```
// Eugene Ngo
// 1/20/2023
// EE 371
// Lab 2 Task 3

// DE1_SoC is the top-level module that defines the I/Os for the DE1 SoC board.
// DE1_SoC is the top-level module for the entire FIFO system. It instantiates the FIFO module. Additionally, it
// connects KEY3 to the read signal, KEY2 to the write signal, and KEY0 to the reset signal. It also displays the
// content of the input bus in hexadecimal on HEX5 and HEX4, and displays the content of the output bus in hexadecimal
// on HEX1 and HEX0.

module DE1_SoC_task3 (HEX0, HEX1, HEX2, HEX3, HEX4, HEX5, KEY, SW, LEDR, CLOCK_50);
   output logic [6:0]  HEX0, HEX1, HEX2, HEX3, HEX4, HEX5;
   output logic [9:0]  LEDR;
   input  logic [3:0]  KEY;
   input  logic [9:0]  SW;
   input logic CLOCK_50;

   // turning unused hex displays off
   assign HEX3 = 7'b1111111;
   assign HEX2 = 7'b1111111;

   // Setting up buses
   logic [7:0] inputBus, outputBus;
   assign inputBus = SW[7:0];

   // Cleaning up button presses to be one clock cycle long.
   logic reset, read, write;
   // ip1-ip3 takes the corresponding buttons as inputs, and outputs to intermediate logics that will be used by
   // the FIFO module.
   input_process ip1 (.A(~KEY[0]), .out(reset), .clk(CLOCK_50));
   input_process ip2 (.A(~KEY[3]), .out(read), .clk(CLOCK_50));
   input_process ip3 (.A(~KEY[2]), .out(write), .clk(CLOCK_50));

   // Setting up the drivers for the hex displays.
   // hex5, hex4, hex1, and hex0 all takes parts of the I/O bus as inputs, and outputs to the corresponding hex
   // displays on the DE1-SoC board.
   seg7 hex5 (.in(inputBus[7:4]), .out(HEX5));
   seg7 hex4 (.in(inputBus[3:0]), .out(HEX4));
   seg7 hex1 (.in(outputBus[7:4]), .out(HEX1));
   seg7 hex0 (.in(outputBus[3:0]), .out(HEX0));

   // FIFO takes the processed input from the input_processing modules, takes the data from the input bus, uses the
   // 50MHz clock, outputs "full" and "empty" signals to the corresponding LEDs, and outputs data onto the output Bus.
   FIFO queue (.clk(CLOCK_50), .reset, .read, .write, .inputBus, .empty(LEDR[8]), .full(LEDR[9]), .outputBus);

endmodule

// DE1_SoC_task3_testbench tests all expected, unexpected, and edgecase behaviors
`timescale 1 ps / 1 ps
module DE1_SoC_task3_testbench();
   logic [6:0]  HEX0, HEX1, HEX2, HEX3, HEX4, HEX5;
   logic [9:0]  LEDR;
   logic [3:0]  KEY;
   logic [9:0]  SW;
   logic clk;

   DE1_SoC_task3 dut (.HEX0, .HEX1, .HEX2, .HEX3, .HEX4, .HEX5, .KEY, .LEDR, .SW, .CLOCK_50(clk));

   // Setting up logics to make the code more readable.
   logic reset, read, write;
   logic [7:0] inputBus;
   assign KEY[0] = ~reset;
   assign KEY[3] = ~read;
   assign KEY[2] = ~write;
   assign SW[7:0] = inputBus;

   // Setting up a simulated clock.
   parameter CLOCK_PERIOD = 100;
   initial begin
      clk <= 0;
      forever #(CLOCK_PERIOD/2) clk <= ~clk; // Forever toggle the clock
   end

   initial begin
      reset <= 1; repeat(5) @(posedge clk);
      // put in excessive data;
      reset <= 0; read <= 0; write <= 1; inputBus <= 8'h00;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h01;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h02;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h03;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h04;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h05;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h06;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h07;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h08;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h09;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0a;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0b;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0c;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0d;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0e;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h0f;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             // attempt to write more data
                             write <= 1; inputBus <= 8'h10;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h11;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h12;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h13;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h14;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
      // attempt to read off 20 sets of data
                  read <= 1; write <= 0;                              @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);
                  read <= 1;                                          @(posedge clk);
                  read <= 0;                                          @(posedge clk);

      // simultaneous read/write
      reset <= 0; read <= 0; write <= 1; inputBus <= 8'h00;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h01;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h02;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h03;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
                             write <= 1; inputBus <= 8'h04;           @(posedge clk);
                             write <= 0;                              @(posedge clk);
      reset <= 0; read <= 1; write <= 1; inputBus <= 8'h05;           @(posedge clk);
                  read <= 0; write <= 0;                              @(posedge clk);
                  read <= 1; write <= 1; inputBus <= 8'h06;           @(posedge clk);
                  read <= 0; write <= 0;                              @(posedge clk);
                  read <= 1; write <= 1; inputBus <= 8'h07;           @(posedge clk);
                  read <= 0; write <= 0;                              @(posedge clk);
                  read <= 1; write <= 1; inputBus <= 8'h08;           @(posedge clk);
                  read <= 0; write <= 0;                              @(posedge clk);
                  read <= 1; write <= 1; inputBus <= 8'h09;           @(posedge clk);
                  read <= 0; write <= 0;                              @(posedge clk);
      $stop;
   end
endmodule
```