



TEC | Tecnológico
de Costa Rica

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Verificación Funcional de Circuitos Integrados

Proyecto 3

Profesor:

Ronny García Ramírez

Integrantes:

Susana Astorga Rodríguez

Mac Alfred Pinnock Chacón

II Semestre 2021

Cartago, Costa Rica

Índice

Plan de pruebas (Test plan)	2
Módulos a utilizar:	2
Caso de uso común:	3
Casos de esquina:	3
Estructura del ambiente	3
Pruebas para los diferentes escenarios	4
Solución del error del DUT	8
Reportes de los escenarios (usando EDA Playground)	9
Cobertura de la prueba	15
Gráficos de reportes con GNUpot	17

Plan de pruebas (Test plan)

GitHub: https://github.com/astorga02/Proyecto3_VFCI

La prueba se corre en EDA Playground o en VCS

En la misma ruta de archivo en donde abre el documento "LEERME.txt", encontrará 2 carpetas: "Para_EDA" y "Para_VCS" estas carpetas difieren en lo siguiente para poder ejecutarlos.

Para_EDA:

- Solo se puede ejecutar en EDA Playground.
- Para ejecutar cada escenario (2 en total) debe hacerlo manualmente, cambiando la línea 42 por el número de escenario a ejecutar :
`"run_test("test_escenario2"); //se coloca el nombre del test a probar (test_escenario1 o test_escenario2)"`

Para_VCS:

- Solo se puede ejecutar en VCS
- Para ejecutar la prueba solo es necesario abrir la terminal en la raíz de los archivos de la carpeta y escribir ./correr.sh y dar enter, la prueba finalizara mostrando la cobertura de la prueba y también puede acceder al reporte.csv. Además de poder consultar la salida de la terminal para verificar cualquier problema con el DUT.

Módulos a utilizar:

- monitor.sv
- design.sv
- agent.sv
- scoreboard.sv
- environment.sv
- driver.sv
- interface.sv
- sequence.sv
- sequence_item.sv
- test.sv
- testbench.sv

Caso de uso común:

Aleatorizando:

- Eventos de datos de entrada.
- Eventos de modo de redondeo.

Casos de esquina:

- Datos de entrada que cubran todos los bits.
- Overflow.
- Underflow.
- No es un número (NaN)
- Caso infinito (inf).

Estructura del ambiente

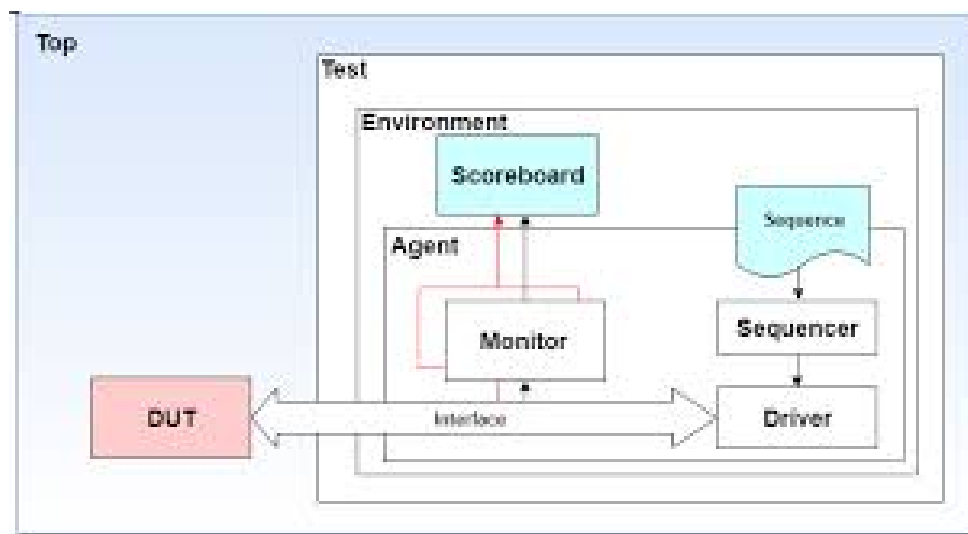


Figura 1. Estructura del ambiente.

Pruebas para los diferentes escenarios

Escenario 1: Envío de mensaje con instrucción del tipo aleatorio
Descripción del test
<p>Test 1: Envío de mensajes tipo aleatorio</p> <p>Objetivo: Verificar el correcto funcionamiento de envío de mensajes aleatorios en el DUT y que su la multiplicación de los valores de entrada sea correcta.</p> <p>Descripción:</p> <p>En este test se envía un número de 100 transacciones con valores de entrada aleatorios que no sobrepasen las capacidades del DUT en cuanto a overflow, underflow y no ser un número, esto debido a que se dejan estos casos como objetivos para tests de caso de esquina.</p> <p>Las transacciones se enviarán con retardos aleatorios independientes para cada uno de los dispositivos.</p> <p>Criterio de aprobación: Se considera que el test es exitoso si la multiplicación de los valores de entrada es correcta comparándolo con un dispositivo simulado que simula el correcto funcionamiento del DUT. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.</p> <p>Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida)</p>
Escenario 2: Comportamiento del DUT en casos de esquina
Descripción del test
Test 2.1: Datos de entrada que cubran todos los bits

Objetivo: Cubrir todos los bits que es capaz de operar el DUT

Descripción:

En este test se enviarán valores para una máxima alternancia de bits con una cantidad de 100 transacciones de estos valores.

Las transacciones se enviarán con retardos aleatorios.

Criterio de aprobación: Se considera que el test es exitoso si la multiplicación de los valores de entrada es correcta comparándolo con un dispositivo simulado que simula el correcto funcionamiento del DUT. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.

Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida)

Test 2.2: Overflow

Objetivo: Verificar el funcionamiento de envío de mensajes con el caso de overflow para los valores multiplicados por el DUT.

Descripción:

En este test el DUT tendrá valores de entrada con valores que vayan a causar un overflow en el resultado de la operación. De este caso se envía un número de 100 transacciones.

El contenido de los datos de transacción será totalmente aleatorio pero que se encuentren fuera del rango de operación del DUT. Las transacciones se enviarán con retardos aleatorios independientes.

Criterio de aprobación: Se considera que el test es exitoso si la multiplicación de los valores de entrada da como resultado la activación de la bandera de overflow y un valor infinito a la salida del DUT. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.

Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida)

Test 3.3: Underflow

Objetivo: Verificar el funcionamiento de envío de mensajes con el caso de underflow para los valores multiplicados por el DUT.

Descripción:

En este test el DUT tendrá valores de entrada con valores que vayan a causar un underflow en el resultado de la operación. De este caso se envía un número de 100 transacciones. El contenido de los datos de transacción será totalmente aleatorio pero que se encuentren fuera del rango de operación del DUT. Las transacciones se enviarán con retardos aleatorios independientes.

Criterio de aprobación: Se considera que el test es exitoso si la multiplicación de los valores de entrada da como resultado la activación de la bandera de underflow y un valor de 0 a la salida del DUT. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.

Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida)

Test 3.4: No es un número

Objetivo: Verificar el funcionamiento de envío de mensajes con salida “NaN”.

Descripción:

En este test se envía un número de 100 transacciones de las cuales se hacen asignando los valores de entrada para que sean cero para la entrada 1 e infinito para la entrada 2 y viceversa

El orden de envío será totalmente aleatorio en donde cero e infinito se asignan aleatoriamente a las entradas con la condición de que las entradas no pueden ser a la vez

cero o a la vez infinitas.

Criterio de aprobación: No existe problema con la verificación de que todo sea NaN en ese caso de esquina. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.

Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida).

Test 3.5: Infinito

Objetivo: Verificar el funcionamiento de envío de mensajes con salida “inf”.

Descripción:

En este test se envía un número de 100 transacciones de las cuales se hacen asignando los valores de entrada para representar el escenario de un resultado infinito

El orden de envío será totalmente aleatorio en donde cero e infinito se asignan aleatoriamente a las entradas con la condición de que las entradas no pueden ser a la vez cero o a la vez infinitas.

Criterio de aprobación: No existe problema con la verificación de que todo sea inf en ese caso de esquina. Se espera una línea indicando “Todo correcto” como salida inmediata para cada verificación.

Reportes: Al final de la prueba se imprimirá un reporte con el contenido de cada una de las operaciones (entradas y salida)

Solución del error del DUT

Existe un error cuando una de las entradas se establece como underflow y el otro no, además que para el caso contrario (overflow), sucede el mismo problema. En el primer caso se obtiene un resultado con underflow sin que la bandera underflow se active, lo mismo con el caso de overflow en el otro caso.

Para solucionarlo se agrega en el DUT en la sección de código de la figura 2, se agrega la función en donde se da la generación de las señales de overflow y underflow en caso de que se cumpla la condición.

```
module EXP(                                //module con cambios agregados
    input norm,
    input [7:0]exp_X, exp_Y,
    output [7:0]exp_Z,
    output ovrf, udrf);

    wire [8:0]buffer;
    wire over_X, over_Y, under_X, under_Y;

    assign buffer = exp_X + exp_Y;

    wire [7:0]bias;

    assign over_X = &exp_X;
    assign over_Y = &exp_Y;
    assign under_X = ~|exp_X;
    assign under_Y = ~|exp_Y;
    assign bias = {7'b0111111, !norm};
    assign ovrf = {{buffer >= {255 + bias}}|{over_X}|{over_Y}};
    assign udrf = {{buffer <= bias}|{under_X}|{under_Y}};

    assign exp_Z = exp_X + exp_Y - bias;

endmodule
```

Figura 2. Secciones de código agregadas usando Visual Studio Code.

Reportes de los escenarios (usando EDA Playground)

Solo se muestran ciertas secciones debido a la gran cantidad de datos obtenidos en el test y para el test aleatorio sí se muestra el encabezado de la simulación de donde se inicializan los componentes del ambiente, esto con el objetivo de evitar redundar en cosas que son siempre iguales para los demás tests.

Escenario 1: Envío de mensaje con instrucción del tipo aleatorio Test 1. Aleatorización

```
UVM_INFO scoreboard.sv(167) @ 1790: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 11100100100000000000000001111111 Entrada y = 1110001010000000000000000001011
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1810: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 01110111000000000000000000000110 Entrada y = 101011101000000000000000000101110
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1830: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 0010111000000000000000000001010101 Entrada y = 1110101110000000000000000001010101
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1850: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 010101001000000000000000000101001 Entrada y = 1100111100000000000000000001100000
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1870: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 0000001010000000000000000001111001 Entrada y = 1100010110000000000000000000000111
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1890: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 11010111000000000000000000010100 Entrada y = 00111010000000000000000000011110
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1910: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 0100001110000000000000000001001101 Entrada y = 100100011000000000000000000111010
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1930: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 001011001000000000000000000100101 Entrada y = 100000010000000000000000000111010
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1950: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 0001011100000000000000000001011000 Entrada y = 110011100000000000000000000101110
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1970: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 0101011100000000000000000001001011 Entrada y = 10001110100000000000000000011010
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 1990: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 101100111000000000000000000101011 Entrada y = 0011011100000000000000000001010011
UVM_INFO sequence.sv(43) @ 1990: uvm_test_top.ambiente_instancia.agente_instancia.secuenciador_instancia@seq.Secuencia_aleatorio [SEQ] Se generaron 100 items del tipo aleatorio
UVM_INFO /apps/vcsmx/vcs/Q-2020.03-SP1-1//etc/uvm-1.2/src/base/uvm_objection.svh(1276) @ 1990: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
UVM_INFO /apps/vcsmx/vcs/Q-2020.03-SP1-1//etc/uvm-1.2/src/base/uvm_report_server.svh(894) @ 1990: reporter [UVM/REPORT/SERVER]
--- UVM Report Summary ---

** Report counts by severity
UVM_INFO : 105
UVM_WARNING : 0
UVM_ERROR : 0
UVM_FATAL : 0

** Report counts by id
[RNTST] 1
[SCBD] 100
[SEQ] 1
[TEST_DONE] 1
[TOP] 1
[UVM/RELNOTES] 1

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 11100110000000000000000001010101 Resultado esperado = 11100110000000000000000001010101 overflow = 0 Underflow = 0

Resultado del DUT = 110101000000000000000000010011111 Resultado esperado = 110101000000000000000000010011111 overflow = 0 Underflow = 0

Resultado del DUT = 111001000000000000000000010001001 Resultado esperado = 111001000000000000000000010001001 overflow = 0 Underflow = 0

Resultado del DUT = 100010001000000000000000010000000 Resultado esperado = 100010001000000000000000010000000 overflow = 0 Underflow = 0

Resultado del DUT = 110110010000000000000000000110011 Resultado esperado = 110110010000000000000000000110011 overflow = 0 Underflow = 0

Resultado del DUT = 100101011000000000000000010000111 Resultado esperado = 100101011000000000000000010000111 overflow = 0 Underflow = 0

Resultado del DUT = 100000000000000000000000000000000 Resultado esperado = 100000000000000000000000000000000 overflow = 0 Underflow = 1

Resultado del DUT = 10100110100000000000000000010110110 Resultado esperado = 10100110100000000000000000010110110 overflow = 0 Underflow = 0

Resultado del DUT = 1010011000000000000000000001100101 Resultado esperado = 1010011000000000000000000001100101 overflow = 0 Underflow = 0

Resultado del DUT = 10101111100000000000000000010111110 Resultado esperado = 10101111100000000000000000010111110 overflow = 0 Underflow = 0
```

Test 2.1. Alternancia de bits en los datos de entrada

10

Test 2.2. Overflow

```
UVM_INFO scoreboard.sv(167) @ 2150: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 11001010011000101000100000011111 Entrada y = 01010101000111101101110101010000
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2170: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 110001011001011110100010010101010 Entrada y = 01111001110001010001110000000100
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2190: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 0110000011100101110100010100100 Entrada y = 110111010000110100111000001110
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2210: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 01001110110101010101110011101101 Entrada y = 01110000101100001000000101010001
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2230: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 11111010101101101010100101010110 Entrada y = 11000100110101011000001000101000
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2250: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 010010000000100111100110111001 Entrada y = 111101101110010001110101000111
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2270: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 1110001110110100101111000011101 Entrada y = 110101111101010101010011001101
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2290: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11011011100000101010110010010010 Entrada y = 11100011010011101110100000000011
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2310: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 01111110011000101101100100000000 Entrada y = 1100000101110000101010110001110
Bandera overflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 2330: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 0110001010011001010101010000100 Entrada y = 010111001011100101010010001011
UVM_INFO sequence.sv(131) @ 2330: uvm_test_top.ambiente_instancia.agente_instancia.secuenciador_instancia@Seq.Secuencia_overflow [SEQ] Se generaron 100 items del tipo overflow

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 Underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 Underflow = 0
```

Test 2.3: Underflow

```
UVM_INFO scoreboard.sv(167) @ 4130: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 1000000000110011011010100000011 Entrada y = 11001111001101101000001011001011
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4150: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 000000000010010010010101000010110 Entrada y = 11001011011110001000011010110
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4170: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 1101010001110100101010111111011 Entrada y = 0000000000001100010101001000101
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4190: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 00011111000000110010000000101110 Entrada y = 000000000000100001110101111110
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4210: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 1111010000110101110111011010110 Entrada y = 0000000000100101110101010100110
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4230: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 00010100100100110101101001111101 Entrada y = 100000000011111010111100111111
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4250: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 1000000001010100010000001000100 Entrada y = 1111010001001110100010101000000
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4270: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 000000000001000010110001010000100 Entrada y = 11110100101011000100101011000001
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4290: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 1000000001001010111010100001000 Entrada y = 110010110000111101000001110000
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4310: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 11011010101010101011110000000100 Entrada y = 000000000011111000110001001010
Bandera underflow activada
Resultado redondeado
UVM_INFO scoreboard.sv(167) @ 4330: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 10100010000100001001000001001011 Entrada y = 10000000000000000001010111100001
UVM_INFO sequence.sv(169) @ 4330: uvm_test_top.ambiente_instancia.agente_instancia.secuenciador_instancia@seq.Seuencia_underflow [SEQ] Se generaron 100 items del tipo underflow
```

Resultado del DUT = 0000000000000000000000000000 Resultado esperado = 0000000000000000000000000000 overflow = 0 Underflow = 1

Resultado del DUT = 10000000000000000000000000000 Resultado esperado = 10000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 1000000000000000000000000000 Resultado esperado = 1000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 00000000000000000000000000000000 Resultado esperado = 00000000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 10000000000000000000000000000 Resultado esperado = 10000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 10000000000000000000000000000 Resultado esperado = 10000000000000000000000000000 overflow = 0 Underflow = 1

Resultado del DUT = 0000000000000000000000000000 Resultado esperado = 0000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 1000000000000000000000000000 Resultado esperado = 1000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 0000000000000000000000000000 Resultado esperado = 0000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 10000000000000000000000000000 Resultado esperado = 10000000000000000000000000000 overflow = 0 underflow = 1

Resultado del DUT = 0000000000000000000000000000 Resultado esperado = 0000000000000000000000000000 overflow = 0 Underflow = 1

Test 2.4: No es un número (NaN)

```

UMW_INFO scoreboard.sv(167) @ 6030: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 0111111110010010111101000011000 Entrada y = 1010010011110101010101100101100001
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6050: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 1101100011101111101100001110010 Entrada y = 011111111010110101010101011001110
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6070: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 01111111110101001110100000011111 Entrada y = 11001111100000111110001010101001
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6090: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 11111111110101010100001110010000 Entrada y = 11010001001000101000111100100101
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6110: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11111111101111001010000100100101 Entrada y = 1100001010000010000011010101110
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6130: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 011111111010111110101000001011 Entrada y = 00001110010101010100111010111011
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6150: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 101110100110100000001001100011 Entrada y = 011111111111110110001001111110
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6170: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 11100001000010011111010101100100 Entrada y = 0111111111110010000101110010010
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6190: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 0101010000010000010110011010011 Entrada y = 01111111100101000001100111101011
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6210: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 1001010100010000101110000100010 Entrada y = 0111111110101001010000110100000
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6230: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 101001000100001111001000101000 Entrada y = 011111111100100011110010101011
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6250: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 101010100010000101100100111001 Entrada y = 11111111110010001000100101000010
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6270: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 100 Entrada x = 1111010101101100001011001100101 Entrada y = 11111111101011000001001111110011
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6290: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 01111000011010000101111011000001 Entrada y = 1111111111011000010010010100010
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6310: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 010 Entrada x = 1111111110010000100000010111110 Entrada y = 010101011111111001100000110111
Resultado redondeado
UMW_INFO scoreboard.sv(167) @ 6330: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 101010100011011000001001100100 Entrada y = 11111111100110101010000110100111
UMW_INFO sequence.sv(207) @ 6330: uvm_test_top.ambiente_instancia.agente_instancia.secuencia_instancia@seq_secuencia_NAN [Seq] Se generaron 100 items del tipo NAN

```

[illegible]

Test 2.4: Infinito (inf)

```
UVM_INFO scoreboard.sv(167) @ 8210: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11111111110000000000000000000000 Entrada y = 00011110110110011010010101000000
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8230: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 01000111101110010111001100011100 Entrada y = 01111111100000000000000000000000
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8250: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 01111111100000000000000000000000 Entrada y = 00111110110111011010000101100011
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8270: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11111111100000000000000000000000 Entrada y = 01000100000001100000011011011011
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8290: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 011 Entrada x = 00001100011010100001001000000001 Entrada y = 01111111100000000000000000000000
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8310: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11111111100000000000000000000000 Entrada y = 011110101010101010100111000010001
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8330: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 000 Entrada x = 11100010111010000101110100011011 Entrada y = 01111111100000000000000000000000
Bandera overflow activada
UVM_INFO scoreboard.sv(167) @ 8350: uvm_test_top.ambiente_instancia.scoreboard_instancia [SCBD] Mode = 001 Entrada x = 01111111100000000000000000000000 Entrada y = 10100010001111100110100000000100
UVM_INFO sequence_sv(246) @ 8350: uvm_test_top.ambiente_instancia.agente_instancia.secuenciador_instancia@seq.Seuencia_inf [SEQ] Se generaron 100 items del tipo inf
UVM_INFO /apps/vcsmx/vcs/Q-2020.03-SPI-1//etc/uvm-1.2/src/base/uvm_objection.svh(1276) @ 8350: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
UVM_INFO /apps/vcsmx/vcs/Q-2020.03-SPI-1//etc/uvm-1.2/src/base/uvm_report_server.svh(894) @ 8350: reporter [[VW/REPORT/SERVER]]

--- UVM Report Summary ---

** Report counts by severity
UVM_INFO :    427
UVM_WARNING :     0
UVM_ERROR   :     0
UVM_FATAL   :     0

** Report counts by id
[RNTST]      1
[SCBD]       418
[SEQ]         5
[Test_DONE]   1
[TOP]          1
[UWM/RELNOTES] 1

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 01111111100000000000000000000000 Resultado esperado = 01111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 underflow = 0

Resultado del DUT = 11111111100000000000000000000000 Resultado esperado = 11111111100000000000000000000000 overflow = 1 underflow = 0
```

Cobertura de la prueba

<Verdi:vdCoverage:1><vdb: salida.vdb>

File View Plan Exclusion Tools Window Help

Summary

Hierarchy Modules Groups Asserts Statistics Tests

Name	Score	Line	Toggle	FSM	Condition	Branch	Assert
top_testbench	99.34%	100.00%	97.85%		100.00%	98.85%	100.00%
if	100.00%		100.00%				100.00%
dut0	99.28%		97.83%		100.00%	100.00%	
FPM	99.27%		97.80%		100.00%	100.00%	
EXC	100.00%		100.00%				
EXP	85.42%		85.42%				
MUL	99.42%		98.25%		100.00%	100.00%	
NET	100.00%		100.00%		100.00%	100.00%	
NORM	98.39%		98.39%				
ROUND	98.89%		96.67%		100.00%	100.00%	

722 wire [7

723 :0] bias;

724

725 assign

over_X

= &exp_

X;

726 assign

over_Y

= &exp_

Y;

727 assign

under_X

= ~|ex

p_X;

728 assign

under_Y

CovSrc...PM.EXP

CovDetail

Line Toggle FSM Condition Branch A

Variable	Type	Cov
▲ bias[7:0]	signal	
▲ buffer[8:0]	signal	
▲ exp_X[7:0]	port	
▲ exp_Y[7:0]	port	
▲ exp_Z[7:0]	port	

Variable	0->1	1->0
▲ bias[7:1]	X	X
▲ bias[0]	✓	✓

CovSrc.1 Hvp

CovDetail HvpDetail

Message

Failed to load tests. Details are as the following:
Load Failed:
Warning-[UCAPI-SNF] Shape Not Found
'Fsm' coverage shape is not there in the vdb 'salida.vdb'. Coverage of this metric will not be reported.
Please check if proper -cm option was used at compile time.

The design 'salida.vdb' was loaded successfully.
The following test is loaded from "salida.vdb",
salida/test

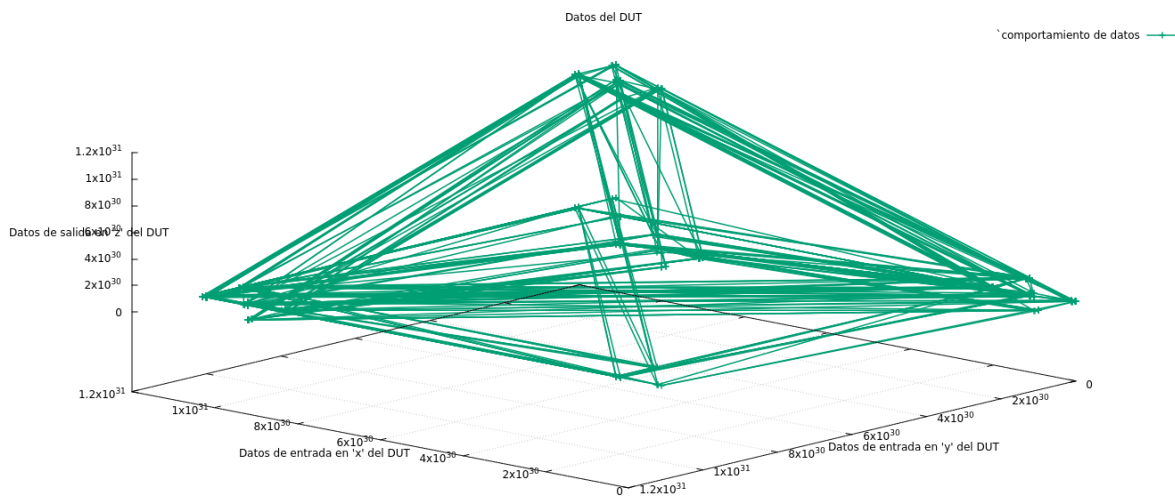
Exclusion Manager Requirements Manager Message

En la imagen anterior se muestra la cobertura de la prueba en la que se cumple la condición de al menos un 95% de cobertura para el toggle y para los casos.

Gráficos de reportes con GNUplot

- **Gráfica reporte de las entradas en “x” y “y” con respecto a la salida “z”**

En el siguiente apartado se grafican los datos obtenidos en el reporte obtenido del test, por lo que la siguiente figura muestra la gráfica de los datos de entrada en “y” del DUT, los datos de entrada en “x” del DUT y los datos de salida en “z” del DUT, mediante una gráfica de datos de las entradas en “x” y “y” con respecto a la salida “z”.



Conclusiones

- Se cumplen las condiciones de aprobación en la verificación para todos los escenarios expuestos en el test plan.
- En el momento en que el DUT sea implementado en la vida real, se debe tener mucho cuidado sobre la forma en que se usan las profundidades de las entradas de datos para evitar la pérdida de datos por overflow o por underflow.
- El fin mismo de la verificación de este DUT fue el poder lograr reparar el error que tenía el dispositivo, que al final fue satisfactoriamente solucionado
- Las gráficas obtenidas mediante el software GNUplot permiten de una manera visual conocer el comportamiento del DUT ante diferentes escenarios.
- Los datos de cobertura de la prueba facilitan en gran medida poder contemplar escenarios de esquina que no son pensados de primera mano a la hora de montar el test plan.