

**Instituto Tecnológico de Costa Rica**

**Escuela de electrónica**

**Verificación funcional de circuitos integrados**

**Proyecto 1**

**Verificación DUT: Bus de datos para FIFOS**

**Profesor:**

Ronny García Ramírez

**Estudiantes:**

Susana Astorga Rodríguez

Mac Alfred Pinnock Chacón

**II Semestre, 2021**

**Cartago, Costa Rica**

**Índice**

[**Plan de pruebas (test plan, en inglés)**](#_idweugb0wbsd) **3**

# **Plan de pruebas (test plan, en inglés)**

**Módulos a utilizar:**

* agente.sv
* ambiente.sv
* checker.sv
* driver.sv
* scoreboard.sv
* interface\_de\_transacciones.sv
* test\_bench.sv

**Interconexión de los módulos:**

**Figura 1.** Conexión de los módulos de la prueba.

**Casos de uso común**

* Aleatorizacion de reset
* Aleatorizacion en el tiempo de envio de los paquetes
* Aleatorizacion de los datos

**Casos de esquina**

Se omite el caso de esquina en el que la fifo tiene los valores lógicos de lectura y escritura al mismo tiempo activos, debido a que se implementa una fifo de solamente software la cual es meramente secuencial. Ni tampoco se contempla una fifo con un tamaño parametrizable lo que implica que tampoco se contemplan errores por overflow o underflow.

Por lo cual se contemplan los siguientes casos de esquina:

* Llenado con únicamente ceros el payload.
* Llenado con únicamente unos y el payload.
* Llenado con ceros y unos.
* Ingreso de una dirección incorrecta en el BUS (fuera de las capacidades del bus).
* Envío de señales con broadcast.
* Envío de señales a un dispositivo de cierta cantidad de manejadores (10 manejadores).
* Envío de señales a un dispositivo al inicio del bus.
* Acceso de al menos dos FIFO’s al mismo tiempo.

**Paquetes de comunicación:**

Se utilizarán estructuras de datos en forma que se puedan acomodar como datos apilables en pilas o colas de datos accedidas y administradas mediante el uso de clases. Así mismo, los correos o “mailboxes” entre dispositivos representan la forma en que los paquetes son enviados, estos “mailboxes” son los siguientes, y tiene por nombre una forma clara para su identificación. Además se puede delimitar en esos “mailboxes” los datos que se están manejando y que están asociados a las clases de cada dispositivo.

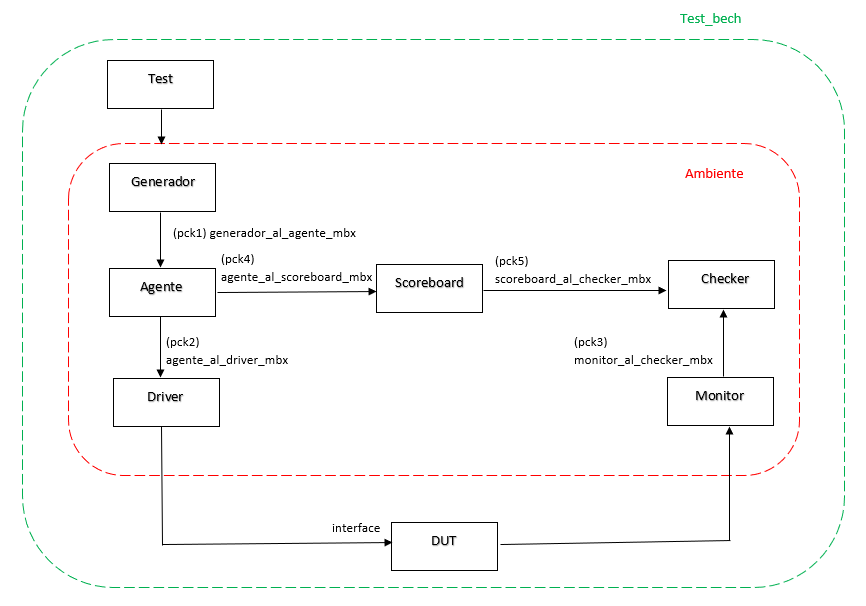
Mailboxes:

* monitor\_al\_checker
* agente\_al\_checker
* generador\_al\_agente
* agente\_al\_controlador
* agente\_al\_scoreboard
* scoreboard\_al\_checker
* agente\_al\_driver

Para lograr conectar el BUS de las FIFO’s con el agente y driver y el monitor se crean dos clases dentro del archivo de interfaz\_de\_transacciones. Estas clases son las siguientes:

* entrada\_del\_DUT
* salida\_del\_DUT

Con esto logramos que se puedan realizar funciones de pop y push a todas las FIFO’s disponibles a través del bus que se está poniendo a prueba. Todo el escenario de prueba está de roeintado en el uso de clases en cada uno de los archivos, y luego estos archivos son agregados como librerías al testbench principal que se encarga de llamar al ambiente y este a su vez inicializa de forma paralela las otras instancias del ambiente de pruebas. Esto se puede evidenciar en la figura 2 a continuación, en donde los módulos son llamados según su orden jerárquico en la estructura del ambiente.



**Figura 2.** Estructura del ambiente.

Paquetes de transaccion:

* pck1: Trans\_scoreboard\_checker
* tama\_paquete
* \*controladores
* caso
* opcion (esquina)
* mensaje
* pck3: simul\_fifo
* tama\_de\_paquete
* controladores
* BITS
* pck4: Agente al scoreboard (solicitud)
* reporte\_completo
* retardo\_completo
* pck5: tipos\_de\_transaccion
* llenado\_aleaotorio
* llenado\_específico
  + opcion:
    - ceros
    - unos
    - cerosyunos
    - 2dat
    - broadcast
    - direccion\_incorrecta
    - cerca\_bus
    - lejos\_bus
* pck6: salida\_DUT
* numero\_fifo
* Dato\_push
* retraso
* pck7: entrada\_DUT
* destino
* numero\_fifo
* Dato\_pop
* contenido (dato creado random)