mac_array.v

```
// Created by prof. Mingu Kang @VVIP Lab in UCSD ECE department
// Please do not spread this code without permission
module mac_array (clk, reset, out_s, in_w, in_n, inst_w, valid);

  parameter bw = 4;
  parameter psum_bw = 16;
  parameter col = 8;
  parameter row = 8;

  input  clk, reset;
  output [psum_bw*col-1:0] out_s;
  input  [row*bw-1:0] in_w; // inst[1]:execute, inst[0]: kernel loading
  input  [1:0] inst_w;
  input  [psum_bw*col-1:0] in_n;
  output [col-1:0] valid;
  wire [psum_bw*col-1:0] temp_psum [row:0];
  wire [col-1:0] temp_valid [row-1:0];
  wire [row*2-1:0] temp_inst [row:0];

  reg  [1:0] temp_w;

  assign temp_psum[0] = in_n;
  assign out_s = temp_psum[row];
  assign valid = temp_valid[row-1];
 // assign temp_w = inst_w;

genvar i;
  for (i=1; i < row+1 ; i=i+1) begin : row_num
          mac_row #(.bw(bw), .psum_bw(psum_bw)) mac_row_instance (
          .clk(clk),
          .reset(reset),
    .out_s(temp_psum[i]),
    .in_w(in_w[bw*i-1:bw*(i-1)]),
    .in_n(temp_psum[i-1]),
    .inst_w(temp_w),
    .valid(temp_valid[i-1]));
  end

  always @ (posedge clk) begin


   // inst_w flows to row0 to row7
   //
   if (reset == 1)
            temp_w <=0;
   else
            temp_w <= inst_w;

  end



endmodule
```