Experiment #2 - FPGA Realization of Radix-4 Multiplier

Ava Mir - Nesa Abbasi,
810199501- 810199457

1.1

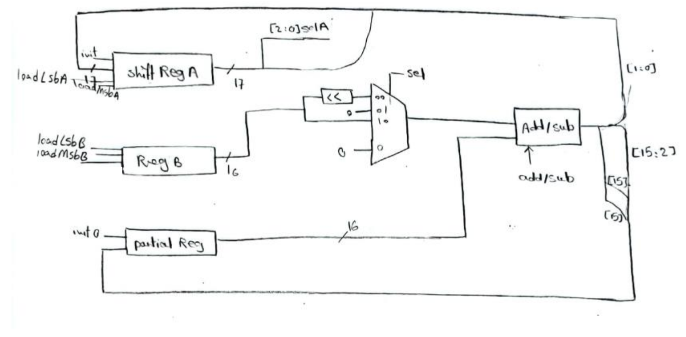


Fig. 1 DataPath

1.2

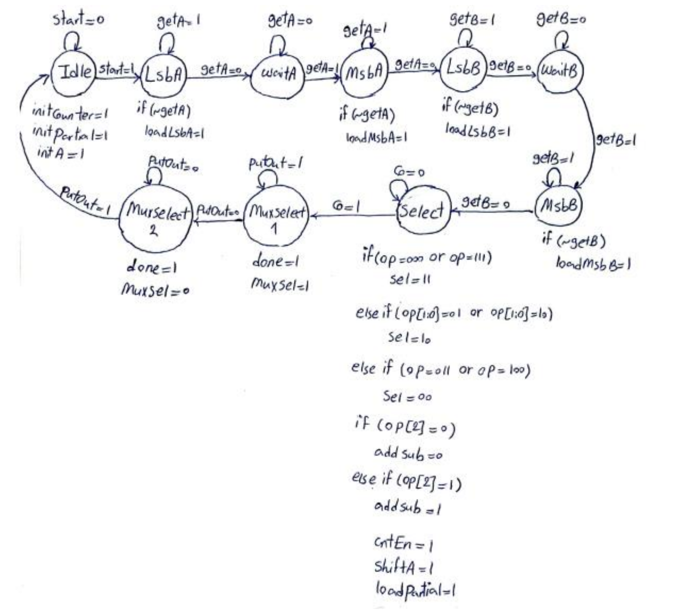
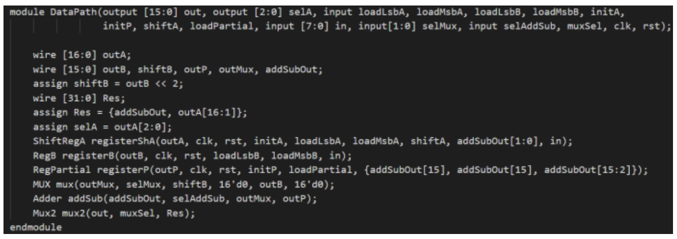


Fig. 2 Controller

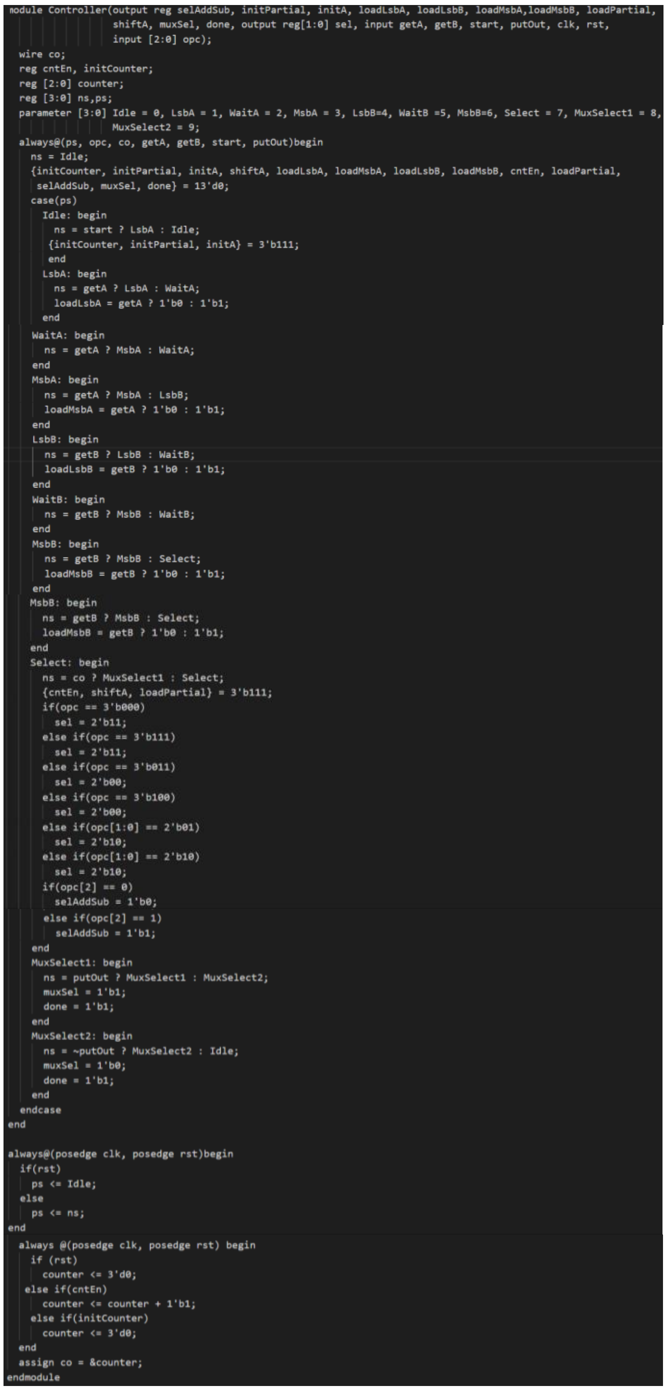
1.3



```verilog
module DataPath(output [15:0] out, output [2:0] selA, input loadLsbA, loadMsbA, loadLsbB, loadMsbB, initA,
                initP, shiftA, loadPartial, input [7:0] in, input[1:0] selMux, input selAddSub, muxSel, clk, rst);

    wire [16:0] outA;
    wire [15:0] outB, shiftB, outP, outMux, addSubOut;
    assign shiftB = outB << 2;
    wire [31:0] Res;
    assign Res = {addSubOut, outA[16:1]};
    assign selA = outA[2:0];
    ShiftRegA registerShA(outA, clk, rst, initA, loadLsbA, loadMsbA, shiftA, addSubOut[1:0], in);
    RegB registerB(outB, clk, rst, loadLsbB, loadMsbB, in);
    RegPartial registerP(outP, clk, rst, initP, loadPartial, {addSubOut[15], addSubOut[15], addSubOut[15:2]});
    MUX mux(outMux, selMux, shiftB, 16'd0, outB, 16'd0);
    Adder addSub(addSubOut, selAddSub, outMux, outP);
    Mux2 mux2(out, muxSel, Res);
endmodule
```

Fig. 3 DataPath Verilog

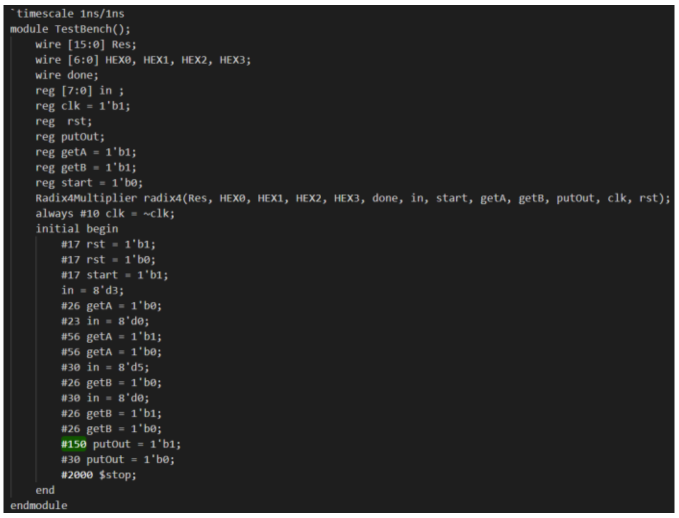


```verilog
module Controller(output reg selAddSub, initPartial, initA, loadLsbA, loadLsbB, loadMsbA,loadMsbB, loadPartial,
                  shiftA, muxSel, done, output reg[1:0] sel, input getA, getB, start, putOut, clk, rst,
                  input [2:0] opc);
  wire co;
  reg cntEn, initCounter;
  reg [2:0] counter;
  reg [3:0] ns,ps;
  parameter [3:0] Idle = 0, LsbA = 1, WaitA = 2, MsbA = 3, LsbB=4, WaitB =5, MsbB=6, Select = 7, MuxSelect1 = 8,
                  MuxSelect2 = 9;
  always@(ps, opc, co, getA, getB, start, putOut)begin
    ns = Idle;
    {initCounter, initPartial, initA, shiftA, loadLsbA, loadMsbA, loadLsbB, loadMsbB, cntEn, loadPartial,
     selAddSub, muxSel, done} = 13'd0;
    case(ps)
      Idle: begin
        ns = start ? LsbA : Idle;
       {initCounter, initPartial, initA} = 3'b111;
       end
      LsbA: begin
        ns = getA ? LsbA : WaitA;
        loadLsbA = getA ? 1'b0 : 1'b1;
      end
     WaitA: begin
       ns = getA ? MsbA : WaitA;
     end
     MsbA: begin
       ns = getA ? MsbA : LsbB;
       loadMsbA = getA ? 1'b0 : 1'b1;
     end
     LsbB: begin
       ns = getB ? LsbB : WaitB;
       loadLsbB = getB ? 1'b0 : 1'b1;
     end
     WaitB: begin
       ns = getB ? MsbB : WaitB;
     end
     MsbB: begin
       ns = getB ? MsbB : Select;
       loadMsbB = getB ? 1'b0 : 1'b1;
     end
    MsbB: begin
      ns = getB ? MsbB : Select;
      loadMsbB = getB ? 1'b0 : 1'b1;
    end
    Select: begin
      ns = co ? MuxSelect1 : Select;
      {cntEn, shiftA, loadPartial} = 3'b111;
      if(opc == 3'b000)
        sel = 2'b11;
      else if(opc == 3'b111)
        sel = 2'b11;
      else if(opc == 3'b011)
        sel = 2'b00;
      else if(opc == 3'b100)
        sel = 2'b00;
      else if(opc[1:0] == 2'b01)
        sel = 2'b10;
      else if(opc[1:0] == 2'b10)
        sel = 2'b10;
      if(opc[2] == 0)
        selAddSub = 1'b0;
      else if(opc[2] == 1)
        selAddSub = 1'b1;
    end
    MuxSelect1: begin
      ns = putOut ? MuxSelect1 : MuxSelect2;
      muxSel = 1'b1;
      done = 1'b1;
    end
    MuxSelect2: begin
      ns = ~putOut ? MuxSelect2 : Idle;
      muxSel = 1'b0;
      done = 1'b1;
    end
  endcase
end

always@(posedge clk, posedge rst)begin
  if(rst)
    ps <= Idle;
  else
    ps <= ns;
end
  always @(posedge clk, posedge rst) begin
    if (rst)
      counter <= 3'd0;
   else if(cntEn)
      counter <= counter + 1'b1;
    else if(initCounter)
      counter <= 3'd0;
  end
  assign co = &counter;
endmodule
```

Fig. 4 Controller Verilog

1.4



```
`timescale 1ns/1ns
module TestBench();
    wire [15:0] Res;
    wire [6:0] HEX0, HEX1, HEX2, HEX3;
    wire done;
    reg [7:0] in ;
    reg clk = 1'b1;
    reg  rst;
    reg putOut;
    reg getA = 1'b1;
    reg getB = 1'b1;
    reg start = 1'b0;
    Radix4Multiplier radix4(Res, HEX0, HEX1, HEX2, HEX3, done, in, start, getA, getB, putOut, clk, rst);
    always #10 clk = ~clk;
    initial begin
        #17 rst = 1'b1;
        #17 rst = 1'b0;
        #17 start = 1'b1;
        in = 8'd3;
        #26 getA = 1'b0;
        #23 in = 8'd0;
        #56 getA = 1'b1;
        #56 getA = 1'b0;
        #30 in = 8'd5;
        #26 getB = 1'b0;
        #30 in = 8'd0;
        #26 getB = 1'b1;
        #26 getB = 1'b0;
        #150 putOut = 1'b1;
        #30 putOut = 1'b0;
        #2000 $stop;
    end
endmodule
```

Fig. 5 TestBench Verilog

1.5

We enter 3 for A input and 5 for B input. First we enter 8’d0 for MsbA and 8’d3 for LsbA, then we enter 8’d0 for MsbB and 8’d5 for LsbB. 8’d15 will be recieved as Lsb of the output and 8’d0 will be received as Msb of the output. The end result is what we expected.

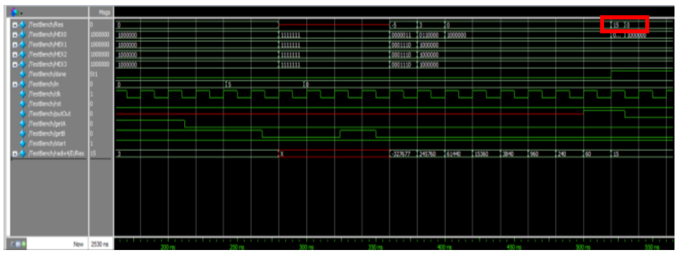


Fig. 6 Waveform

2.5

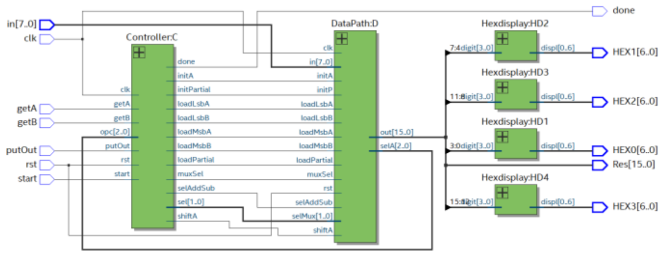


Fig. 7 RTL View

2.7

We pressed buttons in following orders:

1. Rst
2. Press Start button(Key2)
3. Show 8’d0 with switches
4. Press GetA button(Key0)
5. Show 8’d3 with switches
6. Press GetA button(Key0)
7. Show 8’d0 with switches
8. Press GetB button(Key1)
9. Show 8’d5 with switches
10. Press GetB button(Key1)
11. Press PutOut button(Key3)
12. Press PutOut button(Key3)

After we pressed the PutOut button for the first time, We received the Lsb of the output which is 4’hF(15 in Decimal). And when we pressed the PutOut button for the second time, We received the Msb of the output which is 4’h0(0 in Decimal). So the final output was 15 as we expected.

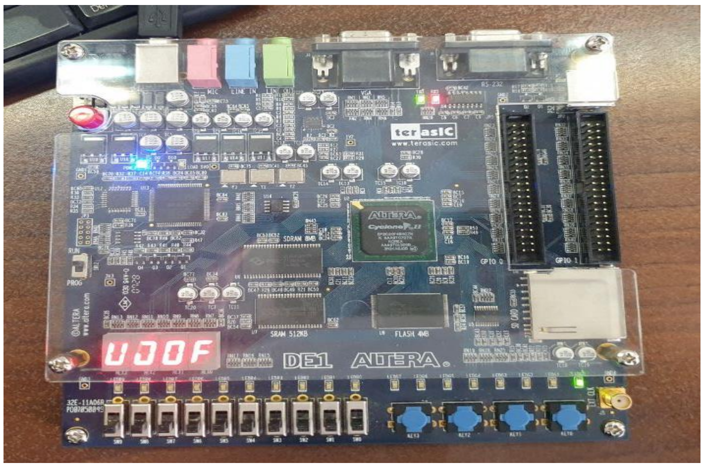


Fig. 8 FPGA Result(Lsb)

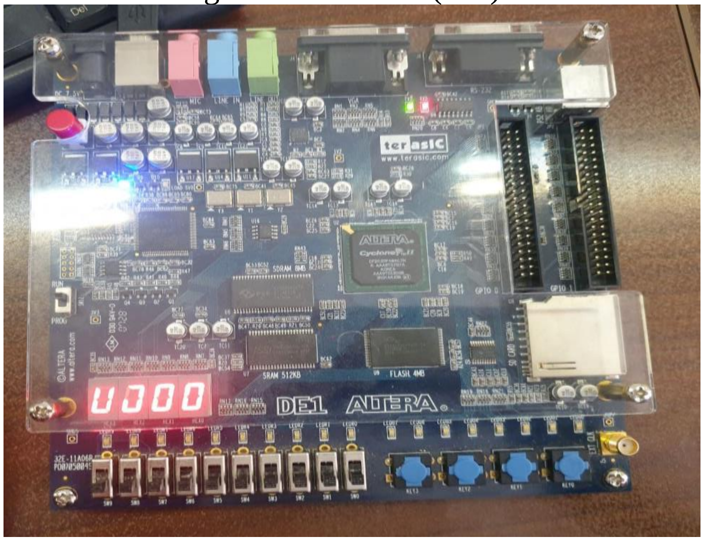


Fig. 9 FPGA Result(Msb)