**REPORT**

**Design and Simulation**

1. **Instantiating and Connecting PEs**

**Requirement:**

//FILL IN REST OF THE CODE (x)

//Three of them are connected below. Connect the others properly.

processing\_element pe00(.reset(effective\_rst), .clk(clk), .in\_a(a0), .in\_b(b0), .out\_a(a00to01), .out\_b(b00to10), .out\_c(matrixC00));

processing\_element pe01(.reset(effective\_rst), .clk(clk), .in\_a(a00to01), .in\_b(b1), .out\_a(a01to02), .out\_b(b01to11), .out\_c(matrixC01));

processing\_element pe02(x);

processing\_element pe03(x);

processing\_element pe10(.reset(effective\_rst), .clk(clk), .in\_a(a1), .in\_b(b00to10), .out\_a(a10to11), .out\_b(b10to20), .out\_c(matrixC10));

processing\_element pe11(x);

processing\_element pe12(x);

processing\_element pe13(x);

processing\_element pe20(x);

processing\_element pe21(x);

processing\_element pe22(x);

processing\_element pe23(x);

processing\_element pe30(x);

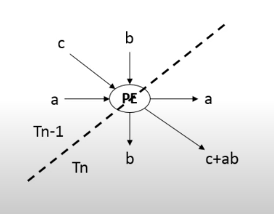
processing\_element pe31(x);

processing\_element pe32(x);

processing\_element pe33(x);

**Solution:**

There are 3 inputs and 3 outputs from Processing Element (PEs). Two inputs from Matrix A and Matrix B will be given to PE and again through FF will be given to another processing element. Multiplication and accumulative result (c+ab) will be store in Matrix varirable.



**Output Code:**

These are the remaining PEs in order to perform a 4 x 4 matmul.

//FILL IN REST OF THE CODE (x)

processing\_element pe00(.reset(effective\_rst), .clk(clk), .in\_a(a0), .in\_b(b0), .out\_a(a00to01), .out\_b(b00to10), .out\_c(matrixC00));

processing\_element pe10(.reset(effective\_rst), .clk(clk), .in\_a(a00to01), .in\_b(b1), .out\_a(a10to11), .out\_b(b10to20), .out\_c(matrixC10));

processing\_element pe20(.reset(effective\_rst), .clk(clk), .in\_a(a10to11), .in\_b(b2), .out\_a(a20to21), .out\_b(b20to30), .out\_c(matrixC20));

processing\_element pe30(.reset(effective\_rst), .clk(clk), .in\_a(a20to21), .in\_b(b3), .out\_a(a30to31), .out\_b(b30to40), .out\_c(matrixC30));

processing\_element pe01(.reset(effective\_rst), .clk(clk), .in\_a(a1), .in\_b(b00to10), .out\_a(a01to02), .out\_b(b01to11), .out\_c(matrixC01));

processing\_element pe11(.reset(effective\_rst), .clk(clk), .in\_a(a01to02), .in\_b(b10to20), .out\_a(a11to12), .out\_b(b11to21), .out\_c(matrixC11));

processing\_element pe21(.reset(effective\_rst), .clk(clk), .in\_a(a11to12), .in\_b(b20to30), .out\_a(a21to22), .out\_b(b21to31), .out\_c(matrixC21));

processing\_element pe31(.reset(effective\_rst), .clk(clk), .in\_a(a21to22), .in\_b(b30to40), .out\_a(a31to32), .out\_b(b31to41), .out\_c(matrixC31));

processing\_element pe02(.reset(effective\_rst), .clk(clk), .in\_a(a2), .in\_b(b01to11), .out\_a(a02to03), .out\_b(b02to12), .out\_c(matrixC02));

processing\_element pe12(.reset(effective\_rst), .clk(clk), .in\_a(a02to03), .in\_b(b11to21), .out\_a(a12to13), .out\_b(b12to22), .out\_c(matrixC12));

processing\_element pe22(.reset(effective\_rst), .clk(clk), .in\_a(a12to13), .in\_b(b21to31), .out\_a(a22to23), .out\_b(b22to32), .out\_c(matrixC22));

processing\_element pe32(.reset(effective\_rst), .clk(clk), .in\_a(a22to23), .in\_b(b31to41), .out\_a(a32to33), .out\_b(b32to42), .out\_c(matrixC32));

processing\_element pe03(.reset(effective\_rst), .clk(clk), .in\_a(a3), .in\_b(b02to12), .out\_a(a03to04), .out\_b(b03to13), .out\_c(matrixC03));

processing\_element pe13(.reset(effective\_rst), .clk(clk), .in\_a(a03to04), .in\_b(b12to22), .out\_a(a13to14), .out\_b(b13to23), .out\_c(matrixC13));

processing\_element pe23(.reset(effective\_rst), .clk(clk), .in\_a(a13to14), .in\_b(b22to32), .out\_a(a23to24), .out\_b(b23to33), .out\_c(matrixC23));

processing\_element pe33(.reset(effective\_rst), .clk(clk), .in\_a(a23to24), .in\_b(b32to42), .out\_a(a33to34), .out\_b(b33to43), .out\_c(matrixC33));

1. **# of Clock Cycles Needed for matmul**

**Requirement:**

//FILL IN THE CODE (x)

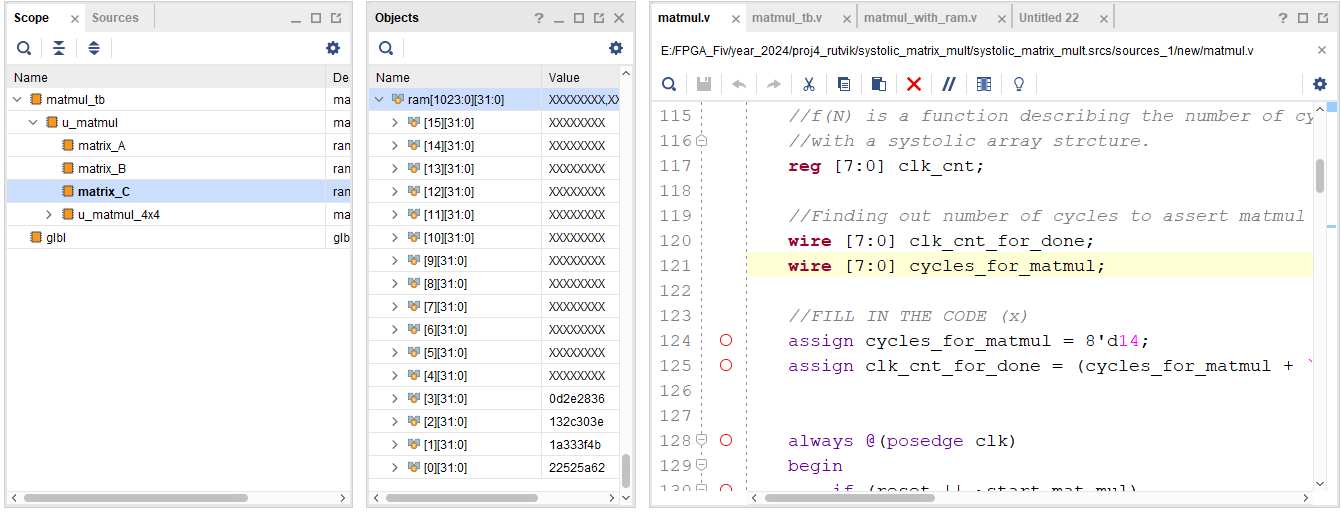
assign cycles\_for\_matmul = 8'dx;

**Solution:**

cycles\_for\_matmul = 8'd14;  
On assigning 8’d14 value, it has exactly N\*N values populated.

**Verification using waveform:**

As we know,if the count is too low, matrix C will not be fully filled and also if the count is too high, matrix C will be overfilled with 0’s. It is fully filled with assign cycles\_for\_matmul = 8'd14;



1. **Instantiating Block RAMs**

**Requirement:**

//FILL IN THE CODE (x)

////////////////////////////////////////////////////////////////

// BRAM matrix A

////////////////////////////////////////////////////////////////

ram matrix\_A (x);

////////////////////////////////////////////////////////////////

// BRAM matrix B

////////////////////////////////////////////////////////////////

ram matrix\_B (x);

////////////////////////////////////////////////////////////////

// BRAM matrix C

////////////////////////////////////////////////////////////////

ram matrix\_C (x);

**Solution:**

////////////////////////////////////////////////////////////////

// BRAM matrix A

////////////////////////////////////////////////////////////////

ram matrix\_A (

.addr0(bram\_addr\_a), // Address input connected to port 0

.d0(bram\_wdata\_a), // Data input connected to port 0

.we0(bram\_we\_a), // Write enable input connected to port 0

.q0(bram\_rdata\_a), // Data output connected to port 0

.addr1(bram\_addr\_a\_ext), // Address input connected to port 1 (external)

.d1(bram\_wdata\_a\_ext), // Data input connected to port 1 (external)

.we1(bram\_we\_a\_ext), // Write enable input connected to port 1 (external)

.q1(bram\_rdata\_a\_ext), // Data output connected to port 1 (external)

.clk(clk) // Clock input

);

////////////////////////////////////////////////////////////////

// BRAM matrix B

////////////////////////////////////////////////////////////////

ram matrix\_B (

.addr0(bram\_addr\_b), // Address input connected to port 0

.d0(bram\_wdata\_b), // Data input connected to port 0

.we0(bram\_we\_b), // Write enable input connected to port 0

.q0(bram\_rdata\_b), // Data output connected to port 0

.addr1(bram\_addr\_b\_ext), // Address input connected to port 1 (external)

.d1(bram\_wdata\_b\_ext), // Data input connected to port 1 (external)

.we1(bram\_we\_b\_ext), // Write enable input connected to port 1 (external)

.q1(bram\_rdata\_b\_ext), // Data output connected to port 1 (external)

.clk(clk) // Clock input

);

////////////////////////////////////////////////////////////////

// BRAM matrix C

////////////////////////////////////////////////////////////////

ram matrix\_C (

.addr0(bram\_addr\_c), // Address input connected to port 0

.d0(bram\_wdata\_c), // Data input connected to port 0

.we0(bram\_we\_c), // Write enable input connected to port 0

.q0(bram\_rdata\_c), // Data output connected to port 0

.addr1(bram\_addr\_c\_ext), // Address input connected to port 1 (external)

.d1(bram\_wdata\_c\_ext), // Data input connected to port 1 (external)

.we1(bram\_we\_c\_ext), // Write enable input connected to port 1 (external)

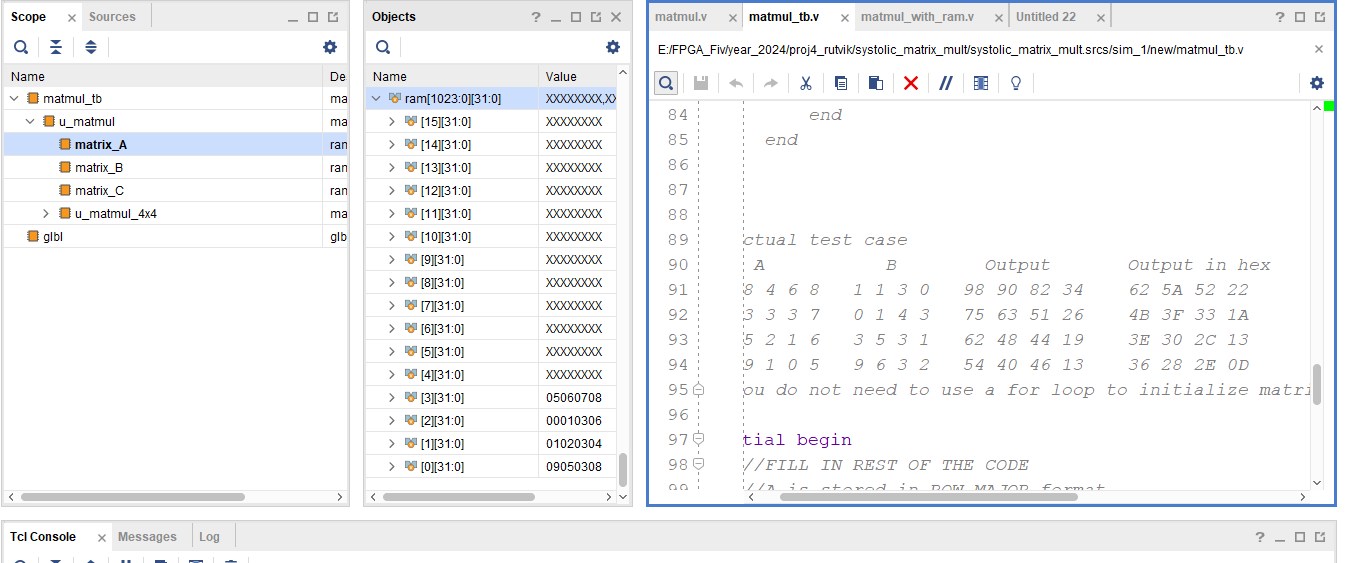
.q1(bram\_rdata\_c\_ext), // Data output connected to port 1 (external)

.clk(clk) // Clock input

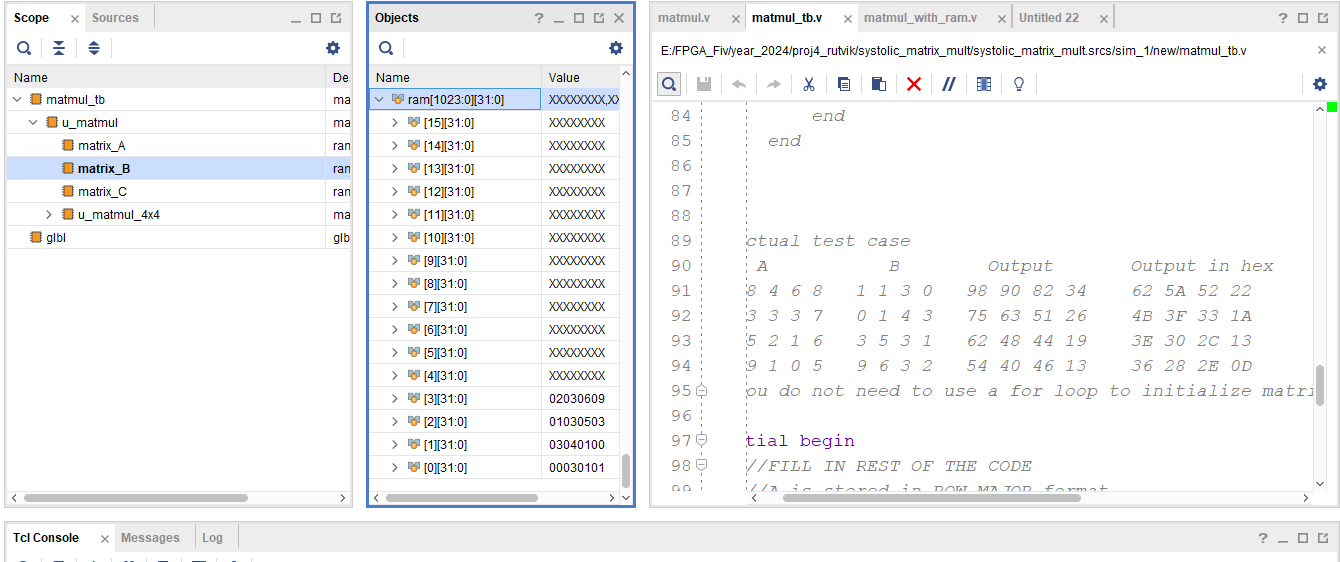
);

**Verification using waveform:**

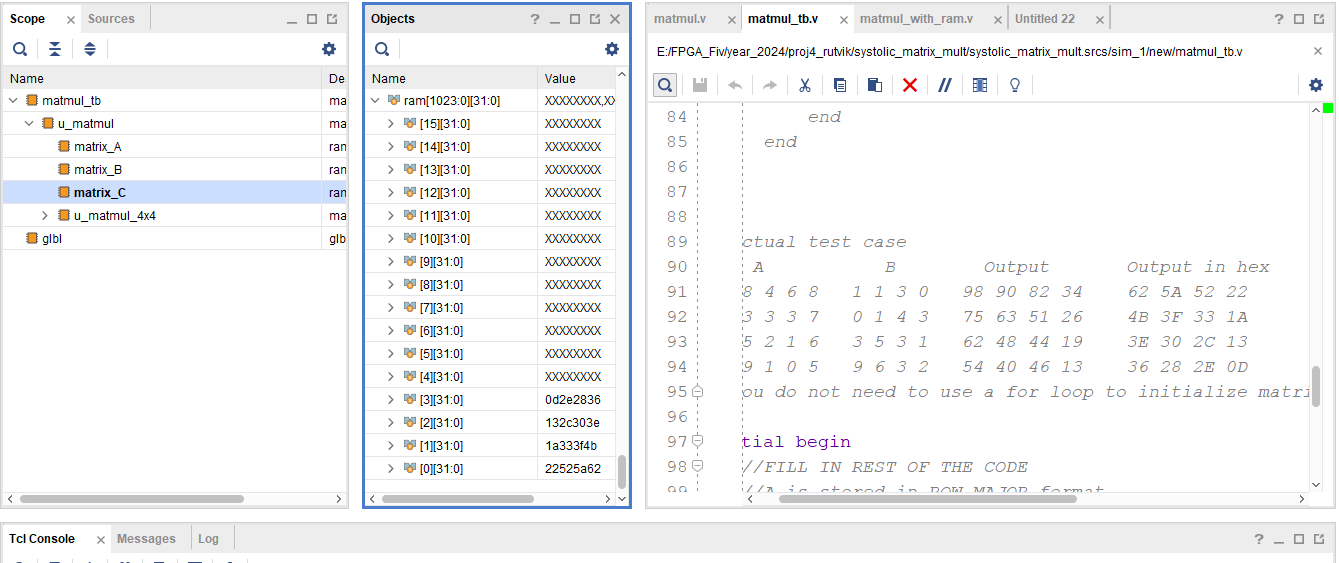
Matrix\_A: Data stored in ram is equal to data in test case



Matrix\_B: Data stored in ram is equal to data in test case



Matrix\_C: Data stored in ram is equal to data in test case



1. **More Testcases in Testbench**

**Test Case 1:**

Test bench code:

integer i;

initial

begin

for (i=0; i<4; i = i + 1)

begin

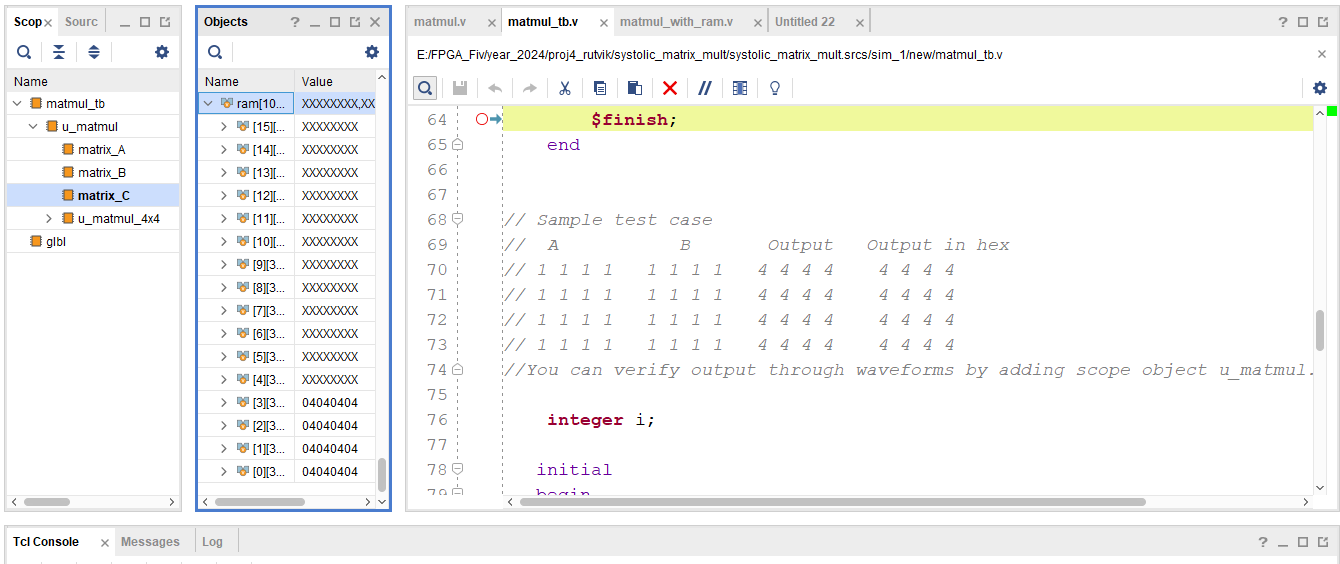
u\_matmul.matrix\_A.ram[i] = {8'h01, 8'h01, 8'h01, 8'h01};

u\_matmul.matrix\_B.ram[i] = {8'h01, 8'h01, 8'h01, 8'h01};

end

end

**Verification using waveform:**



**Test Case 2:**

Test bench code:

initial begin

u\_matmul.matrix\_A.ram[0] = {8'h09, 8'h05, 8'h03, 8'h08};

u\_matmul.matrix\_A.ram[1] = {8'h01, 8'h02, 8'h03, 8'h04};

u\_matmul.matrix\_A.ram[2] = {8'h00, 8'h01, 8'h03, 8'h06};

u\_matmul.matrix\_A.ram[3] = {8'h05, 8'h06, 8'h07, 8'h08};

u\_matmul.matrix\_B.ram[0] = {8'h00, 8'h03, 8'h01, 8'h01};

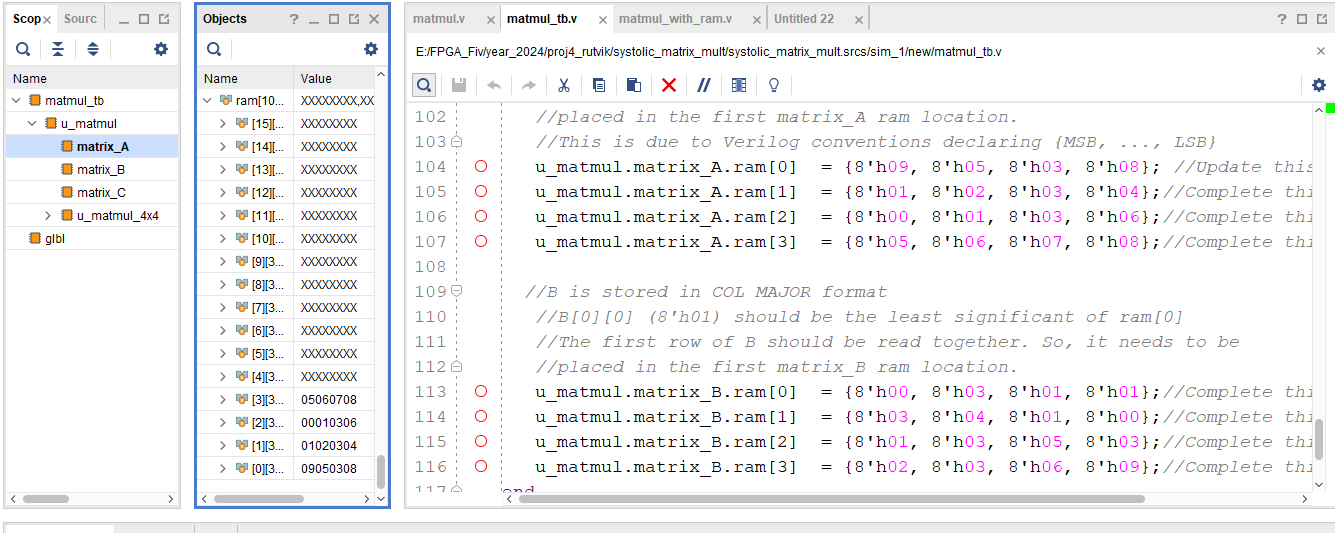
u\_matmul.matrix\_B.ram[1] = {8'h03, 8'h04, 8'h01, 8'h00};

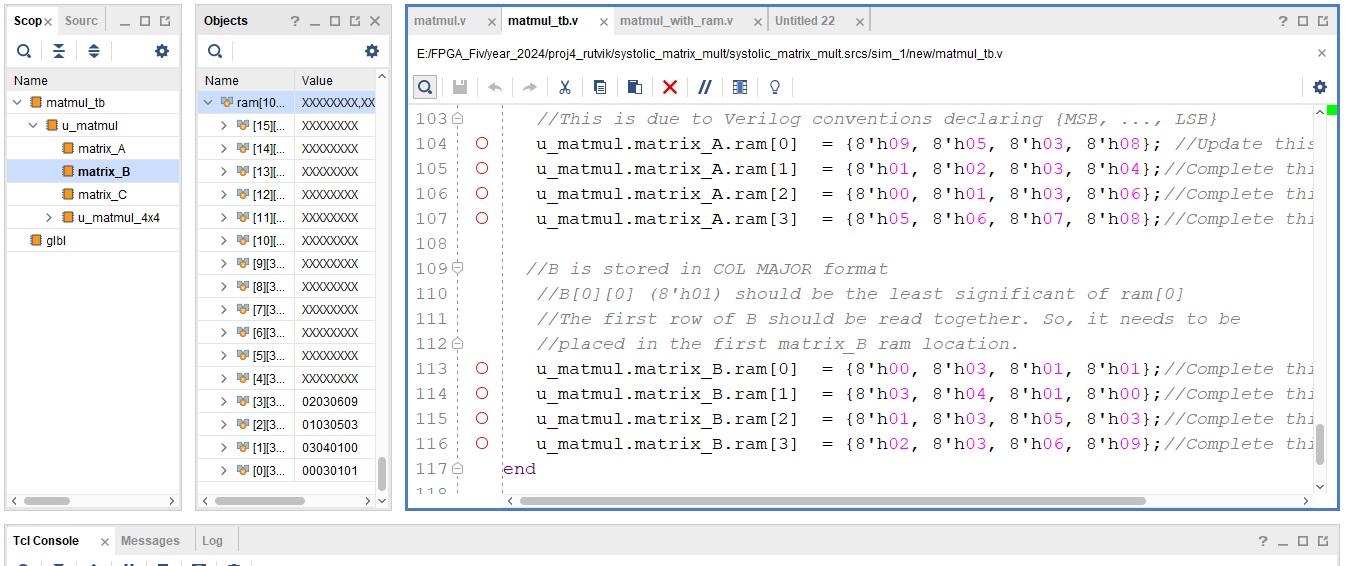
u\_matmul.matrix\_B.ram[2] = {8'h01, 8'h03, 8'h05, 8'h03};

u\_matmul.matrix\_B.ram[3] = {8'h02, 8'h03, 8'h06, 8'h09};

end

**Verification using waveform:**



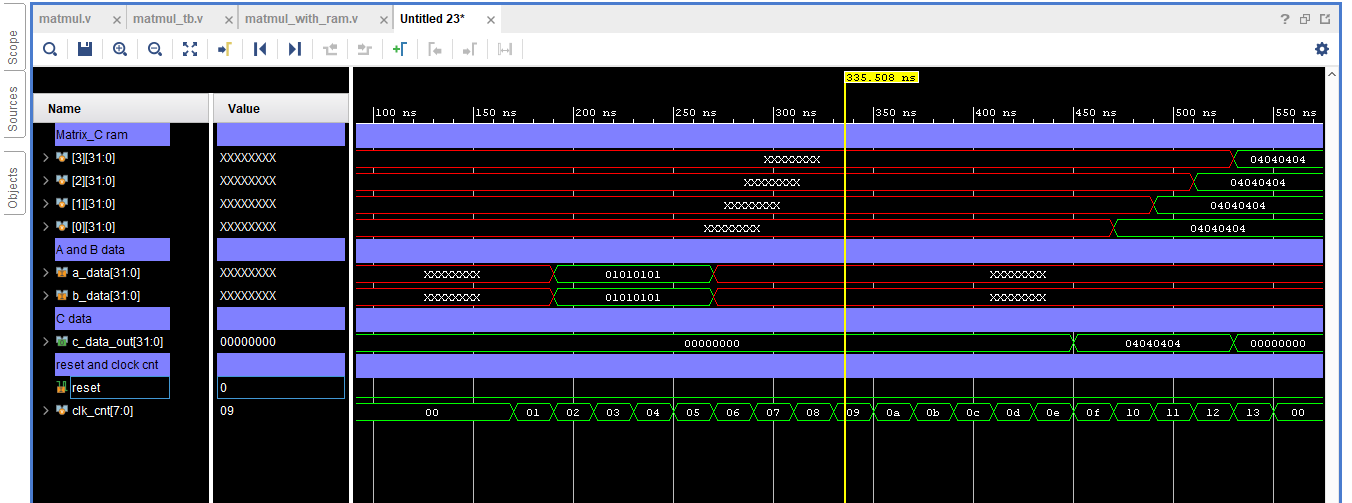


1. **Simulation**

Add the following signals from the scope to the wave window:

* u\_matmul.matrix\_C.ram[7:0][31:0]
* u\_matmul.u\_matmul\_4x4.a\_data[31:0]
* u\_matmul.u\_matmul\_4x4.b\_data[31:0]
* u\_matmul.u\_matmul\_4x4.c\_data\_out[31:0]
* u\_matmul.u\_matmul\_4x4.reset
* u\_matmul.u\_matmul\_4x4.clk\_cnt[7:0]

**For Test Case 1:**



**For Test Case 2:**

