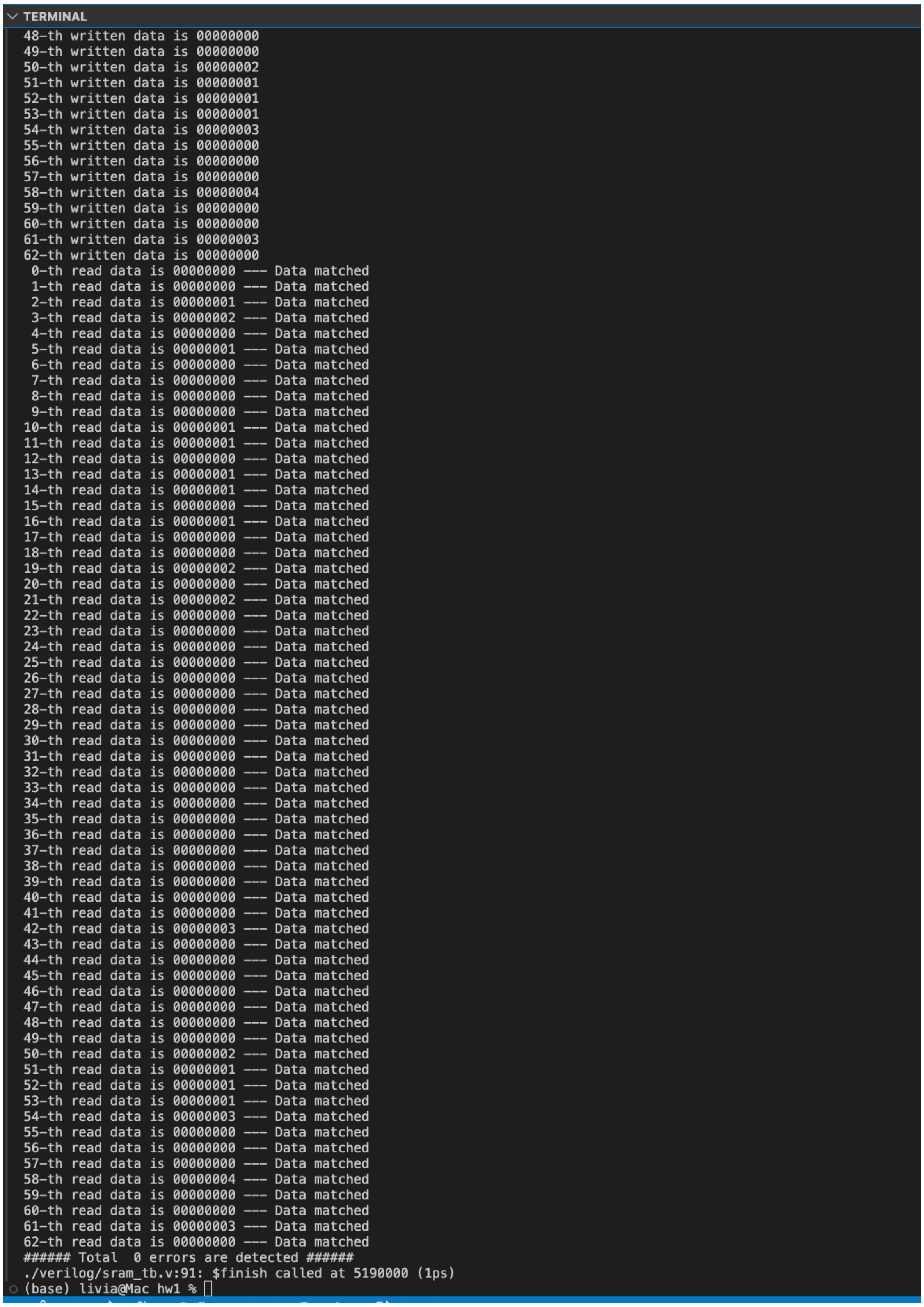
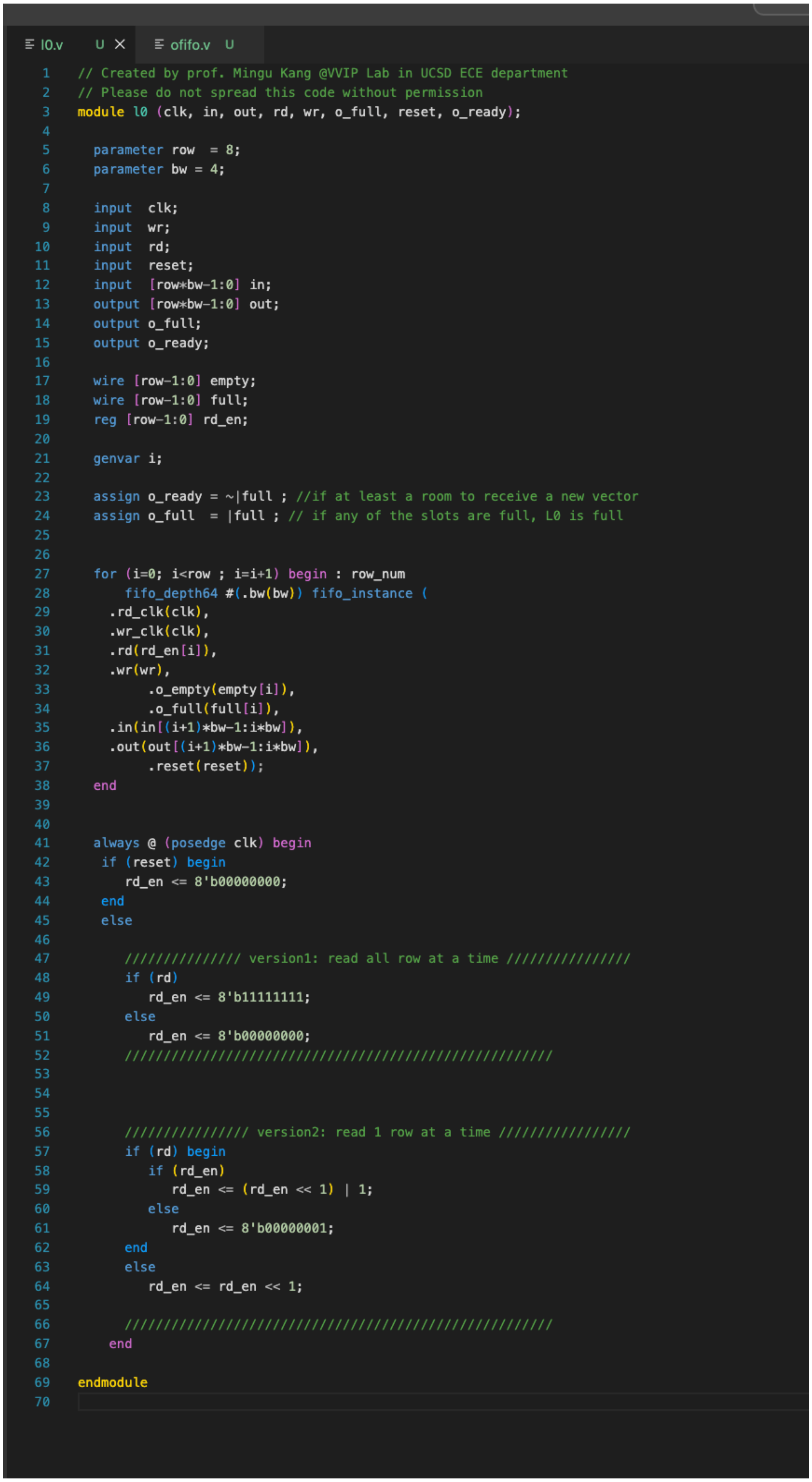

TERMINAL

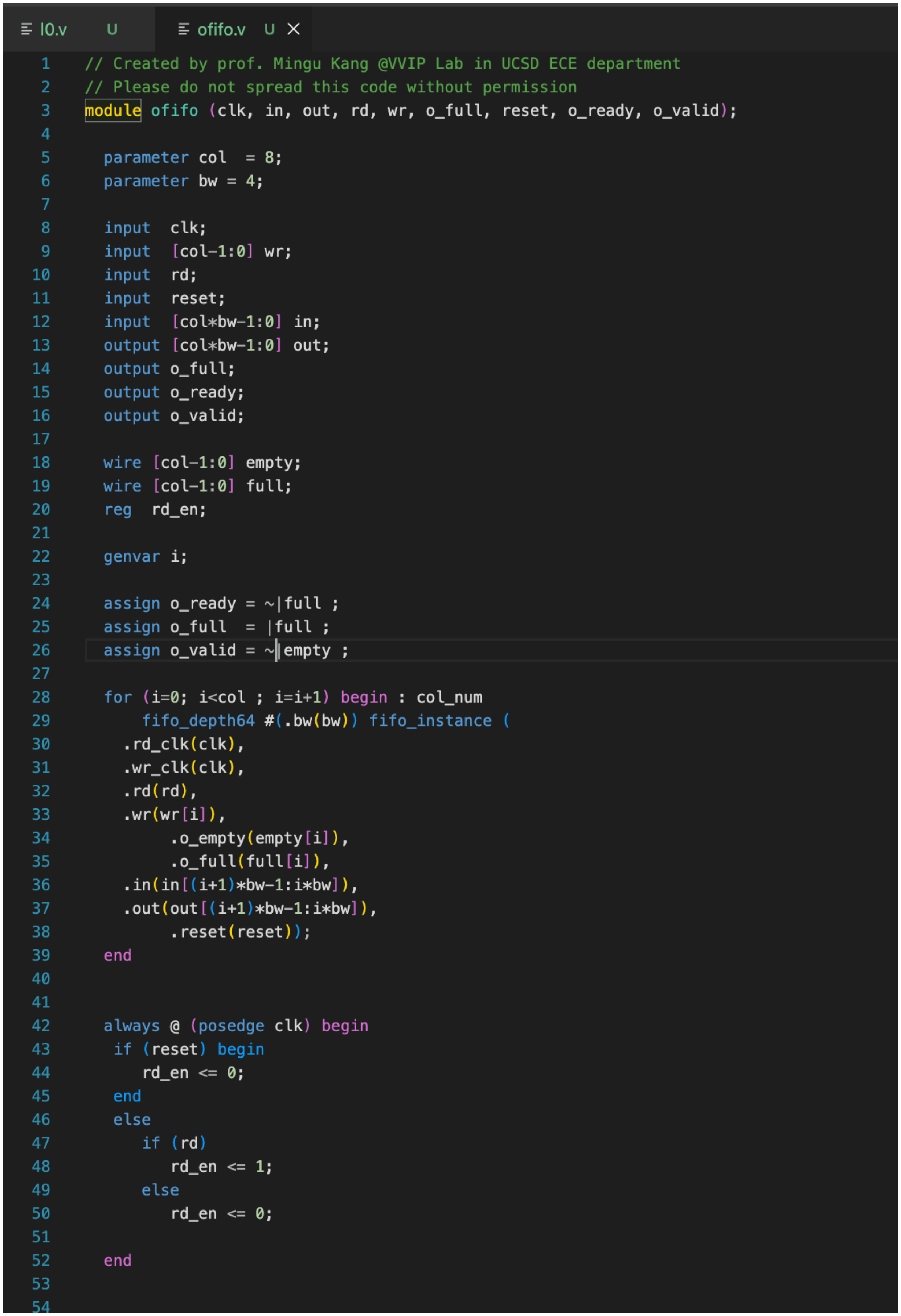
```
48-th written data is 00000000
49-th written data is 00000000
50-th written data is 00000002
51-th written data is 00000001
52-th written data is 00000001
53-th written data is 00000001
54-th written data is 00000003
55-th written data is 00000000
56-th written data is 00000000
57-th written data is 00000000
58-th written data is 00000004
59-th written data is 00000000
60-th written data is 00000000
61-th written data is 00000003
62-th written data is 00000000
 0-th read data is 00000000 ---- Data matched
 1-th read data is 00000000 ---- Data matched
 2-th read data is 00000001 ---- Data matched
 3-th read data is 00000002 ---- Data matched
 4-th read data is 00000000 ---- Data matched
 5-th read data is 00000001 ---- Data matched
 6-th read data is 00000000 ---- Data matched
 7-th read data is 00000000 ---- Data matched
 8-th read data is 00000000 ---- Data matched
 9-th read data is 00000000 ---- Data matched
10-th read data is 00000001 ---- Data matched
11-th read data is 00000001 ---- Data matched
12-th read data is 00000000 ---- Data matched
13-th read data is 00000001 ---- Data matched
14-th read data is 00000001 ---- Data matched
15-th read data is 00000000 ---- Data matched
16-th read data is 00000001 ---- Data matched
17-th read data is 00000000 ---- Data matched
18-th read data is 00000000 ---- Data matched
19-th read data is 00000002 ---- Data matched
20-th read data is 00000000 ---- Data matched
21-th read data is 00000002 ---- Data matched
22-th read data is 00000000 ---- Data matched
23-th read data is 00000000 ---- Data matched
24-th read data is 00000000 ---- Data matched
25-th read data is 00000000 ---- Data matched
26-th read data is 00000000 ---- Data matched
27-th read data is 00000000 ---- Data matched
28-th read data is 00000000 ---- Data matched
29-th read data is 00000000 ---- Data matched
30-th read data is 00000000 ---- Data matched
31-th read data is 00000000 ---- Data matched
32-th read data is 00000000 ---- Data matched
33-th read data is 00000000 ---- Data matched
34-th read data is 00000000 ---- Data matched
35-th read data is 00000000 ---- Data matched
36-th read data is 00000000 ---- Data matched
37-th read data is 00000000 ---- Data matched
38-th read data is 00000000 ---- Data matched
39-th read data is 00000000 ---- Data matched
40-th read data is 00000000 ---- Data matched
41-th read data is 00000000 ---- Data matched
42-th read data is 00000003 ---- Data matched
43-th read data is 00000000 ---- Data matched
44-th read data is 00000000 ---- Data matched
45-th read data is 00000000 ---- Data matched
46-th read data is 00000000 ---- Data matched
47-th read data is 00000000 ---- Data matched
48-th read data is 00000000 ---- Data matched
49-th read data is 00000000 ---- Data matched
50-th read data is 00000002 ---- Data matched
51-th read data is 00000001 ---- Data matched
52-th read data is 00000001 ---- Data matched
53-th read data is 00000001 ---- Data matched
54-th read data is 00000003 ---- Data matched
55-th read data is 00000000 ---- Data matched
56-th read data is 00000000 ---- Data matched
57-th read data is 00000000 ---- Data matched
58-th read data is 00000004 ---- Data matched
59-th read data is 00000000 ---- Data matched
60-th read data is 00000000 ---- Data matched
61-th read data is 00000003 ---- Data matched
62-th read data is 00000000 ---- Data matched
###### Total  0 errors are detected ######
./verilog/sram_tb.v:91: $finish called at 5190000 (1ps)
(base) livia@Mac hw1 %
```

l0.v



```verilog
// Created by prof. Mingu Kang @VVIP Lab in UCSD ECE department
// Please do not spread this code without permission
module l0 (clk, in, out, rd, wr, o_full, reset, o_ready);

  parameter row  = 8;
  parameter bw = 4;

  input  clk;
  input  wr;
  input  rd;
  input  reset;
  input  [row*bw-1:0] in;
  output [row*bw-1:0] out;
  output o_full;
  output o_ready;

  wire [row-1:0] empty;
  wire [row-1:0] full;
  reg [row-1:0] rd_en;

  genvar i;

  assign o_ready = ~|full ; //if at least a room to receive a new vector
  assign o_full  = |full ; // if any of the slots are full, L0 is full


  for (i=0; i<row ; i=i+1) begin : row_num
      fifo_depth64 #(.bw(bw)) fifo_instance (
    .rd_clk(clk),
    .wr_clk(clk),
    .rd(rd_en[i]),
    .wr(wr),
         .o_empty(empty[i]),
         .o_full(full[i]),
    .in(in[(i+1)*bw-1:i*bw]),
    .out(out[(i+1)*bw-1:i*bw]),
         .reset(reset));
  end


  always @ (posedge clk) begin
   if (reset) begin
      rd_en <= 8'b00000000;
   end
   else

      /////////////// version1: read all row at a time ////////////////
      if (rd)
         rd_en <= 8'b11111111;
      else
         rd_en <= 8'b00000000;
      ///////////////////////////////////////////////////////



      //////////////// version2: read 1 row at a time /////////////////
      if (rd) begin
         if (rd_en)
            rd_en <= (rd_en << 1) | 1;
         else
            rd_en <= 8'b00000001;
      end
      else
         rd_en <= rd_en << 1;

      ///////////////////////////////////////////////////////
    end

endmodule
```

ofifo.v



```
// Created by prof. Mingu Kang @VVIP Lab in UCSD ECE department
// Please do not spread this code without permission
module ofifo (clk, in, out, rd, wr, o_full, reset, o_ready, o_valid);

  parameter col  = 8;
  parameter bw = 4;

  input  clk;
  input  [col-1:0] wr;
  input  rd;
  input  reset;
  input  [col*bw-1:0] in;
  output [col*bw-1:0] out;
  output o_full;
  output o_ready;
  output o_valid;

  wire [col-1:0] empty;
  wire [col-1:0] full;
  reg  rd_en;

  genvar i;

  assign o_ready = ~|full ;
  assign o_full  = |full ;
  assign o_valid = ~|empty ;

  for (i=0; i<col ; i=i+1) begin : col_num
      fifo_depth64 #(.bw(bw)) fifo_instance (
    .rd_clk(clk),
    .wr_clk(clk),
    .rd(rd),
    .wr(wr[i]),
         .o_empty(empty[i]),
         .o_full(full[i]),
    .in(in[(i+1)*bw-1:i*bw]),
    .out(out[(i+1)*bw-1:i*bw]),
         .reset(reset));
  end


  always @ (posedge clk) begin
   if (reset) begin
      rd_en <= 0;
   end
   else
      if (rd)
         rd_en <= 1;
      else
         rd_en <= 0;

  end


```