mac_tile

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

parameter bw = 4;
parameter psum_bw = 16;

output [psum_bw-1:0] out_s;
input  [bw-1:0] in_w; // inst[1]:execute, inst[0]: kernel loading
output [bw-1:0] out_e;
input  [1:0] inst_w;
output [1:0] inst_e;
input  [psum_bw-1:0] in_n;
input  clk;
input  reset;

reg [1:0]inst_q;
reg [bw-1:0] a_q;
reg [bw-1:0] b_q;
reg [psum_bw-1:0] c_q;
reg load_ready_q;
//reg [psum_bw-1:0] mac_out;

mac #(.bw(bw), .psum_bw(psum_bw)) mac_instance (
        .a(a_q),
        .b(b_q),
        .c(c_q),
    .out(out_s)
);
assign inst_e = inst_q;
assign out_e = a_q;

//assign mac_out = out_s;

always @ (posedge clk)
begin
    // reset logic
    if (reset == 1 )
    begin
        inst_q <= 0;
        load_ready_q <= 1;
```

```
    end
    // if not reset, always accept inst_q[1] into inst_w[1], a_q to out_e
    else
    begin
        inst_q[1] <= inst_w[1]; // inst_q[1] into inst_w[1]
        c_q <= in_n;

        if (inst_w[0] == 1 || inst_w[1]==1)
            begin
                a_q <= in_w ; // accept new in_w into a_q latch.
            end

        if (inst_w[0] == 1 && load_ready_q == 1)
            begin
                b_q <= in_w; //accept weight into b_q latch via in_w port.
                load_ready_q <= 0; //load_ready_q = 0

            end
        else if (load_ready_q == 0)
            begin
                inst_q[0] <= inst_w[0]; //latch inst_w[0] into inst_q[0].
            end
    end




end


endmodule
```