Verificación Funcional De Circuitos Integrados

Proyecto 2

Estudiantes

Ivannia Fernández Rodríguez 2020026764

Irán Medina Aguilar 2020146906

Profesor: Ronny García Ramírez

Semestre II 2023

La aleatorización del número de transacciones, de la terminal que envía, de la terminal que recibe, del dato a enviar, y del retardo se realizó en el agente. En el siguiente fragmento se puede ver como se aleatoriza el número de transacciones para que vaya en un rango entre 1 y la profundidad, y en el caso de lo demás se aleatoriza con la instrucción transaccion.randomize.

```
varios_dispositivos_envio_recibido: begin //Caso en el que se envían varios paquetes aleatorios desde cualquier dispositivo hacia cualquier otro
  num_transacciones = $urandom_range(1, 32); //Define un número aleatorio de transacciones
    for(int i = 0; i< num_transacciones; i++)begin
        espera = 0;
        transaccion = new();
        transaccion.randomize();
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin
          @(posedge vif.clk_i)
            espera = espera +1;
        end

        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia);
    end

end
```

El parámetro de la profundidad de las fifos de entrada se aleatorizó en el testbench con el uso de my_package por lo que este es aleatorio pero una vez se definen para el resto de la prueba se mantienen fijos.

```
module test_bench();

    reg clk = 0;

    import my_package::*;

    //Definición de la variable virtual
    mesh_if #(.ROWS(my_package::ROWS), .COLUMNS(my_package::COLUMNS), .PAKG_SIZE(my_package::PAKG_SIZE), .FIFO_DEPTH(my_package::FIFO_DEPTH)) _if(.c

    always #5 clk = ~clk;

    //Definición del test
    test #(.ROWS(my_package::ROWS), .COLUMNS(my_package::COLUMNS), .PAKG_SIZE(my_package::PAKG_SIZE), .FIFO_DEPTH(my_package::FIFO_DEPTH)) test_inst

    //Instancia del dispositivo
    mesh_gnrtr #(.ROWS(my_package::ROWS), .COLUMS(my_package::COLUMNS), .pckg_sz(my_package::PAKG_SIZE), .fifo_depth(my_package::FIFO_DEPTH), .bdcst
    (
        .pndng(_if.pndng),
        .data_out(_if.data_out),
        .popin(_if.popin),
        .pop(_if.pop),
        .data_out_i_in(_if.dato_out_i_in),
        .pndng_i_in(_if.pdng_i_in),
        .clk(_if.clk_i),
        .reset(_if.rst_i)
    );
```

```
module generador_parametros;

    class A;

        integer f;

        rand int unsigned ROWS;
        rand int unsigned COLUMNS;
        rand int unsigned PAKG_SIZE;
        rand int unsigned FIFO_DEPTH;
        rand int unsigned BROADCAST;


        constraint C0 { ROWS == 4;}
        constraint C1 { COLUMNS == 4;}
        constraint C2 { PAKG_SIZE == 32;}
        constraint C3 { FIFO_DEPTH > 0 ; FIFO_DEPTH < 17;}
        constraint C4 { BROADCAST == {8{1'b1}};}

        function void printPackage;
            f = $fopen("my_package.sv", "w");
            $fdisplay(f, "package my_package;");
            $fdisplay(f, "  parameter ROWS = %0d;", ROWS);
            $fdisplay(f, "  parameter COLUMNS = %0d;", COLUMNS );
            $fdisplay(f, "  parameter PAKG_SIZE = %0d;", PAKG_SIZE );
            $fdisplay(f, "  parameter FIFO_DEPTH = %0d;", FIFO_DEPTH );
            $fdisplay(f, "  parameter broadcast = %0d;", BROADCAST );
            $fdisplay(f, "endpackage");
        endfunction

    endclass

A a;

initial begin
    a = new();
    a.randomize();
    a.printPackage();
end
```

Implementación de los escenarios de uso común

Los escenarios de uso común se implementaron en el agente de la siguiente forma:

- Envío de un solo paquete aleatorio por parte de cualquier dispositivo a cualquiera de los otros terminales dentro del rango aleatorio de terminales existentes.
-

```
un_paquete: begin //Caso en el que se envía un solo paquete aleatorio desde cualquier dispositivo hacia cualquier otro dispositivo
        espera = 0;
        transaccion = new();
        transaccion.randomize(); //Vuelve aleatorios los valores de la transacción
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin //Hace el retardo antes del envío
          @(posedge vif.clk_i)
            espera = espera + 1;
        end
        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion); //Envía la transacción al mailbox del agente al driver en la posición de la ter
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia); //Envía la transacción al mailbox del agente al scoreboard

end
```

- Envío de una cantidad aleatoria de paquetes aleatorios por parte uno o varios terminales cualesquiera a cualquiera de los otros terminales dentro de un rango aleatorio de terminales existentes.

```
varios_dispositivos_envio_recibido: begin //Caso en el que se envían varios paquetes aleatorios desde cualquier dispositivo hacia cualquier otro
  num_transacciones = $urandom_range(1, 32); //Define un número aleatorio de transacciones
    for(int i = 0; i< num_transacciones; i++)begin
        espera = 0;
        transaccion = new();
        transaccion.randomize();
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin
          @(posedge vif.clk_i)
            espera = espera +1;
        end

        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia);
    end

end
```

- Llenado de todas las FIFOS de todos los dispositivos existentes con datos aleatorios.

```
llenado_fifos: begin //Caso en el que se llenan las FIFOS de todos los drivers disponibles

    for(int i = 0; i < 16 ; i++)begin //For para recorrer todas las filas disponibles
        for (int j = 0; j < FIFO_DEPTH; j++)begin//For para llenar una por una las FIFOS
          transaccion = new();
          transaccion.randomize(); //Vuelve aleatorios los valores de la transacción
          transaccion.fun_pckg;
          transaccion.tiempo_envio  = $time;
          agente_drv_mbx[i].put(transaccion); //Envía la transacción al mailbox del agente al driver en la posición de la terminal de envío
          transaccion_copia = new();
          transaccion_copia = transaccion;
          agente_sb_mbx.put(transaccion_copia);
        end
    end
end
```

Implementación de los escenarios de esquina

Los escenarios de esquina se implementaron en el agente de la siguiente forma:

- Reset antes del envío de una cantidad aleatoria de paquetes aleatorios por parte uno o varios terminales cualesquiera a cualquiera de los otros terminales dentro de un rango aleatorio de terminales existentes.
-

```
reset_inicio: begin //Caso de esquina en el que se realiza un reset antes de que se envíe alguna transacción
  vif.rst_i = '1;
  @(posedge vif.clk_i);
  vif.rst_i = '0;
  num_transacciones = $urandom_range(1, 32); //Define un número aleatorio de transacciones
    for(int i = 0; i< num_transacciones; i++)begin
        espera = 0;
        transaccion = new();
        transaccion.randomize();
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin
          @(posedge vif.clk_i)
            espera = espera +1;
        end

        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia);
    end

end
```

- Reset a la mitad del envío de una cantidad aleatoria de paquetes aleatorios por parte uno o varios terminales cualesquiera a cualquiera de los otros terminales dentro de un rango aleatorio de terminales existentes.
-

```
reset_mitad: begin //Caso de esquina en el que se realiza un reset a la mitad del envío de las transacciones
  num_transacciones = $urandom_range(1,FIFO_DEPTH);
    for(int i = 0; i< num_transacciones/2; i++)begin
        espera = 0;
        transaccion = new();
        transaccion.randomize();
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin
          @(posedge vif.clk_i)
            espera = espera +1;
        end

        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia);
    end

    vif.rst_i = '1;
    @(posedge vif.clk_i);
    vif.rst_i = '0;

    for(int i = 0; i< num_transacciones/2; i++)begin
        espera = 0;
        transaccion = new();
        transaccion.randomize();
        transaccion.fun_pckg;
        while (espera < transaccion.tiempo_retardo) begin
          @(posedge vif.clk_i)
            espera = espera +1;
        end
```

```
        transaccion.tiempo_envio  = $time;
        transaccion.print();
        agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
        transaccion_copia = new();
        transaccion_copia = transaccion;
        agente_sb_mbx.put(transaccion_copia);
    end
```

- Reset después del envío de una cantidad aleatoria de paquetes aleatorios por parte uno o varios terminales cualesquiera a cualquiera de los otros terminales dentro de un rango aleatorio de terminales existentes.

```
reset_final: begin  //Caso de esquina en el que se realiza un reset anteal final de que se envían las transacciones

  num_transacciones = $urandom_range(1,FIFO_DEPTH);
   for(int i = 0; i< num_transacciones/2; i++)begin
      espera = 0;
      transaccion = new();
      transaccion.randomize();
      transaccion.fun_pckg;
      while (espera < transaccion.tiempo_retardo) begin
        @(posedge vif.clk_i)
          espera = espera +1;
      end

      transaccion.tiempo_envio  = $time;
      transaccion.print();
      agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
      transaccion_copia = new();
      transaccion_copia = transaccion;
      agente_sb_mbx.put(transaccion_copia);
   end
  vif.rst_i = '1;
  @(posedge vif.clk_i);
  vif.rst_i = '0;
end
```

- Envío de una cantidad aleatoria de paquetes aleatorios por parte de uno o varios terminales cualesquiera a cualquier dirección fuera de un rango aleatorio de terminales existentes.

```
envio_fuera_de_rango: begin //Caso de esquina en el que se envían transacciones a una dirección fuera del rango de terminales existentes
  num_transacciones = $urandom_range(1,FIFO_DEPTH);
   for(int i = 0; i< num_transacciones; i++)begin
      espera = 0;
      transaccion = new;
      transaccion.randomize();
      transaccion.row = $urandom_range(ROWS+1,ROWS +20) ;
      transaccion.colum = $urandom_range(COLUMNS + 1, COLUMNS +20);
      transaccion.fun_pckg;
      while (espera < transaccion.tiempo_retardo) begin
        @(posedge vif.clk_i)
          espera = espera +1;
      end
      transaccion.tiempo_envio  = $time;
      transaccion.print();
      agente_drv_mbx[transaccion.terminal_envio].put(transaccion);
      transaccion_copia = new();
      transaccion_copia = transaccion;
      agente_sb_mbx.put(transaccion_copia);
   end

end
```

Generación de datos

El retraso promedio en la entrega de paquetes por terminal y general en función de la cantidad de dispositivos y la profundidad de las fifos, el ancho de banda promedio máximo y mínimo en función de la cantidad de dispositivos y la profundidad de las fifos, y la generación del reporte de los paquetes enviados y recibidos en formato

csv con el tiempo de envío, terminal de procedencia, terminal de destino, tiempo de recibido, y el retraso en el envío, se generaron en el scoreboard donde el código con lo anterior se encuentra a continuación.

```
forever begin

    // Inicializa los datos
   #1;

    bw = 0;
    if(test_sb_mailbox.num() > 0)begin

        test_sb_mailbox.get(transaccion_test);

        case (transaccion_test)

            reporte: begin
                //$display("Mae si era una transacciones de reporte");
                tiempo = 0;
                linea = "";
                linea_agregar = "";
                informacion = {};


                for (int i=0; i < verificadas.size(); ++i) begin


                    transaccion_auxiliar = new();
                    transaccion_auxiliar = verificadas[i];
                    tiempo = tiempo + transaccion_auxiliar.latencia;

                    $sformat(linea_agregar, "%h,%g,%g,%g,%g,%g,%g,%g\n",

                    transaccion_auxiliar.pckg,
                    transaccion_auxiliar.t_envio,
                    transaccion_auxiliar.t_recibido,
                    transaccion_auxiliar.terminal_envio,
                    transaccion_auxiliar.terminal_recibido,
                    transaccion_auxiliar.latencia,
                    transaccion_auxiliar.prof,
                    transaccion_auxiliar.dispositivos

                    );
```

```
            informacion.push_back(linea_agregar);


        end
        archivo_1 = $fopen("Reporte_transacciones.csv", "a" );
        for (int i=0; i<informacion.size(); ++i) begin
            $fwrite(archivo_1, "%s", informacion[i]);
        end
        $fclose(archivo_1);
        tpromedio = tiempo / verificadas.size();
        bw = width * 10e9 / (tpromedio);

        $display("Para una prueba con Terminales: [%g], Profundidad: [%g], Ancho de palabra: [%g], Tiempo promedio: [%g], Ancho de banda: [
        archivo_2 = $fopen("Reporte_Anchos_de_banda_Tiempo_promedio.csv", "a" );
        $sformat (linea, "\n%g,%g,%g,%g,%g", ROWS, COLUMNS, FIFO_DEPTH, PAKG_SIZE, tpromedio, bw);
        $fwrite(archivo_2, "%s", linea);
        $fclose(archivo_2);

    end
    default: begin
        $display("No se recibio ninguna instrucción de reporte valida desde el test");
    end
```

Resultados de las pruebas

Reporte de transacciones en .csv

Tiempo promedio y ancho de banda en .csv según la profundidad, el ancho, y los dispositivos

Se implemento código en el scoreboard para imprimir los resultados, sin embargo, este no está cumpliendo con imprimir el reporte. Por lo tanto, no se está logrando registrar los reportes.

Hay que modificar el scoreboard.