Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
Carrera de Maestría en Electrónica

Verificación Funcional
MP6134
Prof. Gerardo Castro Jiménez

Tarea #2

Notas generales:

- La tarea puede realizarse en grupos de 3-4 personas.
- Se calificará con una nota de cero si se demuestra cualquier tipo de plagio.
- Fecha de entrega: 29 de noviembre del 2018, a las 6pm.

Entregable:

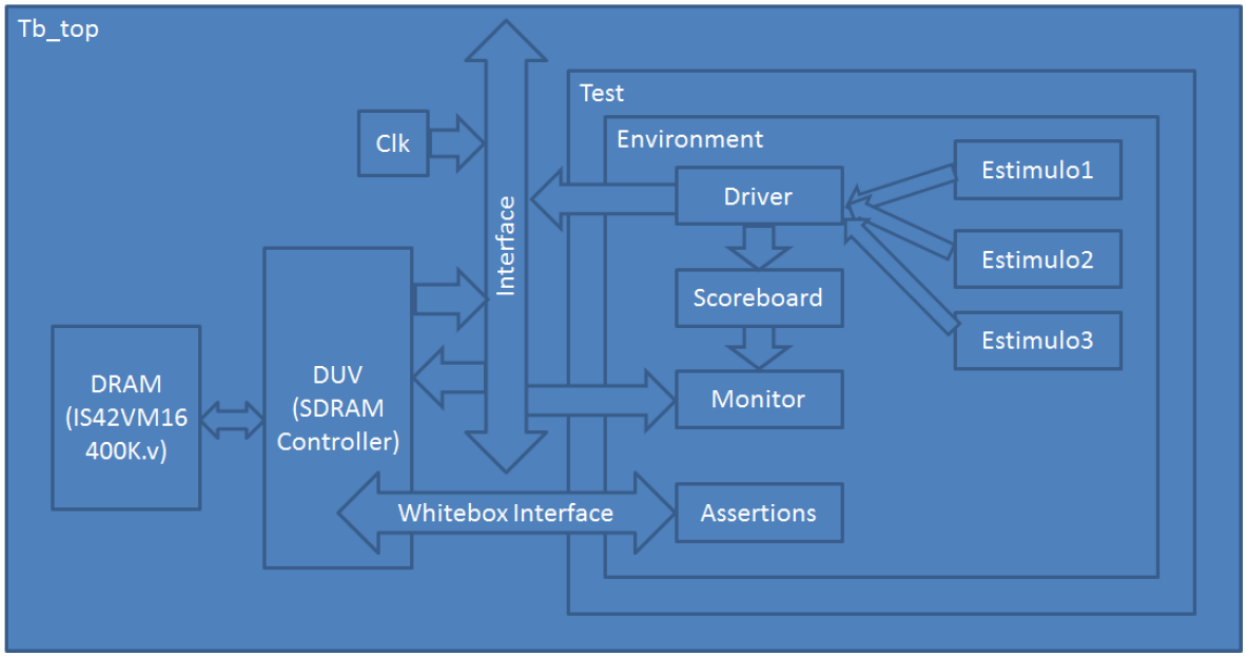
- Archivo comprimido extensión .tar.gz con los archivos del proyecto.

Evaluación:

- Se realizará una revisión del proyecto el día de la fecha de entrega con cada grupo de trabajo.

Con base en la tarea programa #1, modifique el código del archivo tb_top.sv para implementar un testbench como se muestra en la siguiente figura:

- **Se debe utilizar herencia para crear una clase "environment2" que reutilice el "environment" de la tarea #1.**



Indicaciones:

- Utilizar restricciones y aleatoriedad para reemplazar el tipo de código mostrado abajo, por clases estimulo que utiliza la clase driver como se muestra en la figura de color azul:

```
$display("---------------------------------------");
  $display(" Case:5 24 Write & 24 Read With Different Bank and Row ");
  $display("---------------------------------------");
  //---------------------------------------
  // Address Decodeing:
  //  with cfg_col bit configured as: 00
  //    <12 Bit Row> <2 Bit Bank> <8 Bit Column> <2'b00>
  //
  burst_write({12'h000,2'b00,8'h00,2'b00},8'h4);   // Row: 0 Bank : 0

burst_write({12'h000,2'b01,8'h00,2'b00},8'h5);   // Row: 0 Bank : 1
burst_write({12'h000,2'b10,8'h00,2'b00},8'h6);   // Row: 0 Bank : 2
burst_write({12'h000,2'b11,8'h00,2'b00},8'h7);   // Row: 0 Bank : 3
burst_write({12'h001,2'b00,8'h00,2'b00},8'h4);   // Row: 1 Bank : 0
burst_write({12'h001,2'b01,8'h00,2'b00},8'h5);   // Row: 1 Bank : 1
burst_write({12'h001,2'b10,8'h00,2'b00},8'h6);   // Row: 1 Bank : 2
burst_write({12'h001,2'b11,8'h00,2'b00},8'h7);   // Row: 1 Bank : 3
```

- Modificar el scoreboard para que sea capaz de manejar escrituras y lecturas de forma aleatoria (en desorden).
- Reemplazar en todos los archivos del RTL el siguiente tipo de código por aserciones:

```
always @(posedge wb_clk_i) begin
      if (cmdfifo_full == 1'b1 && cmdfifo_wr == 1'b1)  begin
     $display("ERROR:%m COMMAND FIFO WRITE OVERFLOW");
     end
end
```

- Crear un bloque interface llamado Whitebox que incluya las señales internas del DUV. Por ejemplo:

```
Parameter TOP_PATH = top.dut;

Interface whitebox();
      logic var;
      assign var = `TOP_PATH.module.submodule.signal;
endinterface
```

- Crear un bloque *assertion* que reciba la interface Whitebox e incluya aserciones concurrentes para la inicialización de la DRAM y comprobar su funcionamiento:

INITIALIZATION

SDRAM must be powered up and initialized in a predefined manner. Operational procedures other than those specified may result in undefined operation. Once power is applied to VDD and the clock is stable, the SDRAM requires a 100 µs delay prior to issuing any command other than a COMMAND INHIBIT or NOP. Starting at some point during this 100 µs period and continuing at least through the end of this period, COMMAND INHIBIT or NOP commands should be applied.

Once the 100 µs delay has been satisfied with at least one COMMAND INHIBIT or NOP command having been applied, a PRECHARGE command should be applied. All device banks must then be precharged, thereby placing the device in the all banks idle state. Once in the idle state, two AUTO REFRESH cycles must be performed. After two refresh cycles are complete, SDRAM ready for mode register programming. Because the mode registers will power up in unknown state, it should be loaded prior to applying any operational command.

- Dentro del bloque *assertion*, crear aserciones para las siguientes reglas del protocolo WISHBONE revisión b4 y comprobar su funcionamiento:
    - Regla 3.00, 3.05, 3.10, 3.25, 3.35.