expr 2001230 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2001230ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
          }
```




```
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2001295} {
puts "ERROR: SEU is being inserted at past
time:
                    SEU 2001295 vs NOW
$runningtime"
} else {
# 2001295 $sample
set run [expr 2001295 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2001295ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
          }
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2001425} {
puts "ERROR: SEU is being inserted at past
time:
                    SEU 2001425 vs NOW
$runningtime"
} else {
# 2001425 $sample
set run [expr 2001425 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2001425ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
          }
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2001945} {
puts "ERROR: SEU is being inserted at past
time:
                    SEU 2001945 vs NOW
$runningtime"
} else {
# 2001945 $sample
set run [expr 2001945 - $sample]

run $run ns
```

```
# define signal path
set wire /firtb/dut/reg(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2001945ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
            }
set runningtime [getactivecursortime]
regsub " ns" $runningtime "" sample
puts "traza $sample"
if {$sample > 2002785} {
puts "ERROR: SEU is being inserted at past
time:
                        SEU 2002785 vs NOW
$runningtime"
} else {
# 2002785 $sample
set run [expr 2002785 - $sample]

run $run ns
# define signal path
set wire /firtb/dut/reg_next(5)
# examine current value
set wire_checked_val [exa $wire]
# flip one bit of the wire
set wire_upset_val [exec perl -S SST_bit_flip.pl
$wire_checked_val $bit_in_m]
if {$wire_upset_val != "undefined"} {
  echo Forcing $wire: $wire_checked_val to
$wire_upset_val @ 2002785ns
  force -deposit $wire $wire_upset_val
} else {
  echo Unable to upset $wire. Undef value.
}
            }
echo {Comparing...}
dataset open $file_name_gold
regsub ".wlf" $file_name_gold "" sample
compare start -hide $sample sim
compare add -r $sample:/firtb/dut/*
compare run
compare info -write $file_name_summary
                        status
```