FPGA 系統設計實務

期末報告

B10702226

李季鴻

Design

Decoder 真值表

		regEn	regSel	regData	reg op	reg 外	readA	readB	outB_in	outImm	LHI
00001	LHI	1	instr[10:8]	ALUout	0	0	instr[10:8]		instr[7:0]	1	1
00010	LLI	1	instr[10:8]	ALUout	0	0	x		instr[7:0]	1	0
00011	LDR	1	instr[10:8]	MEMout	1	1	instr[7:5]		instr[4:0]	1	0
00101	STR	0	x	x	0	0	instr[7:5]	instr[10:8]	instr[4:0]	1	0
00000	ADD	1	instr[10:8]	ALUout	0	0	instr[7:5]	instr[4:2]		0	0
00000	ADC	1	instr[10:8]	ALUout	0	0	instr[7:5]	instr[4:2]		0	0
00000	SUB	1	instr[10:8]	ALUout	0	0	instr[7:5]	instr[4:2]		0	0
00000	SBB	1	instr[10:8]	ALUout	0	0	instr[7:5]			0	0
00110	CMP	0	x	x	0	0	instr[7:5]	instr[4:2]		0	0
00111	ADDI	1	instr[10:8]	ALUout	0	0	instr[7:5]		instr[4:0]	1	0
01000	SUBI	1	instr[10:8]	ALUout	0	0	instr[7:5]		instr[4:0]	1	0
01011	MOV	1	instr[10:8]	ALUout	0	0	instr[7:5]		0x0000	1	0
C3	BCC	0	x	x	0	0	x	x		0	0
C2	BCS	0	x	x	0	0	x	x		0	0
C1	BNE	0	x	x	0	0	x	x		0	0
C0	BEQ	0	x	x	0	0	x	x		0	0
CE	B[AL]	0	x	x	0	0	x	x		0	0
10000	JMP	0	x	x	0	0	x	x		0	0
10001	JAL	1	instr[10:8]	pcaddr	2	1	x	x		0	0
10010	JAL	1	instr[10:8]	pcaddr	2	1	instr[7:5]		0x0000	1	0
10011	JR	0	x	x	0	0	instr[7:5]		0x0000	1	0
11100	OutR	0	x	x	0	0	instr[7:5]		0x0000	1	0
11100	HLT	0	x	x	0	0	x	x	0	0	0

add/sub	logicm	memread	memEn	memAddr	memData	done	pcen	pcext	pc inc bra
0	1	0	0	x	x	0		1	0
0	1	0	0	x	x	0		1	0
0	0	1	0	ALUout	x	0		1	0
0	0	0	1	ALUout	RB	0		1	0
0	0	0	0	x	x	0		1	0
0	0	0	0	x	x	0		1	0
1	0	0	0	x	x	0		1	0
1	0	0	0	x	x	0		1	0
1	0	0	0	x	x	0		1	0
0	0	0	0	x	x	0		1	0
1	0	0	0	x	x	0		1	0
0	0	0	0	x	x	0		1	0
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		instr[10:0]	1
0	0	0	0	x	x	0		disp	0
0	0	0	0	x	x	0		ALUout	1
0	0	0	0	x	x	0		ALUout	1
0	0	0	0	x	x	0		1	0
0	0	0	0	x	x	1		0	0

Verilog HDL

1. CPU

```
module RISC_CPU_16bits(
   input clk,rst_n,test_normal,
   input ext_wen,
   input [7:0] ext_addr,
   input [15:0] ext_data,
   output [15:0] OutR,mem_out,
   output done,
   output [1:0] tb_step,
   output [15:0] ins,
   output pc_en,
   output [15:0] pc_ext
    );
   wire [7:0] mem_addr;
   wire [15:0] mem_data;
   wire [2:0] rf_addr;
   wire [2:0] rf_readA,rf_readB;
   wire [15:0] rf_B;
   wire [15:0] rf_data;
   wire [15:0] alu_imm_B;
   wire [15:0] alu_o;
   wire [15:0] pc_addr;
   wire [1:0] rf_op;
   assign mem_clk = (ext_wen)? clk:clk_sl;
   Datapath ut(
      .mem_clk(mem_clk),
      .mem_wen(mem_wen),
      .mem_ren(mem_ren),
      .mem_addr(mem_addr),
      .mem_datain(mem_data),
      .mem_dataout(mem_out),
      .rf_clk(clk_sl),
      .rst_n(rst_n),
      .rf_en(rf_en),
      .rf_op(rf_op),
      .rf_addr(rf_addr),
      .rf_readA(rf_readA),
      .rf_readB(rf_readB),
      .rf_B(rf_B),
      .add0_sub1(alu_add0_sub1),
      .ext_imm(alu_ext_imm),
      .ext_immB(alu_imm_B),
      .LHI(alu_LHI),.LLI(alu_LLI),
      .alu_out(alu_o),
      .N(alu_N),.Z(alu_Z),.C(alu_C),.V(alu_V),
      .ctro_outR(ctro_outR),
      .OutR(OutR),
      .pc_clk(clk_sl),
      .pc_ext(pc_ext),
      .pc_inc0_jum1(pc_inc0_jum1),
      .pc_en(pc_en),
      .pc_addr(pc_addr)
   );
   Controller uut(
      .rst_n(rst_n),
      .instruction(mem_out),
      .alu_N(alu_N),.alu_Z(alu_Z),.alu_C(alu_C),.alu_V(alu_V),
      .alu_o(alu_o),
      .rf_B(rf_B),
      .pc_addr(pc_addr),
      .alu_add0_sub1(alu_add0_sub1),
      .alu_LHI(alu_LHI),
        .alu_LLI(alu_LLI),
        .alu_ext_imm(alu_ext_imm),
        .alu_imm_B(alu_imm_B),
        .rf_en(rf_en),
        .rf_op(rf_op),
        .rf_addr(rf_addr),
        .rf_readA(rf_readA),
        .rf_readB(rf_readB),
        .pc_en(pc_en),
        .pc_inc0_jum1(pc_inc0_jum1),
        .pc_ext(pc_ext),
        .ext_wen(ext_wen),
        .ext_addr(ext_addr),
        .ext_data(ext_data),
        .mem_wen(mem_wen),
        .mem_ren(mem_ren),
        .mem_addr(mem_addr),
        .mem_data(mem_data),
        .ctro_outR(ctro_outR),
        .done(done),
        .ins(ins),
        .clk(clk),
        .timer_en(~(done|ext_wen|test_normal)),
        .step(step),
        .clk_s0(clk_s0),
        .clk_s1(clk_s1),
        .tb_step(tb_step)
        );
endmodule
```

2. Controller

```
module Controller(
   input clk,rst_n,timer_en,
   input ext_wen,
   input [7:0] ext_addr,
   input [15:0] ext_data,
   output step,clk_s0,clk_s1,
   input [15:0] instruction,
   input alu_N,alu_Z,alu_C,alu_V,
   input [15:0] alu_o,
   input [15:0] rf_B,
   input [15:0] pc_addr,
   output alu_add0_sub1,alu_LHI,alu_LLI,alu_ext_imm,
   output [15:0] alu_imm_B,
   output rf_en,
   output [1:0] rf_op,
   output [2:0] rf_addr,rf_readA,rf_readB,
   output pc_en,pc_inc0_jum1,
   output [15:0] pc_ext,
   output mem_wen,mem_ren,
   output [7:0] mem_addr,
   output [15:0] mem_data,
   output ctro_outR,
   output done,
   output [15:0] ins,
   output [1:0] tb_step
    );
   Instruction_decoder uut(
      .clk(clk_s0),.clk_s1(clk_s1),
      .rst_n(rst_n),
      .step(step),
      .instruction(instruction),
      .alu_N(alu_N),.alu_Z(alu_Z),.alu_C(alu_C),.alu_V(alu_V),
      .alu_o(alu_o),
      .rf_B(rf_B),
      .pc_addr(pc_addr),
      .alu_add0_sub1(alu_add0_sub1),
      .alu_LHI(alu_LHI),
      .alu_LLI(alu_LLI),
      .alu_ext_imm(alu_ext_imm),
      .alu_imm_B(alu_imm_B),
      .rf_en(rf_en),
      .rf_op(rf_op),
      .rf_addr(rf_addr),
      .rf_readA(rf_readA),
      .rf_readB(rf_readB),
      .pc_en(pc_en),
      .pc_inc0_jum1(pc_inc0_jum1),
      .pc_ext(pc_ext),
      .ext_mem_wen(ext_wen),
      .ext_mem_addr(ext_addr),
      .ext_mem_data(ext_data),
      .mem_wen(mem_wen),
      .mem_ren(mem_ren),
      .mem_addr(mem_addr),
      .mem_data(mem_data),
      .ctro_outR(ctro_outR),
      .done(done),
      .ins(ins)
   );
   Timer_generator ut(
      .clk(clk),
      .rst_n(rst_n),
      .E(timer_en),
      .step(step),
      .clk_s0(clk_s0),
      .clk_s1(clk_s1),
      .div_count(tb_step));

endmodule
```

3. Datapath

```
module Datapath(
   input mem_clk,rf_clk,pc_clk,rst_n,
   input mem_wen,mem_ren,rf_en,pc_en,
   input [7:0] mem_addr,
   input [15:0] mem_datain,
   output [15:0] mem_dataout,
   input pc_inc0_jum1,
   input [15:0] pc_ext,
   output [15:0] pc_addr,
   input [2:0] rf_addr,
   input [1:0] rf_op,
   input [2:0] rf_readA,rf_readB,
   output [15:0] rf_B,
   input add0_sub1,LHI,LLI,ext_imm,
   input [15:0] ext_immB,
   output [15:0] alu_out,
   output N,Z,C,V,
   input ctro_outR,
   output [15:0] OutR
    );
   wire [7:0] mem_address;
   assign mem_address = (mem_wen|mem_ren)? mem_addr:pc_addr[7:0];
   wire [15:0] rf_data;
   assign rf_data =  (rf_op==2'b00)? alu_out:
                     (rf_op==2'b01)? mem_dataout : pc_addr;
   memory_256x16 my_mem(
      .clk(mem_clk),
      .write_en(mem_wen),
      .address(mem_address),
      .data_in(mem_datain),
      .data_out(mem_dataout)
   );
   RF_ALU_16bits my_rfalu(
      .clk(rf_clk),
      .rst_n(rst_n),
      .RF_en(rf_en),
      .RF_addr(rf_addr),
      .read_A(rf_readA),
      .read_B(rf_readB),
      .rB(rf_B),
      .RF_data(rf_data),
      .add0_sub1(add0_sub1),
      .ext_imm(ext_imm),
      .ext_B_data(ext_immB),
      .LHI(LHI),.LLI(LLI),
      .S(alu_out),
      .N(N),.Z(Z),.C(C),.V(V),
      .ctro_outR(ctro_outR),
      .OutR(OutR)
   );
   ProgramCounter my_pc(
      .clk(pc_clk),
      .rst_n(rst_n),
      .ext(pc_ext),
      .inc0_jum1(pc_inc0_jum1),
      .E(pc_en),
      .address(pc_addr)
   );
endmodule
```

4. Instruction Decoder

```verilog
module Instruction_decoder(
   input clk,rst_n,clk_sl,
   input step,
   input [15:0] instruction,
   input alu_N,alu_Z,alu_C,alu_V,
   input [15:0] alu_o,
   input [15:0] rf_B,
   input [15:0] pc_addr,
   output reg alu_add0_sub1,alu_LHI,alu_LLI,alu_ext_imm,
   output reg [15:0] alu_imm_B,
   output reg rf_en,
   output reg [1:0] rf_op,
   output reg [2:0] rf_addr,rf_readA,rf_readB,
   output reg pc_en,
   output reg pc_inc0_jum1,
   output reg [15:0] pc_ext,
   input ext_mem_wen,
   input [7:0] ext_mem_addr,
   input [15:0] ext_mem_data,
   output reg mem_wen,mem_ren,
   output reg [7:0] mem_addr,
   output reg [15:0] mem_data,
   output reg ctro_outR,
   output reg done,
   output reg [15:0] ins
    );
   always@(posedge clk or negedge rst_n)
      if(!rst_n) ins<=0;
      else ins <= instruction;
   reg LHI,LLI,LDR,STR,ADD,ADC,SUB,SBB,CMP,ADDI,
      SUBI,MOV,BCC,BCS,BNE,BEQ,BAL,JMP,JAL,JALR,JR,OutR,HLT;
   reg N,Z,C,V;
   reg [15:0] S;
   always@(posedge clk_sl or negedge rst_n)
      if(!rst_n)begin
         N<=0;
         Z<=0;
         C<=0;
         V<=0;
         S<=0;
      end
      else begin
         N<=alu_N;
         Z<=alu_Z;
         C<=alu_C;
         V<=alu_V;
         S<=alu_o;
      end
   always@(*)begin

      if(SUB|SBB|CMP|SUBI) alu_add0_sub1 = 1'b1;
      else alu_add0_sub1 = 1'b0;

      if(LHI) alu_LHI = 1'b1;
      else alu_LHI = 1'b0;

      if(LLI) alu_LLI = 1'b1;
      else alu_LLI = 1'b0;

      if(LHI|LLI|LDR|STR|ADDI|SUBI|MOV|JALR|JR|OutR) alu_ext_imm = 1'b1;
      else alu_ext_imm = 1'b0;

      if(LHI|LLI) alu_imm_B = {8'h00,ins[7:0]};
      else if(LDR|STR|ADDI|SUBI) alu_imm_B = {11'h000,ins[4:0]};
      else alu_imm_B = 16'h0;

      if(LHI|LLI|LDR|ADD|ADC|SUB|SBB|ADDI|SUBI|MOV|JAL|JALR) rf_en = 1'b1;
      else rf_en = 1'b0;

      if(LDR) rf_op = 2'b01;
      else if(JAL|JALR) rf_op = 2'b10;
      else rf_op = 2'b00;

      rf_addr = ins[10:8];

      if(LHI) rf_readA = ins[10:8];
      else rf_readA = ins[7:5];

      if(STR) rf_readB = ins[10:8];
      else rf_readB = ins[4:2];

      if(step) pc_en = 1'b1;
      else pc_en = 1'b0;

      if(JMP|JAL|JALR) pc_inc0_jum1 = 1'b1;
      else pc_inc0_jum1 = 1'b0;

      if(JALR|JR) pc_ext = alu_o;
      else if(JMP) pc_ext = {pc_addr[15:11],ins[10:0]};
      else if(BCC|BCS|BNE|BEQ|BAL|JAL) begin
         if (ins[7]==1'b1)
            pc_ext = ~{8'h00,~ins[7:0]+1'b1}+1'b1;
         else
            pc_ext = {8'h00,ins[7:0]};
      end
      else pc_ext = 16'h1;

      if((STR&&step!=0)|ext_mem_wen) mem_wen = 1'b1;
      else mem_wen = 1'b0;

      if(LDR&&step!=0) mem_ren = 1'b1;
      else mem_ren = 1'b0;

      if(ext_mem_wen) mem_addr = ext_mem_addr;
      else mem_addr = alu_o;

      if(ext_mem_wen) mem_data = ext_mem_data;
      else mem_data = rf_B;

      if(OutR) ctro_outR = 1'b1;
      else ctro_outR = 1'b0;

      if(HLT) done = 1'b1;
      else done = 1'b0;

   end




   always@(ins,N,Z,C,V) begin
      //ins[15]==0
      if(ins[15:11]==5'b00001) LHI = 1'b1;
      else LHI = 1'b0;
      if(ins[15:11]==5'b00010) LLI = 1'b1;
      else LLI = 1'b0;
      if(ins[15:11]==5'b00011) LDR = 1'b1;
      else LDR = 1'b0;
      if(ins[15:11]==5'b00101) STR = 1'b1;
      else STR = 1'b0;
      if(ins[15:11]==5'b00000&&ins[1:0]==2'b00) ADD = 1'b1;
      else ADD = 1'b0;
      if(ins[15:11]==5'b00000&&ins[1:0]==2'b01) ADC = 1'b1;
      else ADC = 1'b0;
      if(ins[15:11]==5'b00000&&ins[1:0]==2'b10) SUB = 1'b1;
      else SUB = 1'b0;
      if(ins[15:11]==5'b00000&&ins[1:0]==2'b11) SBB = 1'b1;
      else SBB = 1'b0;
      if(ins[15:11]==5'b00110&&ins[1:0]==2'b01) CMP = 1'b1;
      else CMP = 1'b0;
      if(ins[15:11]==5'b00111) ADDI = 1'b1;
      else ADDI = 1'b0;
      if(ins[15:11]==5'b01000) SUBI = 1'b1;
      else SUBI = 1'b0;
      if(ins[15:11]==5'b01011) MOV = 1'b1;
      else MOV = 1'b0;
      //ins[15:13]==3'b100
      if(ins[15:11]==5'b10000) JMP = 1'b1;
      else JMP = 1'b0;
      if(ins[15:11]==5'b10001) JAL = 1'b1;
      else JAL = 1'b0;
      if(ins[15:11]==5'b10010) JALR = 1'b1;
      else JALR = 1'b0;
      if(ins[15:11]==5'b10011) JR = 1'b1;
      else JR = 1'b0;
      //ins[15:13]==3'b111
      if(ins[15:11]==5'b11100&&ins[1:0]==2'b00) OutR = 1'b1;
      else OutR = 1'b0;
      if(ins[15:11]==5'b11100&&ins[1:0]==2'b01) HLT = 1'b1;
      else HLT = 1'b0;
      //ins[15:8]==8'hCx
      if(ins[15:8]==8'hC3&&~C) BCC = 1'b1;
      else BCC = 1'b0;
      if(ins[15:8]==8'hC2&&C) BCS = 1'b1;
      else BCS = 1'b0;
      if(ins[15:8]==8'hC1&&~Z) BNE = 1'b1;
      else BNE = 1'b0;
      if(ins[15:8]==8'hC0&&Z) BEQ = 1'b1;
      else BEQ = 1'b0;
      if(ins[15:8]==8'hCE) BAL = 1'b1;
      else BAL = 1'b0;

   end

endmodule
```

5. Timing Generator

```
module Timer_generator(
   input clk,rst_n,E,
   output reg step,
   output reg clk_s0,clk_s1,
   output [1:0] div_count
    );

   parameter div0 = 0,
             div1 = 1,
             div2 = 2,
             div3 = 3;

   reg [1:0] div_count;
   always@(posedge clk or negedge rst_n)
      if(!rst_n)
         div_count <= div0;
      else if(div_count==div3)
         div_count <= div0;
      else if(E)
         div_count <= div_count+1;

   always@(div_count)
      if(div_count==div3)clk_s1=1;
      else clk_s1=0;

   always@(div_count)
      if(div_count==div1)clk_s0=1;
      else clk_s0=0;

   always@(div_count)
      if(div_count==div0||div_count==div1)
         step=0;
      else
         step=1;
endmodule
```

6. Program Counter

```
module ProgramCounter(
   input clk,rst_n,
   input [15:0] ext,
   input inc0_juml,
   input E,
   output reg [15:0] address
   );

   always@(posedge clk or negedge rst_n) begin
      if(!rst_n) address<=0;
      else if(E) begin
         if(inc0_juml) address <= ext;
         else address <= address+ext;
      end
   end

endmodule
```

7. 256x16 Memory

```
module memory_256x16(
   input clk,write_en,
   input [7:0] address,
   input [15:0] data_in,
   output [15:0] data_out
    );

   reg [15:0] memory [0:255];
   assign data_out = memory[address];

   always@(posedge clk)
      if(write_en) memory[address] <= data_in;


endmodule
```

8. Register File + ALU

```verilog
module RF_ALU_16bits(
   input clk,rst_n,RF_en,
   input [2:0] RF_addr,read_A,read_B,
   input [15:0] RF_data,
   input add0_sub1,
   input ext_imm,
   input [15:0] ext_B_data,
   input LHI,LLI,
   output [15:0] S,
   output N,Z,C,V,

   output [15:0] rA,rB,

   input ctro_outR,
   output reg [15:0] OutR
    );

wire [15:0] alu_Bin;
wire [15:0] alu_S;
assign alu_Bin = (ext_imm)? ext_B_data:rB;
wire [15:0] logicm_S; // logicm ALU
assign S = (LHI|LLI)? logicm_S:alu_S;
assign logicm_S = (LHI)? {ext_B_data[7:0],rA[7:0]}:{8'h00,ext_B_data[7:0]};

RegisterFile_16bits mu_rf(
   .clk(clk),
   .rst_n(rst_n),
   .write_en(RF_en),
   .read_A(read_A),
   .read_B(read_B),
   .write_addr(RF_addr),
   .write_data(RF_data),
   .out_A(rA),
   .out_B(rB)
    );
ALU_16bits my_alu(
   .A(rA),
   .B(alu_Bin),
   .add0_sub1(add0_sub1),
   .S(alu_S),
   .N(N),.Z(Z),.C(C),.V(V)
    );
   always@(posedge clk or negedge rst_n)
      if(!rst_n)  OutR<=16'h0000;
      else if(ctro_outR) OutR<=rA;
endmodule
```

9. Eight Register File

```
module RegisterFile_16bits(
   input clk,rst_n,write_en,
   input [2:0] read_A,read_B,write_addr,
   input [15:0] write_data,
   output [15:0] out_A,out_B
    );

   wire [7:0] rf_en;
   wire [15:0] Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7;
   decoder_NtoM #(.N(3)) decoder1(
      .S(write_addr),
      .E(write_en),
      .O(rf_en)
      );

   multiplexer_8to1_16bits mult1(
      .S0(read_A[0]),.S1(read_A[1]),.S2(read_A[2]),
      .A0(Q0),.A1(Q1),.A2(Q2),.A3(Q3),.A4(Q4),.A5(Q5),.A6(Q6),.A7(Q7),
      .O(out_A)
      );

   multiplexer_8to1_16bits mult2(
      .S0(read_B[0]),.S1(read_B[1]),.S2(read_B[2]),
      .A0(Q0),.A1(Q1),.A2(Q2),.A3(Q3),.A4(Q4),.A5(Q5),.A6(Q6),.A7(Q7),
      .O(out_B)
      );

   DFF_16bits dff1(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[0]),
      .Din(write_data),
      .Qout(Q0)
      );
   DFF_16bits dff2(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[1]),
      .Din(write_data),
      .Qout(Q1)
      );
   DFF_16bits dff3(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[2]),
      .Din(write_data),
      .Qout(Q2)
      );
   DFF_16bits dff4(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[3]),
      .Din(write_data),
      .Qout(Q3)
      );
   DFF_16bits dff5(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[4]),
      .Din(write_data),
      .Qout(Q4)
      );
   DFF_16bits dff6(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[5]),
      .Din(write_data),
      .Qout(Q5)
      );
   DFF_16bits dff7(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[6]),
      .Din(write_data),
      .Qout(Q6)
      );
   DFF_16bits dff8(
      .clk(clk),.rst_n(rst_n),
      .E(rf_en[7]),
      .Din(write_data),
      .Qout(Q7)
      );


endmodule
```

10. ALU

```
module ALU_16bits(
   input [15:0] A,B,
   input add0_sub1,
   output [15:0] S,
   output N,Z,C,V
    );

   wire [15:0] C_buf;
   wire [15:0] B_comp;
   assign B_comp = (add0_sub1)? B^16'hffff:B;
   assign N = S[15];
   assign Z = ~(|S);
   assign C = C_buf[15];
   assign V = ((add0_sub1^B[15])~^A[15])&(S[15]^C);
   full_adder fadd0(
      .A(A[0]),
      .B(B_comp[0]),
      .C_in(add0_sub1),
      .S(S[0]),
      .C_out(C_buf[0])
      );
   full_adder fadd1(
      .A(A[1]),
      .B(B_comp[1]),
      .C_in(C_buf[0]),
      .S(S[1]),
      .C_out(C_buf[1])
      );
   full_adder fadd2(
      .A(A[2]),
      .B(B_comp[2]),
      .C_in(C_buf[1]),
      .S(S[2]),
      .C_out(C_buf[2])
      );
   full_adder fadd3(
      .A(A[3]),
      .B(B_comp[3]),
      .C_in(C_buf[2]),
      .S(S[3]),
      .C_out(C_buf[3])
      );
   full_adder fadd4(
      .A(A[4]),
      .B(B_comp[4]),
      .C_in(C_buf[3]),
      .S(S[4]),
      .C_out(C_buf[4])
      );
   full_adder fadd5(
      .A(A[5]),
      .B(B_comp[5]),
      .C_in(C_buf[4]),
      .S(S[5]),
      .C_out(C_buf[5])
      );
   full_adder fadd6(
      .A(A[6]),
      .B(B_comp[6]),
      .C_in(C_buf[5]),
      .S(S[6]),
        .C_out(C_buf[6])
        );
    full_adder fadd7(
        .A(A[7]),
        .B(B_comp[7]),
        .C_in(C_buf[6]),
        .S(S[7]),
        .C_out(C_buf[7])
        );
    full_adder fadd8(
        .A(A[8]),
        .B(B_comp[8]),
        .C_in(C_buf[7]),
        .S(S[8]),
        .C_out(C_buf[8])
        );
    full_adder fadd9(
        .A(A[9]),
        .B(B_comp[9]),
        .C_in(C_buf[8]),
        .S(S[9]),
        .C_out(C_buf[9])
        );
    full_adder fadd10(
        .A(A[10]),
        .B(B_comp[10]),
        .C_in(C_buf[9]),
        .S(S[10]),
        .C_out(C_buf[10])
        );
    full_adder fadd11(
        .A(A[11]),
        .B(B_comp[11]),
        .C_in(C_buf[10]),
        .S(S[11]),
        .C_out(C_buf[11])
        );
    full_adder fadd12(
        .A(A[12]),
        .B(B_comp[12]),
        .C_in(C_buf[11]),
        .S(S[12]),
        .C_out(C_buf[12])
        );
    full_adder fadd13(
        .A(A[13]),
        .B(B_comp[13]),
        .C_in(C_buf[12]),
        .S(S[13]),
        .C_out(C_buf[13])
        );
    full_adder fadd14(
        .A(A[14]),
        .B(B_comp[14]),
        .C_in(C_buf[13]),
        .S(S[14]),
        .C_out(C_buf[14])
        );
    full_adder fadd15(
        .A(A[15]),
        .B(B_comp[15]),
        .C_in(C_buf[14]),
        .S(S[15]),
        .C_out(C_buf[15])
        );


endmodule

```

11. full Adder

```verilog
module full_adder
// #(parameter N=16)(
// input [N-1:0] A,B,
// input C_in,
// output reg [N-1:0] S,
// output reg C_out
//    );
//
// genvar i;
// wire [N-2:0] C_buf;
// assign {C_out,S[N-1]}=A[N-1]+B[N-1]+C_buf[N-2];
// assign {C_buf[0],S[0]}=A[0]+B[0]+C_in;
// generate for(i=1;i<N-1;i=i+1) begin: gener_fadder
//    assign {C_buf[i],S[i]}=A[i]+B[i]+C_buf[i-1];
// end
// endgenerate
   (
   input A,B,
   input C_in,
   output reg S,
   output reg C_out
    );

   always@(A,B,C_in)
      {C_out,S}=A+B+C_in;

endmodule
```

12. Decoder

```verilog
module decoder_NtoM
    #(parameter N=3)(
    input [N-1:0] S,
    input E,
    output reg [2**N-1:0] O
    );

   integer i;
   always@(S,E)
   begin
      for(i=0;i<2**N;i=i+1)begin
         if(S==i&&E==1'b1) O[i]=1'b1;
         else O[i]=1'b0;
      end
   end

endmodule
```

13. Multiplexer 8to1 16bits

```
module multiplexer_8to1_16bits(
    input S2,S1,S0,
    input [15:0] A7,A6,A5,A4,A3,A2,A1,A0,
    output reg [15:0] O
     );
     always@(*)begin
       case({S2,S1,S0})
       0:O=A0;
       1:O=A1;
       2:O=A2;
       3:O=A3;
       4:O=A4;
       5:O=A5;
       6:O=A6;
       7:O=A7;
       endcase
     end
endmodule
```

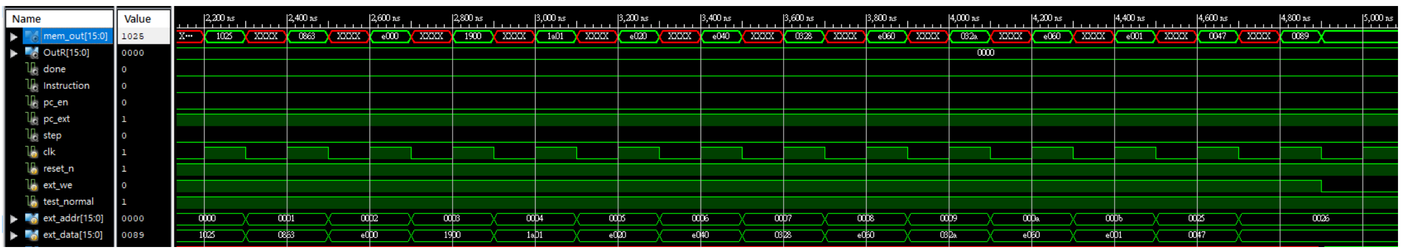
14. D-Flip Flop 16bits

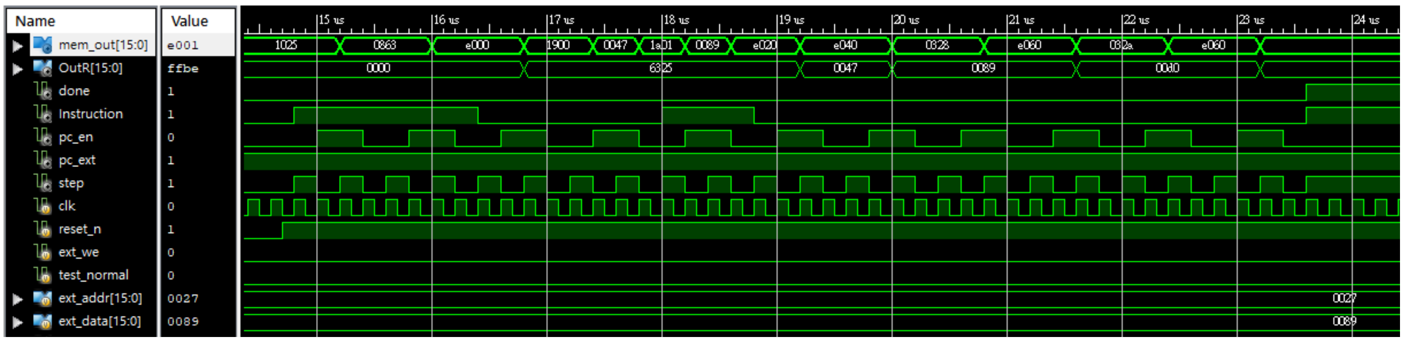
```
module DFF_16bits(
    input clk,E,rst_n,
    input [15:0] Din,
    output reg [15:0] Qout
     );
    always@(posedge clk or negedge rst_n)
    begin
       if(!rst_n)
          Qout<=16'h0000;
       else if(E)
          Qout<=Din;
    end

endmodule
```

Verification

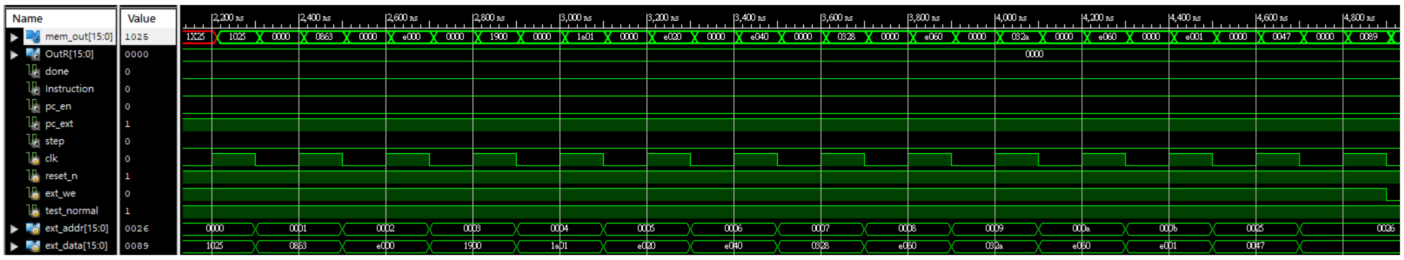
1. CPU

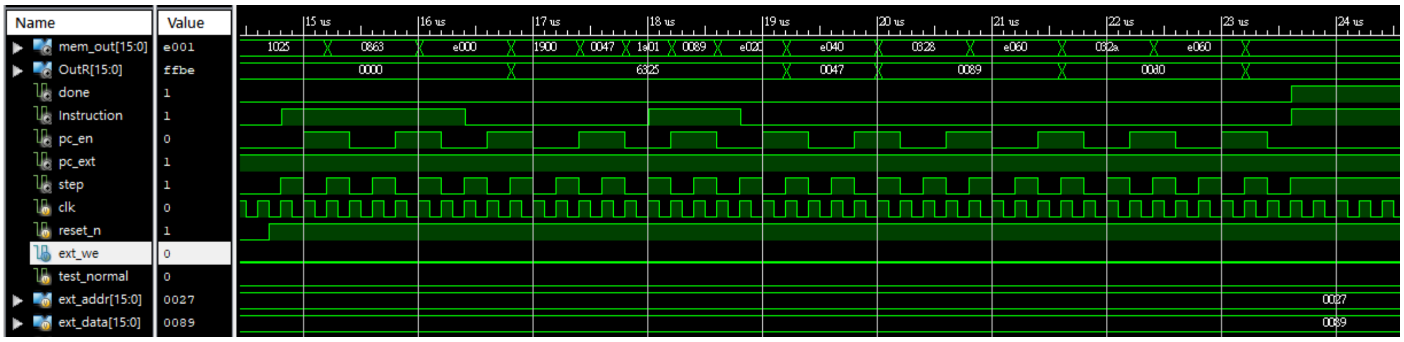
behavioral



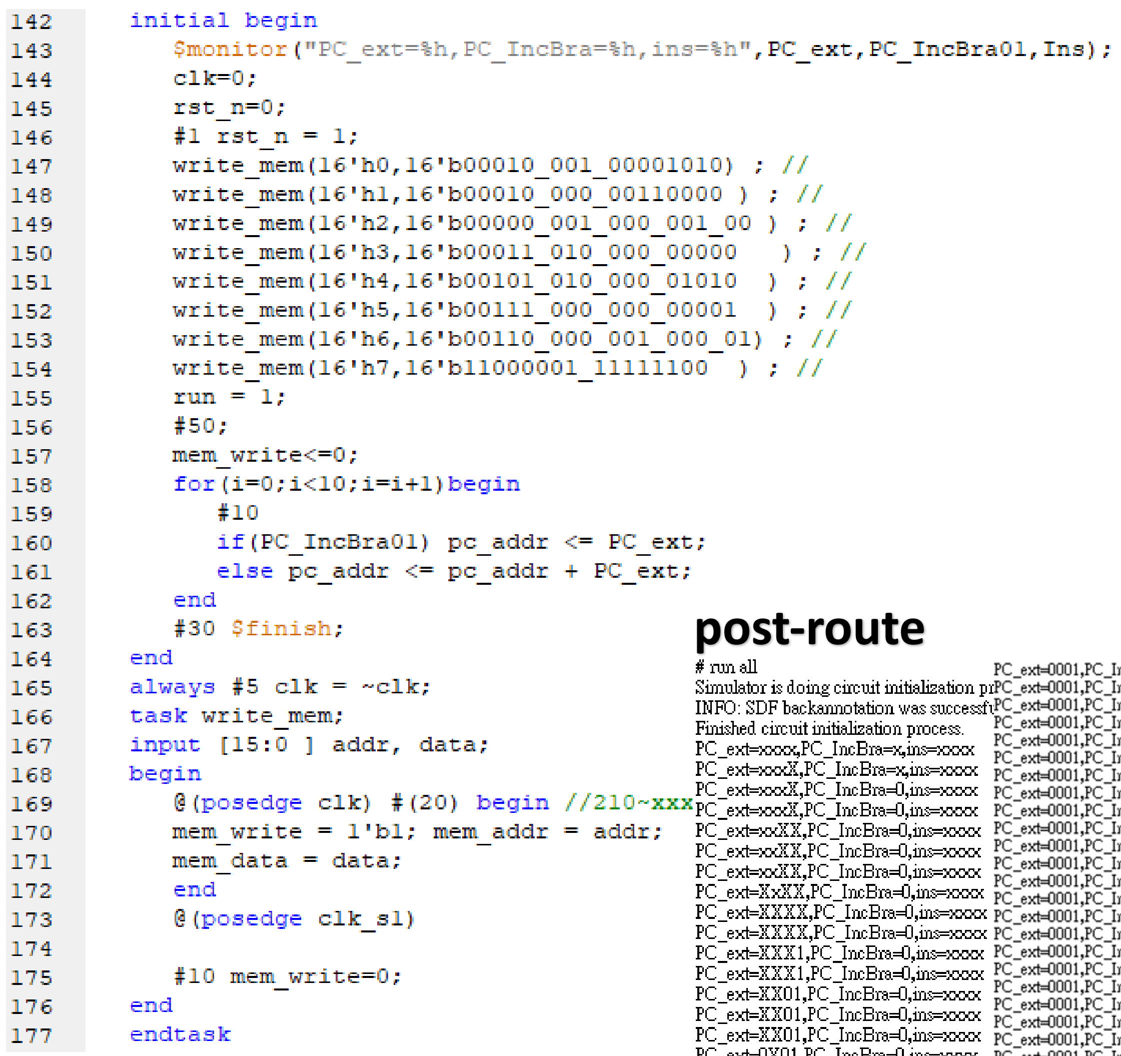


post-route



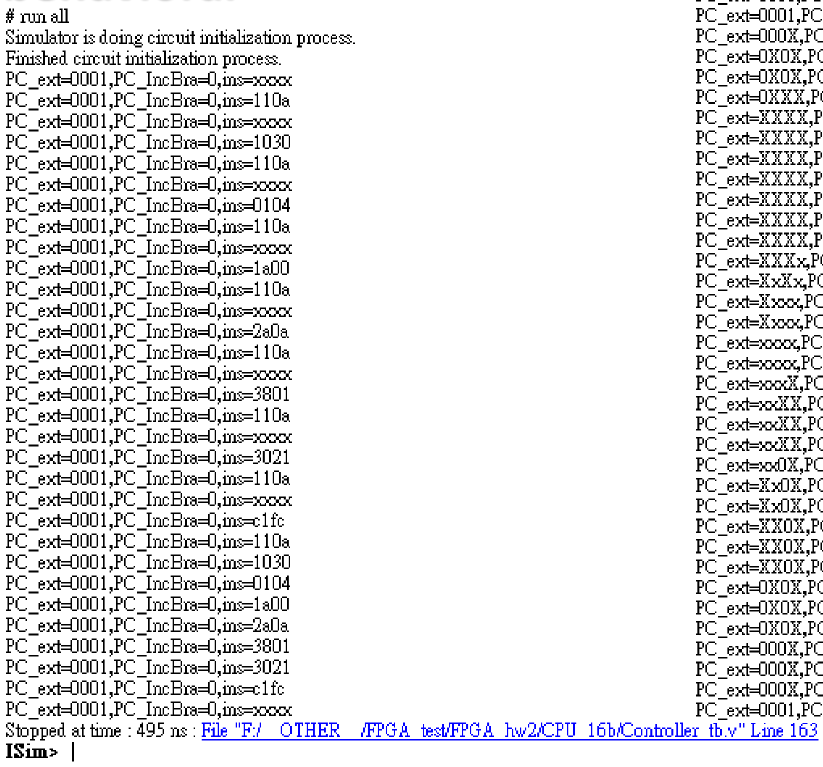


2. Controller



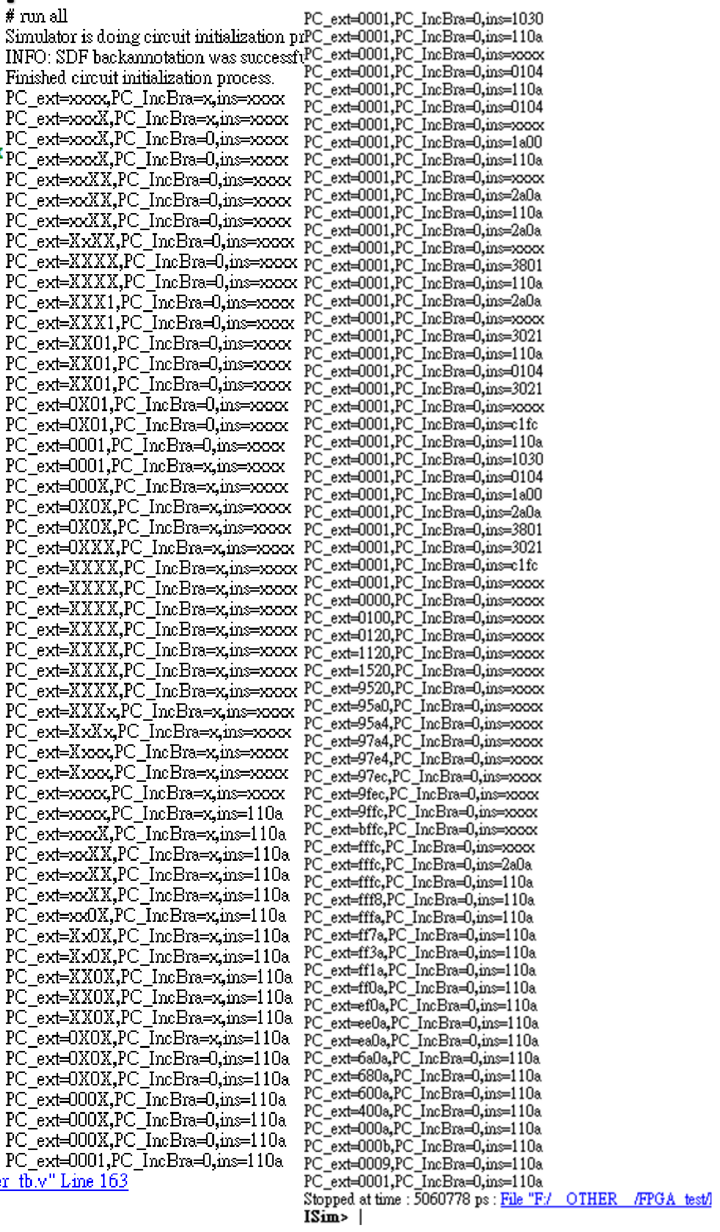
```
142    initial begin
143       $monitor("PC_ext=%h,PC_IncBra=%h,ins=%h",PC_ext,PC_IncBra01,Ins);
144       clk=0;
145       rst_n=0;
146       #1 rst_n = 1;
147       write_mem(16'h0,16'b00010_001_00001010) ; //
148       write_mem(16'h1,16'b00010_000_00110000 ) ; //
149       write_mem(16'h2,16'b00000_001_000_001_00 ) ; //
150       write_mem(16'h3,16'b00011_010_000_00000   ) ; //
151       write_mem(16'h4,16'b00101_010_000_01010  ) ; //
152       write_mem(16'h5,16'b00111_000_000_00001  ) ; //
153       write_mem(16'h6,16'b00110_000_001_000_01) ; //
154       write_mem(16'h7,16'b11000001_11111100  ) ; //
155       run = 1;
156       #50;
157       mem_write<=0;
158       for(i=0;i<10;i=i+1)begin
159          #10
160          if(PC_IncBra01) pc_addr <= PC_ext;
161          else pc_addr <= pc_addr + PC_ext;
162       end
163       #30 $finish;
164    end
165    always #5 clk = ~clk;
166    task write_mem;
167    input [15:0 ] addr, data;
168    begin
169       @(posedge clk) #(20) begin //210~xxx
170       mem_write = 1'b1; mem_addr = addr;
171       mem_data = data;
172       end
173       @(posedge clk_s1)
174
175       #10 mem_write=0;
176    end
177    endtask
```

behavioral



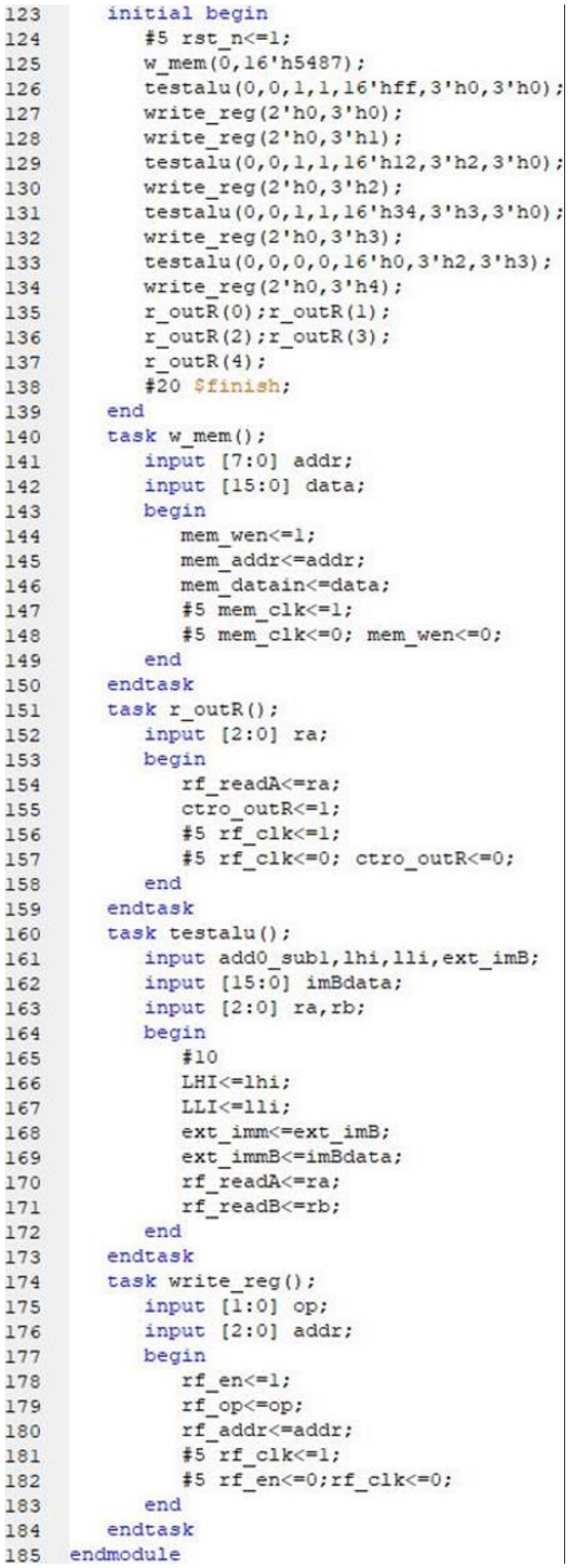
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=1030
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=1a00
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=3801
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=3021
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=c1fc
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=1030
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=1a00
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=3801
PC_ext=0001,PC_IncBra=0,ins=3021
PC_ext=0001,PC_IncBra=0,ins=c1fc
PC_ext=0001,PC_IncBra=0,ins=xxxx
Stopped at time : 495 ns : File "F:/   OTHER   /FPGA_test/FPGA_hw2/CPU_16b/Controller_tb.v" Line 163
ISim> |
```

post-route



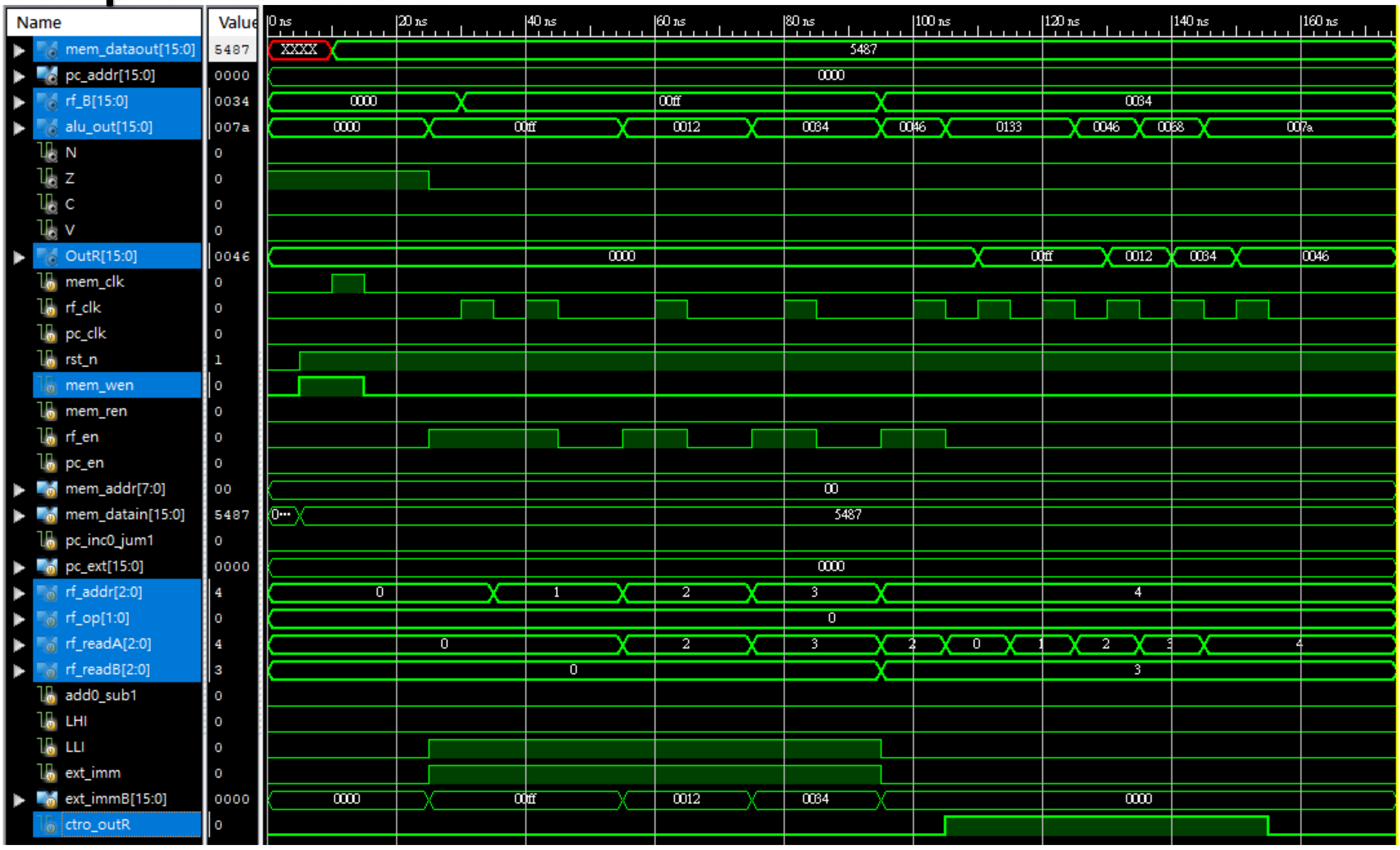
```
# run all
Simulator is doing circuit initialization pr
INFO: SDF backannotation was successfu
Finished circuit initialization process.
PC_ext=xxxx,PC_IncBra=x,ins=xxxx
PC_ext=xxxX,PC_IncBra=x,ins=xxxx
PC_ext=xxxX,PC_IncBra=0,ins=xxxx
PC_ext=xxxX,PC_IncBra=0,ins=xxxx
PC_ext=xxXX,PC_IncBra=0,ins=xxxx
PC_ext=xxXX,PC_IncBra=0,ins=xxxx
PC_ext=xxXX,PC_IncBra=0,ins=xxxx
PC_ext=XxXX,PC_IncBra=0,ins=xxxx
PC_ext=XXXX,PC_IncBra=0,ins=xxxx
PC_ext=XXXX,PC_IncBra=0,ins=xxxx
PC_ext=XXX1,PC_IncBra=0,ins=xxxx
PC_ext=XXX1,PC_IncBra=0,ins=xxxx
PC_ext=XX01,PC_IncBra=0,ins=xxxx
PC_ext=XX01,PC_IncBra=0,ins=xxxx
PC_ext=XX01,PC_IncBra=0,ins=xxxx
PC_ext=0X01,PC_IncBra=0,ins=xxxx
PC_ext=0X01,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=x,ins=xxxx
PC_ext=000X,PC_IncBra=x,ins=xxxx
PC_ext=0X0X,PC_IncBra=x,ins=xxxx
PC_ext=0X0X,PC_IncBra=x,ins=xxxx
PC_ext=0XXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXX,PC_IncBra=x,ins=xxxx
PC_ext=XXXx,PC_IncBra=x,ins=xxxx
PC_ext=XxXx,PC_IncBra=x,ins=xxxx
PC_ext=Xxxx,PC_IncBra=x,ins=xxxx
PC_ext=Xxxx,PC_IncBra=x,ins=xxxx
PC_ext=xxxx,PC_IncBra=x,ins=xxxx
PC_ext=xxxx,PC_IncBra=x,ins=110a
PC_ext=xxxX,PC_IncBra=x,ins=110a
PC_ext=xxXX,PC_IncBra=x,ins=110a
PC_ext=xxXX,PC_IncBra=x,ins=110a
PC_ext=xxXX,PC_IncBra=x,ins=110a
PC_ext=xx0X,PC_IncBra=x,ins=110a
PC_ext=Xx0X,PC_IncBra=x,ins=110a
PC_ext=Xx0X,PC_IncBra=x,ins=110a
PC_ext=XX0X,PC_IncBra=x,ins=110a
PC_ext=XX0X,PC_IncBra=x,ins=110a
PC_ext=XX0X,PC_IncBra=x,ins=110a
PC_ext=0X0X,PC_IncBra=x,ins=110a
PC_ext=0X0X,PC_IncBra=0,ins=110a
PC_ext=0X0X,PC_IncBra=0,ins=110a
PC_ext=000X,PC_IncBra=0,ins=110a
PC_ext=000X,PC_IncBra=0,ins=110a
PC_ext=000X,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=1030
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=1a00
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=3801
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=3021
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=3021
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0001,PC_IncBra=0,ins=c1fc
PC_ext=0001,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=1030
PC_ext=0001,PC_IncBra=0,ins=0104
PC_ext=0001,PC_IncBra=0,ins=1a00
PC_ext=0001,PC_IncBra=0,ins=2a0a
PC_ext=0001,PC_IncBra=0,ins=3801
PC_ext=0001,PC_IncBra=0,ins=3021
PC_ext=0001,PC_IncBra=0,ins=c1fc
PC_ext=0001,PC_IncBra=0,ins=xxxx
PC_ext=0000,PC_IncBra=0,ins=xxxx
PC_ext=0100,PC_IncBra=0,ins=xxxx
PC_ext=0120,PC_IncBra=0,ins=xxxx
PC_ext=1120,PC_IncBra=0,ins=xxxx
PC_ext=1520,PC_IncBra=0,ins=xxxx
PC_ext=9520,PC_IncBra=0,ins=xxxx
PC_ext=95a0,PC_IncBra=0,ins=xxxx
PC_ext=95a4,PC_IncBra=0,ins=xxxx
PC_ext=97a4,PC_IncBra=0,ins=xxxx
PC_ext=97e4,PC_IncBra=0,ins=xxxx
PC_ext=97ec,PC_IncBra=0,ins=xxxx
PC_ext=9fec,PC_IncBra=0,ins=xxxx
PC_ext=9ffc,PC_IncBra=0,ins=xxxx
PC_ext=bffc,PC_IncBra=0,ins=xxxx
PC_ext=fffc,PC_IncBra=0,ins=xxxx
PC_ext=fffc,PC_IncBra=0,ins=2a0a
PC_ext=fffc,PC_IncBra=0,ins=110a
PC_ext=fff8,PC_IncBra=0,ins=110a
PC_ext=fffa,PC_IncBra=0,ins=110a
PC_ext=ff7a,PC_IncBra=0,ins=110a
PC_ext=ff3a,PC_IncBra=0,ins=110a
PC_ext=ff1a,PC_IncBra=0,ins=110a
PC_ext=ff0a,PC_IncBra=0,ins=110a
PC_ext=ef0a,PC_IncBra=0,ins=110a
PC_ext=ee0a,PC_IncBra=0,ins=110a
PC_ext=ea0a,PC_IncBra=0,ins=110a
PC_ext=6a0a,PC_IncBra=0,ins=110a
PC_ext=680a,PC_IncBra=0,ins=110a
PC_ext=600a,PC_IncBra=0,ins=110a
PC_ext=400a,PC_IncBra=0,ins=110a
PC_ext=000a,PC_IncBra=0,ins=110a
PC_ext=000b,PC_IncBra=0,ins=110a
PC_ext=0009,PC_IncBra=0,ins=110a
PC_ext=0001,PC_IncBra=0,ins=110a
Stopped at time : 5060778 ps : File "F:/   OTHER   /FPGA_test/
ISim> |
```

3. Datapath

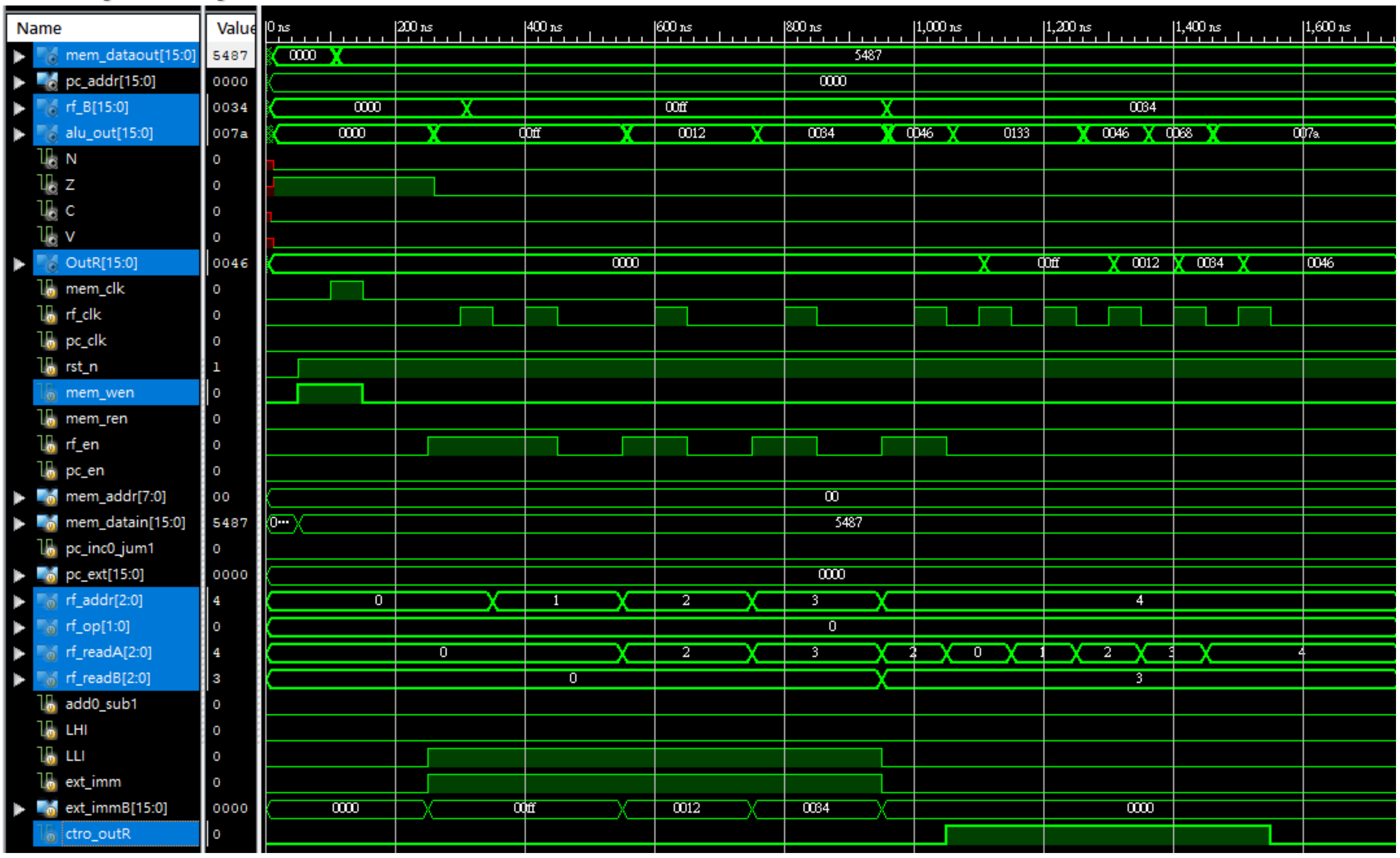


```verilog
   initial begin
      #5 rst_n<=1;
      w_mem(0,16'h5487);
      testalu(0,0,1,1,16'hff,3'h0,3'h0);
      write_reg(2'h0,3'h0);
      write_reg(2'h0,3'h1);
      testalu(0,0,1,1,16'h12,3'h2,3'h0);
      write_reg(2'h0,3'h2);
      testalu(0,0,1,1,16'h34,3'h3,3'h0);
      write_reg(2'h0,3'h3);
      testalu(0,0,0,0,16'h0,3'h2,3'h3);
      write_reg(2'h0,3'h4);
      r_outR(0);r_outR(1);
      r_outR(2);r_outR(3);
      r_outR(4);
      #20 $finish;
   end
   task w_mem();
      input [7:0] addr;
      input [15:0] data;
      begin
         mem_wen<=1;
         mem_addr<=addr;
         mem_datain<=data;
         #5 mem_clk<=1;
         #5 mem_clk<=0; mem_wen<=0;
      end
   endtask
   task r_outR();
      input [2:0] ra;
      begin
         rf_readA<=ra;
         ctro_outR<=1;
         #5 rf_clk<=1;
         #5 rf_clk<=0; ctro_outR<=0;
      end
   endtask
   task testalu();
      input add0_sub1,lhi,lli,ext_imB;
      input [15:0] imBdata;
      input [2:0] ra,rb;
      begin
         #10
         LHI<=lhi;
         LLI<=lli;
         ext_imm<=ext_imB;
         ext_immB<=imBdata;
         rf_readA<=ra;
         rf_readB<=rb;
      end
   endtask
   task write_reg();
      input [1:0] op;
      input [2:0] addr;
      begin
         rf_en<=1;
         rf_op<=op;
         rf_addr<=addr;
         #5 rf_clk<=1;
         #5 rf_en<=0;rf_clk<=0;
      end
   endtask
endmodule
```

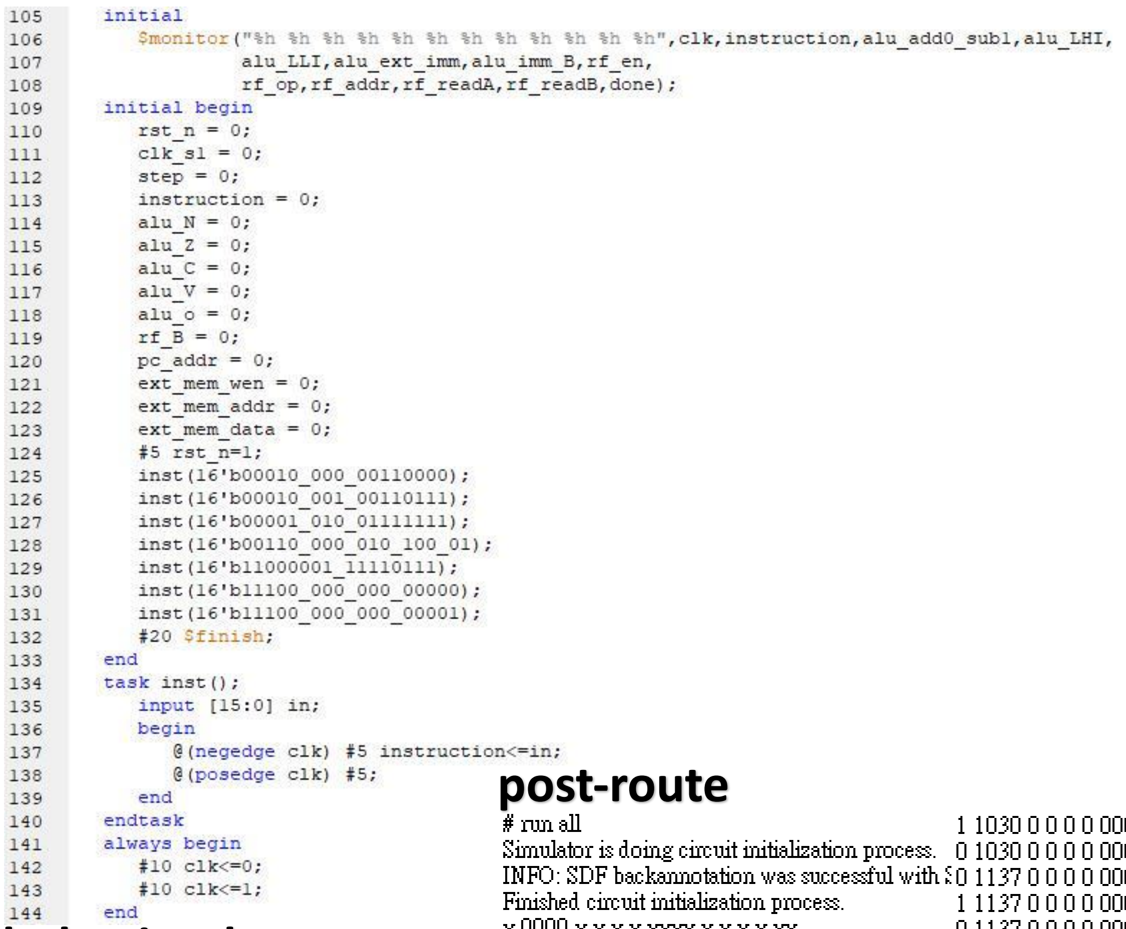
Datapath behavioral



Datapath post-route

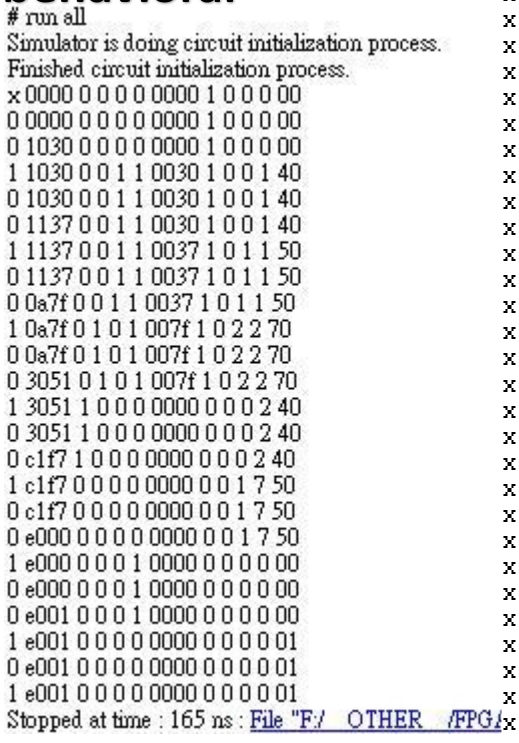


4. Instruction decoder



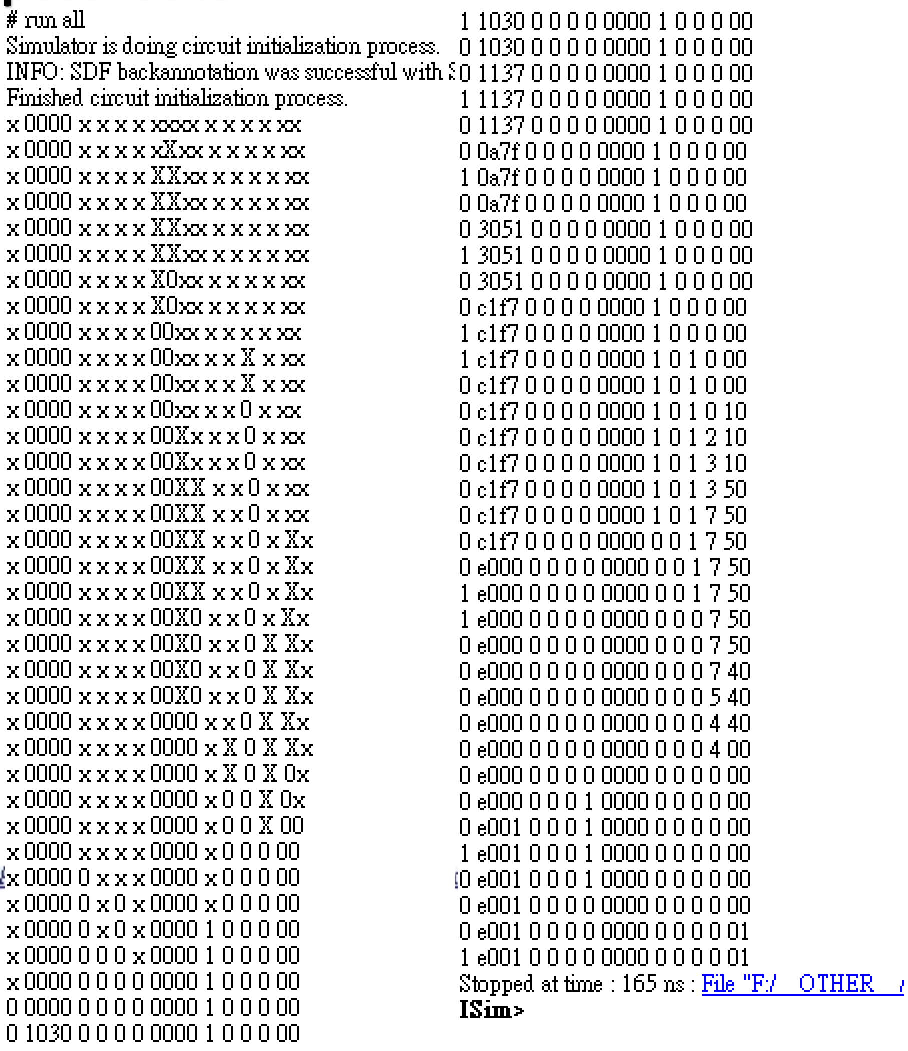
```
initial
   $monitor("%h %h %h %h %h %h %h %h %h %h %h %h",clk,instruction,alu_add0_sub1,alu_LHI,
            alu_LLI,alu_ext_imm,alu_imm_B,rf_en,
            rf_op,rf_addr,rf_readA,rf_readB,done);
initial begin
   rst_n = 0;
   clk_s1 = 0;
   step = 0;
   instruction = 0;
   alu_N = 0;
   alu_Z = 0;
   alu_C = 0;
   alu_V = 0;
   alu_o = 0;
   rf_B = 0;
   pc_addr = 0;
   ext_mem_wen = 0;
   ext_mem_addr = 0;
   ext_mem_data = 0;
   #5 rst_n=1;
   inst(16'b00010_000_00110000);
   inst(16'b00010_001_00110111);
   inst(16'b00001_010_01111111);
   inst(16'b00110_000_010_100_01);
   inst(16'b11000001_11110111);
   inst(16'b11100_000_000_00000);
   inst(16'b11100_000_000_00001);
   #20 $finish;
end
task inst();
   input [15:0] in;
   begin
      @(negedge clk) #5 instruction<=in;
      @(posedge clk) #5;
   end
endtask
always begin
   #10 clk<=0;
   #10 clk<=1;
end
```

behavioral



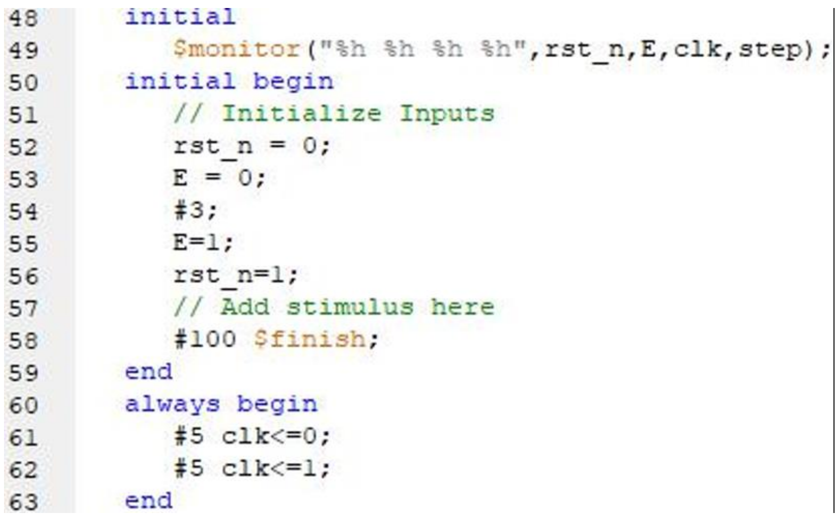
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
x 0000 0 0 0 0 0000 1 0 0 0 00
0 0000 0 0 0 0 0000 1 0 0 0 00
0 1030 0 0 0 0 0000 1 0 0 0 00
1 1030 0 0 1 1 0030 1 0 0 1 40
0 1030 0 0 1 1 0030 1 0 0 1 40
0 1137 0 0 1 1 0030 1 0 0 1 40
1 1137 0 0 1 1 0037 1 0 1 1 50
0 1137 0 0 1 1 0037 1 0 1 1 50
0 0a7f 0 0 1 1 0037 1 0 1 1 50
1 0a7f 0 1 0 1 007f 1 0 2 2 70
0 0a7f 0 1 0 1 007f 1 0 2 2 70
0 3051 0 1 0 1 007f 1 0 2 2 70
1 3051 1 0 0 0 0000 0 0 0 2 40
0 3051 1 0 0 0 0000 0 0 0 2 40
0 c1f7 1 0 0 0 0000 0 0 0 2 40
1 c1f7 0 0 0 0 0000 0 0 1 7 50
0 c1f7 0 0 0 0 0000 0 0 1 7 50
0 e000 0 0 0 0 0000 0 0 1 7 50
1 e000 0 0 0 1 0000 0 0 0 0 00
0 e000 0 0 0 1 0000 0 0 0 0 00
0 e001 0 0 0 1 0000 0 0 0 0 00
1 e001 0 0 0 0 0000 0 0 0 0 01
0 e001 0 0 0 0 0000 0 0 0 0 01
1 e001 0 0 0 0 0000 0 0 0 0 01
Stopped at time : 165 ns : File "F:/   OTHER   /FPGA
```

post-route



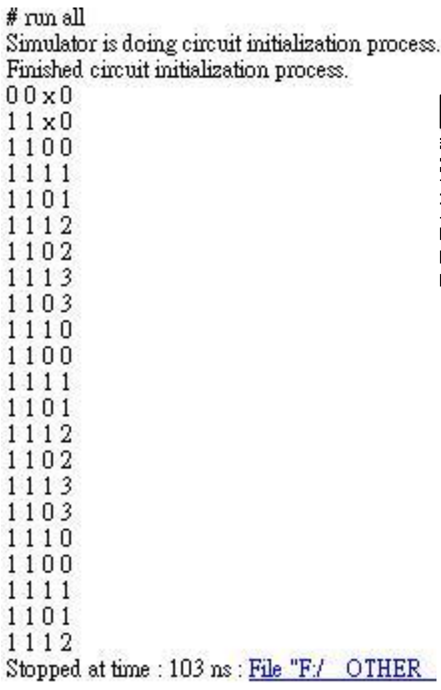
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with S
Finished circuit initialization process.
x 0000 x x x x xxxx x x x x xx
x 0000 x x x x xXxx x x x x xx
x 0000 x x x x XXxx x x x x xx
x 0000 x x x x XXxx x x x x xx
x 0000 x x x x XXxx x x x x xx
x 0000 x x x x XXxx x x x x xx
x 0000 x x x x X0xx x x x x xx
x 0000 x x x x X0xx x x x x xx
x 0000 x x x x 00xx x x x x xx
x 0000 x x x x 00xx x x X x xx
x 0000 x x x x 00xx x x X x xx
x 0000 x x x x 00xx x x 0 x xx
x 0000 x x x x 00Xx x x 0 x xx
x 0000 x x x x 00Xx x x 0 x xx
x 0000 x x x x 00XX x x 0 x xx
x 0000 x x x x 00XX x x 0 x xx
x 0000 x x x x 00XX x x 0 x Xx
x 0000 x x x x 00XX x x 0 x Xx
x 0000 x x x x 00XX x x 0 x Xx
x 0000 x x x x 00X0 x x 0 x Xx
x 0000 x x x x 00X0 x x 0 X Xx
x 0000 x x x x 00X0 x x 0 X Xx
x 0000 x x x x 00X0 x x 0 X Xx
x 0000 x x x x 0000 x x 0 X Xx
x 0000 x x x x 0000 x X 0 X Xx
x 0000 x x x x 0000 x X 0 X 0x
x 0000 x x x x 0000 x 0 0 X 0x
x 0000 x x x x 0000 x 0 0 X 00
x 0000 x x x x 0000 x 0 0 0 00
x 0000 0 x x x 0000 x 0 0 0 00
x 0000 0 x 0 x 0000 x 0 0 0 00
x 0000 0 x 0 x 0000 1 0 0 0 00
x 0000 0 0 0 x 0000 1 0 0 0 00
x 0000 0 0 0 0 0000 1 0 0 0 00
0 0000 0 0 0 0 0000 1 0 0 0 00
0 1030 0 0 0 0 0000 1 0 0 0 00
1 1030 0 0 0 0 0000 1 0 0 0 00
0 1030 0 0 0 0 0000 1 0 0 0 00
0 1137 0 0 0 0 0000 1 0 0 0 00
1 1137 0 0 0 0 0000 1 0 0 0 00
0 1137 0 0 0 0 0000 1 0 0 0 00
0 0a7f 0 0 0 0 0000 1 0 0 0 00
1 0a7f 0 0 0 0 0000 1 0 0 0 00
0 0a7f 0 0 0 0 0000 1 0 0 0 00
0 3051 0 0 0 0 0000 1 0 0 0 00
1 3051 0 0 0 0 0000 1 0 0 0 00
0 3051 0 0 0 0 0000 1 0 0 0 00
0 c1f7 0 0 0 0 0000 1 0 0 0 00
1 c1f7 0 0 0 0 0000 1 0 0 0 00
1 c1f7 0 0 0 0 0000 1 0 1 0 00
0 c1f7 0 0 0 0 0000 1 0 1 0 00
0 c1f7 0 0 0 0 0000 1 0 1 0 10
0 c1f7 0 0 0 0 0000 1 0 1 2 10
0 c1f7 0 0 0 0 0000 1 0 1 3 10
0 c1f7 0 0 0 0 0000 1 0 1 3 50
0 c1f7 0 0 0 0 0000 1 0 1 7 50
0 c1f7 0 0 0 0 0000 0 0 1 7 50
0 e000 0 0 0 0 0000 0 0 1 7 50
1 e000 0 0 0 0 0000 0 0 1 7 50
1 e000 0 0 0 0 0000 0 0 0 7 50
0 e000 0 0 0 0 0000 0 0 0 7 50
0 e000 0 0 0 0 0000 0 0 0 7 40
0 e000 0 0 0 0 0000 0 0 0 5 40
0 e000 0 0 0 0 0000 0 0 0 4 40
0 e000 0 0 0 0 0000 0 0 0 4 00
0 e000 0 0 0 0 0000 0 0 0 0 00
0 e000 0 0 0 1 0000 0 0 0 0 00
0 e001 0 0 0 1 0000 0 0 0 0 00
1 e001 0 0 0 1 0000 0 0 0 0 00
0 e001 0 0 0 1 0000 0 0 0 0 00
0 e001 0 0 0 0 0000 0 0 0 0 00
0 e001 0 0 0 0 0000 0 0 0 0 01
1 e001 0 0 0 0 0000 0 0 0 0 01
Stopped at time : 165 ns : File "F:/   OTHER   /
ISim>
```

5. Timer Generator



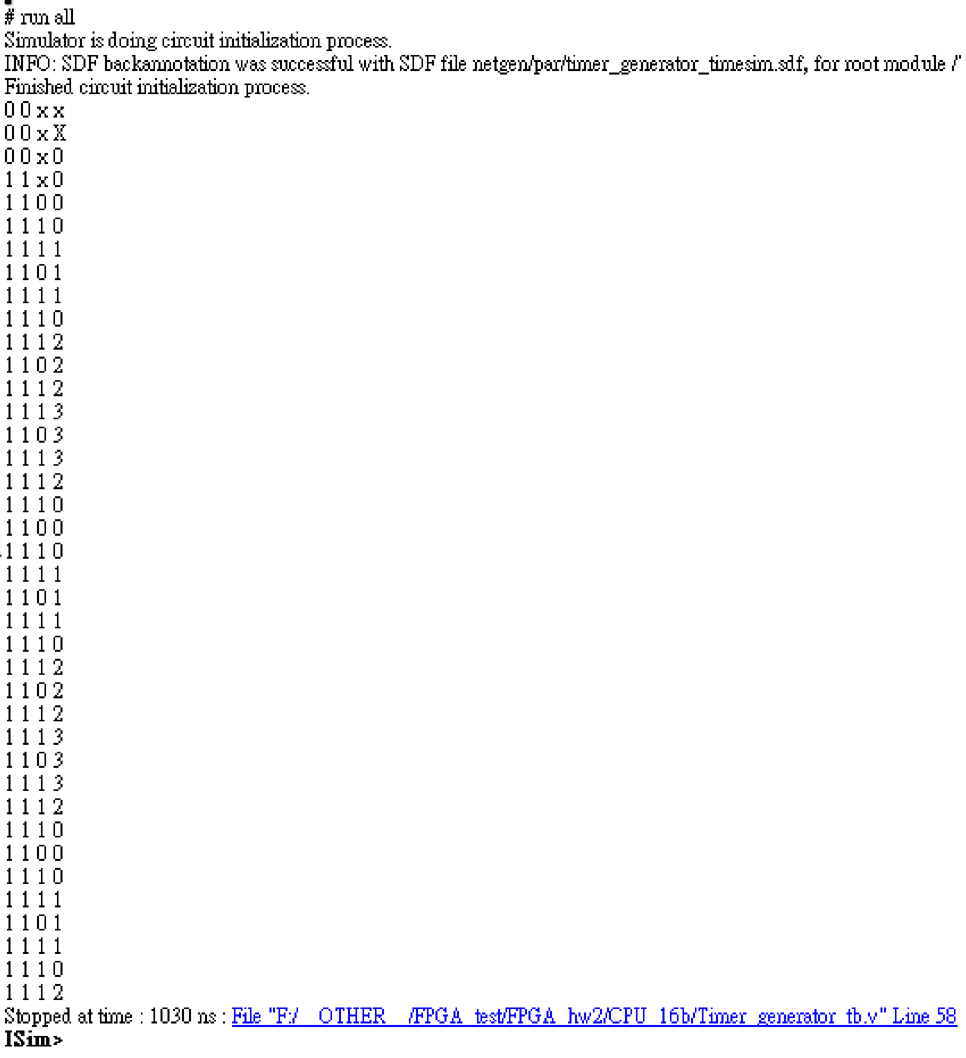
```
48    initial
49        $monitor("%h %h %h %h",rst_n,E,clk,step);
50    initial begin
51        // Initialize Inputs
52        rst_n = 0;
53        E = 0;
54        #3;
55        E=1;
56        rst_n=1;
57        // Add stimulus here
58        #100 $finish;
59    end
60    always begin
61        #5 clk<=0;
62        #5 clk<=1;
63    end
```

behavioral



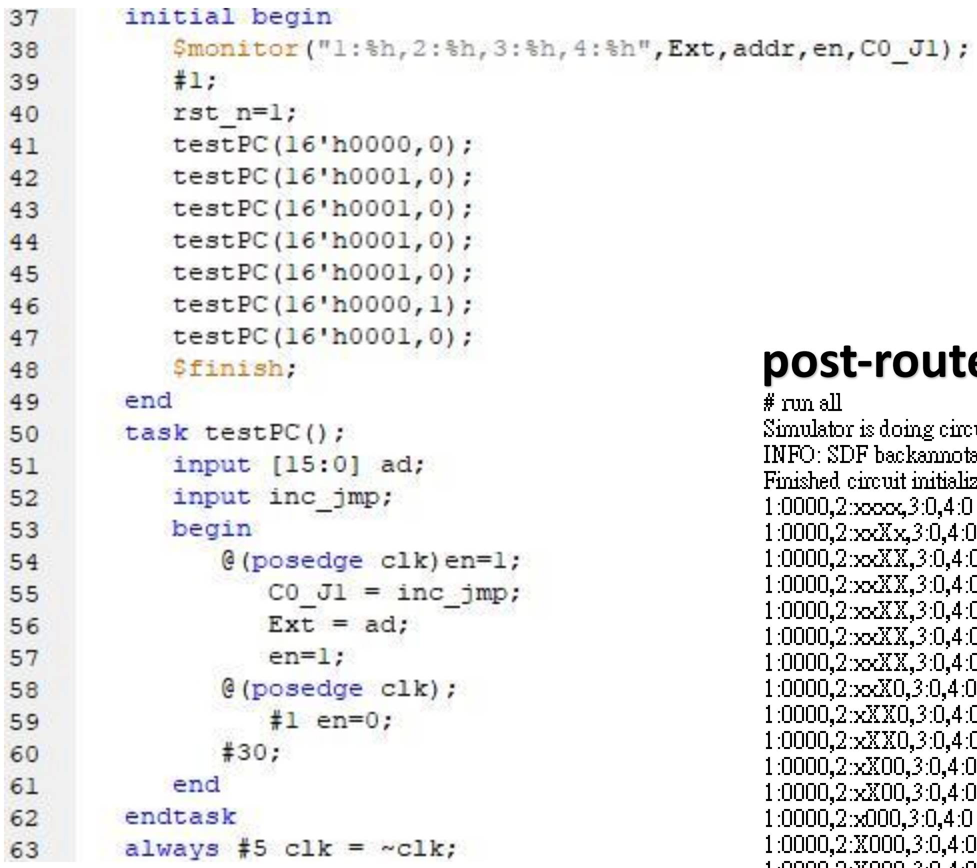
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
0 0 x 0
1 1 x 0
1 1 0 0
1 1 1 1
1 1 0 1
1 1 1 2
1 1 0 2
1 1 1 3
1 1 0 3
1 1 1 0
1 1 0 0
1 1 1 1
1 1 0 1
1 1 1 2
1 1 0 2
1 1 1 3
1 1 0 3
1 1 1 0
1 1 0 0
1 1 1 1
1 1 0 1
1 1 1 2
Stopped at time : 103 ns : File "F:/   OTHER
```

post-route



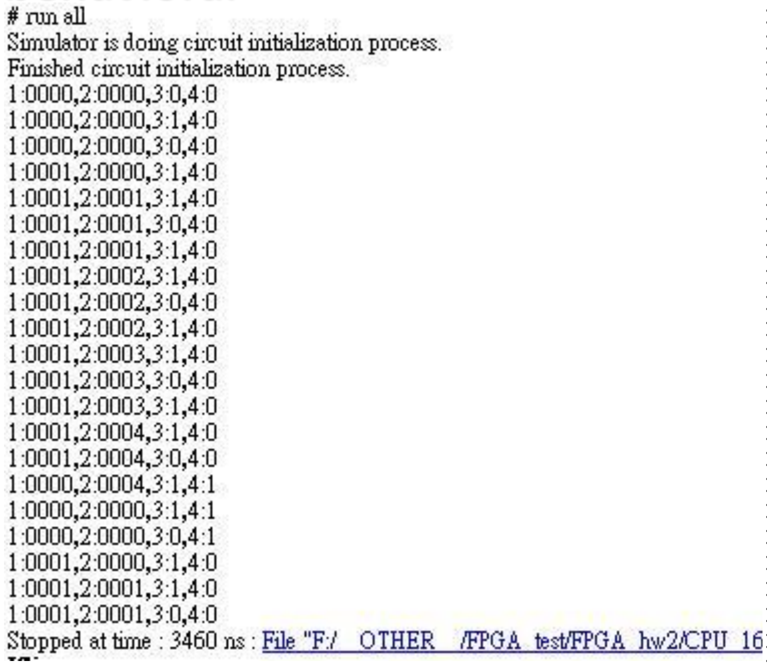
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/timer_generator_timesim.sdf, for root module '/'
Finished circuit initialization process.
0 0 x x
0 0 x X
0 0 x 0
1 1 x 0
1 1 0 0
1 1 1 0
1 1 1 1
1 1 0 1
1 1 1 1
1 1 1 0
1 1 1 2
1 1 0 2
1 1 1 2
1 1 1 3
1 1 0 3
1 1 1 3
1 1 1 2
1 1 1 0
1 1 0 0
1 1 1 0
1 1 1 1
1 1 0 1
1 1 1 1
1 1 1 0
1 1 1 2
1 1 0 2
1 1 1 2
1 1 1 3
1 1 0 3
1 1 1 3
1 1 1 2
1 1 1 0
1 1 0 0
1 1 1 0
1 1 1 1
1 1 0 1
1 1 1 1
1 1 1 0
1 1 1 2
Stopped at time : 1030 ns : File "F:/   OTHER   /FPGA_test/FPGA_hw2/CPU_16b/Timer_generator_tb.v" Line 58
ISim>
```

6. Program Counter



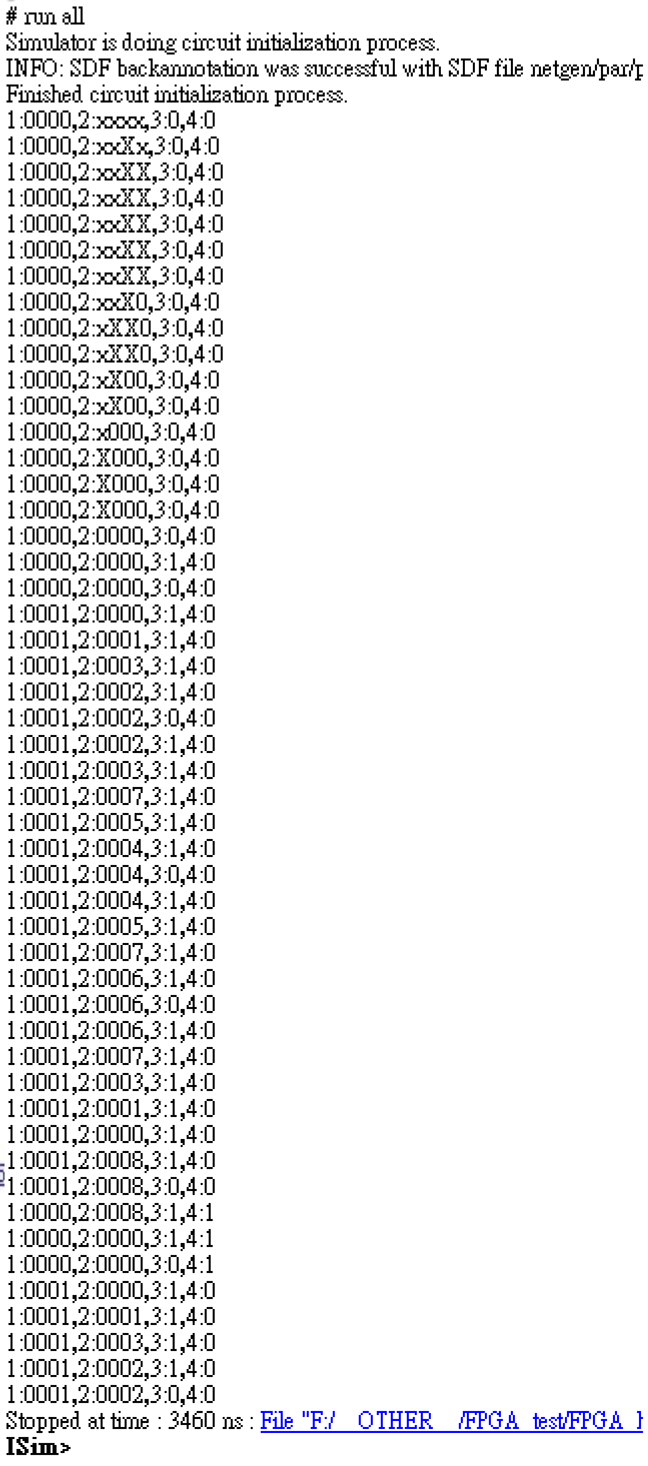
```
initial begin
   $monitor("1:%h,2:%h,3:%h,4:%h",Ext,addr,en,C0_J1);
   #1;
   rst_n=1;
   testPC(16'h0000,0);
   testPC(16'h0001,0);
   testPC(16'h0001,0);
   testPC(16'h0001,0);
   testPC(16'h0001,0);
   testPC(16'h0000,1);
   testPC(16'h0001,0);
   $finish;
end
task testPC();
   input [15:0] ad;
   input inc_jmp;
   begin
      @(posedge clk)en=1;
         C0_J1 = inc_jmp;
         Ext = ad;
         en=1;
      @(posedge clk);
         #1 en=0;
      #30;
   end
endtask
always #5 clk = ~clk;
```

behavioral



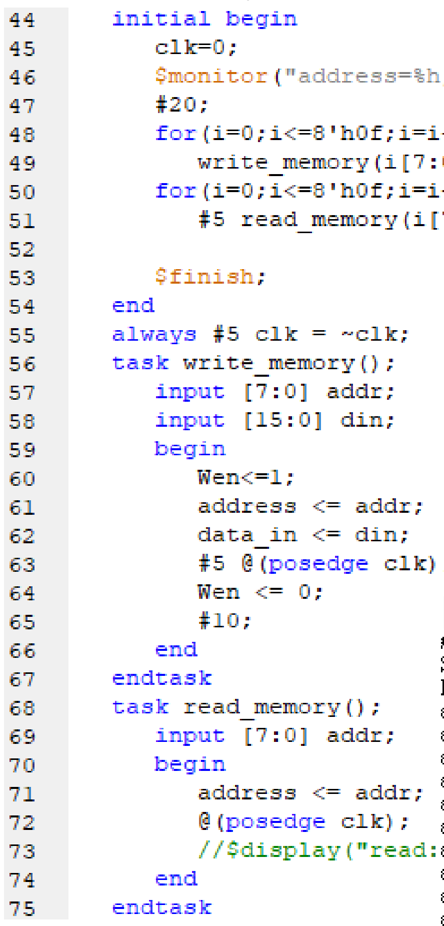
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
1:0000,2:0000,3:0,4:0
1:0000,2:0000,3:1,4:0
1:0000,2:0000,3:0,4:0
1:0001,2:0000,3:1,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0001,3:0,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0002,3:1,4:0
1:0001,2:0002,3:0,4:0
1:0001,2:0002,3:1,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0003,3:0,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0004,3:1,4:0
1:0001,2:0004,3:0,4:0
1:0000,2:0004,3:1,4:1
1:0000,2:0000,3:1,4:1
1:0000,2:0000,3:0,4:1
1:0001,2:0000,3:1,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0001,3:0,4:0
Stopped at time : 3460 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16
```

post-route



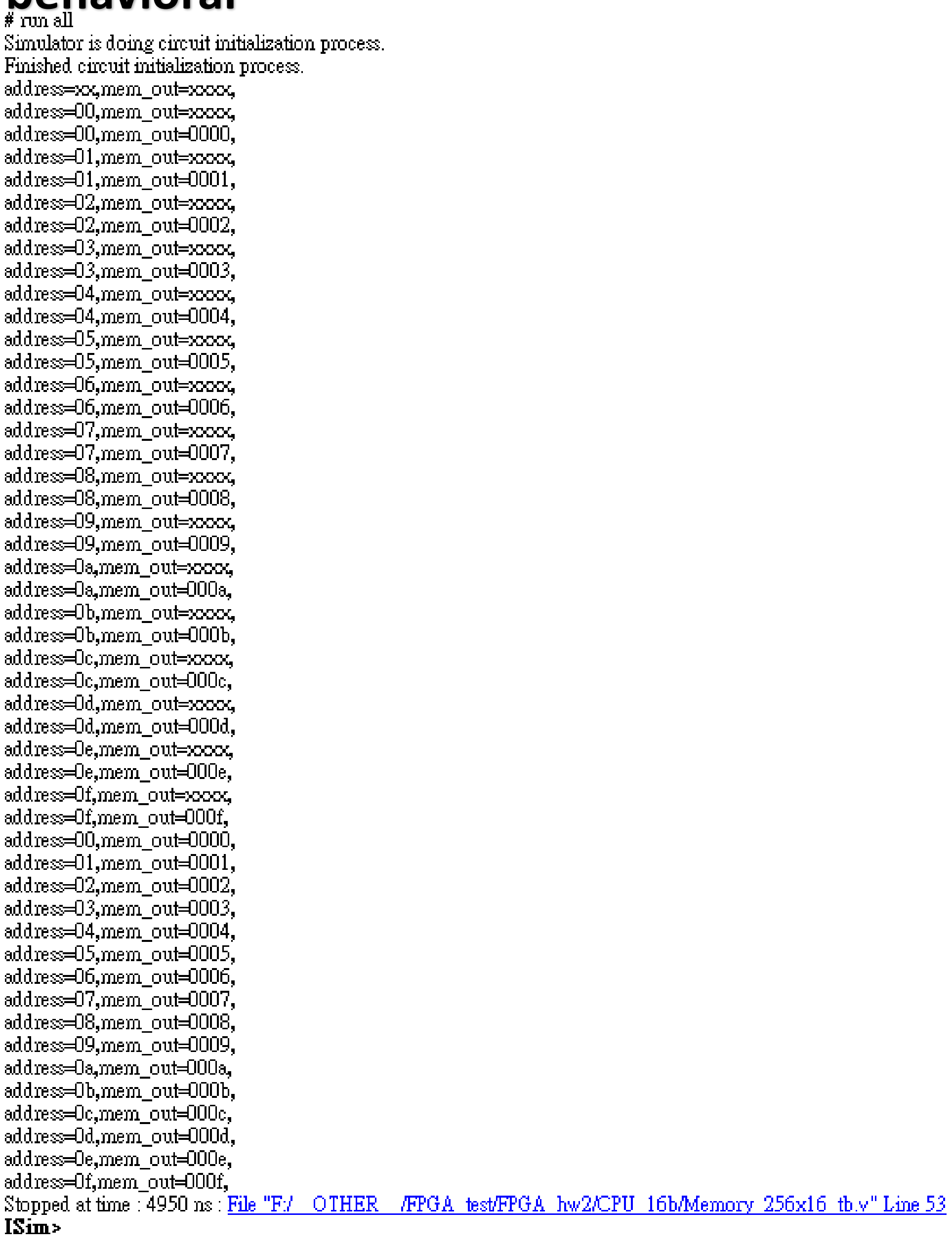
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/p
Finished circuit initialization process.
1:0000,2:xxxx,3:0,4:0
1:0000,2:xxXx,3:0,4:0
1:0000,2:xxXX,3:0,4:0
1:0000,2:xxXX,3:0,4:0
1:0000,2:xxXX,3:0,4:0
1:0000,2:xxXX,3:0,4:0
1:0000,2:xxXX,3:0,4:0
1:0000,2:xxX0,3:0,4:0
1:0000,2:xXX0,3:0,4:0
1:0000,2:xXX0,3:0,4:0
1:0000,2:xX00,3:0,4:0
1:0000,2:xX00,3:0,4:0
1:0000,2:x000,3:0,4:0
1:0000,2:X000,3:0,4:0
1:0000,2:X000,3:0,4:0
1:0000,2:X000,3:0,4:0
1:0000,2:0000,3:0,4:0
1:0000,2:0000,3:1,4:0
1:0000,2:0000,3:0,4:0
1:0001,2:0000,3:1,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0002,3:1,4:0
1:0001,2:0002,3:0,4:0
1:0001,2:0002,3:1,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0007,3:1,4:0
1:0001,2:0005,3:1,4:0
1:0001,2:0004,3:1,4:0
1:0001,2:0004,3:0,4:0
1:0001,2:0004,3:1,4:0
1:0001,2:0005,3:1,4:0
1:0001,2:0007,3:1,4:0
1:0001,2:0006,3:1,4:0
1:0001,2:0006,3:0,4:0
1:0001,2:0006,3:1,4:0
1:0001,2:0007,3:1,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0000,3:1,4:0
1:0001,2:0008,3:1,4:0
1:0001,2:0008,3:0,4:0
1:0000,2:0008,3:1,4:1
1:0000,2:0000,3:1,4:1
1:0000,2:0000,3:0,4:1
1:0001,2:0000,3:1,4:0
1:0001,2:0001,3:1,4:0
1:0001,2:0003,3:1,4:0
1:0001,2:0002,3:1,4:0
1:0001,2:0002,3:0,4:0
Stopped at time : 3460 ns : File "F:/__OTHER__/FPGA_test/FPGA_
ISim>
```

7. 256x16 Memory



```verilog
initial begin
   clk=0;
   $monitor("address=%h,mem_out=%h,",address,data_out);
   #20;
   for(i=0;i<=8'h0f;i=i+1)
      write_memory(i[7:0],i[15:0]);
   for(i=0;i<=8'h0f;i=i+1)
      #5 read_memory(i[7:0]);

   $finish;
end
always #5 clk = ~clk;
task write_memory();
   input [7:0] addr;
   input [15:0] din;
   begin
      Wen<=1;
      address <= addr;
      data_in <= din;
      #5 @(posedge clk);
      Wen <= 0;
      #10;
   end
endtask
task read_memory();
   input [7:0] addr;
   begin
      address <= addr;
      @(posedge clk);
      //$display("read:
   end
endtask
```

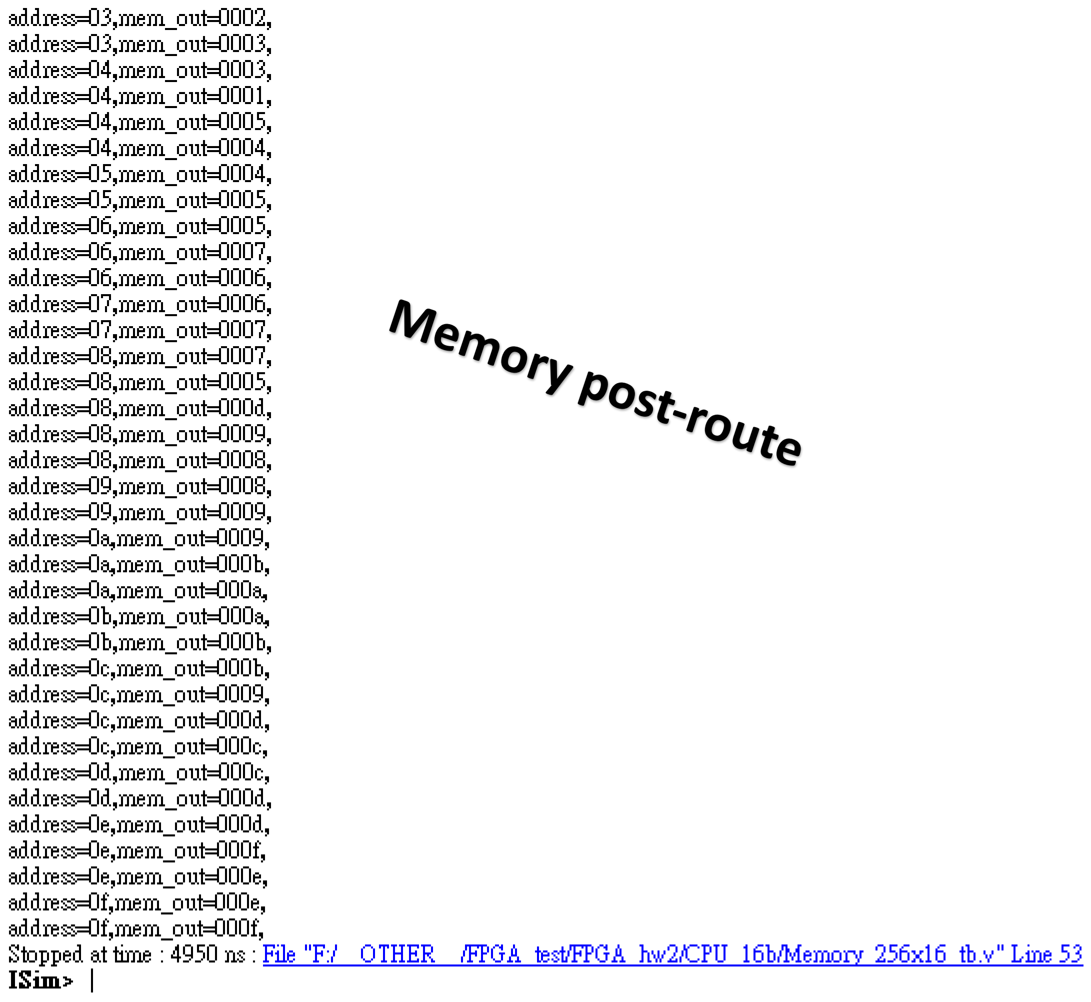
behavioral



```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
address=xx,mem_out=xxxx,
address=00,mem_out=xxxx,
address=00,mem_out=0000,
address=01,mem_out=xxxx,
address=01,mem_out=0001,
address=02,mem_out=xxxx,
address=02,mem_out=0002,
address=03,mem_out=xxxx,
address=03,mem_out=0003,
address=04,mem_out=xxxx,
address=04,mem_out=0004,
address=05,mem_out=xxxx,
address=05,mem_out=0005,
address=06,mem_out=xxxx,
address=06,mem_out=0006,
address=07,mem_out=xxxx,
address=07,mem_out=0007,
address=08,mem_out=xxxx,
address=08,mem_out=0008,
address=09,mem_out=xxxx,
address=09,mem_out=0009,
address=0a,mem_out=xxxx,
address=0a,mem_out=000a,
address=0b,mem_out=xxxx,
address=0b,mem_out=000b,
address=0c,mem_out=xxxx,
address=0c,mem_out=000c,
address=0d,mem_out=xxxx,
address=0d,mem_out=000d,
address=0e,mem_out=xxxx,
address=0e,mem_out=000e,
address=0f,mem_out=xxxx,
address=0f,mem_out=000f,
address=00,mem_out=0000,
address=01,mem_out=0001,
address=02,mem_out=0002,
address=03,mem_out=0003,
address=04,mem_out=0004,
address=05,mem_out=0005,
address=06,mem_out=0006,
address=07,mem_out=0007,
address=08,mem_out=0008,
address=09,mem_out=0009,
address=0a,mem_out=000a,
address=0b,mem_out=000b,
address=0c,mem_out=000c,
address=0d,mem_out=000d,
address=0e,mem_out=000e,
address=0f,mem_out=000f,
Stopped at time : 4950 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/Memory_256x16_tb.v" Line 53
ISim>
```

Memory post-route

```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/memory_256x16_timesim.sdf, for root module /Memory_256x16_Memory_256x16_sch_tb/UUT/.
Finished circuit initialization process.
address=xx,mem_out=xxxx,
address=00,mem_out=xxxx,
address=00,mem_out=xxXx,
address=00,mem_out=xxXX,
address=00,mem_out=xxXX,
address=00,mem_out=xxXX,
address=00,mem_out=xxXX,
address=00,mem_out=xx0X,
address=00,mem_out=xx0X,
address=00,mem_out=xX0X,
address=00,mem_out=xX0X,
address=00,mem_out=xX00,
address=00,mem_out=xX00,
address=00,mem_out=XX00,
address=00,mem_out=XX00,
address=00,mem_out=XX00,
address=00,mem_out=X000,
address=00,mem_out=0000,
address=01,mem_out=0000,
 WARNING: at 354272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:354261 ps, WADR0:354272 ps,42 ps,462 ps)


address=01,mem_out=0001,
address=02,mem_out=0001,
 WARNING: at 554272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:554261 ps, WADR0:554272 ps,42 ps,462 ps)


address=02,mem_out=0000,
address=02,mem_out=0002,
address=03,mem_out=0002,
 WARNING: at 754272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:754261 ps, WADR0:754272 ps,42 ps,462 ps)


address=03,mem_out=0003,
address=04,mem_out=0003,
 WARNING: at 954272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:954261 ps, WADR0:954272 ps,42 ps,462 ps)


address=04,mem_out=0001,
address=04,mem_out=0000,
address=04,mem_out=0004,
address=05,mem_out=0004,
 WARNING: at 1154272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:1154261 ps, WADR0:1154272 ps,42 ps,462 ps)

address=05,mem_out=0005,
address=06,mem_out=0005,
 WARNING: at 1354272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:1354261 ps, WADR0:1354272 ps,42 ps,462 ps)


address=06,mem_out=0004,
address=06,mem_out=0006,
address=07,mem_out=0006,
 WARNING: at 1554272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:1554261 ps, WADR0:1554272 ps,42 ps,462 ps)
address=07,mem_out=0007,
address=08,mem_out=0007,
 WARNING: at 1754272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:1754261 ps, WADR0:1754272 ps,42 ps,462 ps)

address=08,mem_out=0005,
address=08,mem_out=0004,
address=08,mem_out=0000,
address=08,mem_out=0008,
address=09,mem_out=0008,
 WARNING: at 1954272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:1954261 ps, WADR0:1954272 ps,42 ps,462 ps)

address=09,mem_out=0009,
address=0a,mem_out=0009,
 WARNING: at 2154272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:2154261 ps, WADR0:2154272 ps,42 ps,462 ps)

address=0a,mem_out=0008,
address=0a,mem_out=000a,
address=0b,mem_out=000a,
 WARNING: at 2354272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:2354261 ps, WADR0:2354272 ps,42 ps,462 ps)

address=0b,mem_out=000b,
address=0c,mem_out=000b,
 WARNING: at 2554272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:2554261 ps, WADR0:2554272 ps,42 ps,462 ps)

address=0c,mem_out=0009,
address=0c,mem_out=0008,
address=0c,mem_out=0000,
address=0c,mem_out=0008,
address=0c,mem_out=000c,
address=0d,mem_out=000c,
 WARNING: at 2754272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:2754261 ps, WADR0:2754272 ps,42 ps,462 ps)

address=0d,mem_out=000d,
address=0e,mem_out=000d,
 WARNING: at 2954272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:2954261 ps, WADR0:2954272 ps,42 ps,462 ps)

address=0e,mem_out=000c,
address=0e,mem_out=000e,
address=0f,mem_out=000e,
 WARNING: at 3154272 ps: Timing violation in /Memory_256x16_Memory_256x16_sch_tb/UUT/\Mram_memory16/D /  $setuphold<hold>( CLK:3154261 ps, WADR0:3154272 ps,42 ps,462 ps)


address=0f,mem_out=000f,
address=00,mem_out=000f,
address=00,mem_out=000d,
address=00,mem_out=0005,
address=00,mem_out=0001,
address=00,mem_out=0000,
address=01,mem_out=0000,
address=01,mem_out=0001,
address=02,mem_out=0001,
address=02,mem_out=0003,
address=02,mem_out=0002,
```

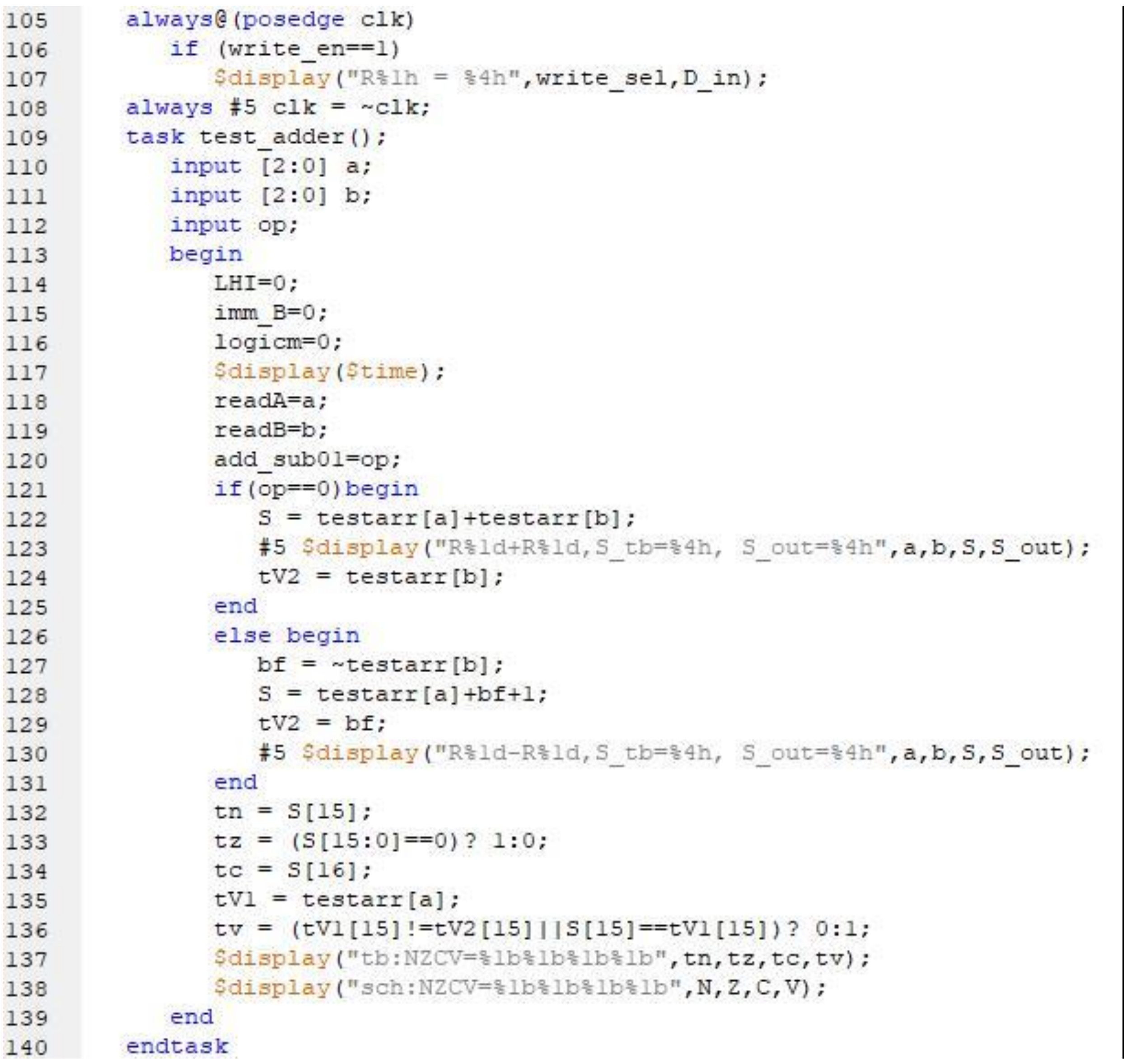


Memory post-route

```
address=03,mem_out=0002,
address=03,mem_out=0003,
address=04,mem_out=0003,
address=04,mem_out=0001,
address=04,mem_out=0005,
address=04,mem_out=0004,
address=05,mem_out=0004,
address=05,mem_out=0005,
address=06,mem_out=0005,
address=06,mem_out=0007,
address=06,mem_out=0006,
address=07,mem_out=0006,
address=07,mem_out=0007,
address=08,mem_out=0007,
address=08,mem_out=0005,
address=08,mem_out=000d,
address=08,mem_out=0009,
address=08,mem_out=0008,
address=09,mem_out=0008,
address=09,mem_out=0009,
address=0a,mem_out=0009,
address=0a,mem_out=000b,
address=0a,mem_out=000a,
address=0b,mem_out=000a,
address=0b,mem_out=000b,
address=0c,mem_out=000b,
address=0c,mem_out=0009,
address=0c,mem_out=000d,
address=0c,mem_out=000c,
address=0d,mem_out=000c,
address=0d,mem_out=000d,
address=0e,mem_out=000d,
address=0e,mem_out=000f,
address=0e,mem_out=000e,
address=0f,mem_out=000e,
address=0f,mem_out=000f,
Stopped at time : 4950 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/Memory_256x16_tb.v" Line 53
ISim> |
```

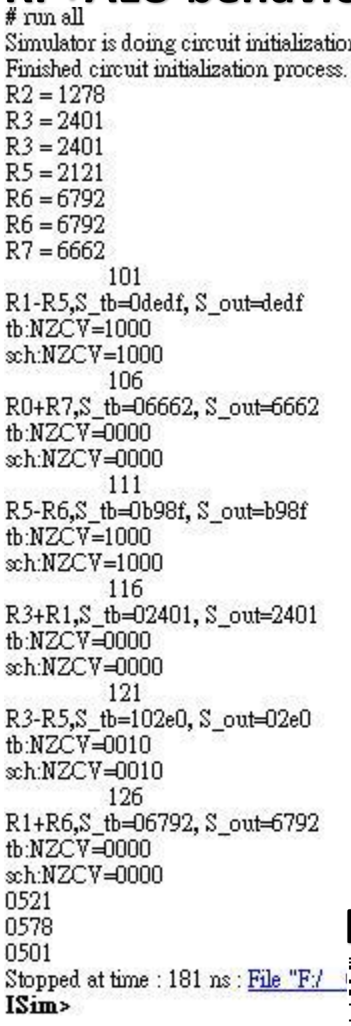
8. Register File + ALU

```verilog
initial begin
   clk=0;
   rst_n=0;
   for(i=0;i<8;i=i+1)
      testarr[i]<=16'h0;
   write_en=0;
   #1 rst_n=1;
   #5 write_register(3'b010,16'h1278);
   write_register(3'b011,16'h2401);
   write_register(3'b101,16'h2121);
   write_register(3'b110,16'h6792);
   write_register(3'b111,16'h6662);
   #20 test_adder(3'b001,3'b101,1); //1
   test_adder(3'b000,3'b111,0); //2
   test_adder(3'b101,3'b110,1); //3
   test_adder(3'b011,3'b001,0); //4
   test_adder(3'b011,3'b101,1); //5
   test_adder(3'b001,3'b110,0); //6
   #20 alu_B_in=5'h5;imm_B=1;
   readA = 3'b101;LHI=1;logicm=0;
   #10 $display("%h",S_out);
   alu_B_in=5'h5;imm_B=1;
   readA = 3'b010;LHI=1;logicm=0;
   #10 $display("%h",S_out);
   alu_B_in=5'h5;imm_B=1;
   readA = 3'b011;LHI=1;logicm=0;
   #10 $display("%h",S_out);
   $finish;
end
task write_register();
   input [2:0] sel;
   input [15:0] D;
   begin
      write_sel <= sel;
      D_in <= D;
      write_en <= 1;
      testarr[sel] <= D;
      #15 write_en = 0;
   end
endtask
```



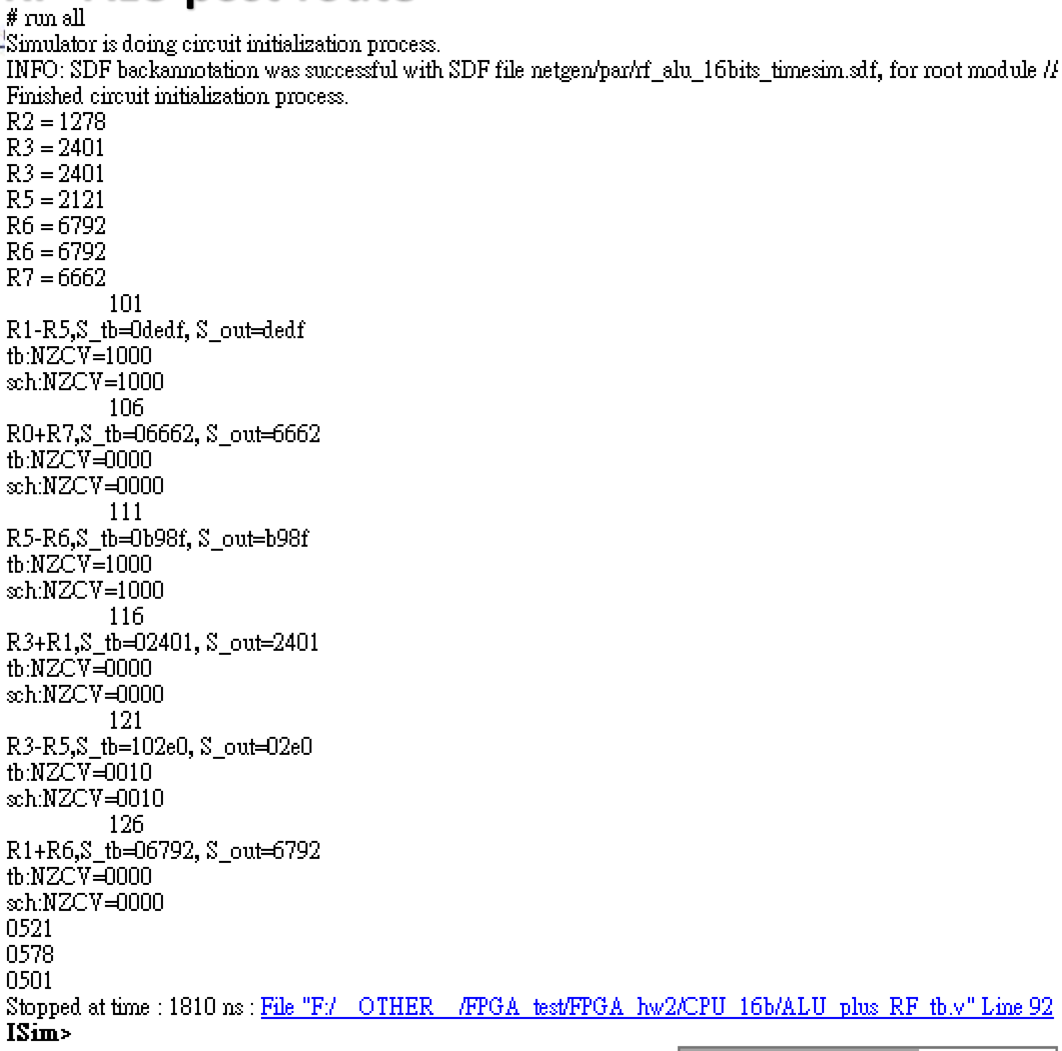
```
always@(posedge clk)
   if (write_en==1)
      $display("R%1h = %4h",write_sel,D_in);
always #5 clk = ~clk;
task test_adder();
   input [2:0] a;
   input [2:0] b;
   input op;
   begin
      LHI=0;
      imm_B=0;
      logicm=0;
      $display($time);
      readA=a;
      readB=b;
      add_sub01=op;
      if(op==0)begin
         S = testarr[a]+testarr[b];
         #5 $display("R%1d+R%1d,S_tb=%4h, S_out=%4h",a,b,S,S_out);
         tV2 = testarr[b];
      end
      else begin
         bf = ~testarr[b];
         S = testarr[a]+bf+1;
         tV2 = bf;
         #5 $display("R%1d-R%1d,S_tb=%4h, S_out=%4h",a,b,S,S_out);
      end
      tn = S[15];
      tz = (S[15:0]==0)? 1:0;
      tc = S[16];
      tV1 = testarr[a];
      tv = (tV1[15]!=tV2[15]||S[15]==tV1[15])? 0:1;
      $display("tb:NZCV=%1b%1b%1b%1b",tn,tz,tc,tv);
      $display("sch:NZCV=%1b%1b%1b%1b",N,Z,C,V);
   end
endtask
```

RF+ALU behavioral



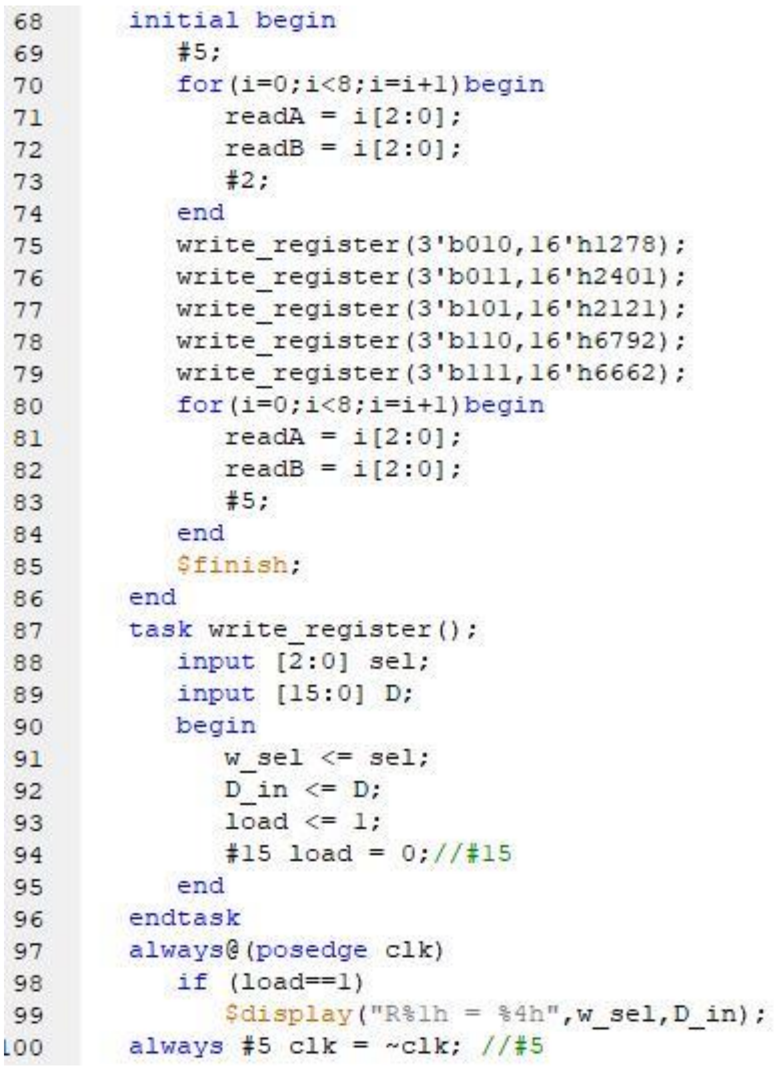
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
R2 = 1278
R3 = 2401
R3 = 2401
R5 = 2121
R6 = 6792
R6 = 6792
R7 = 6662
                101
R1-R5,S_tb=0dedf, S_out=dedf
tb:NZCV=1000
sch:NZCV=1000
                106
R0+R7,S_tb=06662, S_out=6662
tb:NZCV=0000
sch:NZCV=0000
                111
R5-R6,S_tb=0b98f, S_out=b98f
tb:NZCV=1000
sch:NZCV=1000
                116
R3+R1,S_tb=02401, S_out=2401
tb:NZCV=0000
sch:NZCV=0000
                121
R3-R5,S_tb=102e0, S_out=02e0
tb:NZCV=0010
sch:NZCV=0010
                126
R1+R6,S_tb=06792, S_out=6792
tb:NZCV=0000
sch:NZCV=0000
0521
0578
0501
Stopped at time : 181 ns : File "F:/
ISim>
```

RF+ALU post-route



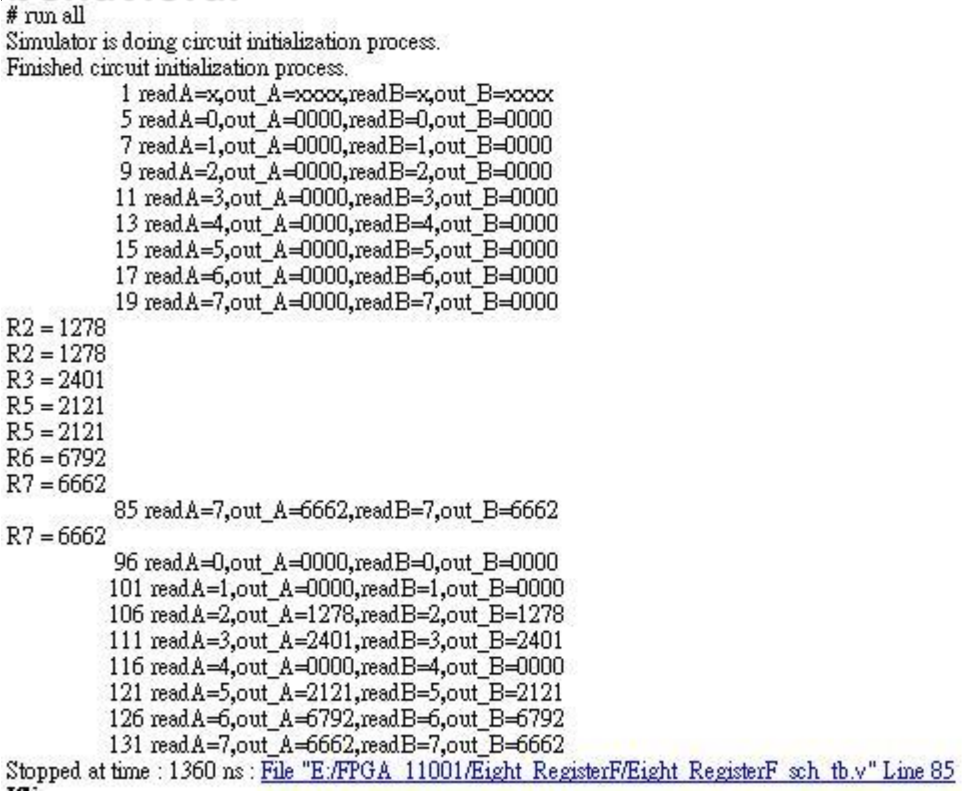
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/rf_alu_16bits_timesim.sdf, for root module /
Finished circuit initialization process.
R2 = 1278
R3 = 2401
R3 = 2401
R5 = 2121
R6 = 6792
R6 = 6792
R7 = 6662
                101
R1-R5,S_tb=0dedf, S_out=dedf
tb:NZCV=1000
sch:NZCV=1000
                106
R0+R7,S_tb=06662, S_out=6662
tb:NZCV=0000
sch:NZCV=0000
                111
R5-R6,S_tb=0b98f, S_out=b98f
tb:NZCV=1000
sch:NZCV=1000
                116
R3+R1,S_tb=02401, S_out=2401
tb:NZCV=0000
sch:NZCV=0000
                121
R3-R5,S_tb=102e0, S_out=02e0
tb:NZCV=0010
sch:NZCV=0010
                126
R1+R6,S_tb=06792, S_out=6792
tb:NZCV=0000
sch:NZCV=0000
0521
0578
0501
Stopped at time : 1810 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/ALU_plus_RF_tb.v" Line 92
ISim>
```

9. Register File



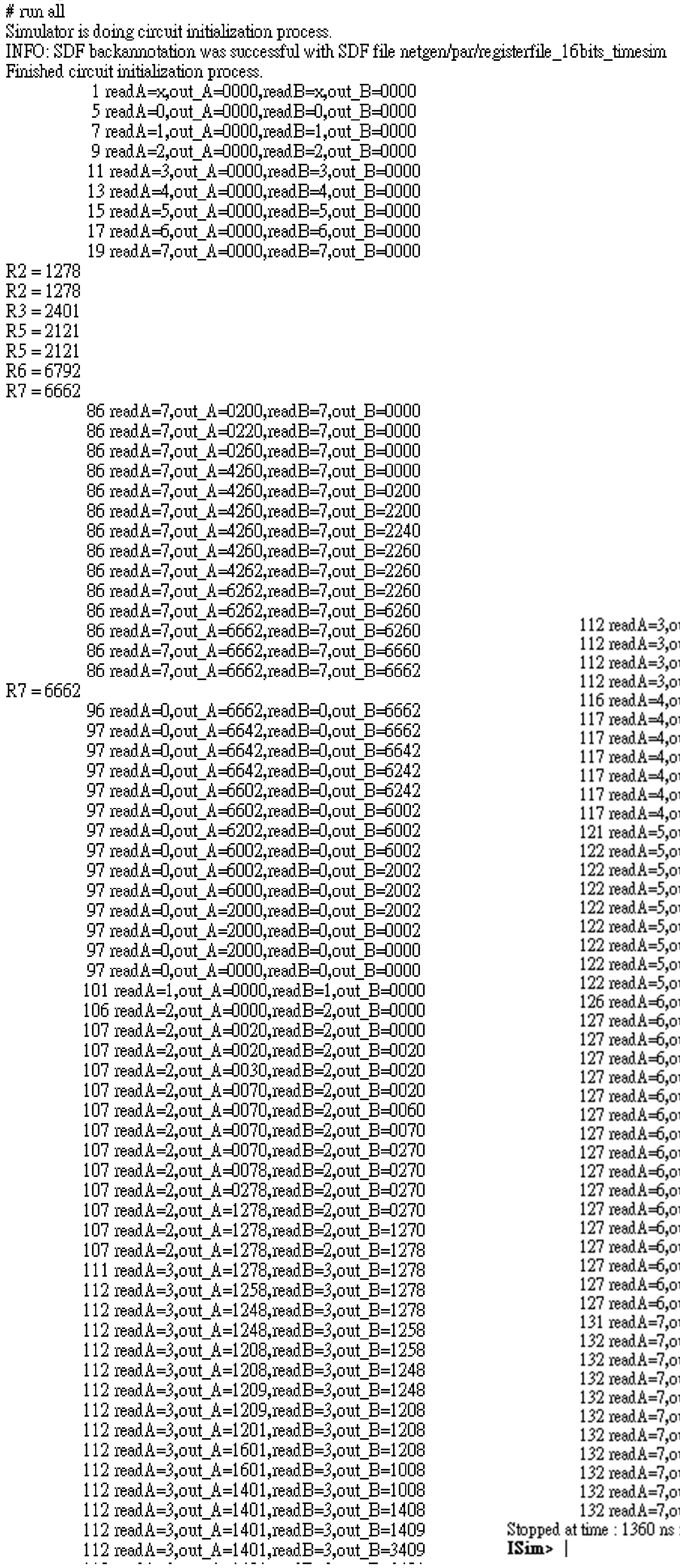
```verilog
initial begin
   #5;
   for(i=0;i<8;i=i+1)begin
      readA = i[2:0];
      readB = i[2:0];
      #2;
   end
   write_register(3'b010,16'h1278);
   write_register(3'b011,16'h2401);
   write_register(3'b101,16'h2121);
   write_register(3'b110,16'h6792);
   write_register(3'b111,16'h6662);
   for(i=0;i<8;i=i+1)begin
      readA = i[2:0];
      readB = i[2:0];
      #5;
   end
   $finish;
end
task write_register();
   input [2:0] sel;
   input [15:0] D;
   begin
      w_sel <= sel;
      D_in <= D;
      load <= 1;
      #15 load = 0;//#15
   end
endtask
always@(posedge clk)
   if (load==1)
      $display("R%1h = %4h",w_sel,D_in);
always #5 clk = ~clk; //#5
```

behavioral

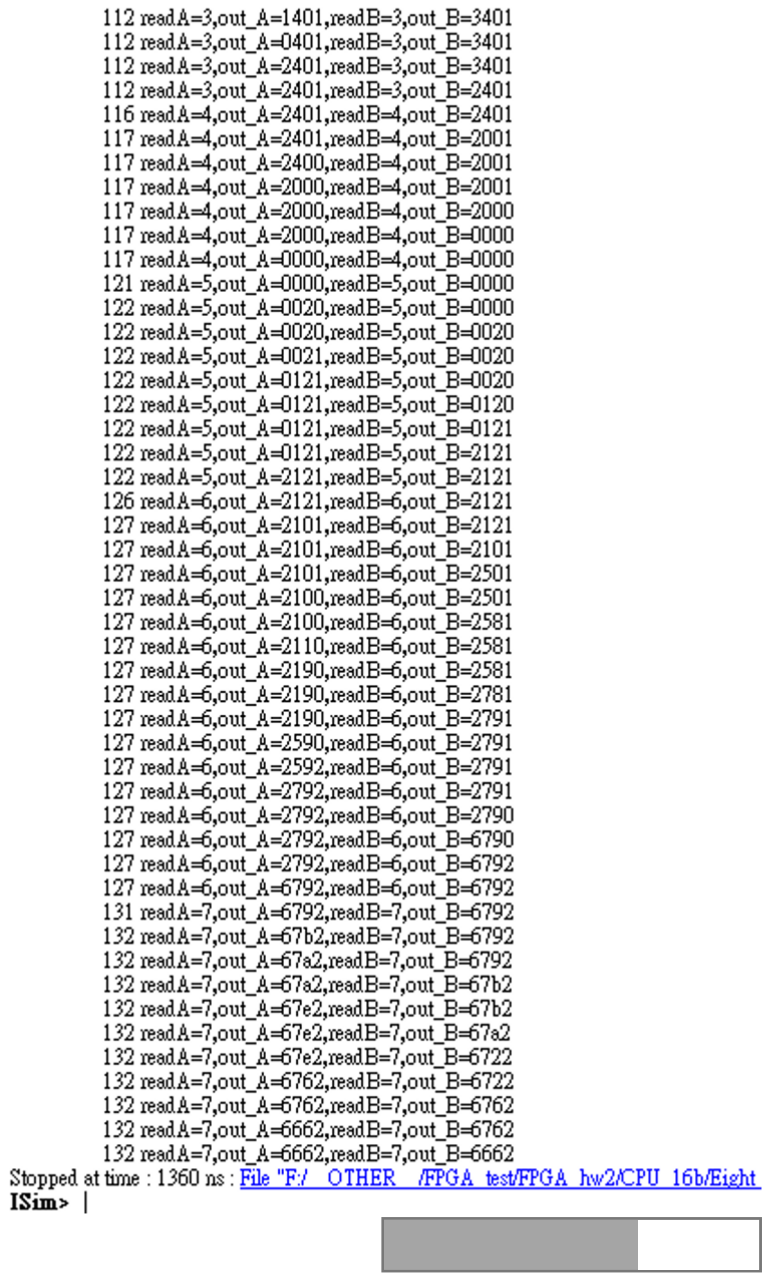


```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
                 1 readA=x,out_A=xxxx,readB=x,out_B=xxxx
                 5 readA=0,out_A=0000,readB=0,out_B=0000
                 7 readA=1,out_A=0000,readB=1,out_B=0000
                 9 readA=2,out_A=0000,readB=2,out_B=0000
                11 readA=3,out_A=0000,readB=3,out_B=0000
                13 readA=4,out_A=0000,readB=4,out_B=0000
                15 readA=5,out_A=0000,readB=5,out_B=0000
                17 readA=6,out_A=0000,readB=6,out_B=0000
                19 readA=7,out_A=0000,readB=7,out_B=0000
R2 = 1278
R2 = 1278
R3 = 2401
R5 = 2121
R5 = 2121
R6 = 6792
R7 = 6662
                85 readA=7,out_A=6662,readB=7,out_B=6662
R7 = 6662
                96 readA=0,out_A=0000,readB=0,out_B=0000
               101 readA=1,out_A=0000,readB=1,out_B=0000
               106 readA=2,out_A=1278,readB=2,out_B=1278
               111 readA=3,out_A=2401,readB=3,out_B=2401
               116 readA=4,out_A=0000,readB=4,out_B=0000
               121 readA=5,out_A=2121,readB=5,out_B=2121
               126 readA=6,out_A=6792,readB=6,out_B=6792
               131 readA=7,out_A=6662,readB=7,out_B=6662
Stopped at time : 1360 ns : File "E:/FPGA_11001/Eight_RegisterF/Eight_RegisterF_sch_tb.v" Line 85
```

RF post-route

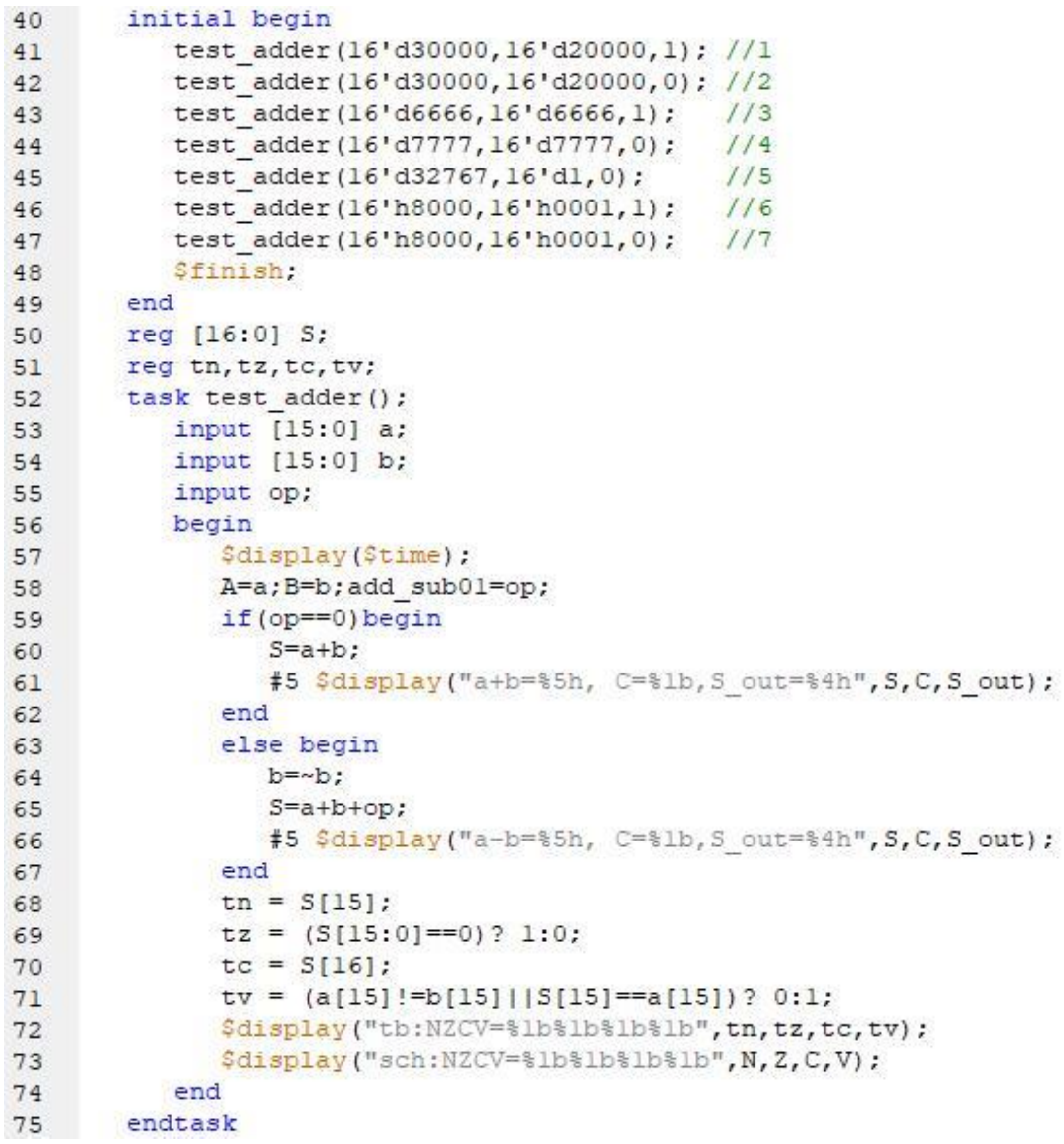


```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/registerfile_16bits_timesim
Finished circuit initialization process.
                    1 readA=x,out_A=0000,readB=x,out_B=0000
                    5 readA=0,out_A=0000,readB=0,out_B=0000
                    7 readA=1,out_A=0000,readB=1,out_B=0000
                    9 readA=2,out_A=0000,readB=2,out_B=0000
                   11 readA=3,out_A=0000,readB=3,out_B=0000
                   13 readA=4,out_A=0000,readB=4,out_B=0000
                   15 readA=5,out_A=0000,readB=5,out_B=0000
                   17 readA=6,out_A=0000,readB=6,out_B=0000
                   19 readA=7,out_A=0000,readB=7,out_B=0000
R2 = 1278
R2 = 1278
R3 = 2401
R5 = 2121
R5 = 2121
R6 = 6792
R7 = 6662
                   86 readA=7,out_A=0200,readB=7,out_B=0000
                   86 readA=7,out_A=0220,readB=7,out_B=0000
                   86 readA=7,out_A=0260,readB=7,out_B=0000
                   86 readA=7,out_A=4260,readB=7,out_B=0000
                   86 readA=7,out_A=4260,readB=7,out_B=0200
                   86 readA=7,out_A=4260,readB=7,out_B=2200
                   86 readA=7,out_A=4260,readB=7,out_B=2240
                   86 readA=7,out_A=4260,readB=7,out_B=2260
                   86 readA=7,out_A=4262,readB=7,out_B=2260
                   86 readA=7,out_A=6262,readB=7,out_B=2260
                   86 readA=7,out_A=6262,readB=7,out_B=6260
                   86 readA=7,out_A=6662,readB=7,out_B=6260
                   86 readA=7,out_A=6662,readB=7,out_B=6660
                   86 readA=7,out_A=6662,readB=7,out_B=6662
R7 = 6662
                   96 readA=0,out_A=6662,readB=0,out_B=6662
                   97 readA=0,out_A=6642,readB=0,out_B=6662
                   97 readA=0,out_A=6642,readB=0,out_B=6642
                   97 readA=0,out_A=6642,readB=0,out_B=6242
                   97 readA=0,out_A=6602,readB=0,out_B=6242
                   97 readA=0,out_A=6602,readB=0,out_B=6002
                   97 readA=0,out_A=6202,readB=0,out_B=6002
                   97 readA=0,out_A=6002,readB=0,out_B=6002
                   97 readA=0,out_A=6002,readB=0,out_B=2002
                   97 readA=0,out_A=6000,readB=0,out_B=2002
                   97 readA=0,out_A=2000,readB=0,out_B=2002
                   97 readA=0,out_A=2000,readB=0,out_B=0002
                   97 readA=0,out_A=2000,readB=0,out_B=0000
                   97 readA=0,out_A=0000,readB=0,out_B=0000
                  101 readA=1,out_A=0000,readB=1,out_B=0000
                  106 readA=2,out_A=0000,readB=2,out_B=0000
                  107 readA=2,out_A=0020,readB=2,out_B=0000
                  107 readA=2,out_A=0020,readB=2,out_B=0020
                  107 readA=2,out_A=0030,readB=2,out_B=0020
                  107 readA=2,out_A=0070,readB=2,out_B=0020
                  107 readA=2,out_A=0070,readB=2,out_B=0060
                  107 readA=2,out_A=0070,readB=2,out_B=0070
                  107 readA=2,out_A=0070,readB=2,out_B=0270
                  107 readA=2,out_A=0078,readB=2,out_B=0270
                  107 readA=2,out_A=0278,readB=2,out_B=0270
                  107 readA=2,out_A=1278,readB=2,out_B=0270
                  107 readA=2,out_A=1278,readB=2,out_B=1270
                  107 readA=2,out_A=1278,readB=2,out_B=1278
                  111 readA=3,out_A=1278,readB=3,out_B=1278
                  112 readA=3,out_A=1258,readB=3,out_B=1278
                  112 readA=3,out_A=1248,readB=3,out_B=1278
                  112 readA=3,out_A=1248,readB=3,out_B=1258
                  112 readA=3,out_A=1208,readB=3,out_B=1258
                  112 readA=3,out_A=1208,readB=3,out_B=1248
                  112 readA=3,out_A=1209,readB=3,out_B=1248
                  112 readA=3,out_A=1209,readB=3,out_B=1208
                  112 readA=3,out_A=1201,readB=3,out_B=1208
                  112 readA=3,out_A=1601,readB=3,out_B=1208
                  112 readA=3,out_A=1601,readB=3,out_B=1008
                  112 readA=3,out_A=1401,readB=3,out_B=1008
                  112 readA=3,out_A=1401,readB=3,out_B=1408
                  112 readA=3,out_A=1401,readB=3,out_B=1409
                  112 readA=3,out_A=1401,readB=3,out_B=3409
```



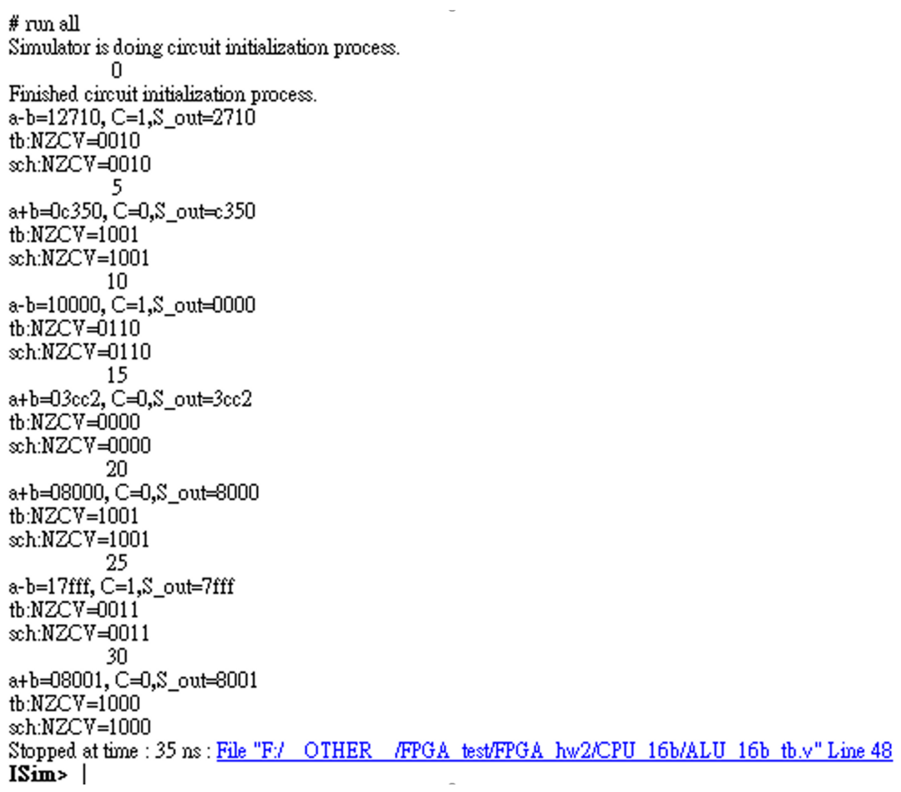
```
              112 readA=3,out_A=1401,readB=3,out_B=3401
              112 readA=3,out_A=0401,readB=3,out_B=3401
              112 readA=3,out_A=2401,readB=3,out_B=3401
              112 readA=3,out_A=2401,readB=3,out_B=2401
              116 readA=4,out_A=2401,readB=4,out_B=2401
              117 readA=4,out_A=2401,readB=4,out_B=2001
              117 readA=4,out_A=2400,readB=4,out_B=2001
              117 readA=4,out_A=2000,readB=4,out_B=2001
              117 readA=4,out_A=2000,readB=4,out_B=2000
              117 readA=4,out_A=2000,readB=4,out_B=0000
              117 readA=4,out_A=0000,readB=4,out_B=0000
              121 readA=5,out_A=0000,readB=5,out_B=0000
              122 readA=5,out_A=0020,readB=5,out_B=0000
              122 readA=5,out_A=0020,readB=5,out_B=0020
              122 readA=5,out_A=0021,readB=5,out_B=0020
              122 readA=5,out_A=0121,readB=5,out_B=0020
              122 readA=5,out_A=0121,readB=5,out_B=0120
              122 readA=5,out_A=0121,readB=5,out_B=0121
              122 readA=5,out_A=0121,readB=5,out_B=2121
              122 readA=5,out_A=2121,readB=5,out_B=2121
              126 readA=6,out_A=2121,readB=6,out_B=2121
              127 readA=6,out_A=2101,readB=6,out_B=2121
              127 readA=6,out_A=2101,readB=6,out_B=2101
              127 readA=6,out_A=2101,readB=6,out_B=2501
              127 readA=6,out_A=2100,readB=6,out_B=2501
              127 readA=6,out_A=2100,readB=6,out_B=2581
              127 readA=6,out_A=2110,readB=6,out_B=2581
              127 readA=6,out_A=2190,readB=6,out_B=2581
              127 readA=6,out_A=2190,readB=6,out_B=2781
              127 readA=6,out_A=2190,readB=6,out_B=2791
              127 readA=6,out_A=2590,readB=6,out_B=2791
              127 readA=6,out_A=2592,readB=6,out_B=2791
              127 readA=6,out_A=2792,readB=6,out_B=2791
              127 readA=6,out_A=2792,readB=6,out_B=2790
              127 readA=6,out_A=2792,readB=6,out_B=6790
              127 readA=6,out_A=2792,readB=6,out_B=6792
              127 readA=6,out_A=6792,readB=6,out_B=6792
              131 readA=7,out_A=6792,readB=7,out_B=6792
              132 readA=7,out_A=67b2,readB=7,out_B=6792
              132 readA=7,out_A=67a2,readB=7,out_B=6792
              132 readA=7,out_A=67a2,readB=7,out_B=67b2
              132 readA=7,out_A=67e2,readB=7,out_B=67b2
              132 readA=7,out_A=67e2,readB=7,out_B=67a2
              132 readA=7,out_A=67e2,readB=7,out_B=6722
              132 readA=7,out_A=6762,readB=7,out_B=6722
              132 readA=7,out_A=6762,readB=7,out_B=6762
              132 readA=7,out_A=6662,readB=7,out_B=6762
              132 readA=7,out_A=6662,readB=7,out_B=6662
Stopped at time : 1360 ns : File "F:/   OTHER   /FPGA test/FPGA hw2/CPU 16b/Eight
ISim> |
```

10. ALU



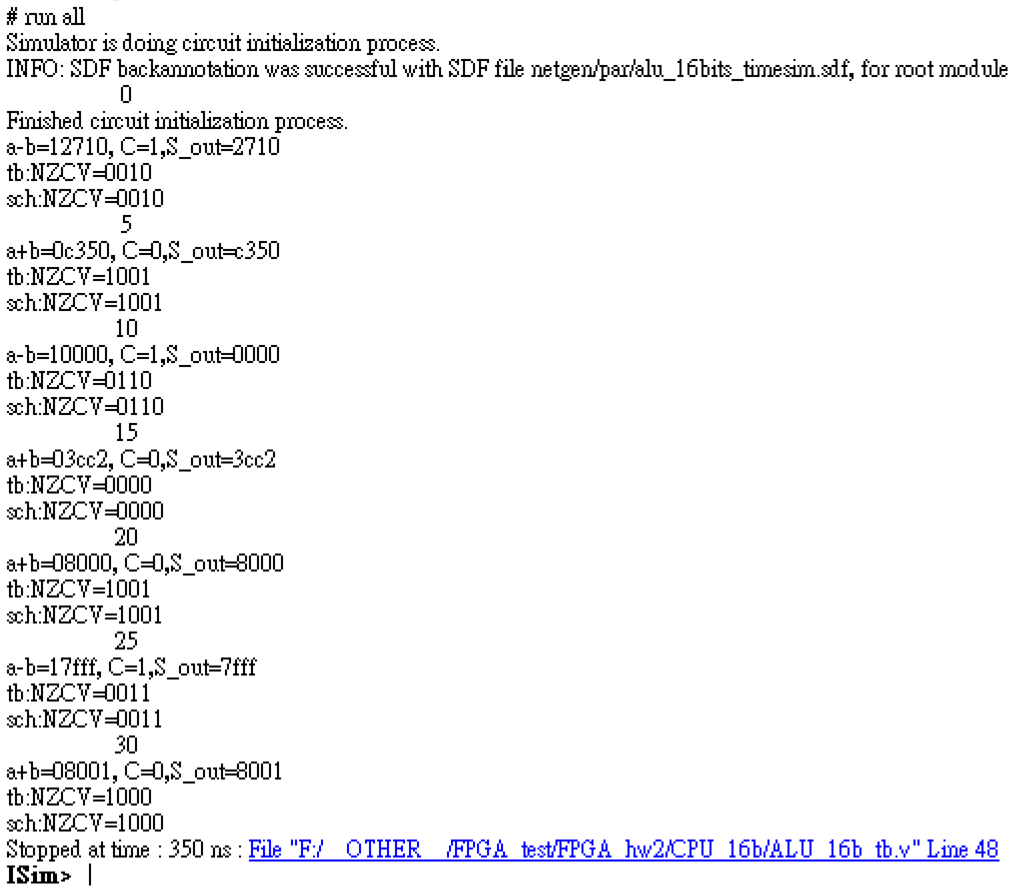
```
initial begin
   test_adder(16'd30000,16'd20000,1); //1
   test_adder(16'd30000,16'd20000,0); //2
   test_adder(16'd6666,16'd6666,1);   //3
   test_adder(16'd7777,16'd7777,0);   //4
   test_adder(16'd32767,16'd1,0);     //5
   test_adder(16'h8000,16'h0001,1);   //6
   test_adder(16'h8000,16'h0001,0);   //7
   $finish;
end
reg [16:0] S;
reg tn,tz,tc,tv;
task test_adder();
   input [15:0] a;
   input [15:0] b;
   input op;
   begin
      $display($time);
      A=a;B=b;add_sub01=op;
      if(op==0)begin
         S=a+b;
         #5 $display("a+b=%5h, C=%1b,S_out=%4h",S,C,S_out);
      end
      else begin
         b=~b;
         S=a+b+op;
         #5 $display("a-b=%5h, C=%1b,S_out=%4h",S,C,S_out);
      end
      tn = S[15];
      tz = (S[15:0]==0)? 1:0;
      tc = S[16];
      tv = (a[15]!=b[15]||S[15]==a[15])? 0:1;
      $display("tb:NZCV=%1b%1b%1b%1b",tn,tz,tc,tv);
      $display("sch:NZCV=%1b%1b%1b%1b",N,Z,C,V);
   end
endtask
```

ALU behavioral



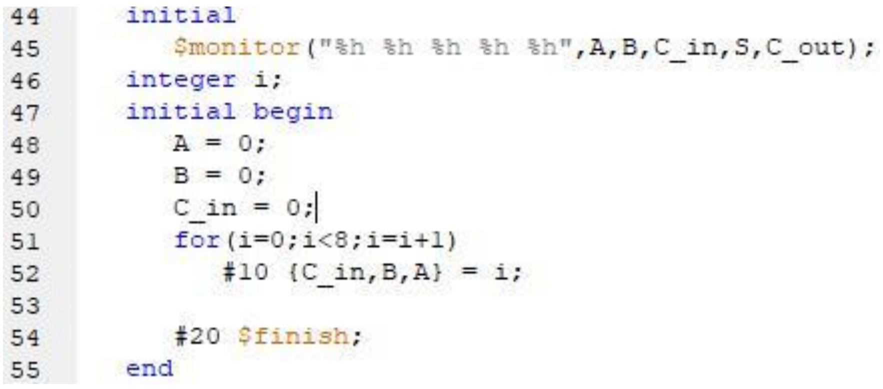
```
# run all
Simulator is doing circuit initialization process.
                   0
Finished circuit initialization process.
a-b=12710, C=1,S_out=2710
tb:NZCV=0010
sch:NZCV=0010
                   5
a+b=0c350, C=0,S_out=c350
tb:NZCV=1001
sch:NZCV=1001
                  10
a-b=10000, C=1,S_out=0000
tb:NZCV=0110
sch:NZCV=0110
                  15
a+b=03cc2, C=0,S_out=3cc2
tb:NZCV=0000
sch:NZCV=0000
                  20
a+b=08000, C=0,S_out=8000
tb:NZCV=1001
sch:NZCV=1001
                  25
a-b=17fff, C=1,S_out=7fff
tb:NZCV=0011
sch:NZCV=0011
                  30
a+b=08001, C=0,S_out=8001
tb:NZCV=1000
sch:NZCV=1000
Stopped at time : 35 ns : File "F:/  OTHER  /FPGA test/FPGA hw2/CPU 16b/ALU 16b tb.v" Line 48
ISim> |
```

ALU post-route



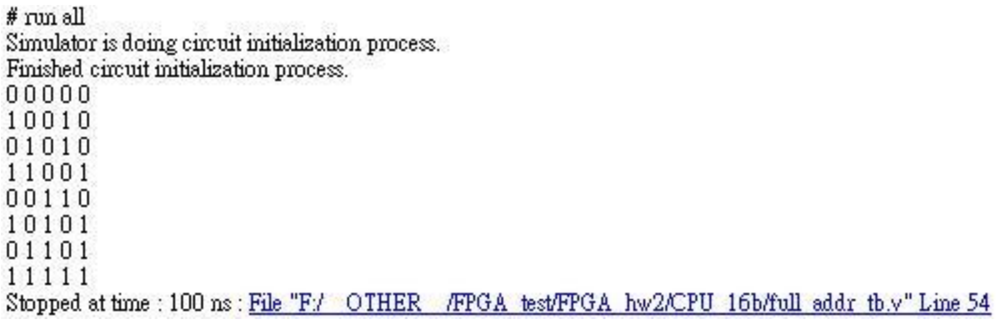
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/alu_16bits_timesim.sdf, for root module
                   0
Finished circuit initialization process.
a-b=12710, C=1,S_out=2710
tb:NZCV=0010
sch:NZCV=0010
                   5
a+b=0c350, C=0,S_out=c350
tb:NZCV=1001
sch:NZCV=1001
                  10
a-b=10000, C=1,S_out=0000
tb:NZCV=0110
sch:NZCV=0110
                  15
a+b=03cc2, C=0,S_out=3cc2
tb:NZCV=0000
sch:NZCV=0000
                  20
a+b=08000, C=0,S_out=8000
tb:NZCV=1001
sch:NZCV=1001
                  25
a-b=17fff, C=1,S_out=7fff
tb:NZCV=0011
sch:NZCV=0011
                  30
a+b=08001, C=0,S_out=8001
tb:NZCV=1000
sch:NZCV=1000
Stopped at time : 350 ns : File "F:/  OTHER  /FPGA test/FPGA hw2/CPU 16b/ALU 16b tb.v" Line 48
ISim> |
```

11. Full adder



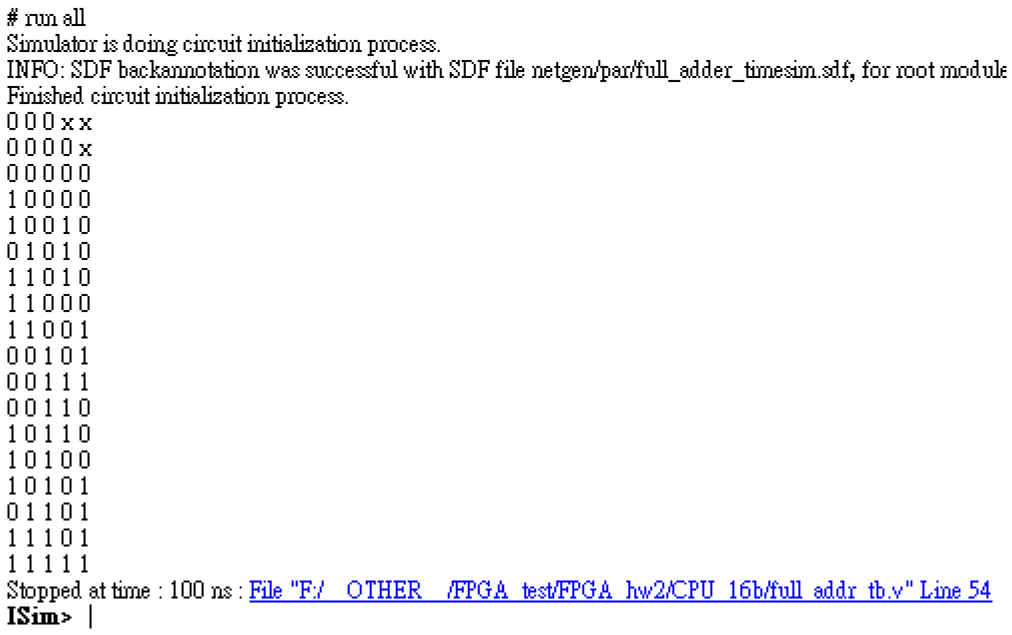
```
initial
   $monitor("%h %h %h %h %h",A,B,C_in,S,C_out);
integer i;
initial begin
   A = 0;
   B = 0;
   C_in = 0;
   for(i=0;i<8;i=i+1)
      #10 {C_in,B,A} = i;

   #20 $finish;
end
```

behavioral



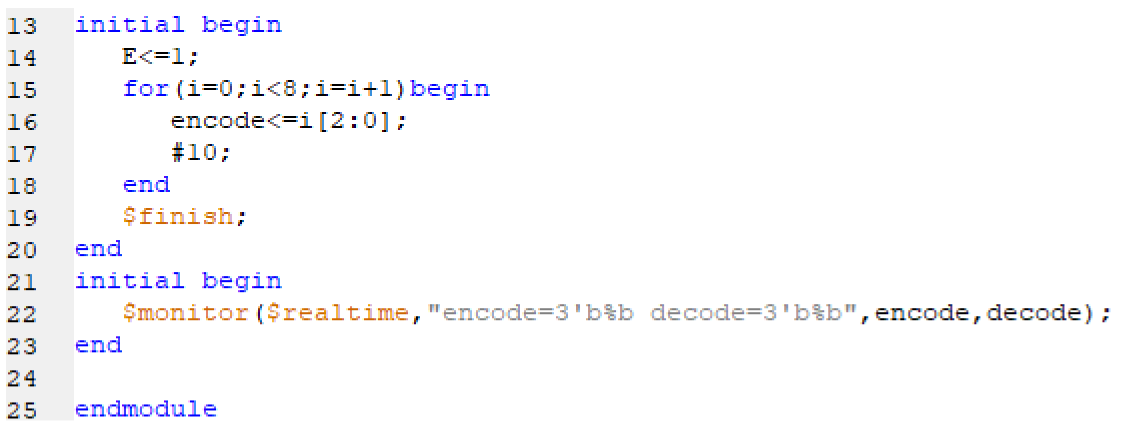
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
0 0 0 0 0
1 0 0 1 0
0 1 0 1 0
1 1 0 0 1
0 0 1 1 0
1 0 1 0 1
0 1 1 0 1
1 1 1 1 1
Stopped at time : 100 ns : File "F:/  OTHER  /FPGA test/FPGA hw2/CPU 16b/full addr tb.v" Line 54
```

post-route



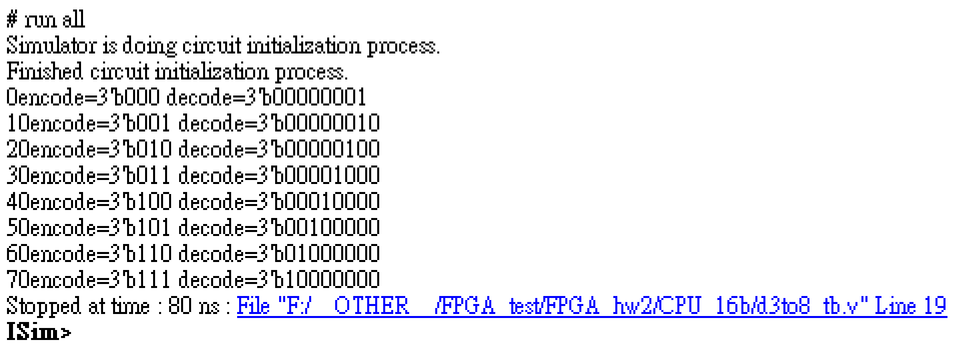
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/full_adder_timesim.sdf, for root module
Finished circuit initialization process.
0 0 0 x x
0 0 0 0 x
0 0 0 0 0
1 0 0 0 0
1 0 0 1 0
0 1 0 1 0
1 1 0 1 0
1 1 0 0 0
1 1 0 0 1
0 0 1 0 1
0 0 1 1 1
0 0 1 1 0
1 0 1 1 0
1 0 1 0 0
1 0 1 0 1
0 1 1 0 1
1 1 1 0 1
1 1 1 1 1
Stopped at time : 100 ns : File "F:/  OTHER  /FPGA test/FPGA hw2/CPU 16b/full addr tb.v" Line 54
ISim> |
```

12. 3to8 decoder



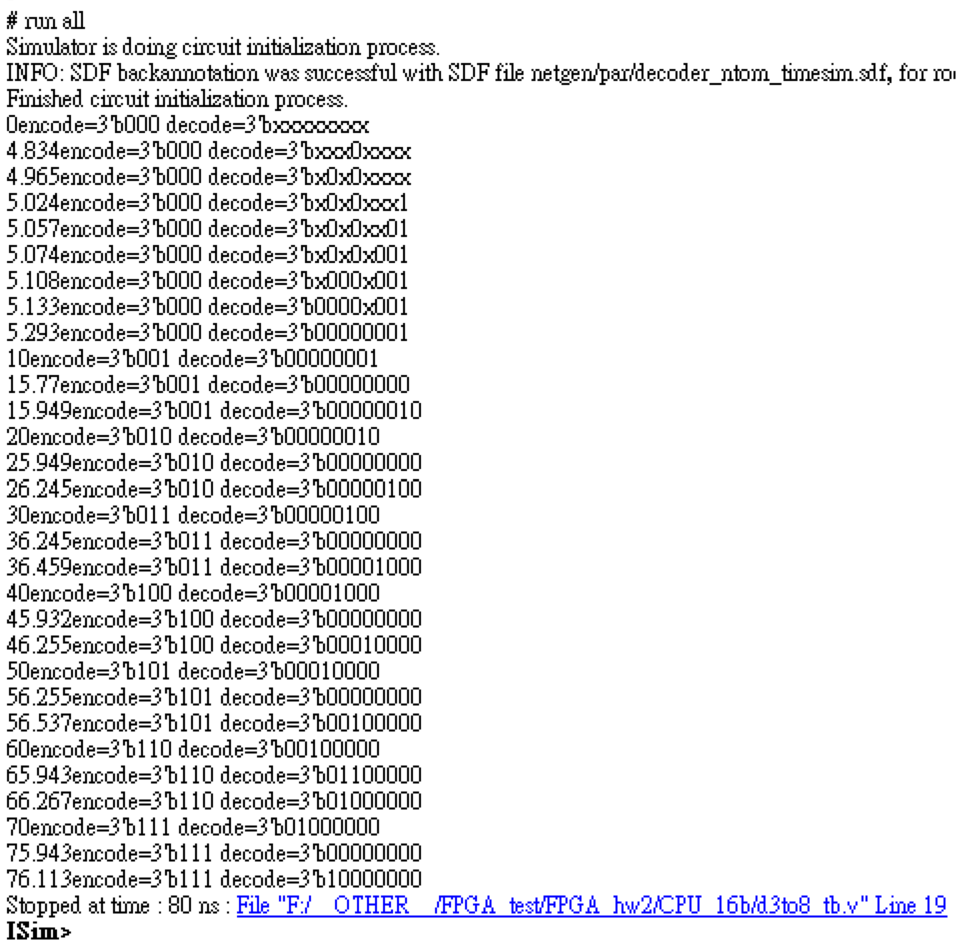
```verilog
initial begin
   E<=1;
   for(i=0;i<8;i=i+1)begin
      encode<=i[2:0];
      #10;
   end
   $finish;
end
initial begin
   $monitor($realtime,"encode=3'b%b decode=3'b%b",encode,decode);
end

endmodule
```

behavioral



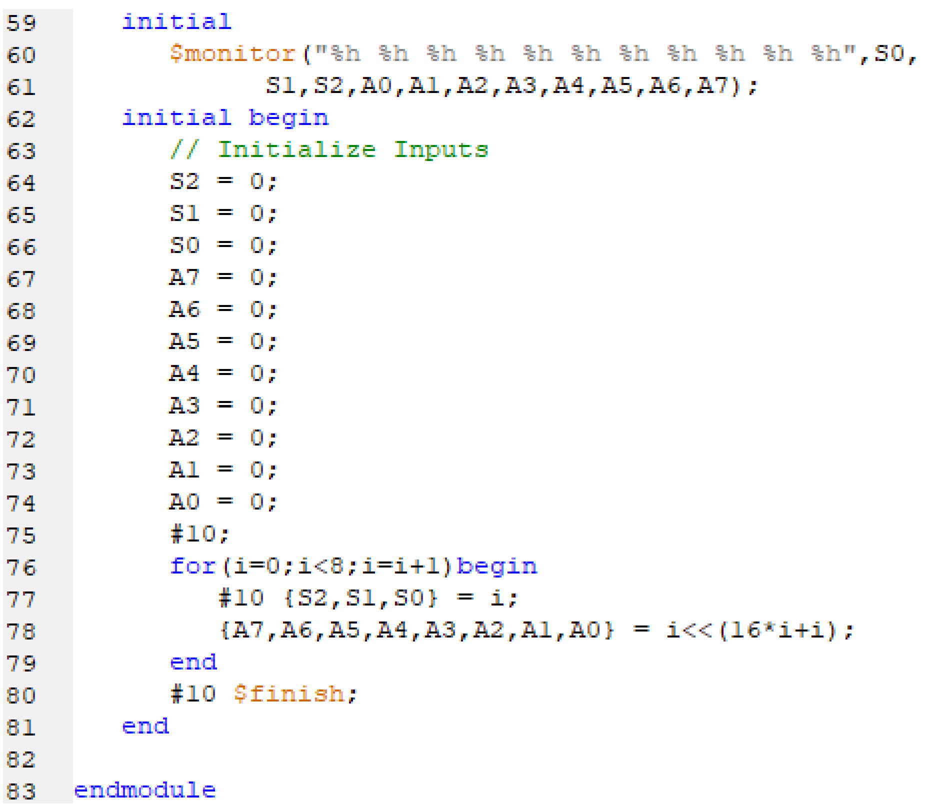
```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
0encode=3'b000 decode=3'b00000001
10encode=3'b001 decode=3'b00000010
20encode=3'b010 decode=3'b00000100
30encode=3'b011 decode=3'b00001000
40encode=3'b100 decode=3'b00010000
50encode=3'b101 decode=3'b00100000
60encode=3'b110 decode=3'b01000000
70encode=3'b111 decode=3'b10000000
Stopped at time : 80 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/d3to8_tb.v" Line 19
ISim>
```

post-route



```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/decoder_ntom_timesim.sdf, for ro
Finished circuit initialization process.
0encode=3'b000 decode=3'bxxxxxxxx
4.834encode=3'b000 decode=3'bxxx0xxxx
4.965encode=3'b000 decode=3'bx0x0xxxx
5.024encode=3'b000 decode=3'bx0x0xxx1
5.057encode=3'b000 decode=3'bx0x0xx01
5.074encode=3'b000 decode=3'bx0x0x001
5.108encode=3'b000 decode=3'bx000x001
5.133encode=3'b000 decode=3'b0000x001
5.293encode=3'b000 decode=3'b00000001
10encode=3'b001 decode=3'b00000001
15.77encode=3'b001 decode=3'b00000000
15.949encode=3'b001 decode=3'b00000010
20encode=3'b010 decode=3'b00000010
25.949encode=3'b010 decode=3'b00000000
26.245encode=3'b010 decode=3'b00000100
30encode=3'b011 decode=3'b00000100
36.245encode=3'b011 decode=3'b00000000
36.459encode=3'b011 decode=3'b00001000
40encode=3'b100 decode=3'b00001000
45.932encode=3'b100 decode=3'b00000000
46.255encode=3'b100 decode=3'b00010000
50encode=3'b101 decode=3'b00010000
56.255encode=3'b101 decode=3'b00000000
56.537encode=3'b101 decode=3'b00100000
60encode=3'b110 decode=3'b00100000
65.943encode=3'b110 decode=3'b01100000
66.267encode=3'b110 decode=3'b01000000
70encode=3'b111 decode=3'b01000000
75.943encode=3'b111 decode=3'b00000000
76.113encode=3'b111 decode=3'b10000000
Stopped at time : 80 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/d3to8_tb.v" Line 19
ISim>
```

13. 8to1 Multiplexer



```
    initial
       $monitor("%h %h %h %h %h %h %h %h %h %h %h",S0,
             S1,S2,A0,A1,A2,A3,A4,A5,A6,A7);
    initial begin
       // Initialize Inputs
       S2 = 0;
       S1 = 0;
       S0 = 0;
       A7 = 0;
       A6 = 0;
       A5 = 0;
       A4 = 0;
       A3 = 0;
       A2 = 0;
       A1 = 0;
       A0 = 0;
       #10;
       for(i=0;i<8;i=i+1)begin
          #10 {S2,S1,S0} = i;
          {A7,A6,A5,A4,A3,A2,A1,A0} = i<<(16*i+i);
       end
       #10 $finish;
    end

 endmodule
```

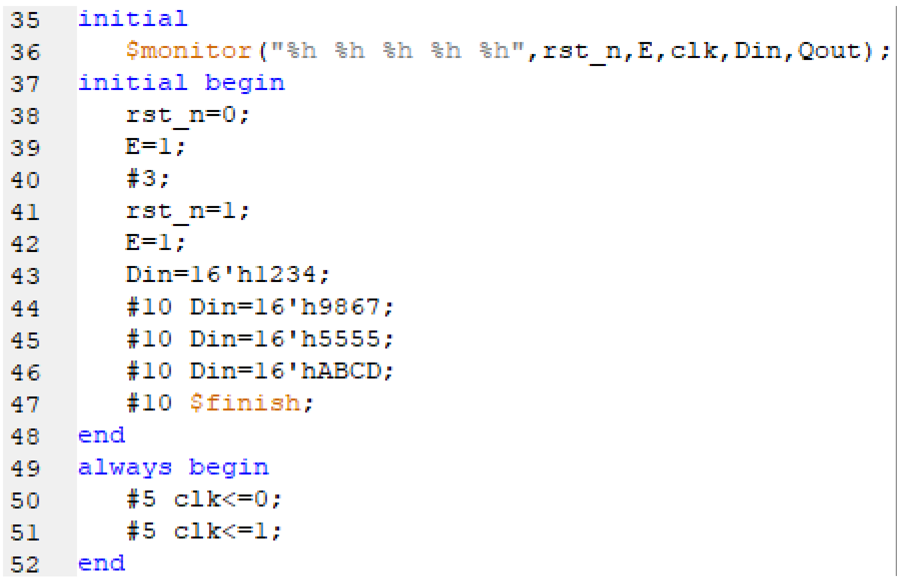
behavioral

```
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
0 0 0 0000 0000 0000 0000 0000 0000 0000 0000
1 0 0 0000 0002 0000 0000 0000 0000 0000 0000
0 1 0 0000 0000 0008 0000 0000 0000 0000 0000
1 1 0 0000 0000 0000 0018 0000 0000 0000 0000
0 0 1 0000 0000 0000 0000 0040 0000 0000 0000
1 0 1 0000 0000 0000 0000 0000 00a0 0000 0000
0 1 1 0000 0000 0000 0000 0000 0000 0180 0000
1 1 1 0000 0000 0000 0000 0000 0000 0000 0380
Stopped at time : 100 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/multiplexer_16bit_tb.v" Line 80
ISim>
```

post-route

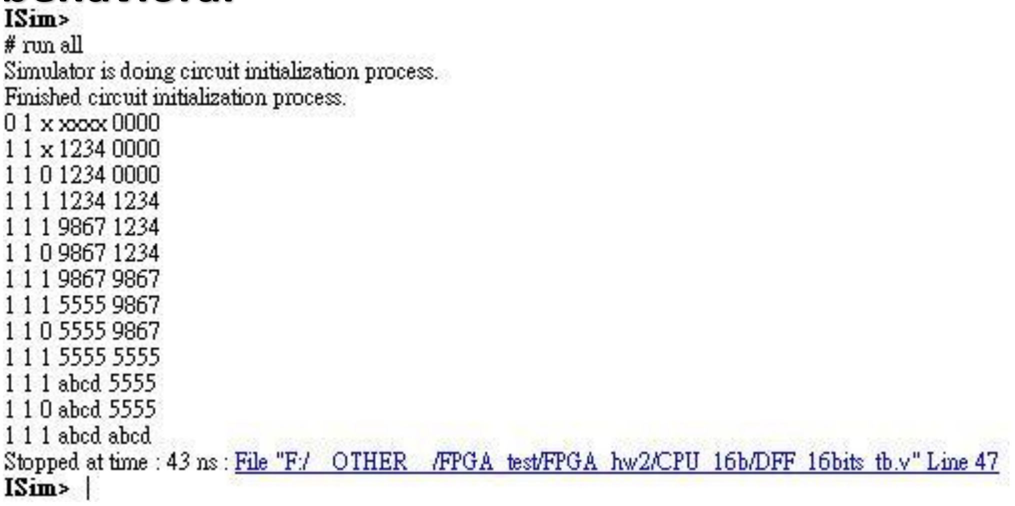
```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/multiplexer_8to1_16bits_timesim.sdf, for root r
Finished circuit initialization process.
0 0 0 0000 0000 0000 0000 0000 0000 0000 0000
1 0 0 0000 0002 0000 0000 0000 0000 0000 0000
0 1 0 0000 0000 0008 0000 0000 0000 0000 0000
1 1 0 0000 0000 0000 0018 0000 0000 0000 0000
0 0 1 0000 0000 0000 0000 0040 0000 0000 0000
1 0 1 0000 0000 0000 0000 0000 00a0 0000 0000
0 1 1 0000 0000 0000 0000 0000 0000 0180 0000
1 1 1 0000 0000 0000 0000 0000 0000 0000 0380
Stopped at time : 100 ns : File "F:/__OTHER__/FPGA_test/FPGA_hw2/CPU_16b/multiplexer_16bit_tb.v" Line 80
ISim>
```

14. D-Filp Flop 16bits



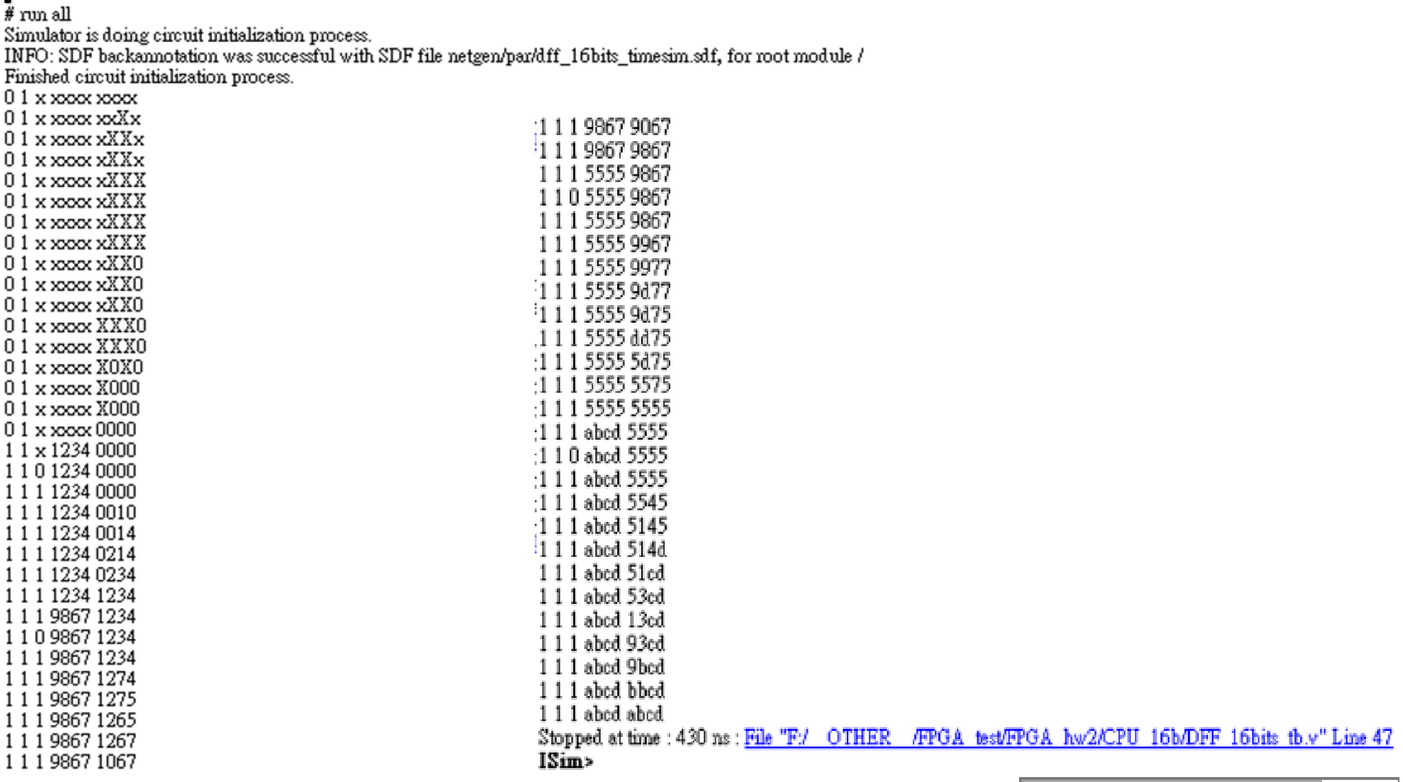
```
initial
    $monitor("%h %h %h %h %h",rst_n,E,clk,Din,Qout);
initial begin
    rst_n=0;
    E=1;
    #3;
    rst_n=1;
    E=1;
    Din=16'h1234;
    #10 Din=16'h9867;
    #10 Din=16'h5555;
    #10 Din=16'hABCD;
    #10 $finish;
end
always begin
    #5 clk<=0;
    #5 clk<=1;
end
```

behavioral



```
ISim>
# run all
Simulator is doing circuit initialization process.
Finished circuit initialization process.
0 1 x xxxx 0000
1 1 x 1234 0000
1 1 0 1234 0000
1 1 1 1234 1234
1 1 1 9867 1234
1 1 0 9867 1234
1 1 1 9867 9867
1 1 1 5555 9867
1 1 0 5555 9867
1 1 1 5555 5555
1 1 1 abcd 5555
1 1 0 abcd 5555
1 1 1 abcd abcd
Stopped at time : 43 ns : File "F:/   OTHER   /FPGA  test/FPGA  hw2/CPU  16b/DFF  16bits  tb.v" Line 47
ISim> |
```

post-route



```
# run all
Simulator is doing circuit initialization process.
INFO: SDF backannotation was successful with SDF file netgen/par/dff_16bits_timesim.sdf, for root module /
Finished circuit initialization process.
0 1 x xxxx xxxx
0 1 x xxxx xxXx
0 1 x xxxx xXXx
0 1 x xxxx xXXx
0 1 x xxxx xXXX
0 1 x xxxx xXXX
0 1 x xxxx xXXX
0 1 x xxxx xXXX
0 1 x xxxx xXX0
0 1 x xxxx xXX0
0 1 x xxxx xXX0
0 1 x xxxx XXX0
0 1 x xxxx XXX0
0 1 x xxxx X0X0
0 1 x xxxx X000
0 1 x xxxx X000
0 1 x xxxx 0000
1 1 x 1234 0000
1 1 0 1234 0000
1 1 1 1234 0000
1 1 1 1234 0010
1 1 1 1234 0014
1 1 1 1234 0214
1 1 1 1234 0234
1 1 1 1234 1234
1 1 1 9867 1234
1 1 0 9867 1234
1 1 1 9867 1234
1 1 1 9867 1274
1 1 1 9867 1275
1 1 1 9867 1265
1 1 1 9867 1267
1 1 1 9867 1067
```

```
1 1 1 9867 9067
1 1 1 9867 9867
1 1 1 5555 9867
1 1 0 5555 9867
1 1 1 5555 9867
1 1 1 5555 9967
1 1 1 5555 9977
1 1 1 5555 9d77
1 1 1 5555 9d75
1 1 1 5555 dd75
1 1 1 5555 5d75
1 1 1 5555 5575
1 1 1 5555 5555
1 1 1 abcd 5555
1 1 0 abcd 5555
1 1 1 abcd 5555
1 1 1 abcd 5545
1 1 1 abcd 5145
1 1 1 abcd 514d
1 1 1 abcd 51cd
1 1 1 abcd 53cd
1 1 1 abcd 13cd
1 1 1 abcd 93cd
1 1 1 abcd 9bcd
1 1 1 abcd bbcd
1 1 1 abcd abcd
Stopped at time : 430 ns : File "F:/   OTHER   /FPGA  test/FPGA  hw2/CPU  16b/DFF  16bits  tb.v" Line 47
ISim>
```

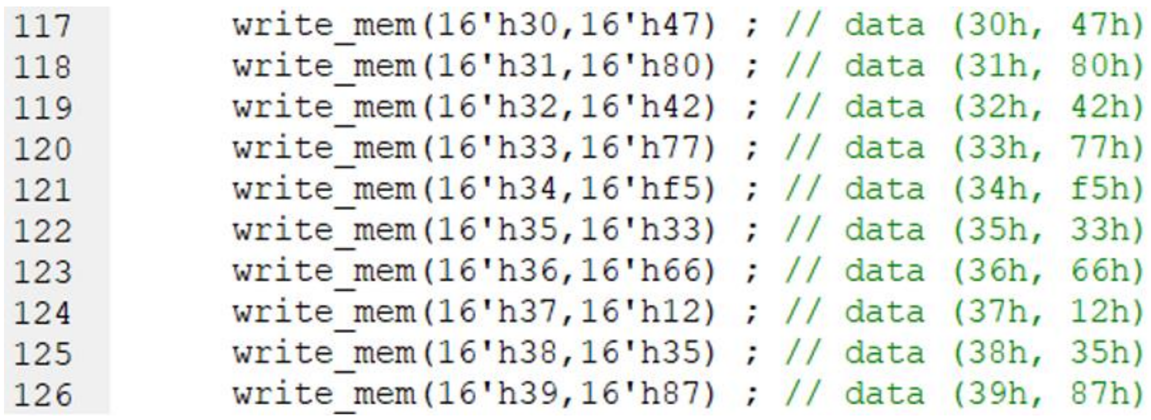
Complete Computer

Write a Verilog HDL module to describe the complete 16-bit RISC computer.

Write a test bench to make sure that your design is correct.

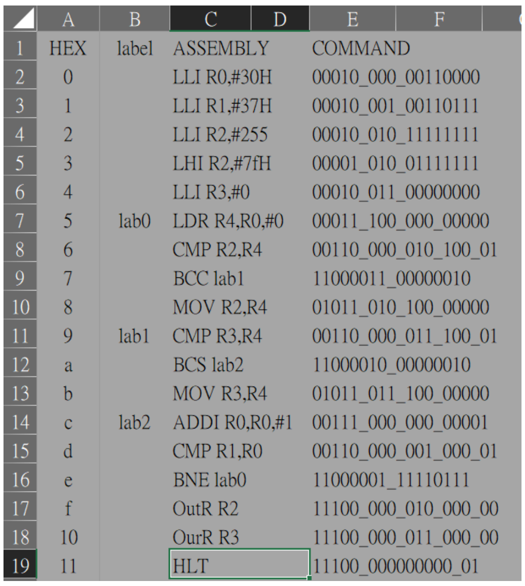
1. Find the minimum and maximum from two numbers in memory.
2. Add two numbers in memory and store the result in another memory location.
3. Add ten numbers in consecutive memory locations.
4. Mov a memory block of N words from one place to another.

除了程式機械碼，將在 Memory 30H~39H 間寫入以下資料，固定使用。



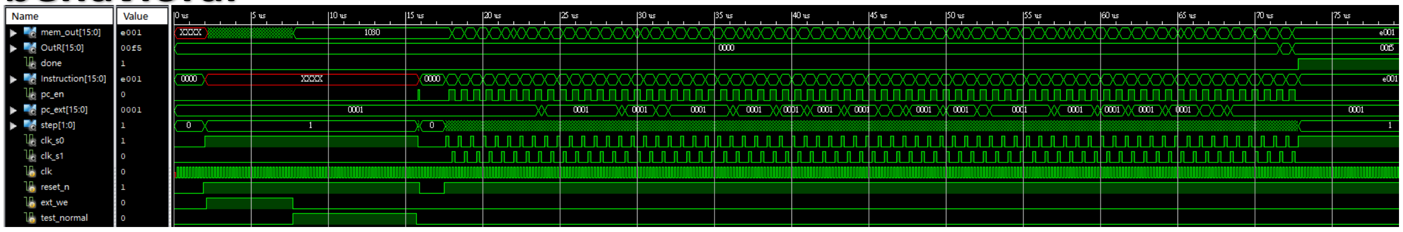
```
        write_mem(16'h30,16'h47) ; // data (30h, 47h)
        write_mem(16'h31,16'h80) ; // data (31h, 80h)
        write_mem(16'h32,16'h42) ; // data (32h, 42h)
        write_mem(16'h33,16'h77) ; // data (33h, 77h)
        write_mem(16'h34,16'hf5) ; // data (34h, f5h)
        write_mem(16'h35,16'h33) ; // data (35h, 33h)
        write_mem(16'h36,16'h66) ; // data (36h, 66h)
        write_mem(16'h37,16'h12) ; // data (37h, 12h)
        write_mem(16'h38,16'h35) ; // data (38h, 35h)
        write_mem(16'h39,16'h87) ; // data (39h, 87h)
```

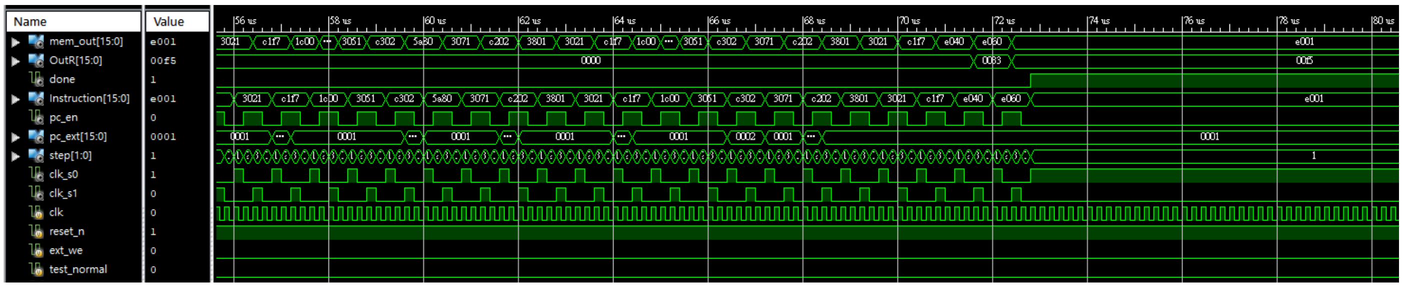
1. Find the minimum and maximum from two numbers in memoey



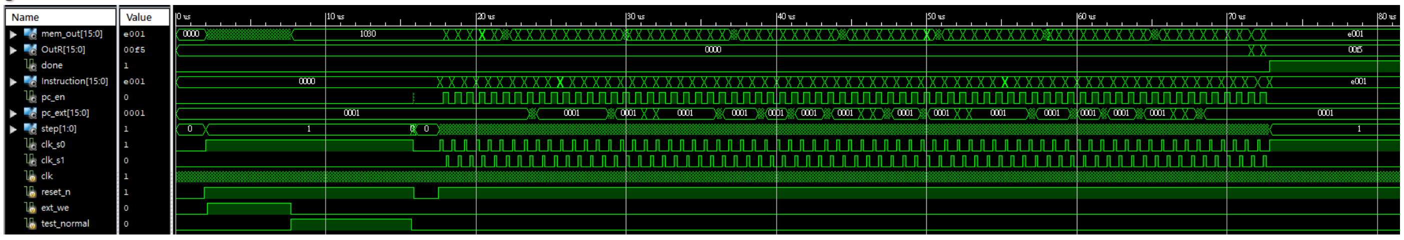
	A	B	C	D	E	F
1	HEX	label	ASSEMBLY		COMMAND	
2	0		LLI R0,#30H		00010_000_00110000	
3	1		LLI R1,#37H		00010_001_00110111	
4	2		LLI R2,#255		00010_010_11111111	
5	3		LHI R2,#7fH		00001_010_01111111	
6	4		LLI R3,#0		00010_011_00000000	
7	5	lab0	LDR R4,R0,#0		00011_100_000_00000	
8	6		CMP R2,R4		00110_000_010_100_01	
9	7		BCC lab1		11000011_00000010	
10	8		MOV R2,R4		01011_010_100_00000	
11	9	lab1	CMP R3,R4		00110_000_011_100_01	
12	a		BCS lab2		11000010_00000010	
13	b		MOV R3,R4		01011_011_100_00000	
14	c	lab2	ADDI R0,R0,#1		00111_000_000_00001	
15	d		CMP R1,R0		00110_000_001_000_01	
16	e		BNE lab0		11000001_11110111	
17	f		OutR R2		11100_000_010_000_00	
18	10		OurR R3		11100_000_011_000_00	
19	11		HLT		11100_000000000_01	

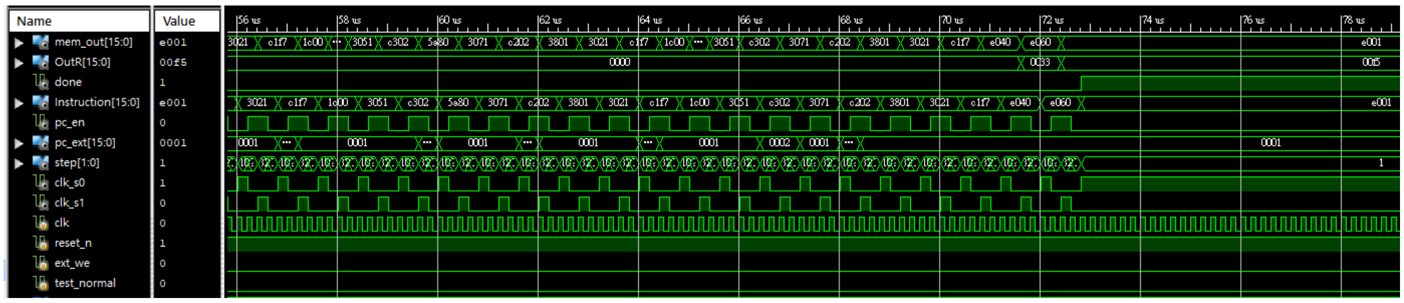
behavioral





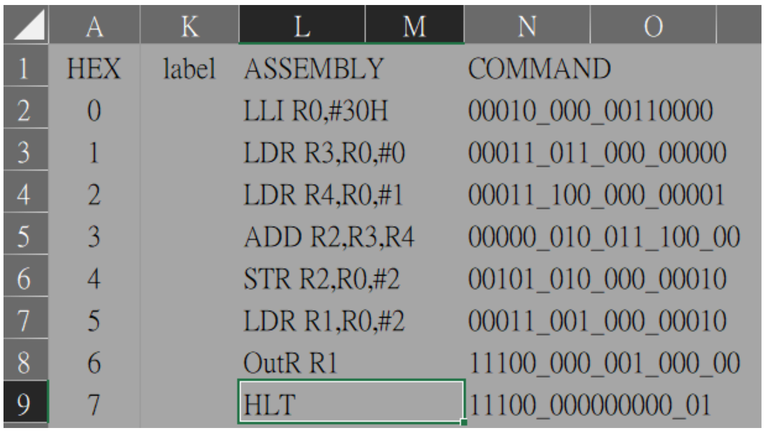
post-route





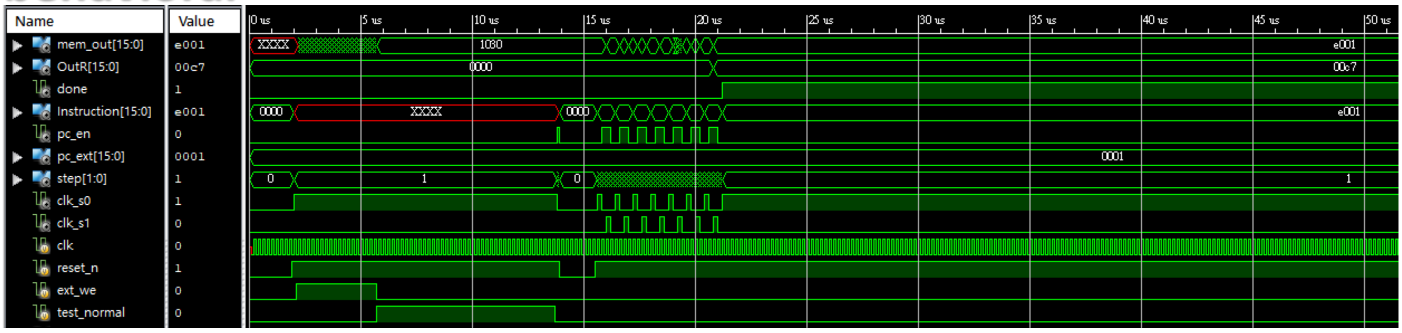
此程式將記憶體簡單讀 7 個 data 比較大小，47H，80H、42H、77H、F5H、33H、66H，做比大小，R2 為最小值，R3 為最大值，可以從波形圖看見 R2 為 33H，R3 為 F5H，結果正確。

2. Add two numbers in memory and store the result in memory location.

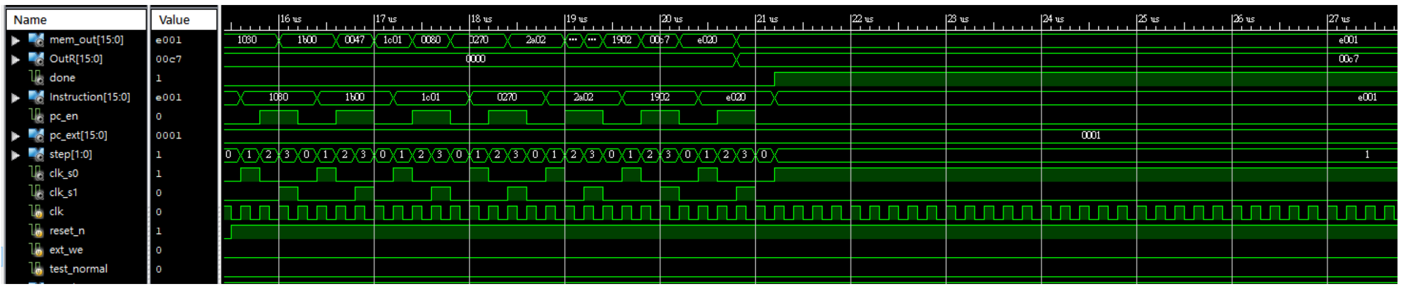


HEX	label	ASSEMBLY	COMMAND
0		LLI R0,#30H	00010_000_00110000
1		LDR R3,R0,#0	00011_011_000_00000
2		LDR R4,R0,#1	00011_100_000_00001
3		ADD R2,R3,R4	00000_010_011_100_00
4		STR R2,R0,#2	00101_010_000_00010
5		LDR R1,R0,#2	00011_001_000_00010
6		OutR R1	11100_000_001_000_00
7		HLT	11100_000000000_01

behavioral

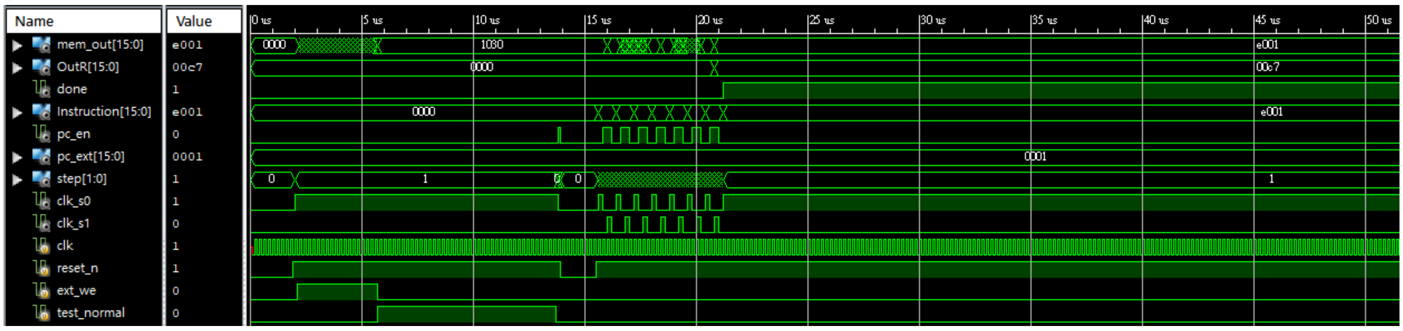


Name	Value
mem_out[15:0]	e001
OutR[15:0]	00c7
done	1
Instruction[15:0]	e001
pc_en	0
pc_ext[15:0]	0001
step[1:0]	1
clk_s0	1
clk_s1	0
clk	0
reset_n	1
ext_we	0
test_normal	0

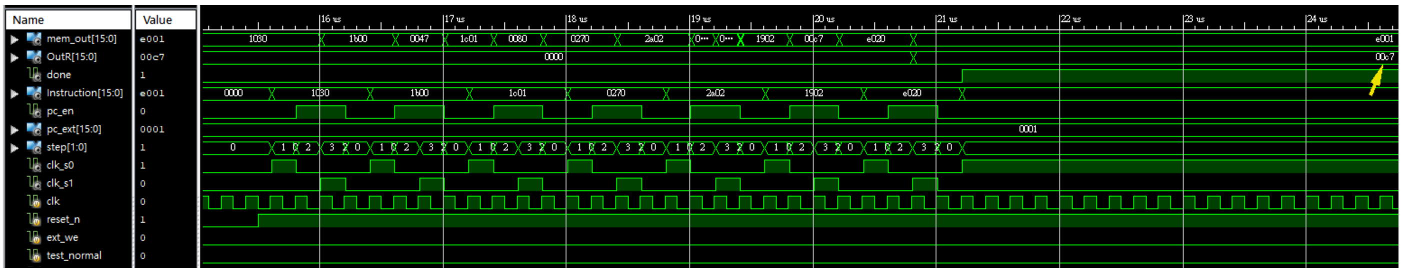


Name	Value
mem_out[15:0]	e001
OutR[15:0]	00c7
done	1
Instruction[15:0]	e001
pc_en	0
pc_ext[15:0]	0001
step[1:0]	1
clk_s0	1
clk_s1	0
clk	0
reset_n	1
ext_we	0
test_normal	0

post-route



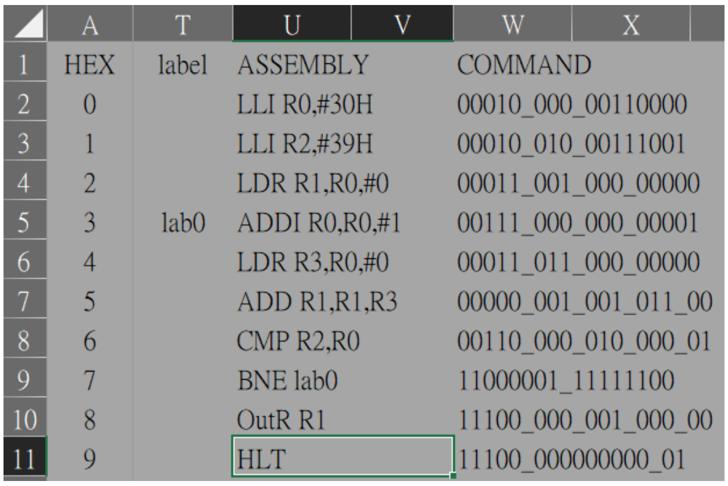
Name	Value
mem_out[15:0]	e001
OutR[15:0]	00c7
done	1
Instruction[15:0]	e001
pc_en	0
pc_ext[15:0]	0001
step[1:0]	1
clk_s0	1
clk_s1	0
clk	1
reset_n	1
ext_we	0
test_normal	0



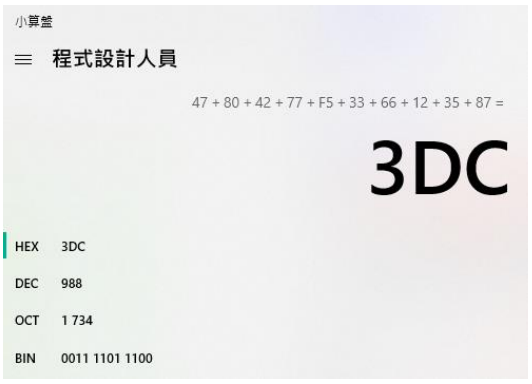
Name	Value
mem_out[15:0]	e001
OutR[15:0]	00c7
done	1
Instruction[15:0]	e001
pc_en	0
pc_ext[15:0]	0001
step[1:0]	1
clk_s0	1
clk_s1	0
clk	0
reset_n	1
ext_we	0
test_normal	0

此程式將記憶體讀兩個數值出來相加後存於另一個記憶體位置，將相加結果存入後，讀出並 OutR 觀看相加數值，可以從波形圖看見 47H add 80H，結果為 C7H，結果正確。

3. Add ten numbers in consecutive memory locations.



	A	T	U	V	W	X
1	HEX	label	ASSEMBLY		COMMAND	
2	0		LLI R0,#30H		00010_000_00110000	
3	1		LLI R2,#39H		00010_010_00111001	
4	2		LDR R1,R0,#0		00011_001_000_00000	
5	3	lab0	ADDI R0,R0,#1		00111_000_000_00001	
6	4		LDR R3,R0,#0		00011_011_000_00000	
7	5		ADD R1,R1,R3		00000_001_001_011_00	
8	6		CMP R2,R0		00110_000_010_000_01	
9	7		BNE lab0		11000001_11111100	
10	8		OutR R1		11100_000_001_000_00	
11	9		HLT		11100_000000000_01	



小算盤

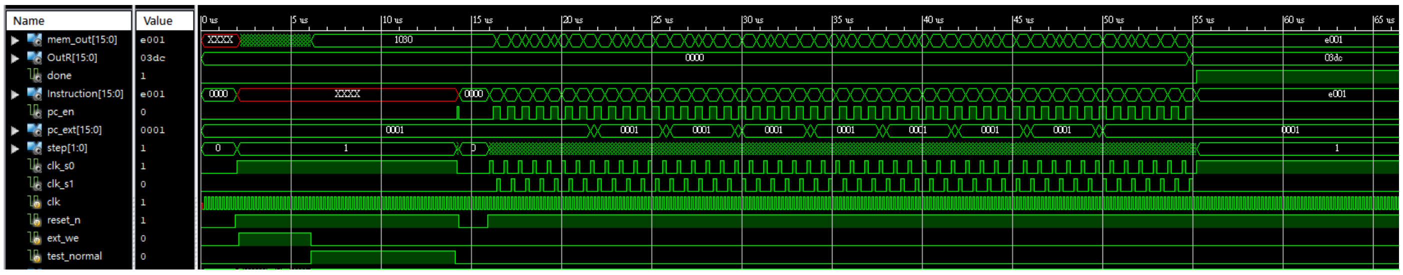
程式設計人員

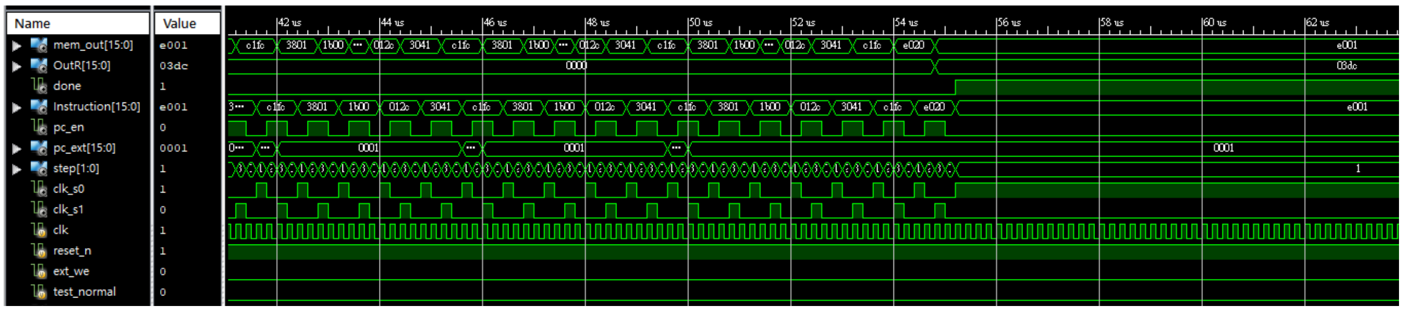
47 + 80 + 42 + 77 + F5 + 33 + 66 + 12 + 35 + 87 =

3DC

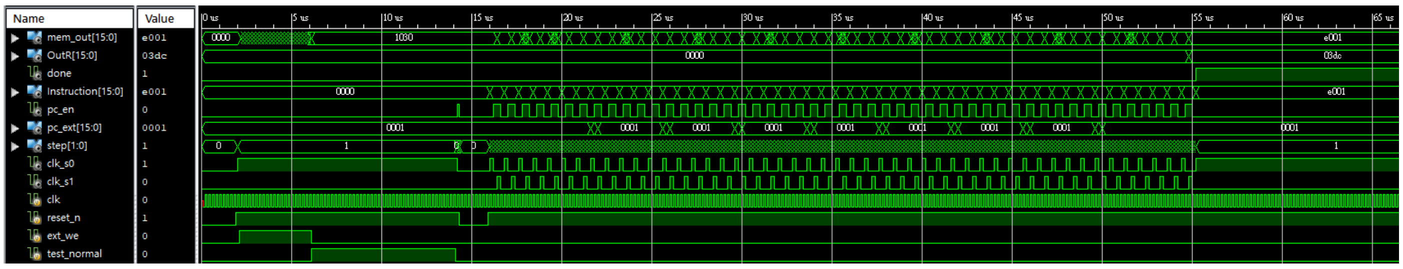
HEX	3DC
DEC	988
OCT	1 734
BIN	0011 1101 1100

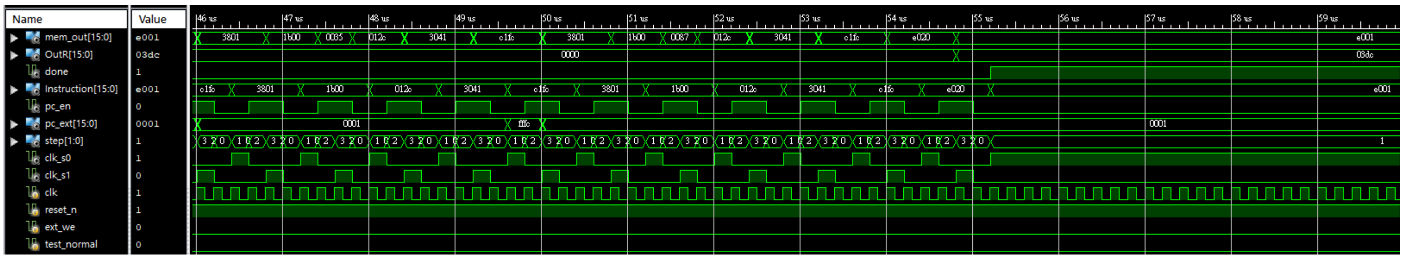
behavioral





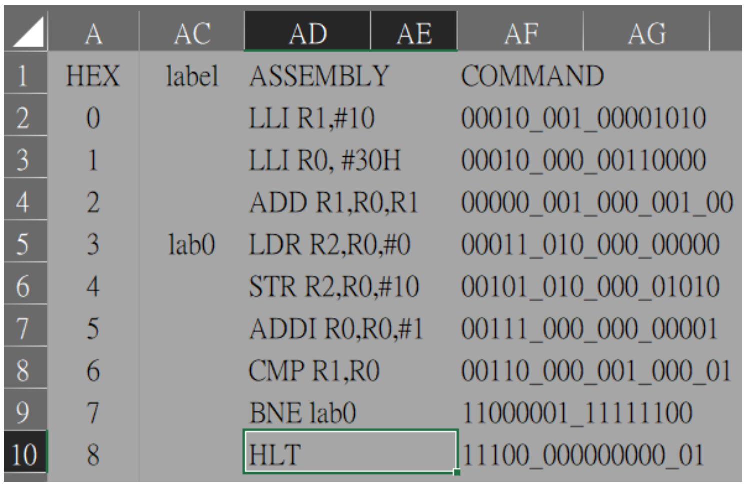
post-route





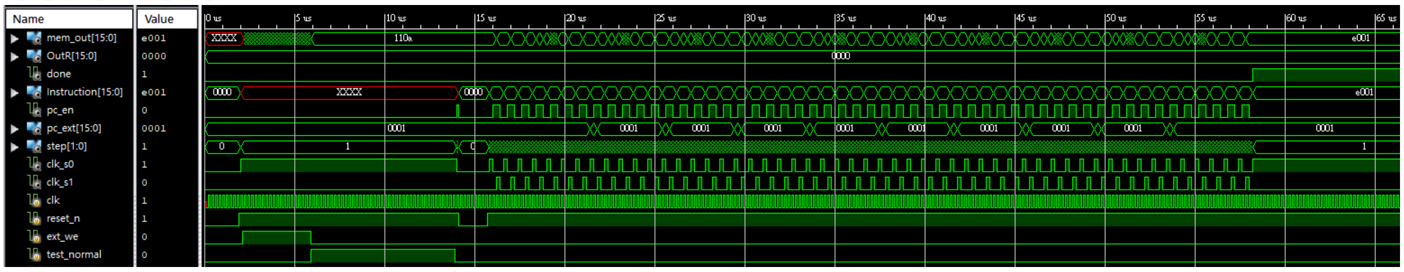
此程式將記憶體裡的十個數字做相加，47H、80H、42H、77H、F5H、33H、66H、12H、35H、87H 的相加結果為 3DCH，可從波形圖看見結果正確。

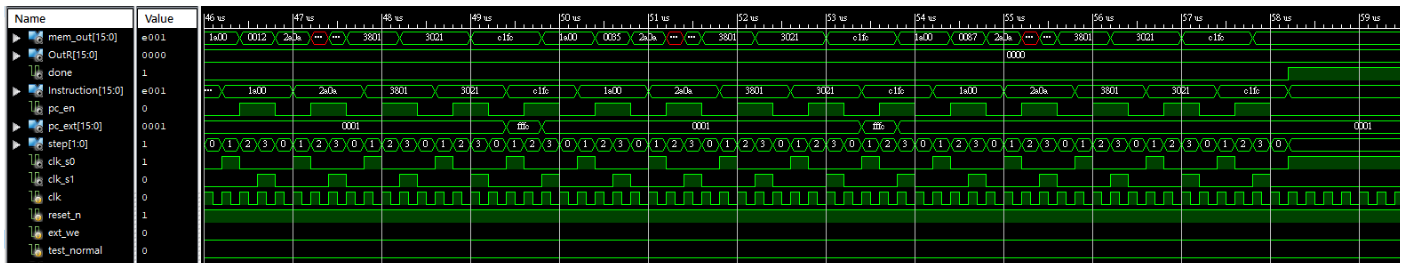
4. Mov a memory block of N words from one place to another.



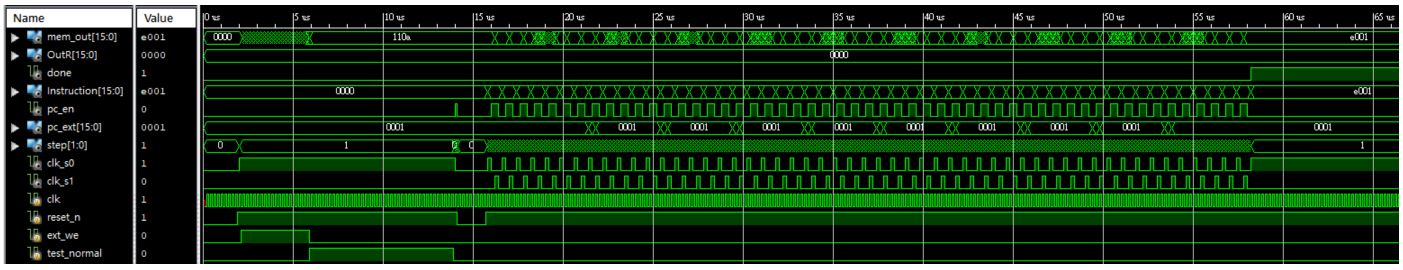
	HEX	label	ASSEMBLY	COMMAND
2	0		LLI R1,#10	00010_001_00001010
3	1		LLI R0, #30H	00010_000_00110000
4	2		ADD R1,R0,R1	00000_001_000_001_00
5	3	lab0	LDR R2,R0,#0	00011_010_000_00000
6	4		STR R2,R0,#10	00101_010_000_01010
7	5		ADDI R0,R0,#1	00111_000_000_00001
8	6		CMP R1,R0	00110_000_001_000_01
9	7		BNE lab0	11000001_11111100
10	8		HLT	11100_000000000_01

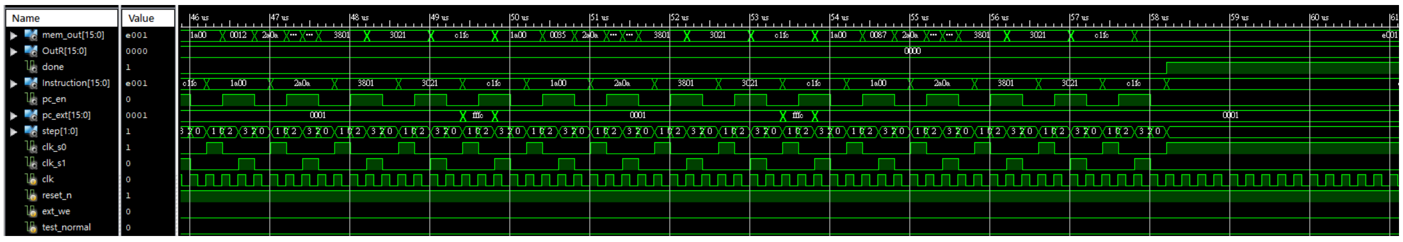
behavioral





post-route





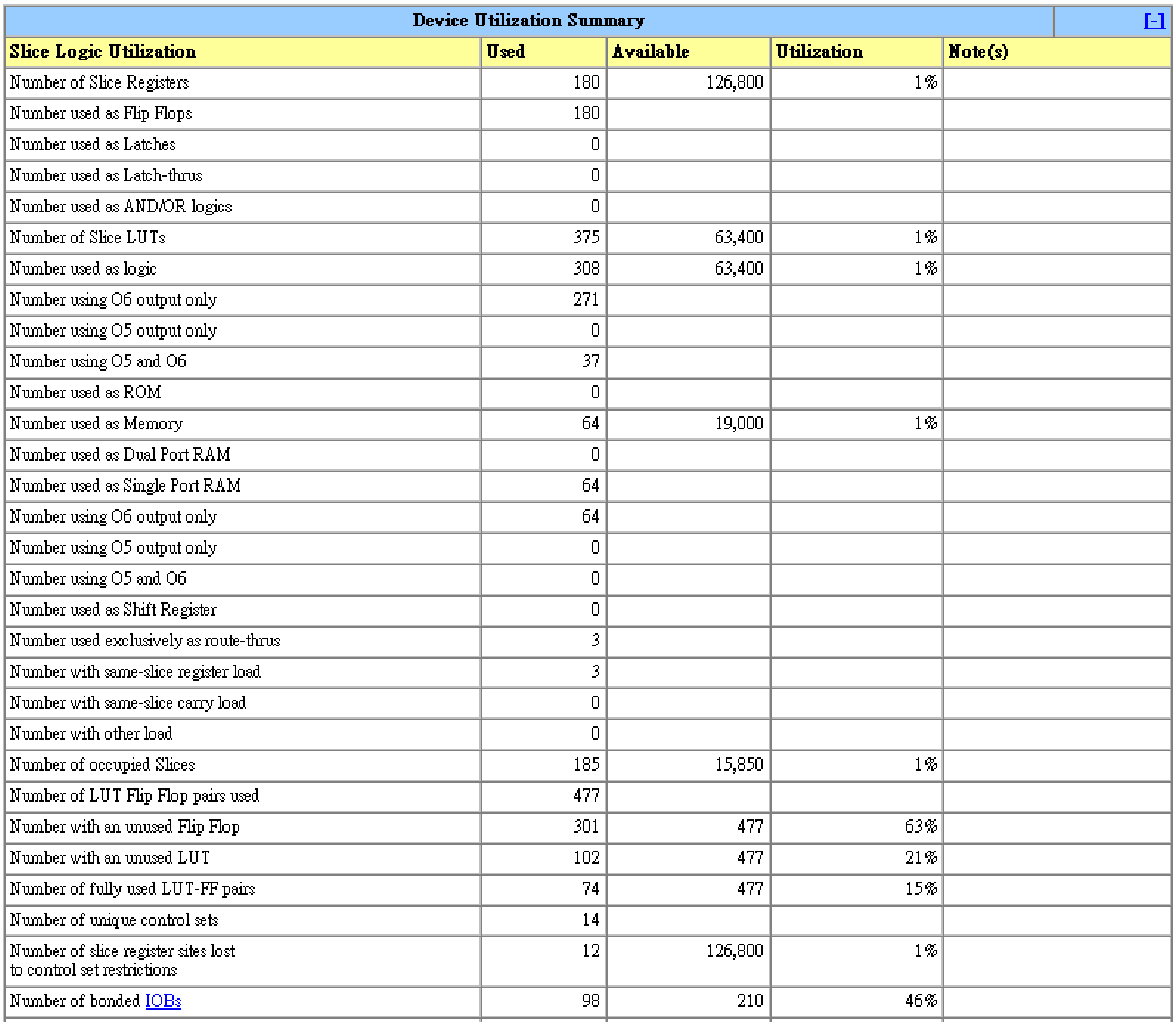
此程式將記憶體中 N 個 word 當作 block 移動到其他位置，我讓組語 R1 為 N=10，波形圖可以看見 Instruction 的重複變化，來觀看確實完成正確的讀取寫入的迴圈。

Hardware Cost

Total number of LUTs **375**

Total number of FFs **180**

16-bits RISC CPU hardware cost



Device Utilization Summary				[-]
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	180	126,800	1%	
Number used as Flip Flops	180			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	375	63,400	1%	
Number used as logic	308	63,400	1%	
Number using O6 output only	271			
Number using O5 output only	0			
Number using O5 and O6	37			
Number used as ROM	0			
Number used as Memory	64	19,000	1%	
Number used as Dual Port RAM	0			
Number used as Single Port RAM	64			
Number using O6 output only	64			
Number using O5 output only	0			
Number using O5 and O6	0			
Number used as Shift Register	0			
Number used exclusively as route-thrus	3			
Number with same-slice register load	3			
Number with same-slice carry load	0			
Number with other load	0			
Number of occupied Slices	185	15,850	1%	
Number of LUT Flip Flop pairs used	477			
Number with an unused Flip Flop	301	477	63%	
Number with an unused LUT	102	477	21%	
Number of fully used LUT-FF pairs	74	477	15%	
Number of unique control sets	14			
Number of slice register sites lost to control set restrictions	12	126,800	1%	
Number of bonded IOBs	98	210	46%	

Discussion

1. Design entry

   利用期中所完成的 Schematic 部分以 Verilog HDL 實現,但是因為我在期中作業 Schematic 部分,沒注意到 run post-route，而在執行下去後發現，波形整個 crush 了，因避免與上一次製作 Schematic 電路時一樣情況發生，我重新反省我的時序控制，避免資料 metastable，並在此期末作業 Verilog HDL 部份成功執行獲得正確結果。

2. Schematic & HDL

   我在修此門 FPGA 系統設計實務中了解到，撰寫 HDL 時，必須了解每一行或每一段都將合成出什麼樣的電路，而此門課中我意識到我在以前寫 Verilog 的時候沒有去確定所有合成出的電路樣子，在做此期末作業的時候因為有期中作業的完成,在寫 Verilog 的時候就非常清楚在描述什麼電路,而功能也正常運行。不過我的總 Hardware Cost 並不相同，這讓我感到困惑，也讓我自己了解我是在騙自己 HDL 的都跟 Schematic 部份一樣，而實質上不同，所以我還需要繼續精進能力才行。

3. Progress

   在獲得錯誤的模擬結果時，我更加的快速在一堆錯誤訊號找到問題點，我想這是不斷的練習所獲得的成果。在期中 Schematic 部分發現 post-route 結果無法正確執行時我非常的低落，在大學依靠程式能力拿取高分的我，竟然無法做好與程式相關的課程，不過在整學期的課程中，了解到很多眉角，必須好好謝謝老師本學期的教學。