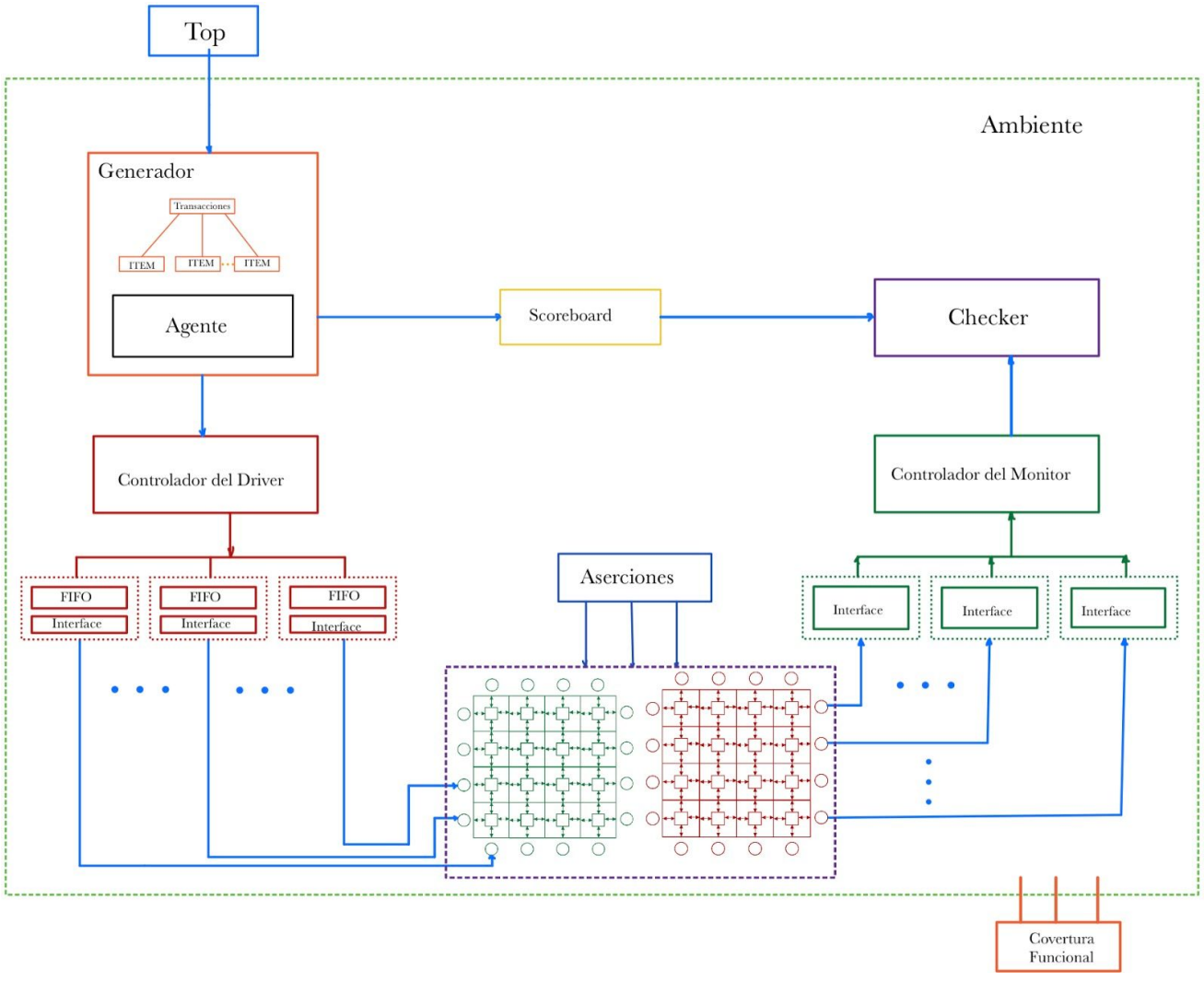
Para este caso se realizó un testbench con la técnica de aleatorización controlada al DUT de un Mesh. Este documento abarca un diagrama general, Test Plan y una serie de pasos a seguir para correr el código.

Para el desarrollo de este banco de pruebas se implementó la siguiente lista de tareas:

- Analizar el DUT.
- Búsqueda de información relacionada al DUT.
- Diseñar el objetivo del Testbench.
- Diseñar un test plan.
- Planificación del desarrollo del Testbench.
- Desarrollo del Ambiente de pruebas.
- Implementación de los diferentes escenarios y pruebas.
- Implementación de los reportes del Testbench.

A continuación se presenta un diagrama de bloques general, donde la clase agente fue combinada con la clase generador.



Test Plan

El Test Plan se diseñó con el objetivo de ejercitar el DUT de forma tal que se cubra la mayor cantidad de estados posibles, tanto para las máquinas de estado como para las señales. Para esto se busca implementar dos escenarios. El primero contiene pruebas en las que se busca alta alternancia entre los dispositivos y entre las señales que ingresan al DUT. El segundo busca cubrir casos en los que las entradas al DUT contienen valores inválidos para ejercitar su comportamiento en dichos casos.

Escenario 1: Envío de paquetes comunes de un dispositivo a otro usando

Test 1.1 Envío de paquetes aleatorios:

Objetivo: Probar la capacidad del DUT que este caso es un Mesh por medio del envío de paquetes de un dispositivo a otro.

Descripción:
En este test se envían un número aleatorio de entre 3 a 300 mensajes desde cada uno de los 16 dispositivos conectados al Mesh hacia un dispositivo destino aleatorio es importante destacar que el dispositivo destino no será el mismo al dispositivo origen.
El tamaño del paquete enviado en el bus será aleatorio entre 20 y 80 bits (payload entre 3 y 63 bits).
Se asumirá una FIFO de entrada de tamaño infinito para asegurar que no se perderán paquetes por overflow.
El contenido de los paquetes de transacción será totalmente aleatorio en el rango de todos los valores posibles para el paquete de información.
Los paquetes se enviarán con retardos aleatorios independientes para cada uno de los dispositivos de entre 0 y 10 ciclos de reloj de diferencia.

Se correrán 100 semillas de esta prueba.

Criterio de aprobación: Se considera que el test será correcto si el payload que llega al dispositivo de destino es igual al del dispositivo de origen.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío en la fuente, tiempo de recibido en el destino, dirección de destino, dirección de fuente y retardo total de transito del paquete.

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, tamaño de los paquetes, contenido de los paquetes.
- Definición de Fifos de entrada de tamaño infinito.
- Constraint de mensaje con error apagado
- Constraint con dispositivo distinto al origen activado.

Test 1.2 Máxima alternancia en payloads:

Objetivo: Maximizar el consumo energético y obtener datos en función del tamaño de los paquetes y la cantidad de dispositivos en el sistema al mismo tiempo que se maximiza la cobertura de alternancia del DUT.

Descripción:
En este test se envían un número aleatorio de entre 3 a 100 mensajes desde cada uno de los dispositivos conectados al bus hacia un destinatario aleatorio que no sea el mismo dispositivo que envía el mensaje.
El número de dispositivos conectados al bus será aleatorizado entre 4 y 16 dispositivos.
El tamaño del paquete enviado en el bus será aleatorio entre 20 y 80 bits.
Se asumirá una FIFO de entrada de tamaño infinito para asegurar que no se perderán paquetes por desbordamiento.
El contenido de los paquetes de transacción será seleccionado de manera que siempre se siga un patrón en el que se cambien la mayor cantidad posible de bits entre paquetes consecutivos de forma que el primer paquete de la secuencia tenga todos los bits del paquete de información en F's, el siguiente en 0's, el siguiente en patrones de A's, el siguiente en patrones de 5's y luego se repetirá la secuencia.
Los paquetes se enviarán con retardos aleatorios entre 3 y 10 ciclos de reloj.

Se correrán 100 semillas de esta prueba.

Criterio de aprobación: Se considera que el test es exitoso si todos los paquetes enviados desde cada uno de los dispositivos alcanzan a su destinatario sin errores en el paquete de información.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío en la fuente, tiempo de recibido en el destino, dirección de destino, dirección de fuente y retardo total de transito del paquete.
Además se creará un reporte de los consumos de potencia de cada una de las pruebas por medio de la estimación de las capacitancias de cada uno de los nodos en el netlist del DUT y los respectivos factores de actividad. Dichos valores de potencia se presentarán en una gráfica de 3 dimensiones de número de dispositivos en el bus, número de bits en el paquete, contra potencia consumida

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, número de dispositivos conectados al bus, tamaño de los paquetes.
- Definición de Fifos de entrada de tamaño infinito.
- Capacidad para generar secuencias de payloads personalizadas
- Constraint de mensaje con error apagado
- Constraint con dispositivo distinto al origen activado.

Test 1.3 Alternancia entre dispositivos destino:

Objetivo: Con esta prueba se pretende ejercitar todos los dispositivos del DUT de manera que se envie un paquete a un dispositivo y después al que le sigue.

Descripción:
Con este test se generan paquetes que se envían desde un dispositivo origen que será aleatorio al igual que el payload, delay y el modo, hacia un dispositivo destino específico el cual irá variando en cada iteración del generador del 0 al 15 y luego se repetirá la secuencia. Por ejemplo en la iteración 0 del generador el dispositivo destino será 1, la iteración 15 el dispositivo destino será 16 y en la iteración 16 el dispositivo destino será 1. Se asumirá una FIFO de entrada de tamaño infinito para asegurar que no se perderán paquetes por desbordamiento. El número de mensajes será aleatorio entre 3 y 300. Cabe destacar que el dispositivo destino no puede ser igual al dispositivo origen. El tamaño del paquete enviado en el bus será aleatorio entre 20 y 80 bits. Los paquetes se enviarán con retardos aleatorios entre 3 y 10 ciclos de reloj. Se correrán un total de 100 semillas de esta prueba.

Criterio de aprobación: Se considera que el test es exitoso si todos los paquetes enviados desde cada uno de los dispositivos alcanzan a su destinatario sin errores en el paquete de información.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío, tiempo de recibido en el destino, dirección de destino, dirección de origen, retardo total de transito del paquete y el ancho de banda.

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, tamaño de los paquetes, destino origen, modo y el contenido del payload.
- Definición de Fifos de entrada de tamaño infinito.
- Capacidad para generar secuencias de destinos origen personalizados.
- Constraint de mensaje con error apagado
- Constraint con dispositivo distinto al origen activado.

Test 1.4 Busca de overflow de la Fifo:

Objetivo: Se pretende verificar el funcionamiento del DUT en casos de overflow de la Fifo de entrada.

Descripción:
Este test es idéntico al test 1.1 con la diferencia que la profundidad de la Fifo es finita y aleatoria. El contenido de los paquetes de transacción será totalmente aleatorio en el rango de todos los valores posibles para el paquete de información.
Los paquetes se enviarán con retardos aleatorios independientes para cada uno de los dispositivos de entre 0 y 10 ciclos de reloj de diferencia. Se correrán 100 semillas de esta prueba.

Criterio de aprobación: Se considera que el test será correcto si el payload que llega al dispositivo de destino es igual al del dispositivo de origen.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío en la fuente, tiempo de recibido en el destino, dirección de destino, dirección de fuente y retardo total de transito del paquete.

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, tamaño de los paquetes, contenido de los paquetes.
- Definición de Fifos de entrada de tamaño finito y aleatorio.
- Constraint de mensaje con error apagado
- Constraint con dispositivo distinto al origen activado.

Escenario 2: Comportamiento del DUT en casos de error

Test 2.1 Dirección de destino inválida

Objetivo: Se pretende ejercitar al DUT con transacciones con error, en este caso con direcciones de dispositivo destino no existentes de manera que se pueda conocer el comportamiento del DUT para estos casos..

Descripción:
En esta prueba se enviarán entre 3 y 300 transacciones desde dispositivos origen elegidos aleatoriamente al igual que el payload, modo y delay. Se diseñará un constraint donde el dispositivo destino puede tomar valor de uno existente (menor probabilidad de escogencia) o una dirección de destino inexistente (mayor probabilidad de escogencia). Los pesos de las 16 direcciones correctas será de 10 y se escogerán 4 direcciones inexistentes (8'b11110000, 8'b00001111, 8'b11111111, 8'b00000000) con un peso de 100.

Criterio de aprobación: Se considera que el test será correcto si el payload que llega al dispositivo de destino es igual al del dispositivo de origen.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío en la fuente, tiempo de recibido en el destino, dirección de destino, dirección de fuente y retardo total de transito del paquete.

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, tamaño de los paquetes, contenido de los paquetes.
- Definición de Fifos de entrada de tamaño infinito y aleatorio.
- Constraint de mensaje con error encendido.
- Constraint con dispositivo distinto al origen activado.

Test 2.2 Envío de un mensaje de un dispositivo a el mismo.

Objetivo: Se pretende conocer el funcionamiento del DUT en el caso de que se envíe un mensaje a sí mismo.

Descripción:
Para este caso el DUT será excitado con mensajes con la misma dirección de origen y destino, en otras palabras se estará enviando paquetes a sí mismo.
En esta prueba se enviará entre 3 y 300 transacciones desde dispositivos origen elegidos aleatoriamente al igual que el payload, modo y delay. Se diseñará un constraint que revise que la dirección de origen sea igual a la de destino.

Criterio de aprobación: Se considera que el test será correcto si el payload que llega al dispositivo de destino es igual al del dispositivo de origen.

Reportes: Al final de la prueba se imprimirá un reporte de cada uno de los paquetes enviados incluidos el tiempo de envío en la fuente, tiempo de recibido en el destino, dirección de destino, dirección de fuente y retardo total de transito del paquete.

Requerimientos para el ambiente:

- Capacidad de aleatorización de: número de paquetes enviados por dispositivo, tiempo de retardo en el envío entre paquetes individuales, tamaño de los paquetes, contenido de los paquetes.
- Definición de Fifos de entrada de tamaño infinito y aleatorio.
- Constraint de mensaje con error apagado.
- Constraint con dispositivo igual al origen activado.

Pasos para Correr el TestBench

Para correr el Testbench es necesario especificar el escenario y el número de prueba que se desea realizar, tales se muestran en el archivo llamado *Top.sv* en la clase Top. En la línea 40 se cambia dicho dato. Una vez compilado el programa se corre el archivo *testbench.sv*, además se debe de generar un archivo .csv como reporte de paquetes enviados y recibidos. Por medio de la aplicación GNUplot se grafican los datos.