TinyMIPS CPU

ECE 4150 Final Project

Nathan Pen and Jaret Williams

Introduction

Created Tiny MIPS CPU in Verilog and used Synopsis and Cadence tools to make one physical chip

Combined all ASIC Design flow steps learned in class and lab into one cumulative project

Previous Presentation

All Verilog modules of different components done and simulated using a testbench

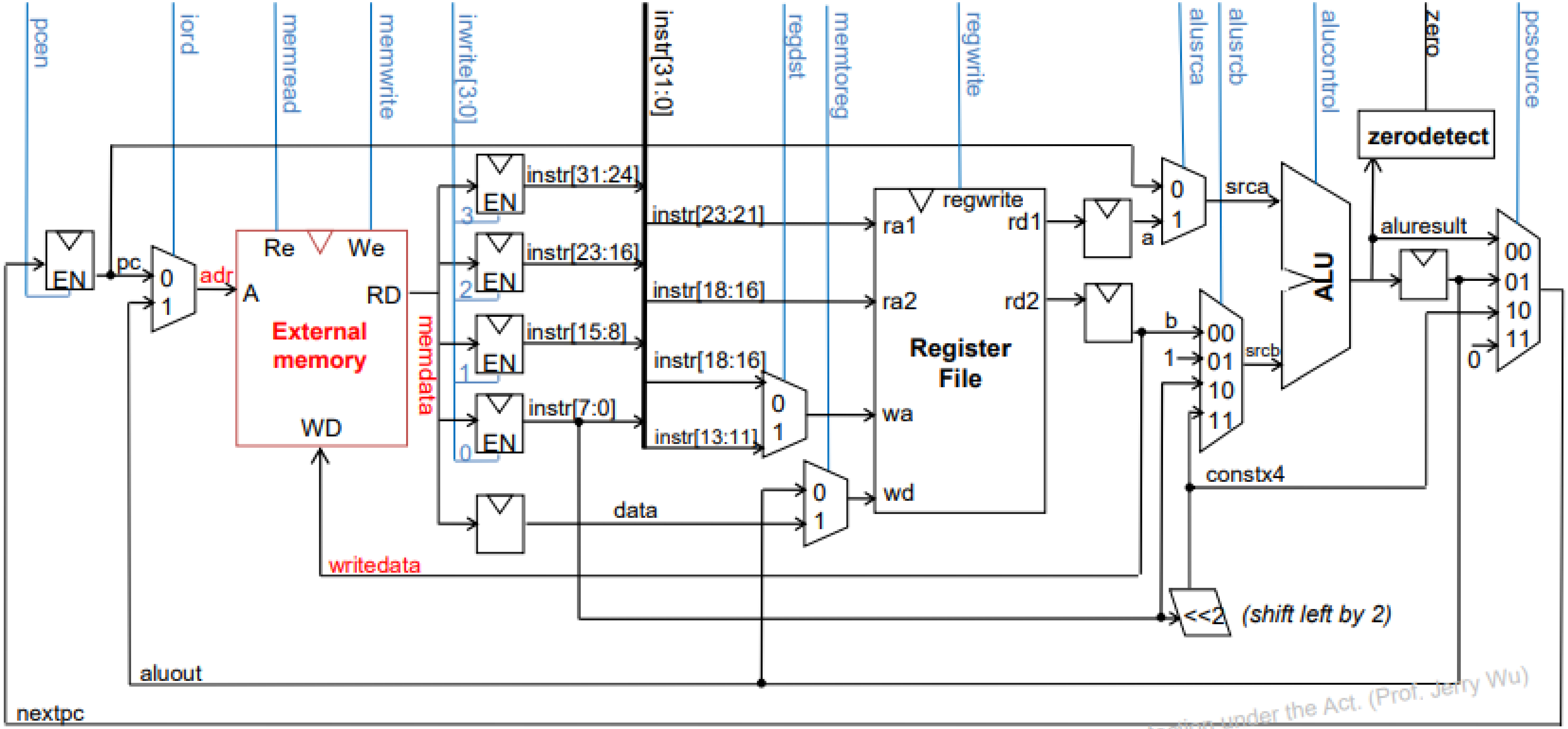
Connected all components in overall MIPS CPU Verilog module

Initial synthesis of MIPS CPU

What we Have done Since

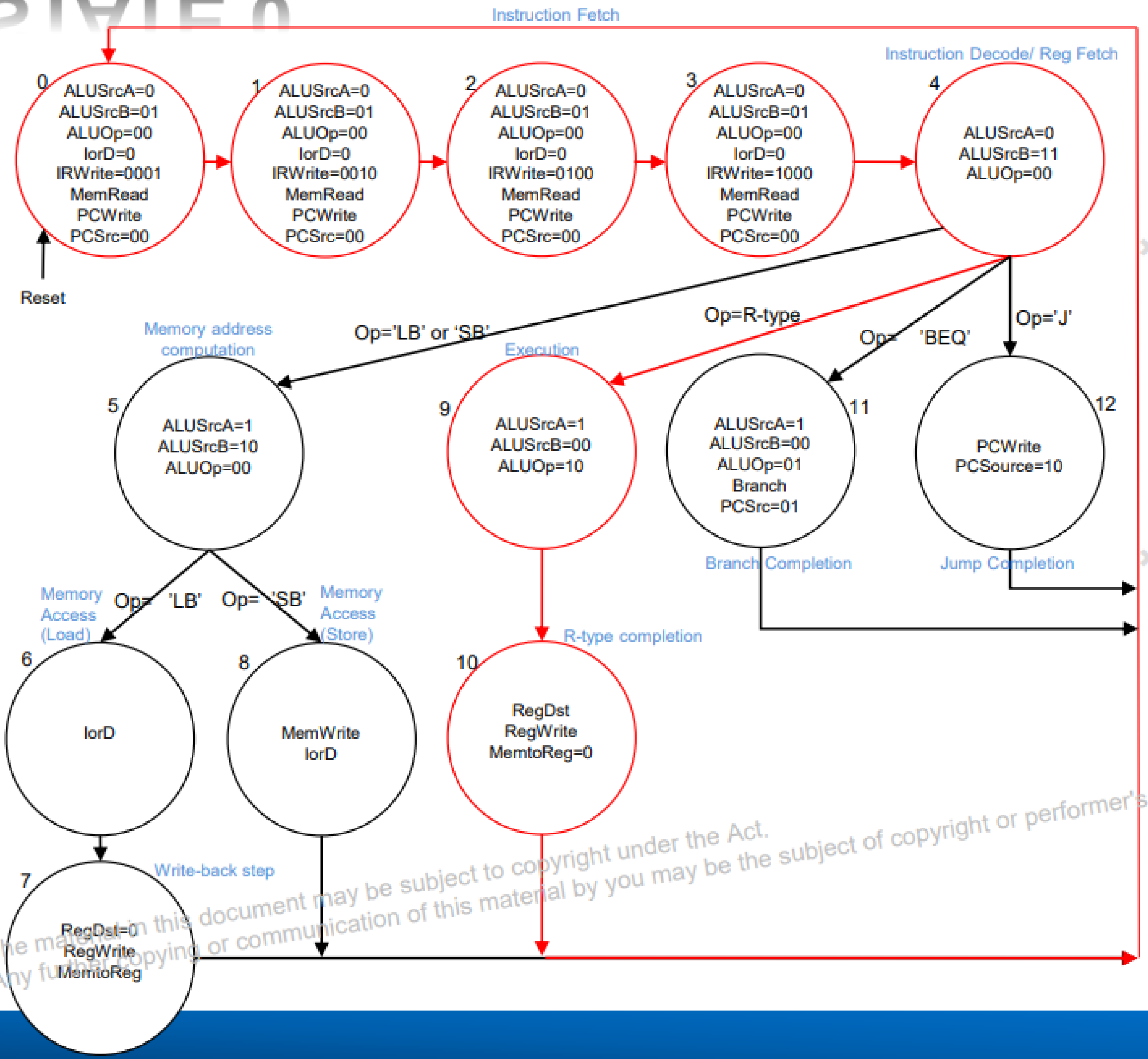
- Tested functionality of MIPS CPU memory file and Fibonacci instructions in overall test bench
- Inserted Scan Cells into CPU design and synthesized schematic with Scan Cell DFF’s
- Performed ATPG test on CPU with Scan Cells inserted and received data for possible Stuck at faults and test patterns
- Completed place and route of synthesized scan cell in Cadence Innovus
- Were able to insert CPU layout and auto route all ports to a pad frame using Cadence

DATAPATH – HOW TO CODE



- Everything EXCEPT: externalMemory is in the datapath verilog module

R-TYPE INSTRUCTION – RETURN TO STATE 0



From State 10 back to State 0

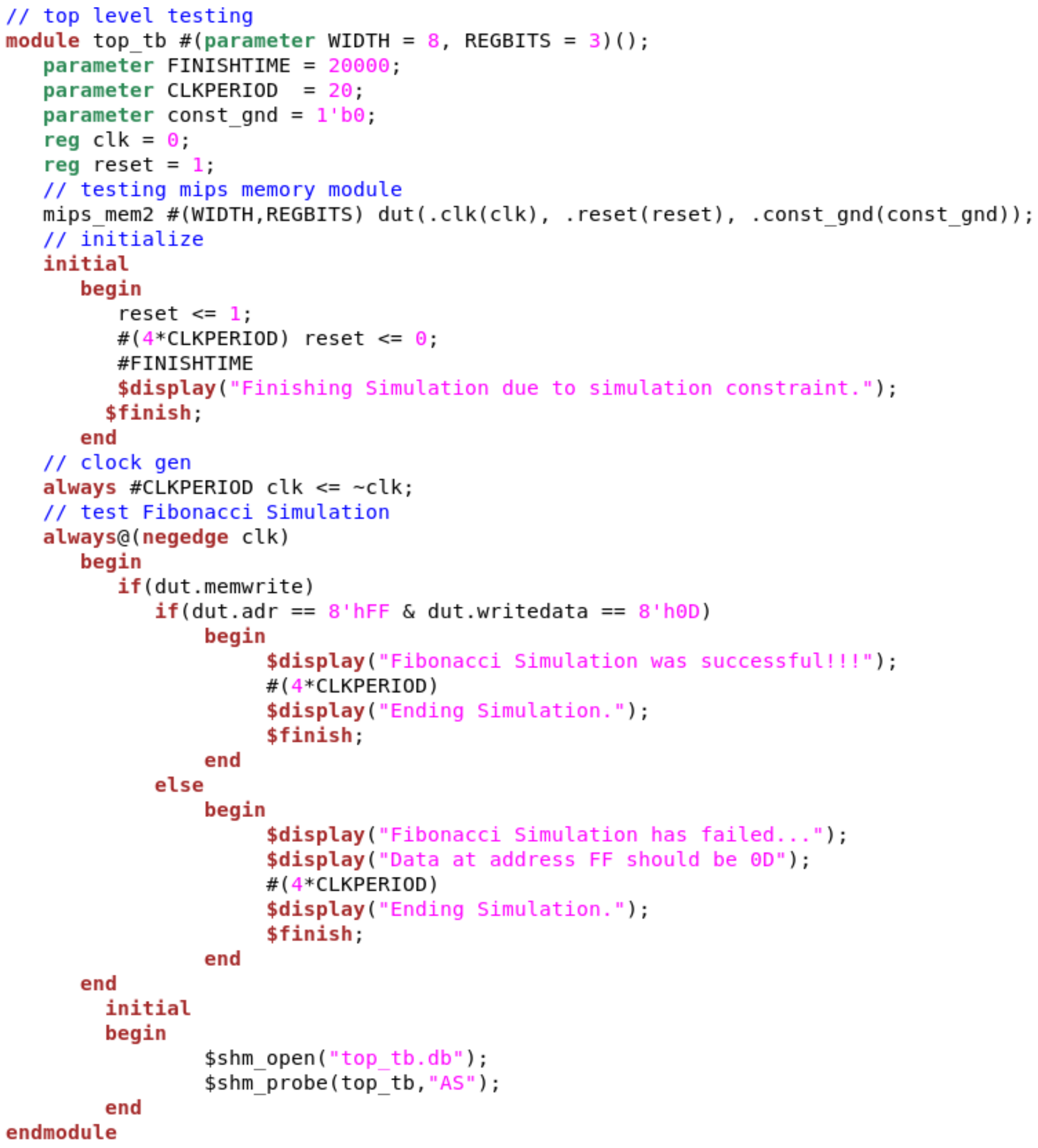
- R-type inst. Is complete
- Answer is stored in destination register in register file

FSM returns to state 0

PC has next instruction's address loaded into it.

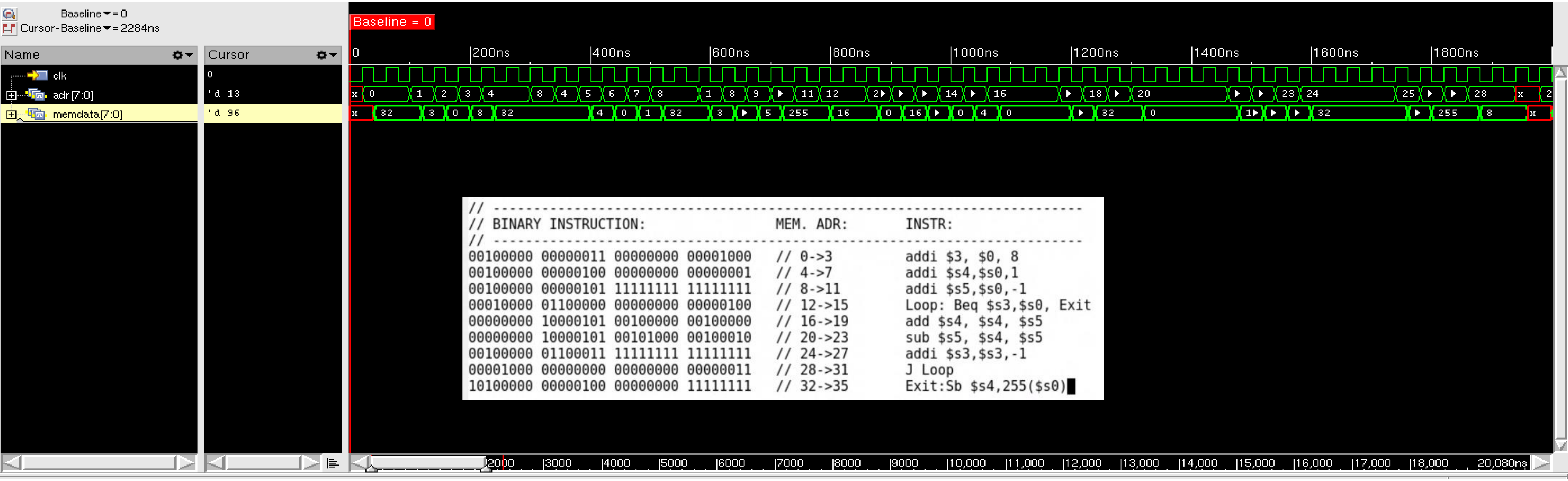
That occurred in state 3 to state 4 transition

Full CPU and Memory Verilog Test Bench



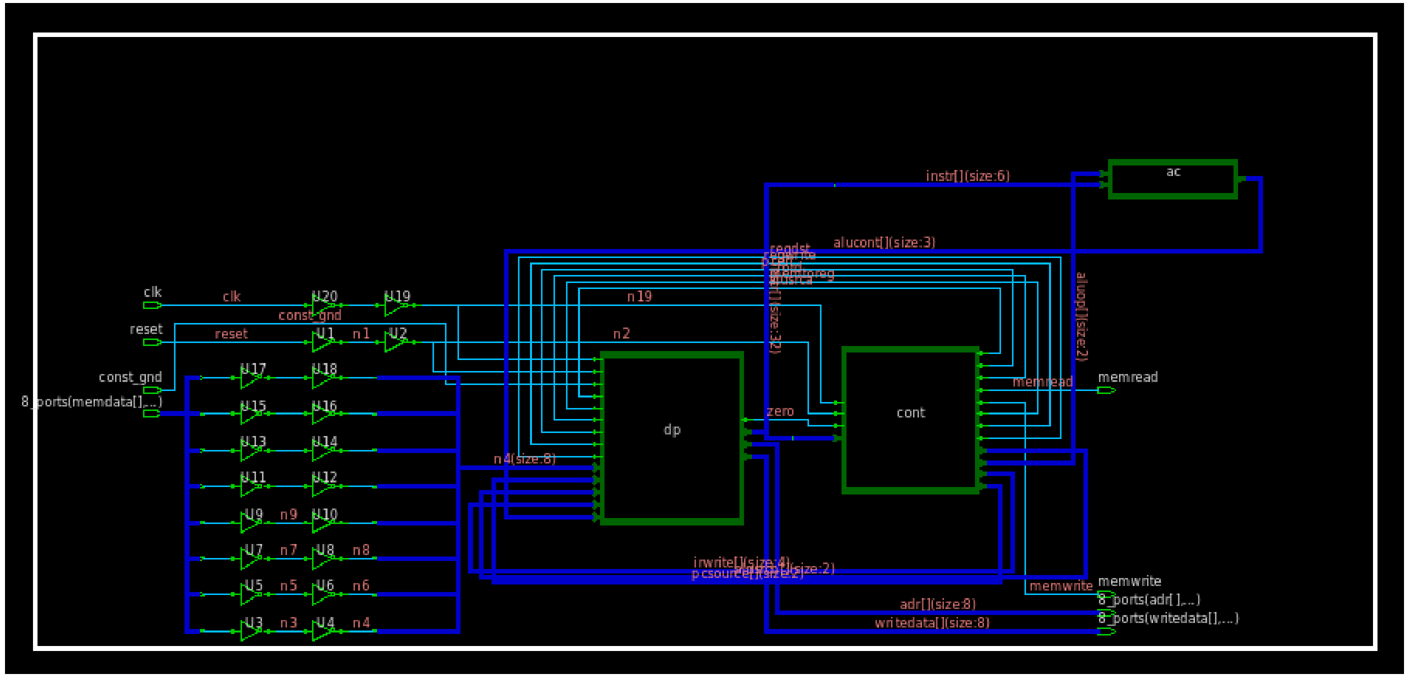
```
// top level testing
module top_tb #(parameter WIDTH = 8, REGBITS = 3)();
   parameter FINISHTIME = 20000;
   parameter CLKPERIOD  = 20;
   parameter const_gnd = 1'b0;
   reg clk = 0;
   reg reset = 1;
   // testing mips memory module
   mips_mem2 #(WIDTH,REGBITS) dut(.clk(clk), .reset(reset), .const_gnd(const_gnd));
   // initialize
   initial
      begin
         reset <= 1;
         #(4*CLKPERIOD) reset <= 0;
         #FINISHTIME
         $display("Finishing Simulation due to simulation constraint.");
        $finish;
      end
   // clock gen
   always #CLKPERIOD clk <= ~clk;
   // test Fibonacci Simulation
   always@(negedge clk)
      begin
         if(dut.memwrite)
            if(dut.adr == 8'hFF & dut.writedata == 8'h0D)
                begin
                     $display("Fibonacci Simulation was successful!!!");
                     #(4*CLKPERIOD)
                     $display("Ending Simulation.");
                     $finish;
                end
            else
                begin
                     $display("Fibonacci Simulation has failed...");
                     $display("Data at address FF should be 0D");
                     #(4*CLKPERIOD)
                     $display("Ending Simulation.");
                     $finish;
                end
      end
        initial
        begin
                $shm_open("top_tb.db");
                $shm_probe(top_tb,"AS");
        end
endmodule
```

Full CPU and Memory Verilog Test Bench

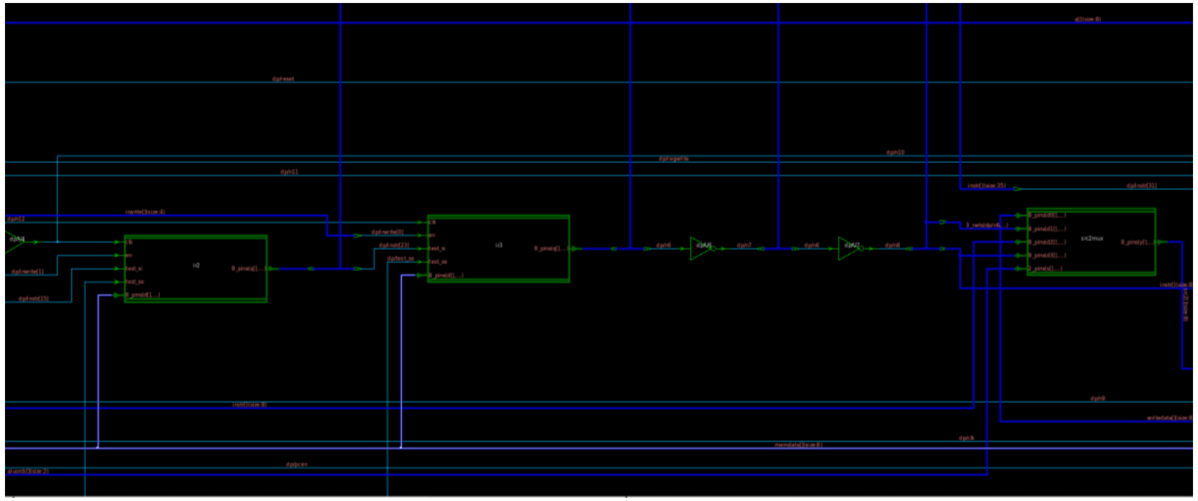


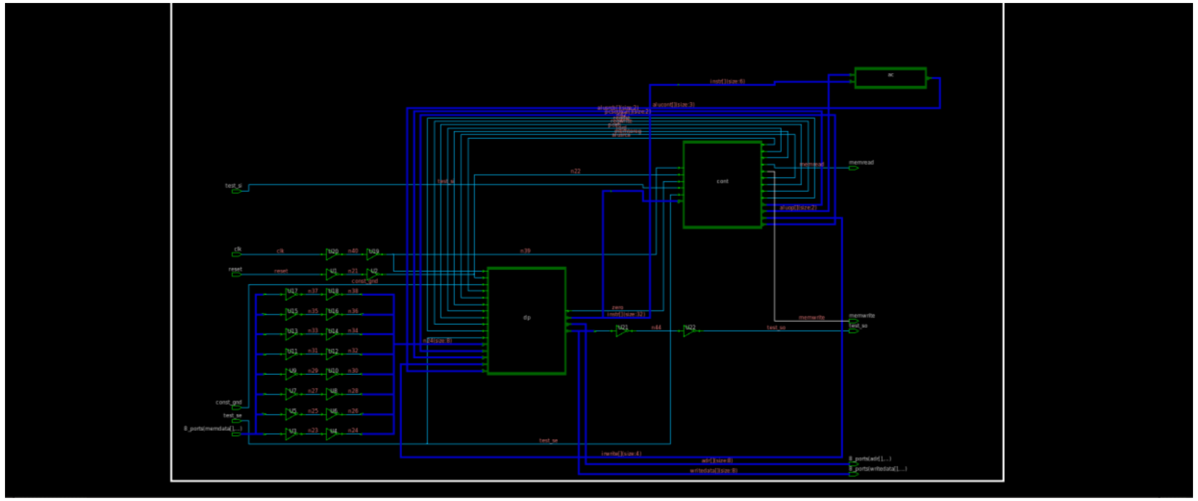
```
// -------------------------------------------------------------------
// BINARY INSTRUCTION:                  MEM. ADR:        INSTR:
// -------------------------------------------------------------------
00100000 00000011 00000000 00001000   // 0->3          addi $3, $0, 8
00100000 00000100 00000000 00000001   // 4->7          addi $s4,$s0,1
00100000 00000101 11111111 11111111   // 8->11         addi $s5,$s0,-1
00010000 01100000 00000000 00000100   // 12->15        Loop: Beq $s3,$s0, Exit
00000000 10000101 00100000 00100000   // 16->19        add $s4, $s4, $s5
00000000 10000101 00101000 00100010   // 20->23        sub $s5, $s4, $s5
00100000 01100011 11111111 11111111   // 24->27        addi $s3,$s3,-1
00001000 00000000 00000000 00000011   // 28->31        J Loop
10100000 00000100 00000000 11111111   // 32->35        Exit:Sb $s4,255($s0)
```

CPU Synthesis

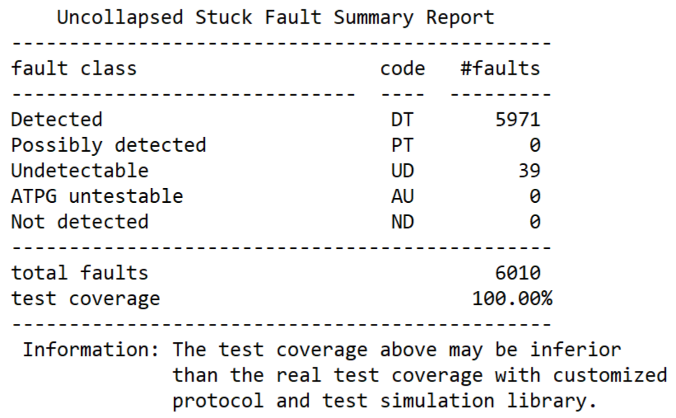


CPU With Scan Cell Synthesis

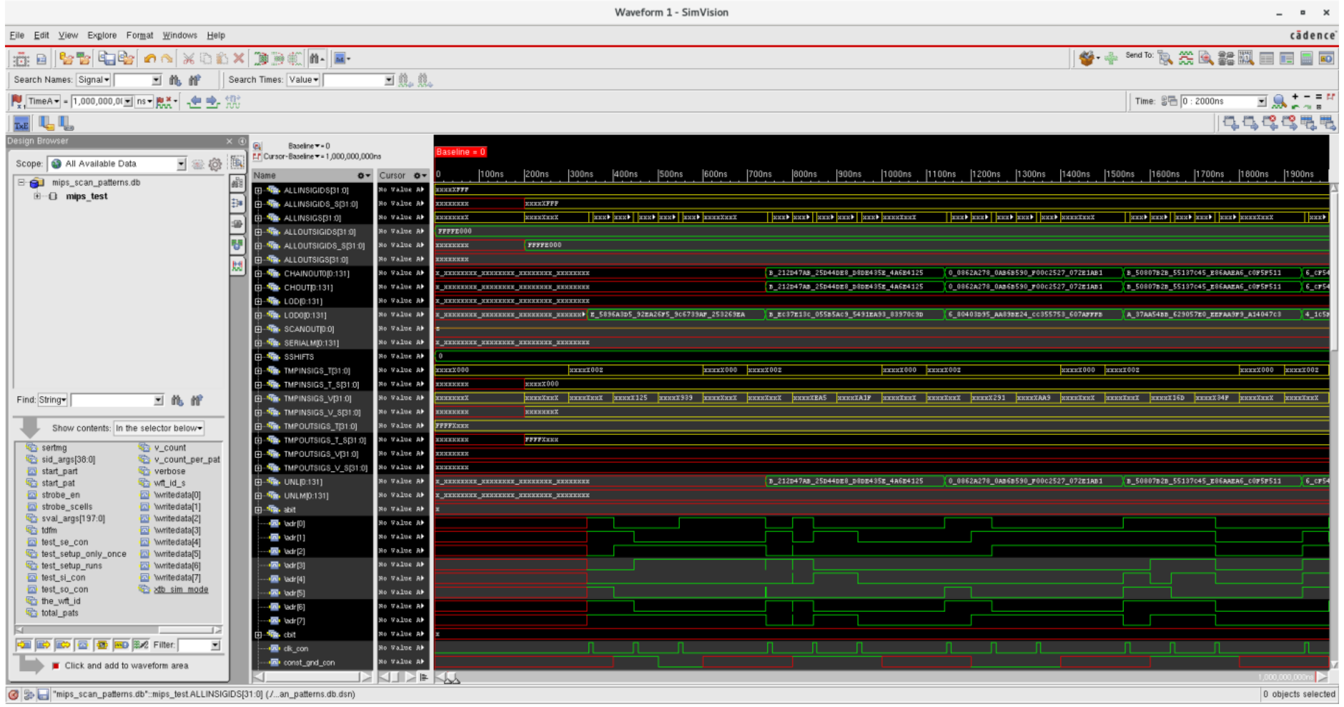




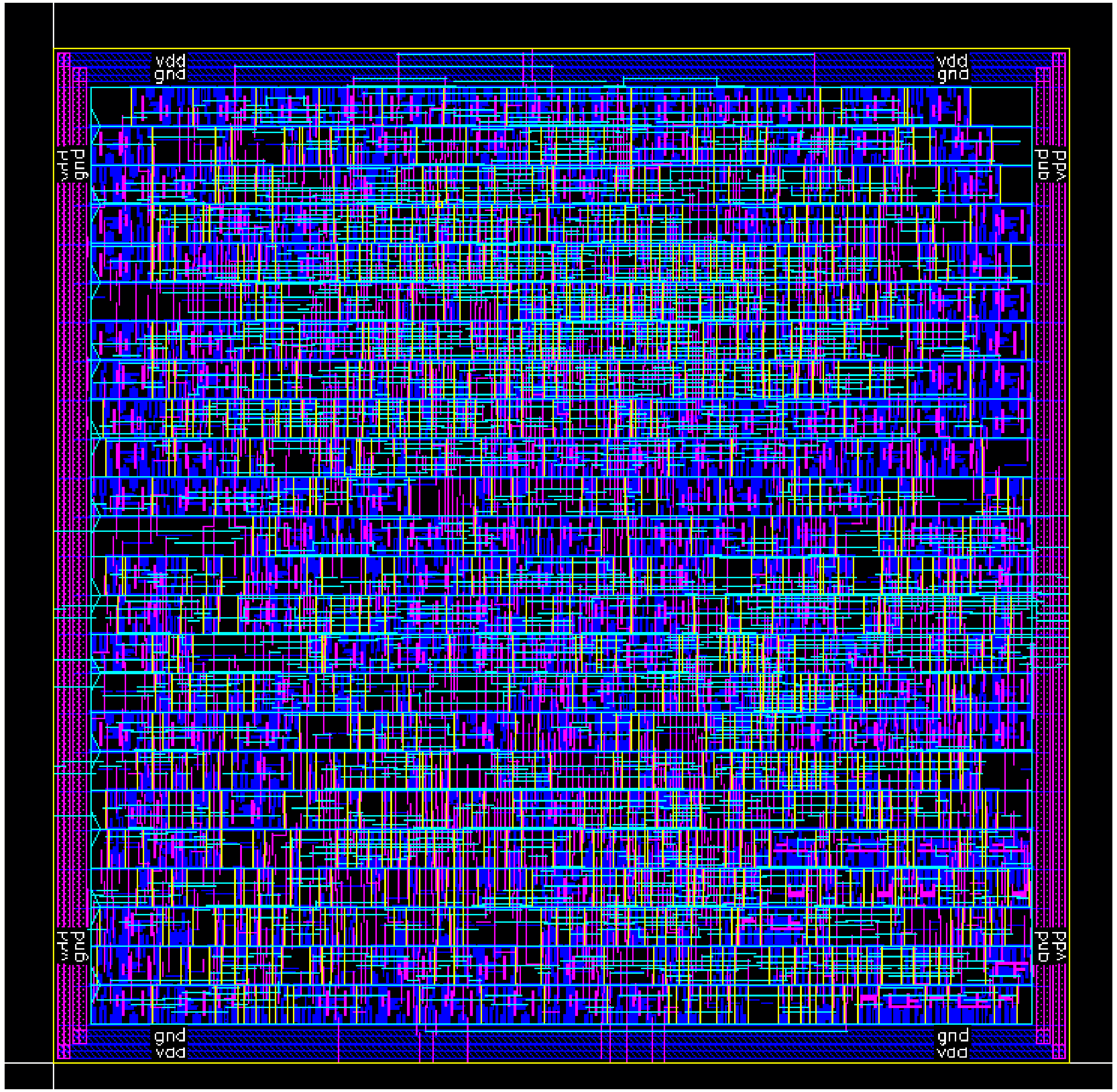
ATPG Test Results



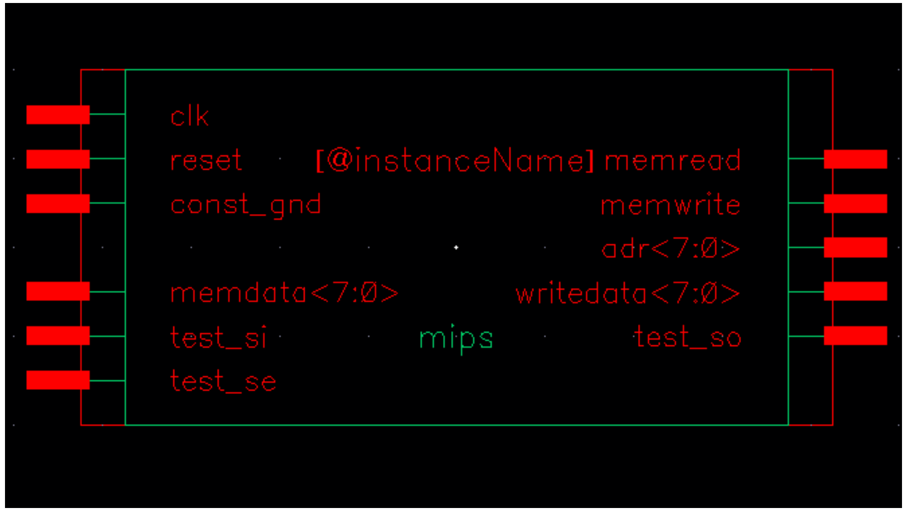
```
   Uncollapsed Stuck Fault Summary Report
-----------------------------------------------
fault class                     code   #faults
------------------------------  ----  ---------
Detected                         DT       5971
Possibly detected                PT          0
Undetectable                     UD         39
ATPG untestable                  AU          0
Not detected                     ND          0
-----------------------------------------------
total faults                              6010
test coverage                           100.00%
-----------------------------------------------
 Information: The test coverage above may be inferior
              than the real test coverage with customized
              protocol and test simulation library.
```

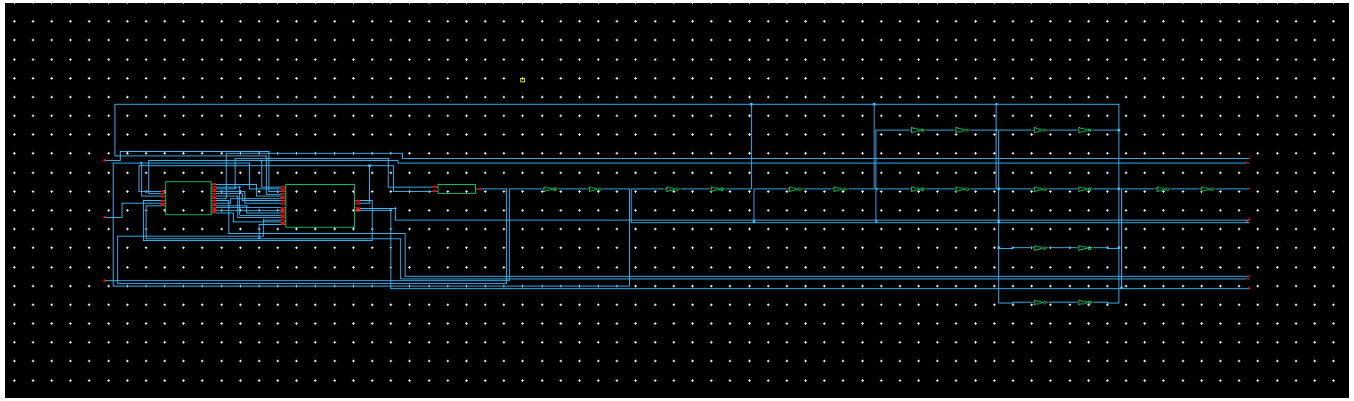


Place and Route of CPU

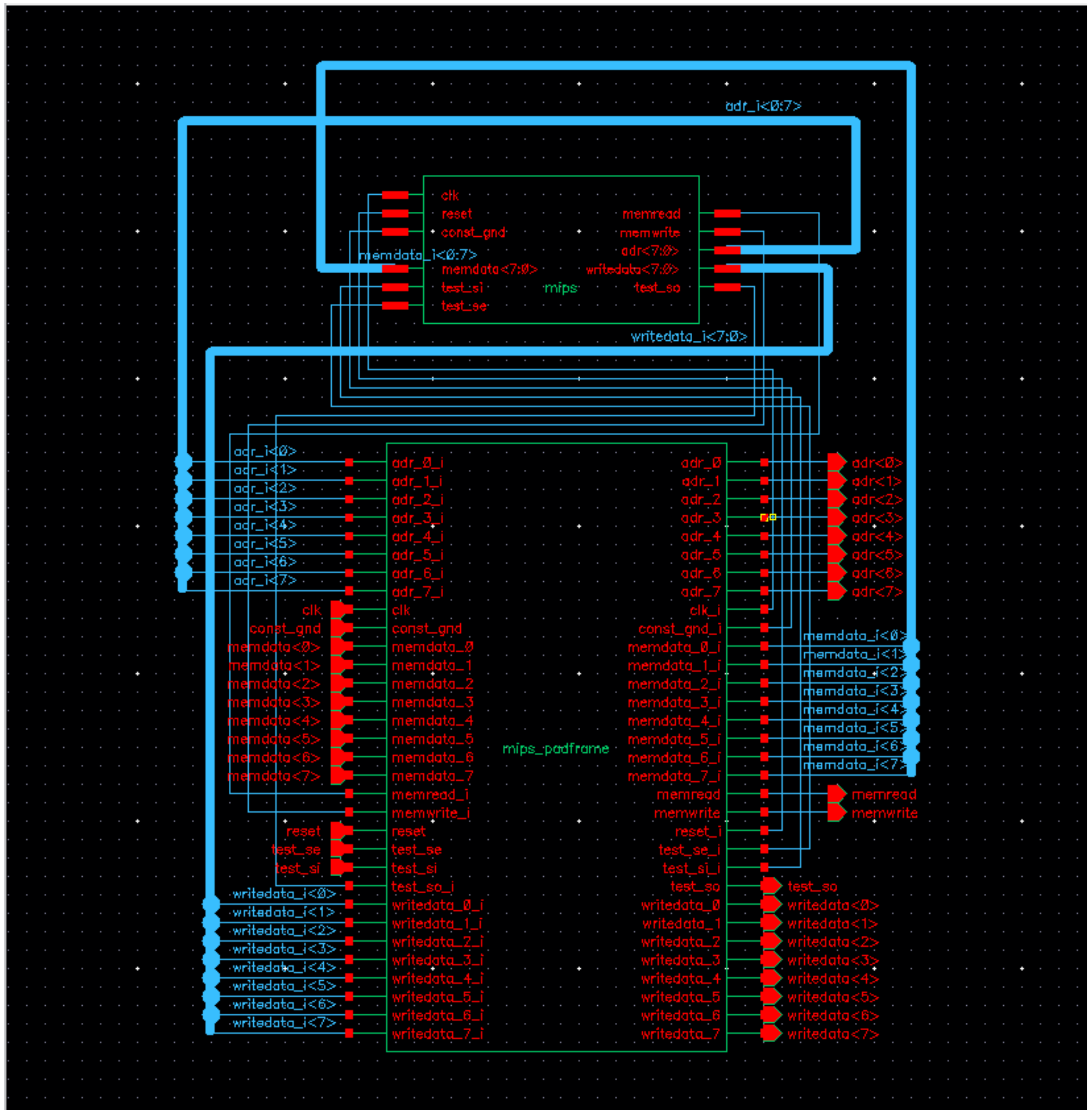


CPU Circuit in Cadence

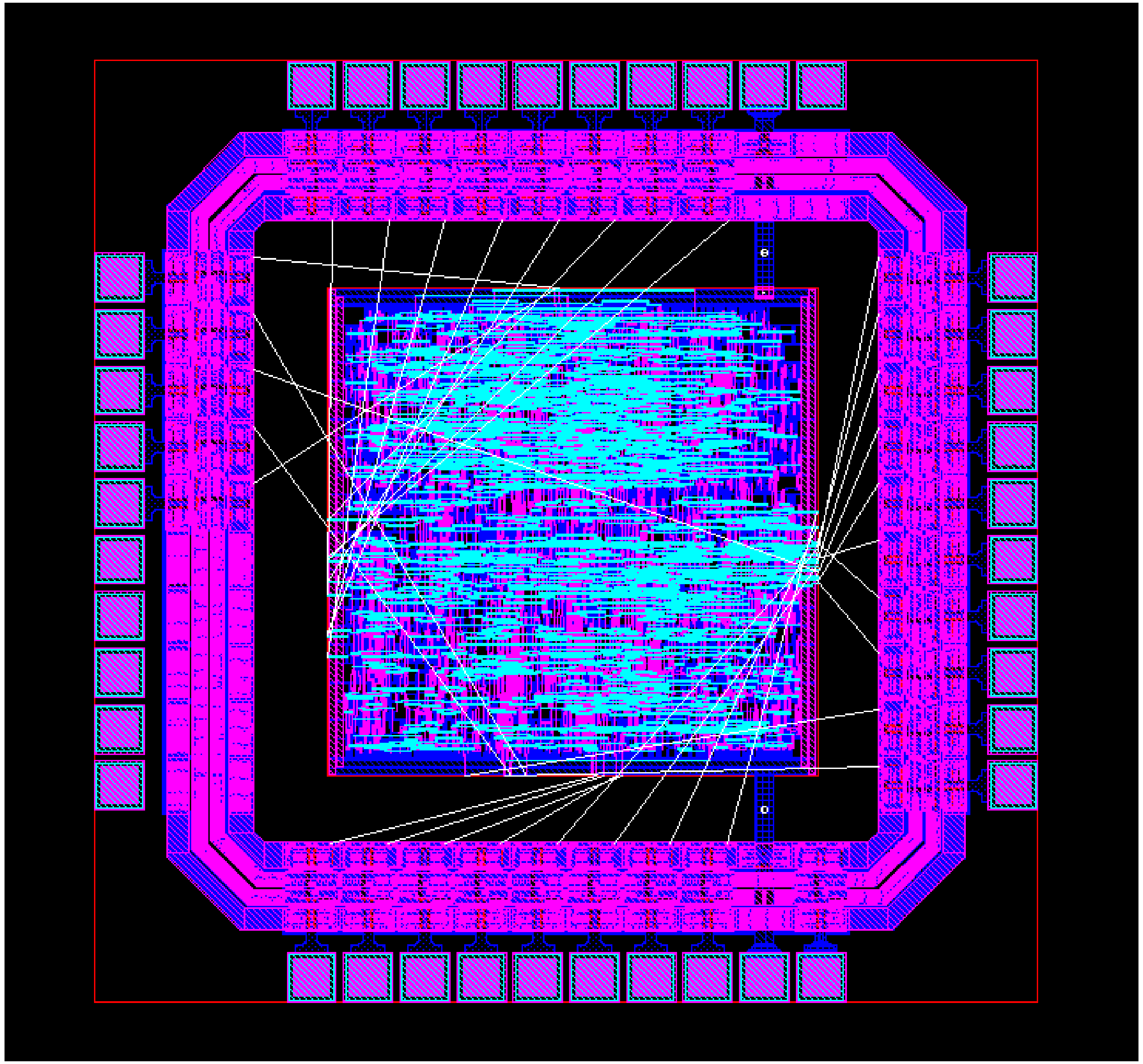




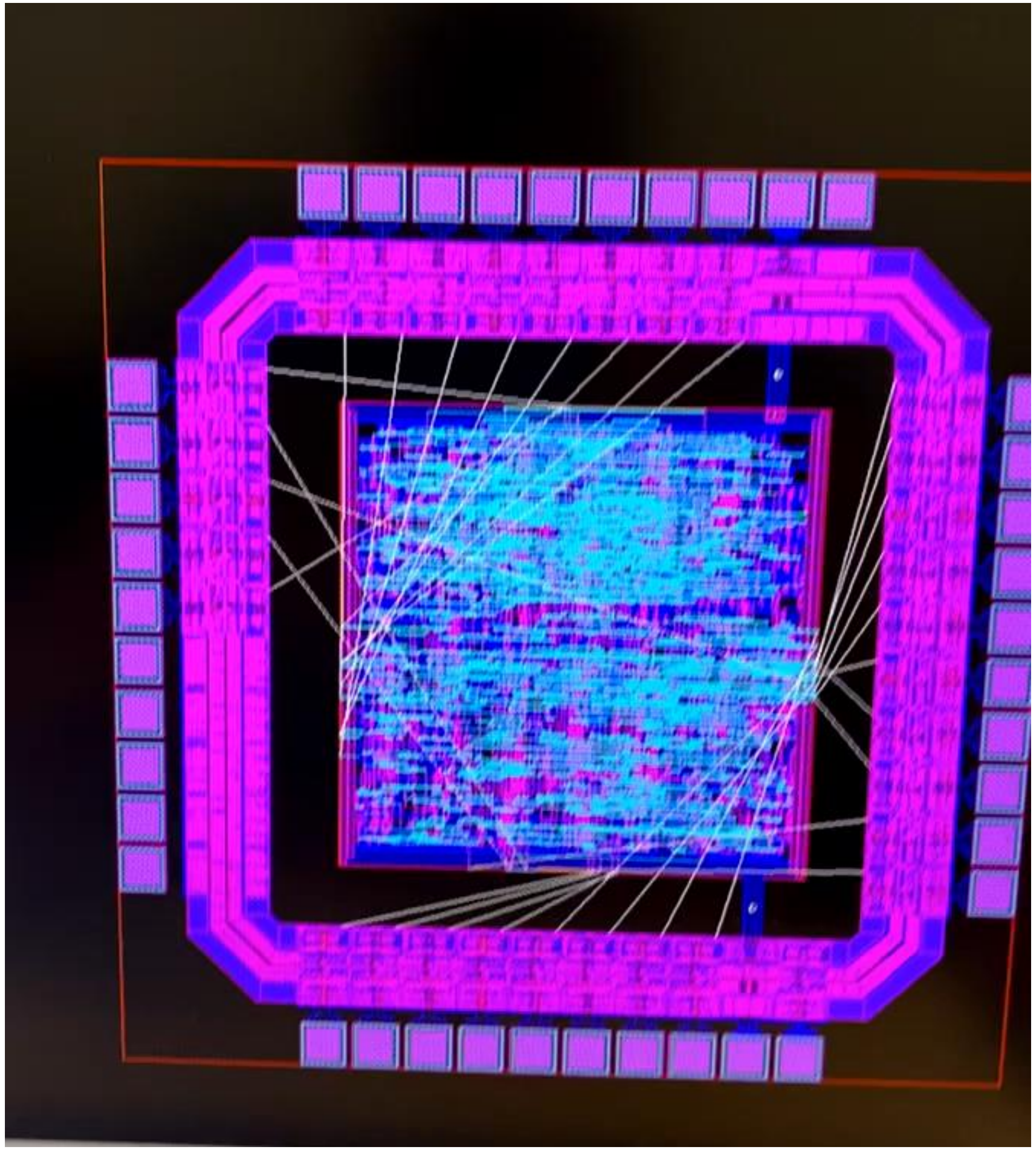
Pad frame Schematic



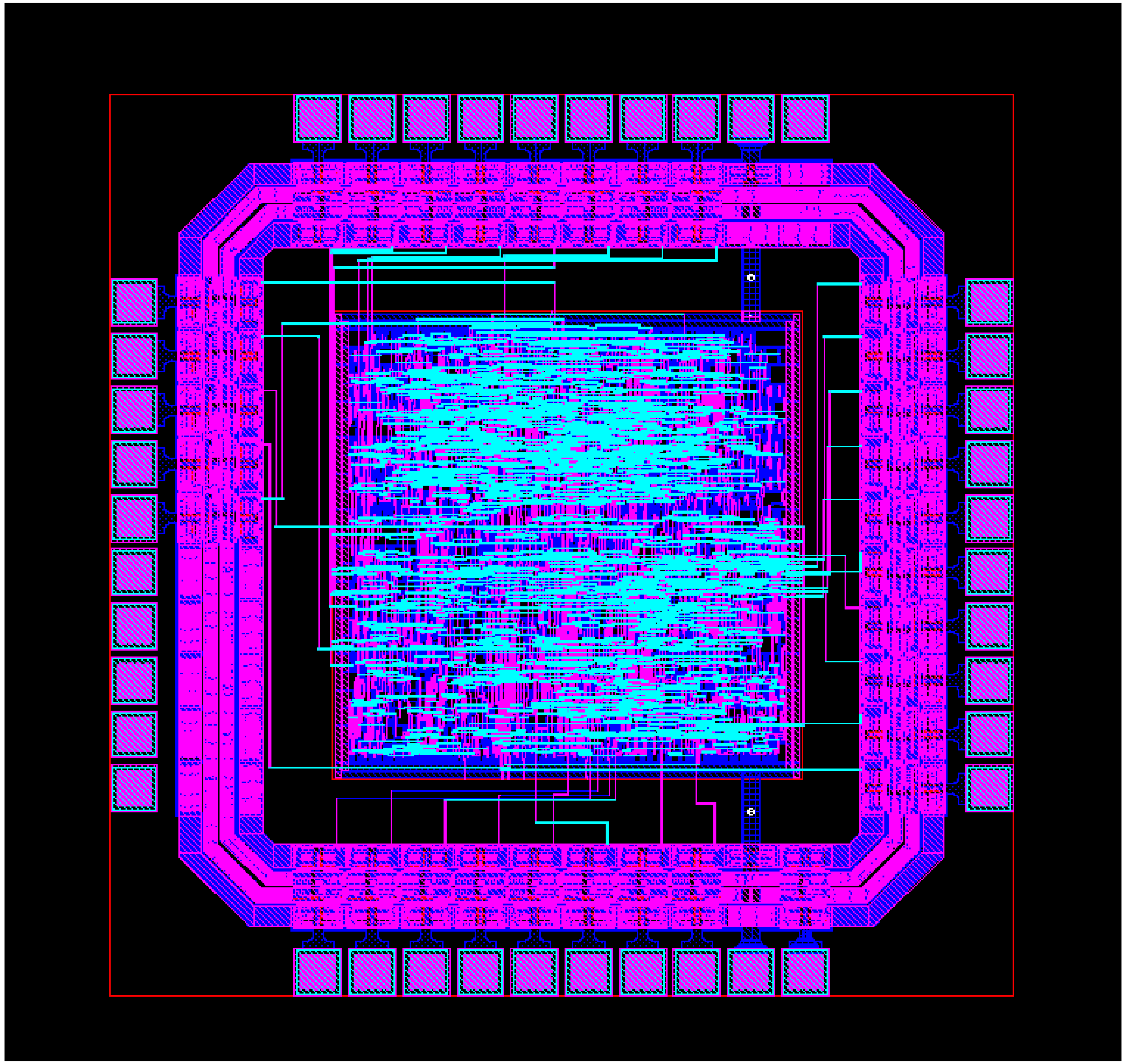
CPU in Pad Frame



Auto Routing Video



CPU Routed in Pad Frame



Challenges

- Following overall test bench simulation to ensure proper operation of CPU with memory file
    - Found error in Datapath file
- Had to redo Synthesis of design with Scan Cell because were missing .sdc file
- Had to redo Place and Route with updated default.view file
- Had to ensure all padframe files were in the same libraries linked to the same tech libraries to complete auto routing

Final Steps

Submit GDS File with final report

Get layout Fabricated if possible

Do testing on physical chip

Questions?