Task 2 - Synchronous RAM Module (Verilog)

Objective: Develop a synchronous RAM module with read and write operations.

Verilog Code (simple_ram.v)

```
module simple_ram #(
    parameter DATA_WIDTH = 8,
    parameter ADDR_WIDTH = 4
)
(
    input                       clk,
    input                       we,
    input      [ADDR_WIDTH-1:0] addr,
    input      [DATA_WIDTH-1:0] d_in,
    output reg [DATA_WIDTH-1:0] d_out
);

  reg [DATA_WIDTH-1:0] mem [(2**ADDR_WIDTH)-1:0];

  always @(posedge clk) begin
    if (we)
      mem[addr] <= d_in;
    d_out <= mem[addr];
  end

endmodule
```

Testbench Code (simple_ram_tb.v)

```
module simple_ram_tb;
  reg clk;
  reg we;
  reg [3:0] addr;
  reg [7:0] d_in;
  wire [7:0] d_out;

  simple_ram #(8, 4) uut (
    .clk(clk),
    .we(we),
    .addr(addr),
    .d_in(d_in),
    .d_out(d_out)
  );

  initial begin
    clk = 0;
    forever #5 clk = ~clk;
  end

  initial begin
      $monitor("Time %t: we=%b, addr=%h, d_in=%h, d_out=%h", $time, we, addr, d_in,
d_out);
    we = 0;
    addr = 0;
    d_in = 0;
    #10;
    we = 1;
    addr = 4'h1; d_in = 8'hAA; #10;
    addr = 4'h2; d_in = 8'h55; #10;
    we = 0;
    #10;
    addr = 4'h1; #10;
    addr = 4'h2; #10;
    $finish;
  end

endmodule
```