

C:/Final_228/Final_228.srzs/sources_1/new/MemoryDemo.v



```
1 `timescale 1ns / 1ps
2 // Company: University of Missouri-Kansas City
3 // Engineer: Kyle Goodman
4 // Create Date: 04/27/2021
5 // Module Name: register_file
6 // Project Name: Final_228
7 ///////////////
8 module register_file(
9     input clk, rw,
10    input [15:0] in_row,
11    input [3:0] AA, AB1, AB2, AB3, AB4, DA,
12    output reg [15:0] OA, OB1, OB2, OB3, OB4
13 );
14
15
16    reg [15:0] rf [15:0];
17
18    reg [3:0] i;
19    always@(posedge clk)
20    begin
21        if (rw == 1) rf[DA]= in_row;
22        else {OA,OB1,OB2,OB3,OB4} = { //output will always be in row form
23            rf[AA],
24            rf[AB1][3:0], //transpose columns of AB into rows of OB
25            rf[AB2][3:0],
26            rf[AB3][3:0],
27            rf[AB4][3:0],
28            rf[AB1][7:4], //transpose column 2 into row 2
29            rf[AB2][7:4],
30            rf[AB3][7:4],
31            rf[AB4][7:4],
32            rf[AB1][11:8], //transpose column 3 into row 3
33            rf[AB2][11:8],
34            rf[AB3][11:8],
35            rf[AB4][11:8],
36            rf[AB1][15:12], //transpose column 4 into row 4
37            rf[AB2][15:12],
38            rf[AB3][15:12],
39            rf[AB4][15:12]
40        };
41    end
42
43 endmodule
44
```

C:/Final_228/Final_228.srcts/sources_1/new/multiply_4b.v



```
1 `timescale 1ns / 1ps
2 //////////////////////////////////////////////////////////////////
3 // Company: University of Missouri-Kansas City
4 // Engineer: Kyle Goodman
5 // Create Date: 04/07/2021 03:00:41 AM
6 // Module Name: multiply_4b
7 // Project Name: Final_228
8 //////////////////////////////////////////////////////////////////
9
10
11 module multiply_4b(
12     input [3:0] A,
13     input [3:0] B,
14     output reg [3:0] P,
15     output reg OF
16 );
17
18     wire [7:0] a0, a1, a2, a3;
19     wire [7:0] P8b;
20
21     assign a0 = {4'b0000, A[3] & B[0], A[2] & B[0], A[1] & B[0], A[0] & B[0]};
22     assign a1 = {3'b000, A[3] & B[1], A[2] & B[1], A[1] & B[1], A[0] & B[1], 1'b0};
23     assign a2 = {2'b00, A[3] & B[2], A[2] & B[2], A[1] & B[2], A[0] & B[2], 2'b00};
24     assign a3 = {1'b0, A[3] & B[3], A[2] & B[3], A[1] & B[3], A[0] & B[3], 3'b000};
25
26     assign P8b = a0 + a1 + a2 + a3;
27
28     always@(*)
29     begin
30         if(P8b > 8'b00001111)
31             begin
32                 P <= P8b[3:0];
33                 OF <= 1'b1;
34             end
35         else
36             begin
37                 P <= P8b[3:0];
38                 OF <= 1'b0;
39             end
40     end
41
42 endmodule
43
```

Project Summary x register_TB.v * x MemoryDemo.v x FA_nb.v x RowMultiplier.v x mac_cell.v x multi_nb.v

C:/Final_228/Final_228.srcc/sources_1/new/multi_nb.v

Q | H | ← | → | X | E | D | X | // | ■ | ? |

```

1 `timescale 1ns / 1ps
2 //////////////////////////////////////////////////////////////////
3 // Company: University of Missouri-Kansas City
4 // Engineer: Kyle Goodman
5 // Create Date: 04/07/2021
6 // Module Name: multi_nb
7 // Project Name: Final_228
8 //////////////////////////////////////////////////////////////////
9
10
11 module multi_nb
12     #(parameter BIT_WIDTH=4)
13     input [BIT_WIDTH-1:0] m_A, m_B,
14     output reg [BIT_WIDTH-1:0] m_P,
15     output reg m_OF
16 );
17
18
19     wire [2*BIT_WIDTH-1:0] m_PLb;
20
21     //also comment/uncomment these lines to change bit width///////////
22     wire [2*BIT_WIDTH-1:0] m_a0, m_a1, m_a2, m_a3/*, a4, a5, a6, a7*/;           //
23     assign m_a0 = {{BIT_WIDTH{1'b0}}, (m_B[0]? m_A : {BIT_WIDTH{1'b0}})};           //
24     assign m_a1 = {{(BIT_WIDTH-1){1'b0}}, (m_B[1]? m_A : {BIT_WIDTH{1'b0}}),1'd0};    //
25     assign m_a2 = {{(BIT_WIDTH-2){1'b0}}, (m_B[2]? m_A : {BIT_WIDTH{1'b0}}),2'd0};    //
26     assign m_a3 = {{(BIT_WIDTH-3){1'b0}}, (m_B[3]? m_A : {BIT_WIDTH{1'b0}}),3'd0};    //
27     //assign a4 = {{(BIT_WIDTH-4){1'b0}}, (B[4]? A : {BIT_WIDTH{1'b0}}),4'd0};          //
28     //assign a5 = {{(BIT_WIDTH-5){1'b0}}, (B[5]? A : {BIT_WIDTH{1'b0}}),5'd0};          //
29     //assign a6 = {{(BIT_WIDTH-6){1'b0}}, (B[6]? A : {BIT_WIDTH{1'b0}}),6'd0};          //
30     //assign a7 = {{(BIT_WIDTH-7){1'b0}}, (B[7]? A : {BIT_WIDTH{1'b0}}),7'd0};          //
31     assign m_PLb = m_a0 + m_a1 + m_a2 + m_a3/* + a4 + a5 + a6 + a7*/;           //
32     //////////////////////////////////////////////////////////////////
33
34     always@(*)
35     begin
36         if(m_PLb > { {BIT_WIDTH{1'b0}}, {BIT_WIDTH{1'b1}} })
37             begin
38                 m_P <= m_PLb[BIT_WIDTH-1:0];
39                 m_OF <= 1'b1;
40             end
41         else
42             begin
43                 m_P <= m_PLb[BIT_WIDTH-1:0];
44                 m_OF <= 1'b0;
45             end
46     end

```

C:/Final_228/Final_228.srsc/sources_1/new/mac_cell.v

The screenshot shows a Verilog code editor window with the following details:

- Toolbar:** Standard icons for search, file operations, and help.
- Code Area:** The main text area contains the Verilog code for the `mac_cell` module. The code includes comments, parameters, and logic for addition and multiplication.
- Line Numbers:** Numerical labels on the left side of the code area indicate the line numbers from 1 to 44.
- Code Content:** The code defines a module `mac_cell` with various input and output ports, including `ENM`, `ENA`, and `CLK`. It uses internal registers `At`, `Bt`, `Ct`, and `prod` to store intermediate values. The module also handles overflow through the `OF` signal and provides enable signals for adder and multiplier.

```
1 `timescale 1ns / 1ps
2 // Company: University of Missouri-Kansas City
3 // Engineer: Kyle Goodman
4 // Create Date: 04/08/2021
5 // Module Name: mac_cell
6 // Project Name: Final_228
7 ///////////////
8 module mac_cell
9     #(parameter BIT_WIDTH=4)
10    input [BIT_WIDTH-1:0] Ai,Bi,Ci,
11    output reg [BIT_WIDTH-1:0] Ao,Bo,Co,
12    output OF, //angry overflow flag from multiplier or adder
13    input ENM, //enable or disable multiply // this allows these features to be
14    input ENA, //enable or disable adder // disabled, turning the cell into a delay
15    input CLK
16    );
17    reg [BIT_WIDTH-1:0] At, Bt, Ct; //to hold values
18    reg [BIT_WIDTH-1:0] prod; //to store product
19
20    always@(posedge CLK)
21    begin
22        At <= Ai; //push those noob values into their prison cells
23        Bt <= Bi;
24        Ct <= Ci;
25    end
26
27
28    wire AF, MF;
29    assign OF = AF | MF; //overflow handling
30
31    wire [BIT_WIDTH-1:0] prodt, sumt;
32    FA_nb mc_FA(.A(Ct), .B(prod), .G(sumt), .OF(AF)); //adder
33    multi_nb mc_MY(.A(Ai), .B(Bi), .P(prodt), .OF(MF)); //multiplier
34
35    always@(posedge CLK)
36    begin
37        if(ENM) prod <= {BIT_WIDTH{1'b0}};
38        else prod <= prodt; //multiplied
39    end
40
41    always@(posedge CLK)
42    begin
43        Ao <= At;
44        Bo <= Bt;
```

```

1 `timescale 1ns / 1ps
2 /////////////////////////////////////////////////////////////////////
3 // Company: University of Missouri-Kansas City
4 // Engineer: Kyle Goodman
5 // Create Date: 04/27/2021
6 // Module Name: row_multiplier
7 // Project Name: Final_228
8 /////////////////////////////////////////////////////////////////////
9
10 module RowMultiplier
11     #(parameter BIT_WIDTH=4)(
12         input [(BIT_WIDTH*4)-1:0] rm_X,rm_Y,
13         output reg [(BIT_WIDTH*4)-1:0] rm_C,
14         output rm_OF
15     );
16
17     wire rm_OF_1, rm_OF_2, rm_OF_3, rm_OF_4; //overflow flags
18     assign rm_OF = (rm_OF_1 | rm_OF_2 | rm_OF_3 | rm_OF_4);
19
20     multi_nb rm_multi_1(.A(rm_X[BIT_WIDTH-1:0]), .B(rm_Y[BIT_WIDTH-1:0]), .P(rm_Z[BIT_WIDTH-1:0]), .OF(rm_OF_1));
21     multi_nb rm_multi_2(.A(rm_X[BIT_WIDTH*2-1:BIT_WIDTH]), .B(rm_Y[BIT_WIDTH*2-1:BIT_WIDTH]), .P(rm_Z[BIT_WIDTH*2-1:BIT_WIDTH]), .OF(rm_OF_2));
22     multi_nb rm_multi_3(.A(rm_X[BIT_WIDTH*3-1:BIT_WIDTH*2]), .B(rm_Y[BIT_WIDTH*3-1:BIT_WIDTH*2]), .P(rm_Z[BIT_WIDTH*3-1:BIT_WIDTH*2]), .OF(rm_OF_3));
23     multi_nb rm_multi_4(.A(rm_X[BIT_WIDTH*4-1:BIT_WIDTH*3]), .B(rm_Y[BIT_WIDTH*4-1:BIT_WIDTH*3]), .P(rm_Z[BIT_WIDTH*4-1:BIT_WIDTH*3]), .OF(rm_OF_4));
24     wire [(BIT_WIDTH*4)-1:0] rm_Z
25     FA_nb rm_add_1();
26
27 endmodule
28

```

C:/Final_228/Final_228.srcc/sources_1/new/FA_nb.v



```

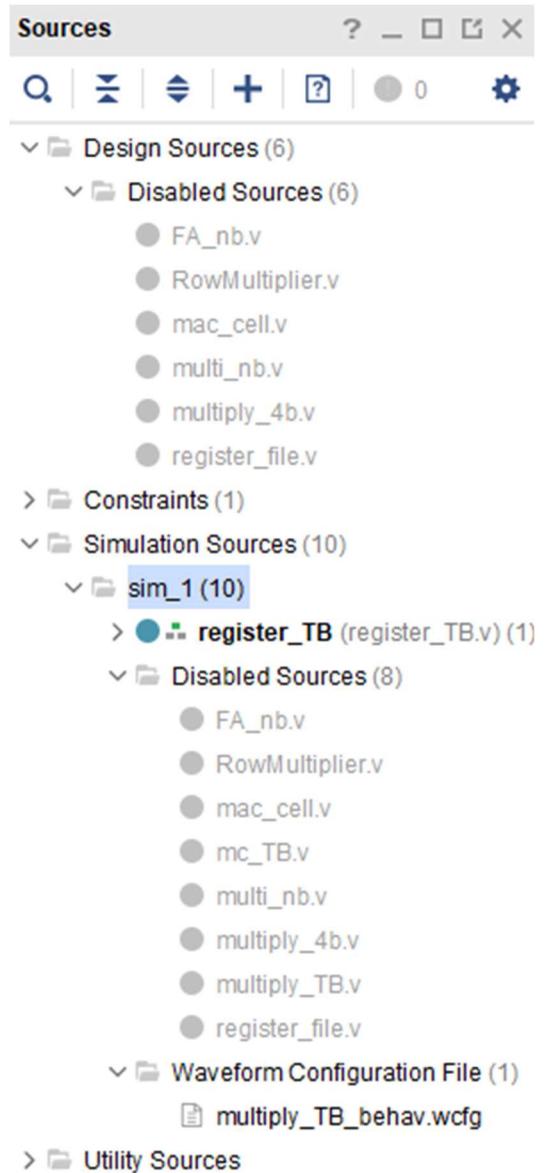
1 `timescale 1ns / 1ps
2 /////////////////////////////////////////////////////////////////////
3 // Company: University of Missouri-Kansas City
4 // Engineer: Kyle Goodman
5 // Create Date: 04/07/2021
6 // Module Name: FA_nb
7 // Project Name: Final_228
8 /////////////////////////////////////////////////////////////////////
9 module FA_nb
10    #(parameter BIT_WIDTH=4)(
11        //input Cin,
12        input [BIT_WIDTH-1:0] fa_A, fa_B,
13        output [BIT_WIDTH-1:0] fa_G,
14        output fa_OF //because we don't care about signs--yet
15    );
16
17    assign {fa_OF,fa_G} = fa_A + fa_B;
18
19 endmodule
20

```

C:/Final_228/Final_228.srcc/sources_1/new/register_file.v

The screenshot shows a Verilog code editor interface with a toolbar at the top containing icons for search, file operations, and other common functions. The main area displays the Verilog source code for a register_file module. The code includes a timescale declaration, module definition, input ports, and an always@(posedge clk) block containing logic to transpose AB columns into OB rows.

```
1 `timescale 1ns / 1ps
2 // Company: University of Missouri-Kansas City
3 // Engineer: Kyle Goodman
4 // Create Date: 04/27/2021
5 // Module Name: register_file
6 // Project Name: Final_228
7 ///////////////
8 module register_file(
9     input clk, rw,
10    input [15:0] in_row,
11    input [3:0] AA, AB1, AB2, AB3, AB4, DA,
12    output reg [15:0] OA, OB1, OB2, OB3, OB4
13 );
14
15
16     reg [15:0] rf [15:0];
17
18     reg [3:0] i;
19     always@(posedge clk)
20     begin
21         if (rw == 1) rf[DA]= in_row;
22         else {OA,OB1,OB2,OB3,OB4} = { //output will always be in row form
23             rf[AA],
24             rf[AB1][3:0], //transpose columns of AB into rows of OB
25             rf[AB2][3:0],
26             rf[AB3][3:0],
27             rf[AB4][3:0],
28             rf[AB1][7:4], //transpose column 2 into row 2
29             rf[AB2][7:4],
30             rf[AB3][7:4],
31             rf[AB4][7:4],
32             rf[AB1][11:8], //transpose column 3 into row 3
33             rf[AB2][11:8],
34             rf[AB3][11:8],
35             rf[AB4][11:8],
36             rf[AB1][15:12], //transpose column 4 into row 4
37             rf[AB2][15:12],
38             rf[AB3][15:12],
39             rf[AB4][15:12]
40         };
41     end
42
43 endmodule
```



BUFF b 1	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	0
t=2	0
t=1	0
t=0	b11

BUFF b2	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	0
t=2	0
t=1	b12
t=0	b21

BUFF b 1	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	0
t=2	b13
t=1	b22
t=0	b31

BUFF a 1	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	0
t=2	0
t=1	0
t=0	a11

MAC			
	a_in	b_in	c_out
t=0	a11	b11	a11b11
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	b21	0
t=1	a11	b12	a11b12
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	b31	0
t=1	0	b22	0
t=2	a11	b13	a11b13
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

BUFF a 2	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	0
t=2	0
t=1	a21
t=0	a12

MAC			
	a_in	b_in	c_out
t=0	a12	0	0
t=1	a21	b11	a21b11
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	a12	b21	a11b11+a12b21
t=2	a21	b12	a21b12
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	b31	0
t=2	a12	b22	a11b12 + a12b22
t=3	a21	b13	a21b13
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

BUFF a 3	
t=8	0
t=7	0
t=6	0

MAC			
	a_in	b_in	c_out
t=0	a13	0	0
t=1	a22	0	0
t=2	a31	b11	a31b11

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	a13	0	0
t=2	a22	b21	a21b11 + a22b21

||
||
||

t=5			0
t=4			0
t=3			0
t=2	a31		
t=1	a22		
t=0	a13		

t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=3	a31	b12	a31b12
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=3	a22	b22	a21b12 + a22b22
t=4	a31	b13	a31b13
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

BUFF a 4			
t=8			0
t=7			0
t=6			0
t=5			0
t=4			0
t=3	a41		
t=2	a32		
t=1	a23		
t=0	a14		

MAC			
	a_in	b_in	c_out
t=0	a14	0	0
t=1	a23	0	0
t=2	a32	0	0
t=3	a41	b11	a41b11
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	
t=1	a14		
t=2	a23		
t=3	a32	b21	a31b11 + a32b21
t=4	a41	b12	a41b12
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	a14		
t=3	a23	b31	a21b11 + a22b21 + a23b31
t=4	a32	b22	a31b12 + a32b22
t=5	a41	b13	a41b13
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

BUFF a 5			
t=8			0
t=7			0
t=6			0
t=5			0
t=4			0
t=3	a42		
t=2	a33		
t=1	a24		
t=0			0

DELAY			
	a_in		
t=0		0	
t=1	a24		
t=2	a33		
t=3	a42		
t=4	0	0	
t=5	0	0	
t=6	0	0	
t=7	0	0	
t=8	0	0	
t=9	0	0	
t=10	0	0	
t=11	0	0	

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	a24		
t=3	a33		
t=4	a42	b21	a41b11 + a42b21
t=5	0	b12	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	a24		
t=4	a33	b31	a31b11 + a32b21 + a33b31
t=5	a42	b22	a41b12 + a42b22
t=6	0	b13	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

BUFF a 6			
t=8			0
t=7			0
t=6			0
t=5			0

DELAY			
	a_in		

<tbl

t=4		0
t=3	a43	
t=2	a34	
t=1		0
t=0		0

t=4		0
t=5		0
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

t=4	a43	
t=5		0
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

t=4	a34	0	0
t=5	a43	b31	$a41b11 + a42b21 + a43b31$
t=6		0	b22
t=7		0	b13
t=8		0	0
t=9		0	0
t=10		0	0
t=11		0	0

BUFF a 7		
t=8		0
t=7		0
t=6		0
t=5		0
t=4		0
t=3	a44	
t=2		0
t=1		0
t=0		0

DELAY		
	a_in	
t=0		0
t=1		0
t=2		0
t=3	a44	
t=4		0
t=5		0
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

DELAY		
	a_in	
t=0		0
t=1		0
t=2		0
t=3		0
t=4	a44	
t=5		0
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

DELAY		
	a_in	
t=0		0
t=1		0
t=2		0
t=3		0
t=4		0
t=5	a44	
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

BUFF b 1		
t=8		0
t=7		0
t=6		0
t=5		0
t=4		0
t=3	b14	
t=2	b23	
t=1	b32	
t=0	b41	

BUFF b 1		
t=8		0
t=7		0
t=6		0
t=5		0
t=4		0
t=3	b24	
t=2	b33	
t=1	b42	
t=0		0

t=8		
t=7		
t=6		
t=5		
t=4		
t=3	b34	
t=2	b43	
t=1		
t=0		

MAC			
	a_in	b_in	c_out
t=0	0	b41	0
t=1	0	b32	0
t=2	0	b23	0
t=3	a11	b14	a11b14
t=4	0	0	0
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

DELAY		
	b_in	
t=0		0
t=1	b42	
t=2	b33	
t=3	b24	
t=4		0
t=5		0
t=6		0
t=7		0
t=8		0
t=9		0
t=10		0
t=11		0

t=0		
t=1		
t=2	b43	
t=3	b34	
t=4		
t=5		
t=6		
t=7		
t=8		
t=9		
t=10		
t=11		

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	b41	0
t=2	0	b32	0
t=3	a12	b23	a11b13 + a12b23
t=4	a21	b14	a21b14
t=5	0	0	0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	b42	0
t=3	0	b33	0
t=4	a12	b24	a11b14 + a12b24
t=5	a21		0
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=0		
t=1		
t=2	b43	
t=3	b34	
t=4		
t=5		
t=6		
t=7		
t=8		
t=9		
t=10		
t=11		

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	b41	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0

t=0		
t=1		
t=2	b43	

t=3	a13	b32	a11b12 + a12b22 + a13b32
t=4	a22	b23	a21b13 + a22b23
t=5	a31	b14	a31b14
t=6	0	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=3	0	b42	0
t=4	a13	b33	a11b13 + a12b23 + a13b33
t=5	a22	b24	a21b14 + a22b24
t=6	a31	0	0
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=3	0	0
t=4	0	b43
t=5	a13	b34
t=6	a22	0
t=7	a31	0
t=8	0	0
t=9	0	0
t=10	0	0
t=11	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	a14	b41	a11b11 + a12b21 + a13b31 + a14b41
t=4	a23	b32	a21b12 + a22b22 + a23b32
t=5	a32	b23	a31b13 + a32b23
t=6	a41	b14	a41b14
t=7	0	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	a14	b42	a11b12 + a12b22 + a13b32 + a14b42
t=5	a23	b33	a21b13 + a22b23 + a23b33
t=6	a32	b24	a31b14 + a32b24
t=7	a41	0	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

	a_in	b_in
t=0	0	0
t=1	0	0
t=2	0	0
t=3	0	0
t=4	0	0
t=5	a14	b43
t=6	a23	b34
t=7	a32	0
t=8	a41	0
t=9	0	0
t=10	0	0
t=11	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	a24	b41	a21b11 + a22b21 + a23b31 + a24b41
t=5	a33	b32	a31b12 + a32b22 + a33b32
t=6	a42	b23	a41b13 + a42b23
t=7	0	b14	0
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	a11b11 + a12b21 + a13b31 + a14b41
t=5	a24	b42	a21b12 + a22b22 + a23b32 + a24b42
t=6	a33	b33	a31b13 + a32b23 + a33b33
t=7	a42	b24	a41b14 + a42b24
t=8	0	0	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

	a_in	b_in
t=0	0	0
t=1	0	0
t=2	0	0
t=3	0	0
t=4	0	0
t=5	0	0
t=6	a24	b43
t=7	a33	b34
t=8	a42	0
t=9	0	0
t=10	0	0
t=11	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0

	a_in	b_in
t=0	0	0
t=1	0	0
t=2	0	0
t=3	0	0

t=4	0	0	0
t=5	a34	b41	a31b11 + a32b21 + a33b31 + a34b41
t=6	a43	b32	a41b12 + a42b22 + a43b32
t=7	0	b23	0
t=8	0	b14	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=4	0	0	0
t=5	0	0	a21b11 + a22b21 + a23b31 + a24b41
t=6	a34	b42	a31b12 + a32b22 + a33b32 + a34b42
t=7	a43	b33	a41b13 + a42b23 + a43b33
t=8	0	b24	0
t=9	0	0	0
t=10	0	0	0
t=11	0	0	0

t=4	0	0
t=5	0	0
t=6	0	0
t=7	a34	b43
t=8	a43	b34
t=9	0	0
t=10	0	0
t=11	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	a44	b41	a41b11 + a42b21 + a43b31 + a44b41
t=7	0	b32	0
t=8	0	b23	0
t=9	0	b14	0
t=10	0	0	0
t=11	0	0	0

MAC			
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	a31b11 + a32b21 + a33b31 + a34b41
t=7	a44	b42	a41b12 + a42b22 + a43b32 + a44b42
t=8	0	b33	0
t=9	0	b24	0
t=10	0	0	0
t=11	0	0	0

	a_in	b_in
t=0	0	0
t=1	0	0
t=2	0	0
t=3	0	0
t=4	0	0
t=5	0	0
t=6	0	0
t=7	0	0
t=8	a44	b43
t=9	0	b34
t=10	0	0
t=11	0	0

OBUF 0	
t=0	0
t=1	0
t=2	0
t=3	0
t=4	0
t=5	0
t=6	0
t=7	a41b11 + a42b21 + a43b31 + a44b41
t=8	
t=9	
t=10	
t=11	

t=0	0
t=1	0
t=2	0
t=3	0
t=4	0
t=5	0
t=6	0
t=7	a31b11 + a32b21 + a33b31 + a34b41
t=8	a41b12 + a42b22 + a43b32 + a44b42
t=9	
t=10	
t=11	

BUFF b 1	
t=8	0
t=7	0
t=6	0
t=5	0
t=4	0
t=3	b44
t=2	0
t=1	0
t=0	0

DELAY	
	b_in
t=0	0
t=1	0
t=2	0
t=3	b44
t=4	0
t=5	0
t=6	0
t=7	0
t=8	0
t=9	0
t=10	0
t=11	0

DELAY	
	b_in
t=0	0
t=1	0
t=2	0
t=3	0
t=4	b44
t=5	0
t=6	0
t=7	0
t=8	0
t=9	0
t=10	0
t=11	0

MAC
c_out
0
0
0

DELAY	
	b_in
t=0	0
t=1	0
t=2	0

0
0
a11b14 + a12b24 + a13b34
0
0
0
0
0
0
0
0
0

t=3	0
t=4	0
t=5	b44
t=6	0
t=7	0
t=8	0
t=9	0
t=10	0
t=11	0

MAC
c_out
0
0
0
0
0
a11b13 + a12b23 + a13b33 + a14b43
a21b14 + a22b24 + a23b34
0
0
0
0
0
0

	MAC		
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	a14	b44	a11b14 + a12b24 + a13b34 + a14b44
t=7	a23		0
t=8	a32		0
t=9	a41		0
t=10	0	0	0
t=11	0	0	0

MAC
c_out
0
0
0
0
0
a11b12 + a12b22 + a13b32 + a14b420
a21b13 + a22b23 + a23b33 + a24b43
a31b14 + a32b24 + a33b34
0
0
0
0
0

	MAC		
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	a11b13 + a12b23 + a13b33 + a14b43
t=7	0	a11b13 + a12b23 + a13b33 + a14b43	
t=8	a24	b44	a21b14 + a22b24 + a23b34 + a24b44
t=9	a33		0
t=10	a42		0
t=11	0	0	0

MAC
c_out
0
0
0
0
0

	MAC		
	a_in	b_in	c_out
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0

	OBUF 6		
t=0	0		
t=1	0		
t=2	0		
t=3	0		
t=4	0		
t=5	0		
t=6	0		
t=7	a11b14 + a12b24 + a13b34 + a14b44		
t=8	0		
t=9	0		
t=10	0		
t=11	0		

	OBUF 5		
t=0	0		
t=1	0		
t=2	0		
t=3	0		
t=4	0		

0
a11b11 + a12b21 + a13b31 + a14b41
a21b12 + a22b22 + a23b32 + a24b42
a31b13 + a32b23 + a33b33 + a34b43
a41b14 + a42b24 + a43b34
0
0
0

t=4	0	0	0
t=5	0	0	0
t=6	0	0	a11b12 + a12b22 + a13b32 + a14b42
t=7	0	0	a21b13 + a22b23 + a23b33 + a24b43
t=8	a34	b44	a31b14 + a32b24 + a33b34 + a34b44
t=9	a43	0	0
t=10	0	0	0
t=11	0	0	0

t=5	0
t=6	0
t=7	a11b13 + a12b23 + a13b33 + a14b43
t=8	a21b14 + a22b24 + a23b34 + a24b44
t=9	0
t=10	0
t=11	0

MAC
c_out
0
0
0
0
0
0
a21b11 + a22b21 + a23b31 + a24b41
a31b12 + a32b22 + a33b32 + a34b42
a41b13 + a42b23 + a43b33 + a44b43
0
0

MAC			
a_in			
t=0	0	0	0
t=1	0	0	0
t=2	0	0	0
t=3	0	0	0
t=4	0	0	0
t=5	0	0	0
t=6	0	0	a11b11 + a12b21 + a13b31 + a14b41
t=7	0	0	a21b12 + a22b22 + a23b32 + a24b42
t=8	0	0	a31b13 + a32b23 + a33b33 + a34b43
t=9	a44	b44	a41b14 + a42b24 + a43b34 + a44b44
t=10	0	0	0
t=11	0	0	0

OBUF 4	
t=0	
t=1	
t=2	
t=3	
t=4	
t=5	
t=6	
t=7	a11b12 + a12b22 + a13b32 + a14b42
t=8	a21b13 + a22b23 + a23b33 + a24b43
t=9	a31b14 + a32b24 + a33b34 + a34b44
t=10	
t=11	

OBUF 1
3b31 + a34b41
3b32 + a44b42

OBUF 2	
t=0	
t=1	
t=2	
t=3	
t=4	
t=5	
t=6	
t=7	a21b11 + a22b21 + a23b31 + a24b41
t=8	a31b12 + a32b22 + a33b32 + a34b42
t=9	a41b13 + a42b23 + a43b33 + a44b43
t=10	
t=11	

OBUF 3	
t=0	
t=1	
t=2	
t=3	
t=4	
t=5	
t=6	
t=7	a11b11 + a12b21 + a13b31 + a14b41
t=8	a21b12 + a22b22 + a23b32 + a24b42
t=9	a31b13 + a32b23 + a33b33 + a34b43
t=10	a41b14 + a42b24 + a43b34 + a44b44
t=11	

Matrix A

A11	a11	a12	a13	a14	A12
	a21	a22	a23	a24	
	a31	a32	a33	a34	
A21	a41	a42	a43	a44	A22

Matrix B

B11	b11	b12	b13	b14	B12
	b21	b22	b23	b24	
	b31	b32	b33	b34	
B21	b41	b42	b43	b44	B22

A11	A12
A21	A22

B11	B12
B21	B22

A11	B11
A21	B11

A11	B12
A21	B12

A12	B12
A22	B12

A11	a11	a12
	a21	a22

B11	b11	b12
	b21	b22

$$a_{11} b_{11} + a_{12} b_{21}$$

$$a_{11} b_{12} + a_{12} b_{22}$$

+

A12	a13	a14
	a23	a24

B21	b31	b32
	b41	b42

$$a_{13} b_{31} + a_{14} b_{41}$$

$$a_{13} b_{32} + a_{14} b_{42}$$

A11	a11	a12
	a21	a22

B12	b13	b14
	b23	b24

$$a_{11} b_{13} + a_{12} b_{23}$$

$$a_{11} b_{14} + a_{12} b_{24}$$

+

A12	a13	a14
	a23	a24

B22	b33	b34
	b43	b44

$$a_{13} b_{33} + a_{14} b_{43}$$

$$a_{13} b_{34} + a_{14} b_{44}$$

A21	a31	a32
	a41	a42

B11	b11	b12
	b21	b22

$$a_{31} b_{11} + a_{32} b_{21}$$

$$a_{31} b_{12} + a_{32} b_{22}$$

+

A22	a33	a34
	a43	a44

B21	b31	b32
	b41	b42

$$a_{33} b_{31} + a_{34} b_{41}$$

$$a_{33} b_{32} + a_{34} b_{42}$$

A21	a31	a32
	a41	a42

B12	b13	b14
	b23	b24

$$a_{31} b_{13} + a_{32} b_{23}$$

$$a_{31} b_{14} + a_{32} b_{24}$$

+

A22	a33	a34
	a43	a44

B22	b33	b34
	b43	b44

$$a_{33} b_{33} + a_{34} b_{43}$$

$$a_{33} b_{34} + a_{34} b_{44}$$

A11	B11	+	A12	B21
a11	b11	+	a12	b21
a21	b11	+	a22	b21

a11	b12	+	a12	b22	+	a13	b32	+	a14	b42
a21	b12	+	a22	b22	+	a23	b32	+	a24	b42

A11	B12	+	A12	B22
a11	b13	+	a12	b23
a21	b13	+	a22	b23

a11	b14	+	a12	b24	+	a13	b34	+	a14	b44
a21	b14	+	a22	b24	+	a23	b34	+	a24	b44

A21	B11	+	A22	B21
a31	b11	+	a32	b21
a41	b11	+	a42	b21

a31	b12	+	a32	b22	+	a33	b32	+	a34	b42
a41	b12	+	a42	b22	+	a43	b32	+	a44	b42

A21	B12	+	A22	B22
a31	b13	+	a32	b23
a41	b13	+	a42	b23

a31	b14	+	a32	b24	+	a33	b34	+	a34	b44
a41	b14	+	a42	b24	+	a43	b34	+	a44	b44

A11	B11	+	A12	B21
a11	b11	+	a12	b21
a21	b11	+	a22	b21

a11	b12	+	a12	b22	+	a13	b32	+	a14	b42
a21	b12	+	a22	b22	+	a23	b32	+	a24	b42

A21	B11	+	A22	B21
a31	b11	+	a32	b21
a41	b11	+	a42	b21

a31	b12	+	a32	b22	+	a33	b32	+	a34	b42
a41	b12	+	a42	b22	+	a43	b32	+	a44	b42

A11	B12	+	A12	B22
a11	b13	+	a12	b23
a21	b13	+	a22	b23

a31	b13	+	a32	b23	+	a33	b33	+	a34	b43
a41	b13	+	a42	b23	+	a43	b33	+	a44	b43

a11	b1+	a12	b24	+	a13	b34	+	a14	b44
a21	b1+	a22	b24	+	a23	b34	+	a24	b44

a31	b1+	a32	b24	+	a33	b34	+	a34	b44
a41	b1+	a42	b24	+	a43	b34	+	a44	b44

Description	Mnemonic
Load to Register	LD
Load Constant (row) to Register	LC
Store to Memory	ST
Multiply <u>A</u> x <u>B</u> (n length row)	MLT
Matrix Add Row A + Row B (n length rows)	ADD
Wait 1 clk cycle	W
Matrix to Column Format	MTC
Matrix to Row Format	MTR

to enable n x n

Syntax

LD DA R/M(addr)

LC DA C1 C2 C3 C4

ST DA R(addr)

MLR DA RA RB n b (where $n \leq 4$, b is the begin flag to reset shift/magic counter)

ADR DA RA RB n (where $n \leq 4$)

W

MTC DA RB

MTR DA RA

A input should be stored in rows

B input should be stored in columns

addition must only be done with row-format matrices

Conversion concept (unverified)

>if we feed a row-form matrix to B we will get a column form matrix as a result

>if we feed a column-form matrix to B, which expects rows, do we get a result in row-form?

Execute 6 Execute 7

MLT R04

MLT R05 MLT R04

MLT R06 MLT R05

MLT R07 MLT R06

MLT R07

MLT R08

MLT R09 MLT R08

MLT R09

DR	RA	RB1	RB2	RB3	RB4
4'b'aaaa	4'bnnnn	4'bffff	4'bffff	4'bffff	4'bffff

MP RA RB DR

		PC (address)															
MP	DR	RA	RB1	RB2	RB3	RB4	M1	t=0	4'd12	4d'4	4d'0	4'd1	4'd5	4'd2	4'd6	4'd3	4'd7
							M2	t=1	4'd12	4d'4	4d'0	4'd1	4'd5	4'd2	4'd6	4'd3	4'd7
							M3	t=2	4'd12	4d'4	4d'0	4'd1	4'd5	4'd2	4'd6	4'd3	4'd7
							M4	t=3	4'd12	4d'4	4d'0	4'd1	4'd5	4'd2	4'd6	4'd3	4'd7

	IF	ID	Buff	Mult	WB
t=0	M1				
t=1	M2	M1			
t=2	M3	M2	M1		
t=3	M4	M3	M2	M1	
		M4	M3	M2	M1
			M4	M3	M2
				M4	M3
					M4

	DA	AA	BA1	BA2	BA3	BA4
t0		M1	M1	M1	M1	M1
t1		M2	M2	M2	M2	M2
t2		M2	M2	M3	M3	M3
t3		M3	M3	M4	M4	M4
t4		M4	M4	M5	M5	M5
t5	M1	M5	M5	M6	M6	M6
t6	M2	M6	M6	M7	M7	M7
t7	M3	M7	M7	M8	M8	M8
t8	M4	M8	M8	M9	M9	M9
t9	M5	M9	M9	M10	M10	M10
t10	M6	M10	M10	M11	M11	M11
t11	M7	M11	M11	M12	M12	M12
t12	M8	M12	M12	M13	M13	M13
t13	M9	M13	M13	M14	M14	M14
t14	M10	M14	M14	M15	M15	M15
t15	M11	M15	M15	M16	M16	M16
t16	M12	M16	M16	M17	M17	M17
				M13		
				M14		
				M15		
				M16		