Design of an Unsigned 4x4 Array Multiplier

1. Verilog/VHDL design of an unsigned 4x4 array multiplier

1.1 Using Verilog or VHDL code, describe the digital logic of an unsigned array multiplier to calculate the unsigned product of two unsigned 4-bit numbers.

For this section the structural verilog description of the unsigned 4x4 array multiplier was written based on section 11.9.1 from the reference textbook:

```
`timescale 1ns / 1ps

// 4x4 unsigned array multiplier (example 11.9.1 from reference textbook)
module multiplier(X, Y, P);
        input [3:0] X, Y;
        output[7:0] P;

        wire [3:0] s0_out, s1_out, s2_out, s3_out, s4_out;
        wire [3:0] c0_out, c1_out, c2_out, c3_out, c4_out;
        wire [3:0] x0y_in, x1y_in, x2y_in, x3y_in;

        // CSA row 0
        and (x0y_in[0], X[0], Y[0]);
        and (x0y_in[1], X[0], Y[1]);
        and (x0y_in[2], X[0], Y[2]);
        and (x0y_in[3], X[0], Y[3]);
        full_adder fa_x0y0 (x0y_in[0], 1'b0, 1'b0, s0_out[0], c0_out[0]);
        full_adder fa_x0y1 (x0y_in[1], 1'b0, 1'b0, s0_out[1], c0_out[1]);
        full_adder fa_x0y2 (x0y_in[2], 1'b0, 1'b0, s0_out[2], c0_out[2]);
        full_adder fa_x0y3 (x0y_in[3], 1'b0, 1'b0, s0_out[3], c0_out[3]);

        // CSA row 1
        and (x1y_in[0], X[1], Y[0]);
        and (x1y_in[1], X[1], Y[1]);
        and (x1y_in[2], X[1], Y[2]);
        and (x1y_in[3], X[1], Y[3]);
        full_adder fa_x1y0 (x1y_in[0], s0_out[1], c0_out[0], s1_out[0], c1_out[0]);
        full_adder fa_x1y1 (x1y_in[1], s0_out[2], c0_out[1], s1_out[1], c1_out[1]);
        full_adder fa_x1y2 (x1y_in[2], s0_out[3], c0_out[2], s1_out[2], c1_out[2]);
        full_adder fa_x1y3 (x1y_in[3], 1'b0, c0_out[3], s1_out[3], c1_out[3]);

        // CSA row 2
        and (x2y_in[0], X[2], Y[0]);
        and (x2y_in[1], X[2], Y[1]);
        and (x2y_in[2], X[2], Y[2]);
        and (x2y_in[3], X[2], Y[3]);
        full_adder fa_x2y0 (x2y_in[0], s1_out[1], c1_out[0], s2_out[0], c2_out[0]);
        full_adder fa_x2y1 (x2y_in[1], s1_out[2], c1_out[1], s2_out[1], c2_out[1]);
        full_adder fa_x2y2 (x2y_in[2], s1_out[3], c1_out[2], s2_out[2], c2_out[2]);
        full_adder fa_x2y3 (x2y_in[3], 1'b0, c1_out[3], s2_out[3], c2_out[3]);

        // CSA row 3
        and (x3y_in[0], X[3], Y[0]);
        and (x3y_in[1], X[3], Y[1]);
        and (x3y_in[2], X[3], Y[2]);
        and (x3y_in[3], X[3], Y[3]);
        full_adder fa_x3y0 (x3y_in[0], s2_out[1], c2_out[0], s3_out[0], c3_out[0]);
        full_adder fa_x3y1 (x3y_in[1], s2_out[2], c2_out[1], s3_out[1], c3_out[1]);
        full_adder fa_x3y2 (x3y_in[2], s2_out[3], c2_out[2], s3_out[2], c3_out[2]);
        full_adder fa_x3y3 (x3y_in[3], 1'b0, c2_out[3], s3_out[3], c3_out[3]);

        // CPA row 4
        full_adder fa_cp4(s3_out[1], c3_out[0], 1'b0, s4_out[0], c4_out[0]);
        full_adder fa_cp5(s3_out[2], c3_out[1], c4_out[0], s4_out[1], c4_out[1]);
        full_adder fa_cp6(s3_out[3], c3_out[2], c4_out[1], s4_out[2], c4_out[2]);
        full_adder fa_cp7(1'b0, c3_out[3], c4_out[2], s4_out[3], c4_out[3]);

        assign P = {s4_out[3], s4_out[2], s4_out[1], s4_out[0], s3_out[0], s2_out[0], s1_out[0], s0_out[0]};

endmodule

module full_adder(x_in, y_in, c_in, s_out, c_out);
        input x_in, y_in, c_in;
        output s_out, c_out;

        wire w1, w2, w3;

        xor(w1, x_in, y_in);
        xor(s_out, w1, c_in);

        and(w2, w1, c_in);
        and(w3, x_in, y_in);
        or(c_out, w2, w3);

endmodule
```

1.2 Provide the output of the test bench dialogue in your report to show the functionality of your multiplier.

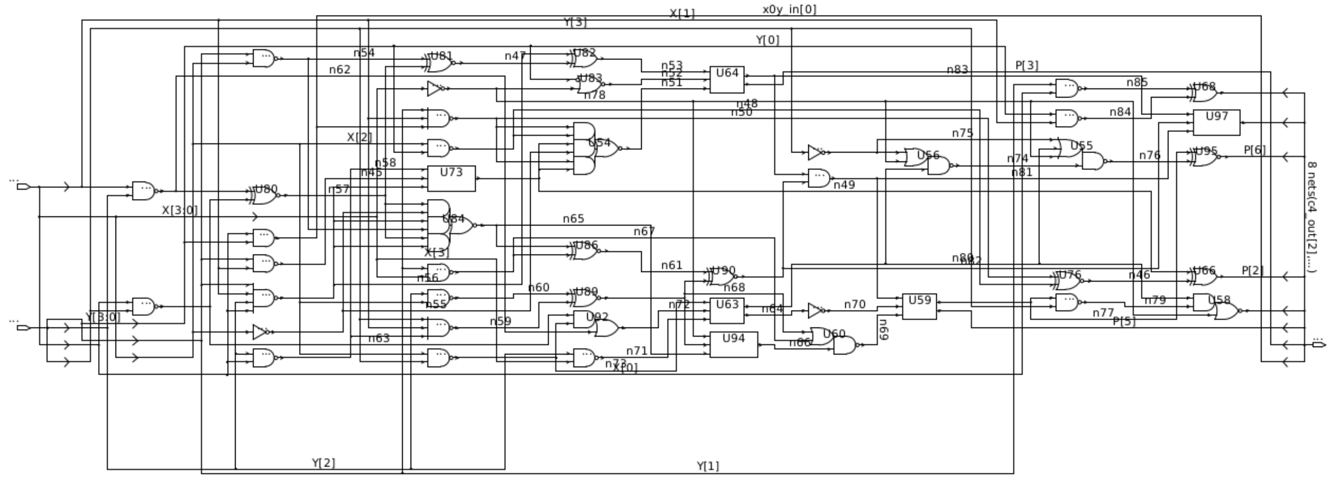
The following are the testbench dialogue results for our multiplier:

```
% ncverilog multiplier.v testbench.v
ncverilog(64): 08.20-s029: (c) Copyright 1995-2011 Cadence Design Systems, Inc.
file: multiplier.v
        module worklib.multiplier:v
                errors: 0, warnings: 0
file: testbench.v
                Caching library 'worklib' ....... Done
        Elaborating the design hierarchy:
        Building instance overlay tables: .................... Done
        Generating native compiled code:
                worklib.multiplier:v <0x28a21907>
                        streams:   1, words:   302
                worklib.testbench:v <0x76f9d4c5>
                        streams:   3, words:  2778
        Loading native compiled code:     .................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                 Instances  Unique
                Modules:                22       3
                Primitives:            116       3
                Registers:               5       5
                Scalar wires:           15       -
                Expanded wires:          8       2
                Vectored wires:          1       -
                Initial blocks:          1       1
                Cont. assignments:       1       1
                Simulation timescale:  1ps
        Writing initial simulation snapshot: worklib.testbench:v
Loading snapshot worklib.testbench:v .................... Done
ncsim> source /ncsimrc
ncsim> run
Test Completed without Errors! :)
ncsim: *W,RNQUIE: Simulation is complete.
ncsim> exit
```

2. Synthesis using standard cells

2.1 Save a copy of the synthesized schematic after synthesis

The .tcl script was executed in synopsys design compiler using 0.13um standard cells, the schematic view can be found below



2.2 Provide the generated estimates for total area and critical path timing

The area estimate report can be found below (**total area in um^2 in bold**):

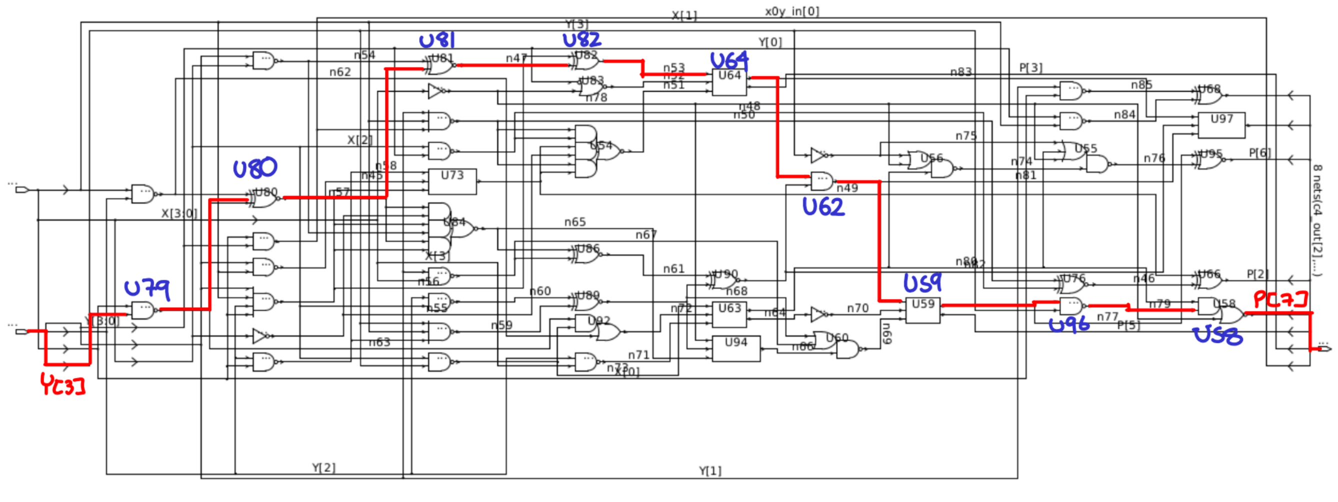
```
****************************************
Report : area
Design : multiplier
Version: F-2011.09-SP4
Date   : Sun Nov  4 19:41:37 2018
****************************************

Library(s) Used:

    scx3_cmos8rf_rvt_tt_1p2v_25c

Number of ports:                          16
Number of nets:                           57
Number of cells:                          46
Number of combinational cells:            46
Number of sequential cells:                 0
Number of macros:                           0
Number of buf/inv:                          4
Number of references:                      16

Combinational area:        437.760010
Noncombinational area:       0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:           437.760010
Total area:                 undefined
```

The critical path timing estimate report can be found below:

```
****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : multiplier
Version: F-2011.09-SP4
Date   : Sun Nov  4 19:41:37 2018
****************************************

Operating Conditions: scx3_cmos8rf_rvt_tt_1p2v_25c
Wire Load Model Mode: top

  Startpoint: Y[3] (input port)
  Endpoint: P[7] (output port)
  Path Group: (none)
  Path Type: max

  Point                                    Incr       Path
  -----------------------------------------------------------
  input external delay                     0.00       0.00 f
  Y[3] (in)                                0.00       0.00 f
  U79/Y (NAND2X1TF)                        0.05       0.05 r
  U80/Y (XNOR2X1TF)                        0.15       0.20 r
  U81/Y (XNOR2X1TF)                        0.16       0.36 r
  U82/Y (XOR2X1TF)                         0.16       0.51 r
  U64/CO (CMPR32X2TF)                      0.30       0.81 r
  U62/Y (AND2X2TF)                         0.10       0.91 r
  U59/CO (CMPR32X2TF)                      0.28       1.19 r
  U96/Y (NAND2X1TF)                        0.04       1.24 f
  U58/Y (AOI21X1TF)                        0.07       1.31 r
  P[7] (out)                               0.00       1.31 r
  data arrival time                                   1.31
  -----------------------------------------------------------
  (Path is unconstrained)
```

As expected the critical path for the unsigned array multiplier follows the path from Y[3] input to the P[7] output, the critical path is greater than others as it has to go through the row of carry propagate full adders before a result is ready at P[7]. The critical path determines the maximum clock rate for this design, because our **critical path timing is 1.31nS**, this **limits our maximum clock rate below 763.36MHz.**

The highlighted critical path can be seen in the synthesized schematic below:



2.3 Verify that your synthesis was successful by running the multiplier test bench with the newly-synthesized netlist and the 0.13um standard cell Verilog technology file:

The following are the testbench dialogue results for our synthesized multiplier netlist:

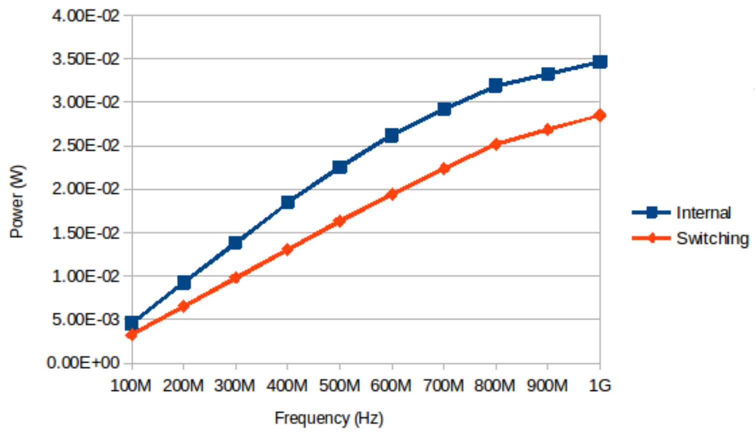
```
ncverilog testbench.v multiplier_syn.v 13rfrvt.v
ncverilog(64): 08.20-s029: (c) Copyright 1995-2011 Cadence Design Systems, Inc.
Recompiling... reason: file './multiplier_syn.v' is newer than expected.
```

```
        expected: Sun Nov  4 19:39:09 2018
        actual:   Sun Nov  4 19:41:37 2018
file: testbench.v
        module worklib.testbench:v
                errors: 0, warnings: 0
file: multiplier_syn.v
file: 13rfrvt.v
                Caching library 'worklib' ....... Done
        Elaborating the design hierarchy:
        Building instance overlay tables: .................... Done
        Generating native compiled code:
                worklib.testbench:v <0x0c46542a>
                        streams:   3, words:  2844
        Loading native compiled code:     .................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                  Instances  Unique
                Modules:                559     529
                UDPs:                   247      12
                Primitives:            1897      10
                Timing outputs:          49      18
                Registers:              162     169
                Scalar wires:           210       -
                Expanded wires:           8       2
                Initial blocks:           1       1
                Cont. assignments:        0       4
                Timing checks:         2132      29
                Simulation timescale:   1ps
        Writing initial simulation snapshot: worklib.testbench:v
Loading snapshot worklib.testbench:v .................... Done
ncsim> source /ncsimrc
ncsim> run
Test Completed without Errors! :)
ncsim: *W,RNQUIE: Simulation is complete.
ncsim> exit
```

3. Place and Route

For this section, we followed the instructions 1-5 as provided in the lab write-up.

2.6 Perform the power analysis for a constant input activity factor of 0.2, over the frequency range of 100MHz to 1GHz. Plot your results for total internal power and total switching power vs. frequency on the same graph.



From looking at the power estimates for internal and switching power we can see there is an approximately linearly increasing slope until approximately 800MHz, where there is a visible change in slope. The power analysis seems to suggest a maximum clocking frequency below 800MHz.

From our critical path we can see our maximum clock frequency is limited to around 763.36MHz.

2.7 Obtain the total area of your placed & routed design

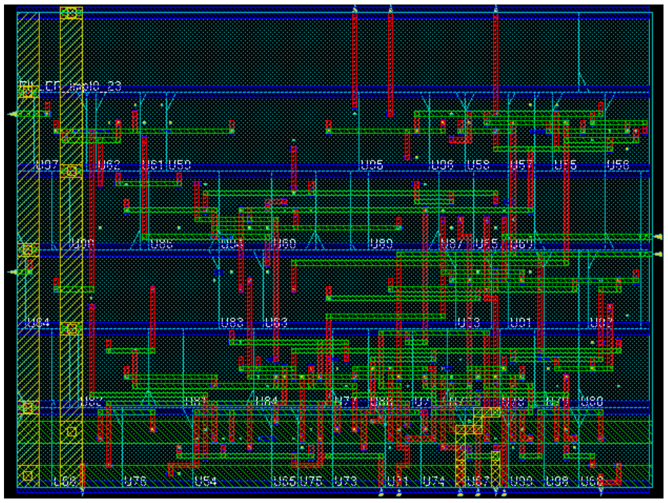
Looking into the floorplan placement information we can see:

```
==============================
Floorplan/Placement Information
==============================
Total area of Standard cells: 622.080 um^2
Total area of Standard cells(Subtracting Physical Cells): 437.760 um^2
Total area of Macros: 0.000 um^2
Total area of Blockages: 0.000 um^2
Total area of Pad cells: 0.000 um^2
Total area of Core: 625.536 um^2
Total area of Chip: 625.536 um^2
Effective Utilization: 1.0000e+00
Number of Cell Rows: 6
```

The total area of the standard cells agree exactly with our prior estimate 437.76um^2, however the total core area is 625.536um^2: this could be due to the added power stripes, routing among cells but most importantly the additional filler cells added for full area utilization.

2.8 Include in your report a screen capture of your placed & routed design in Cadence Encounter.

The following is the screen-shot from cadence encounter after place and route



2.9 Test the newly-synthesized netlist and the 0.13um standard cell Verilog technology file, provide the output of the NCVerilog test bench dialogue in your report.

```
% ncverilog testbench.v multiplier_netlist.v 13rfrvt.v
ncverilog(64): 08.20-s029: (c) Copyright 1995-2011 Cadence Design Systems, Inc.
file: testbench.v
file: multiplier_netlist.v
        module worklib.multiplier:v
                errors: 0, warnings: 0
file: 13rfrvt.v
                Caching library 'worklib' ....... Done
        Elaborating the design hierarchy:
        Building instance overlay tables: .................... Done
        Generating native compiled code:
                worklib.multiplier:v <0x579b66f6>
                        streams:   0, words:     0
        Loading native compiled code:     .................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                  Instances  Unique
                Modules:                559     529
                UDPs:                   247      12
                Primitives:            1897      10
                Timing outputs:          49      18
                Registers:              162     169
                Scalar wires:           210       -
                Expanded wires:           8       2
                Initial blocks:           1       1
                Cont. assignments:        0       4
                Timing checks:         2132      29
                Simulation timescale:   1ps
        Writing initial simulation snapshot: worklib.testbench:v
Loading snapshot worklib.testbench:v .................... Done
ncsim> source /ncsimrc
ncsim> run
Test Completed without Errors! :)
ncsim: *W,RNQUIE: Simulation is complete.
ncsim> exit
```

As can be seen the test-bench simulation of the net-list after place and route completed successfully.