Quiz UVM0

Emanuel Hernández Cepeda

2017134835

Lo primero que hice fue definir el elemento de transacción (sequence item) que se iba a enviar y recibir durante la prueba, con base en el funcionamiento del DUT. En este punto se puso una restricción en cuanto a la probabilidad de los valores de la señal in, para forzar a que el patrón "1011" aparezca más seguido.

Luego, con base en el elemento de transacción se define la secuencia, la cual consiste en una aleatorización controlada de los elementos de la secuencia. Se define una función que convierte a string el contenido del objeto para poder visualizarlo de mejor forma

Lo siguiente fue hacer el driver, con un protocolo de handshake *get-done*, donde el driver primero espera que le llegue la siguiente transacción y cuando la obtiene notifica al secuenciador que ya se recibió esa transacción para proseguir con la siguiente.

Posteriormente se hizo el monitor, declarando la conexión con el scoreboard a través del puerto de análisis.

Hecho el driver y el monitor, luego quedaba hacer el agente. Para el secuenciador se utilizó el que trae por defecto UVM. Luego de instanciar los 3 módulos, se conectó el secuenciador con el driver a través del puerto TLM.

Luego, hice el scoreboard, definiendo unas variables para poder realizar el chequeo del DUT, tomando en cuenta que el patrón a revisar de 4 bits. El scoreboard analiza el patrón que hay y si es necesario cambiar la señal, modifica la señal esperada para realizar el chequeo comparándola con la obtenida desde el monitor.

Listo el scoreboard, hice el ambiente, instanciando el agente y el scoreboard, conectando este último con el monitor a través del puerto de análisis.

Seguidamente, se hizo el test. Para darle mayor flexibilidad y con miras a darle mayor reutilización al código, hice una clase de test base parametrizable para cualquier patrón de 4 bits, y luego una clase hija que fuera en concreto para el patrón de 1011.

Antes de hacer el test_bench hice un archivo definiendo la interfaz para conectar el DUT con el agente.

En el test_bench, se instancian la interfaz y el DUT y se ejecuta el comando run_test() para empezar a correr la prueba.

El resultado de la simulación indica que no hubo errores en el funcionamiento del DUT, y que cada vez que detectaba el patrón, indicaba que la salida debía ser puesta en 1. En el siguiente ciclo de reloj luego de detectar el patrón, se nota que la salida está puesta en 1, indicando que el DUT funciona correctamente

El log de la simulación se muestra en las siguientes imágenes:

En esta imagen se nota que no hubo ningún error durante la prueba

```
UVM_INFO driver.sv(31) @ 7690: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7710: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1101
UVM_INFO driver.sv(31) @ 7710: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7730: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1011
UVM_INFO scoreboard.sv(52) @ 7730: uvm_test_top.e0.s0 [SCBD] El patrón coincide, la salida esperada es 1
UVM_INFO driver.sv(31) @ 7730: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7750: uvm_test_top.e0.s0 [SCBD] in=01 out=01 ref=0b1011 act=0b111
UVM_INFO driver.sv(31) @ 7750: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7770: uvm_test_top.e0.s0 [SCBD] in=00 out=00 ref=0b1011 act=0b1110
UVM_INFO driver.sv(31) @ 7770: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7790: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1101
UVM_INFO driver.sv(31) @ 7790: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7810: uvm_test_top.e0.s0 [SCBD] in=00 out=00 ref=0b1011 act=0b1010
UVM_INFO driver.sv(31) @ 7810: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7830: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b101
UVM_INFO driver.sv(31) @ 7830: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7850: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1011
UVM_INFO scoreboard.sv(52) @ 7850: uvm_test_top.e0.s0 [SCBD] El patrón coincide, la salida esperada es 1
UVM_INFO driver.sv(31) @ 7850: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7870: uvm_test_top.e0.s0 [SCBD] in=01 out=01 ref=0b1011 act=0b111
UVM_INFO driver.sv(31) @ 7870: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO scoreboard.sv(38) @ 7890: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1111
UVM_INFO driver.sv(31) @ 7890: uvm_test_top.e0.a0.d0 [DRV] Espera por un item del secuenciador
UVM_INFO secuencia.sv(27) @ 7890: uvm_test_top.e0.a0.s0@@seq [SEQ] Se han generado 381 items
UVM_INFO scoreboard.sv(38) @ 7910: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1111
UVM_INFO scoreboard.sv(38) @ 7930: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1111
UVM_INFO scoreboard.sv(38) @ 7950: uvm_test_top.e0.s0 [SCBD] in=01 out=00 ref=0b1011 act=0b1111
```

Esta es una muestra de lo que sucede cuando se detecta el patrón y cómo se cambia la salida a uno en el siguiente ciclo de reloj.

Adjunto el link del EDA playground donde simule la prueba en caso de alguna revisión adicional: https://www.edaplayground.com/x/XHRs