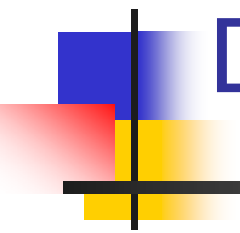


EE4313 COMPUTER ENGINEERING DESIGN PROJECT - I

Lab 2

Lab 2: RC5 Encryption (partial implementation)

- Round keys are fixed and given
  - S[2],…, S[25]
- Pre-round not included
- 64-bit data input, 64-bit output

```
~~A = A + S[0];~~
~~B = B + S[1];~~
for i=1 to 12 do
        A = ((A XOR B) <<< B) + S[2*i];
        B = ((B XOR A) <<< A) + S[2*1+1];
```

Entity Definition

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;   -- we will use CONV_INTEGER

ENTITY rc5_enc IS
    PORT
    (
            clr         : IN STD_LOGIC;  -- Asynchronous reset
            clk         : IN STD_LOGIC;  -- Clock signal
            din         : IN STD_LOGIC_VECTOR(63 DOWNTO 0);  -- 64-bit input
            dout        : OUT STD_LOGIC_VECTOR(63 DOWNTO 0) -- 64-bit output
     );
END rc5_enc;
```

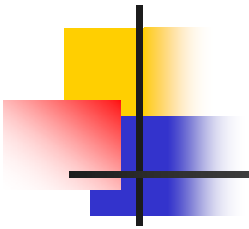
Internal Signals and Round Keys

```
ARCHITECTURE rtl OF rc5_enc IS
    SIGNAL i_cnt     : STD_LOGIC_VECTOR(3 DOWNTO 0); -- round counter

    SIGNAL ab_xor  : STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL a_rot     : STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL a         : STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL a_reg     : STD_LOGIC_VECTOR(31 DOWNTO 0); -- register A

    SIGNAL ba_xor: STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL b_rot   : STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL b         : STD_LOGIC_VECTOR(31 DOWNTO 0);
    SIGNAL b_reg   : STD_LOGIC_VECTOR(31 DOWNTO 0); -- register B

    -- define a type for round keys
    TYPE rom IS ARRAY (2 TO 25) OF STD_LOGIC_VECTOR(31 DOWNTO 0);
    CONSTANT skey : rom:=rom'(X"46F8E8C5", X"460C6085", ..., XX"65046380");
```

Round Operations

```
BEGIN
    -- A=((A XOR B)<<<B) + S[2*i];
    ab_xor <= a_reg XOR b_reg;
    WITH b_reg(4 DOWNTO 0) SELECT
        a_rot<=ab_xor(30 DOWNTO 0) & ab_xor(31) WHEN "00001",
                    ……
                    ab_xor WHEN OTHERS;
    a<=a_rot + skey(CONV_INTEGER(i_cnt & '0'));  -- S[2*i]

    -- B=((B XOR A) <<<A)   + S[2*i+1]
    ab1_xor <= b_reg XOR a;
    WITH a(4 DOWNTO 0) SELECT
        b_rot<=ab1_xor(30 DOWNTO 0) & ab1_xor(31) WHEN "00001",
                    ……
                    ab1_xor WHEN OTHERS;
    b<=b_rot + skey(CONV_INTEGER(i_cnt & '1'));  -- S[2*i+1]
```

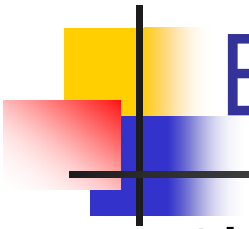
Sequential Circuits



```
-- A register
PROCESS(clr, clk)  BEGIN
    IF(clr='0') THEN
       a_reg<=din(63 DOWNTO 32);
    ELSIF(clk'EVENT AND clk='1') THEN
       a_reg<=a;
    END IF;
END PROCESS;

-- B register
PROCESS(clr, clk)  BEGIN
    IF(clr='0') THEN
       b_reg<=din(31 DOWNTO 0);
    ELSIF(clk'EVENT AND clk='1') THEN
       b_reg<=b;
    END IF;
END PROCESS;
```

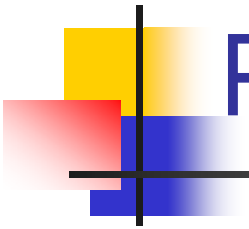
Sequential Circuits

```
    -- round counter
    PROCESS(clr, clk)  BEGIN
        IF(clr='0') THEN
           i_cnt<="0001";
        ELSIF(clk'EVENT AND clk='1') THEN
           IF(i_cnt="1100") THEN
              i_cnt<="0001";
           ELSE
              i_cnt<=i_cnt+'1';
           END IF;
        END IF;
    END PROCESS;

    dout<=a_reg & b_reg;
END rtl;
```



Exercise 1

- Simulate RC5 encryption VHDL Model
  - Functional Simulation using ModelSim
  - Code debugging
- Synthesize the Design
  - Perform Timing Simulation
- Understand following concepts/operations
  - How to compute *2, *2+1
  - CONV_INTEGER
  - Why we need sequential circuits?



Functional Simulation

- Create project folder
  - C:\ee4313\proj2
- Save your VHDL code in project folder
  - rc5_enc.vhd
- Start ModelSim

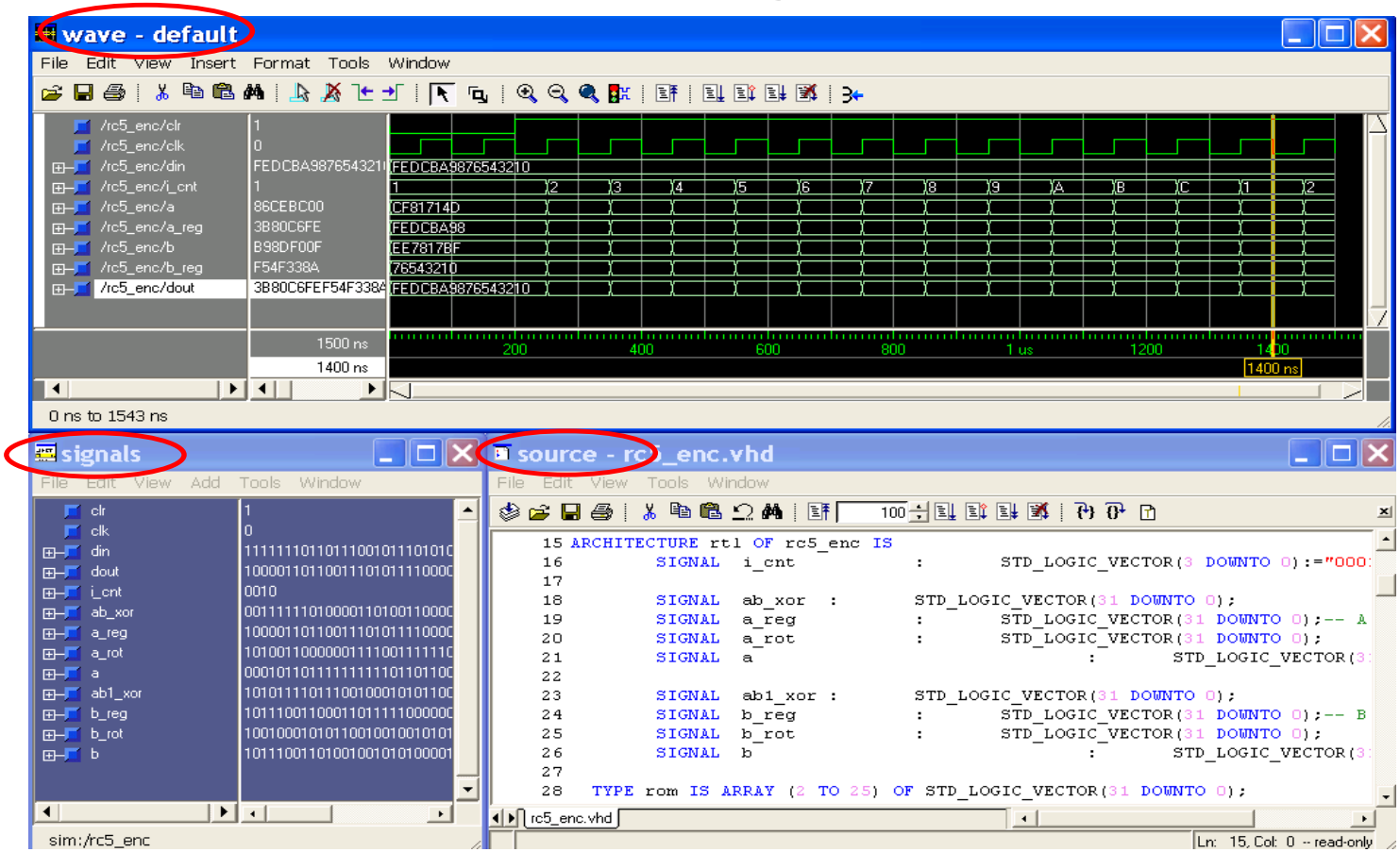
Functional Simulation

- In ModelSim console window, type following commands in sequence

```
cd {c:/ee4313/proj2}
vlib work
vcom rc5_enc.vhd -93                          -- enable VHDL'93 support
vsim rc5_enc
view wave
view source
view signal
add wave clr clk din dout a a_reg b b_reg     -- you can view internal signals
force clr 0 0, 1 200                          -- Asynchronous reset
force clk 0 0, 1 50 –repeat 100               -- clock signal
force din 16#FEDCBA9876543210 0
run 1500
```

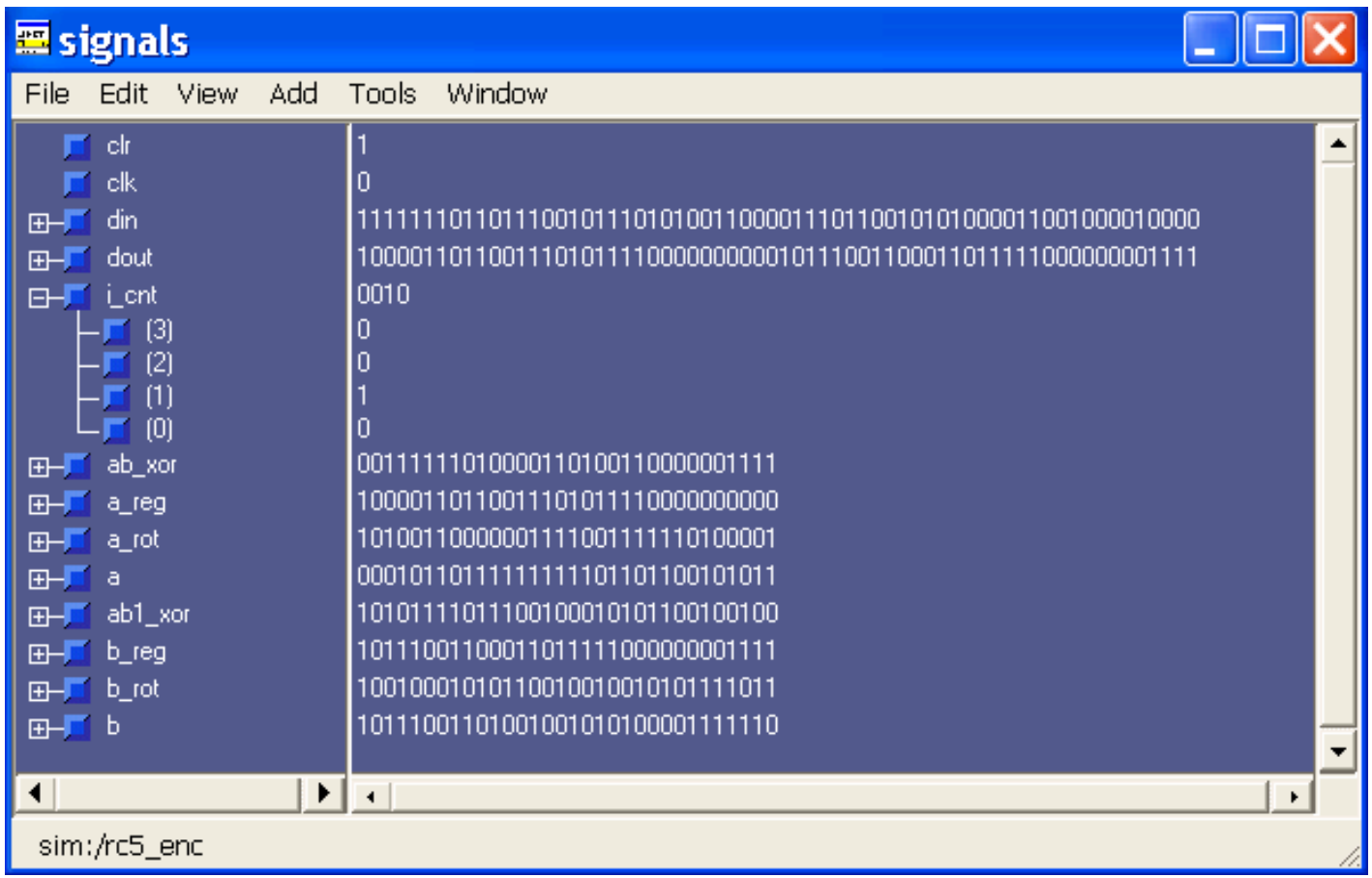
Functional Simulation

- Two more windows – signals and source

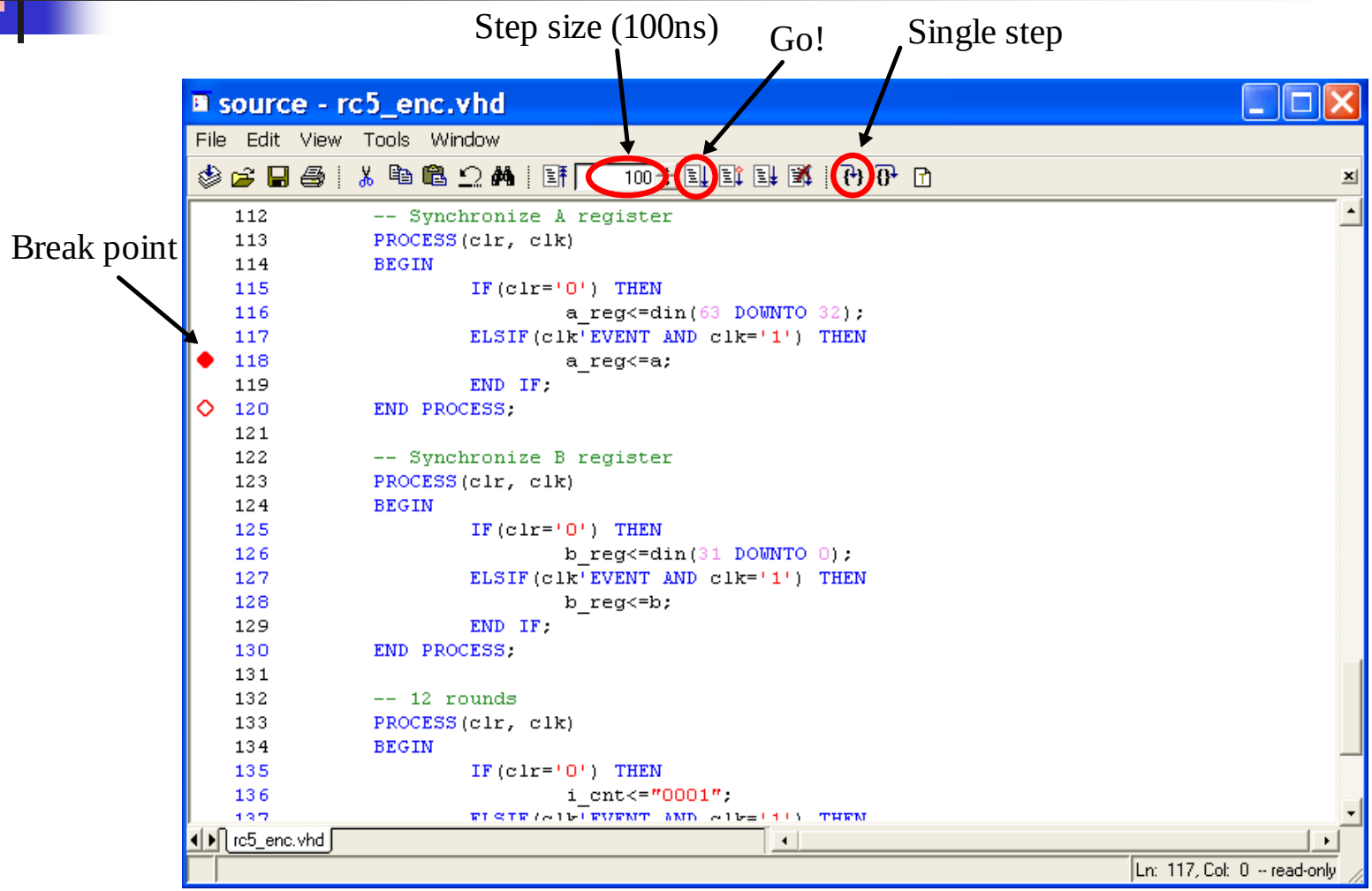


Functional Simulation

- signals window – list dynamic value of signals



Functional Simulation

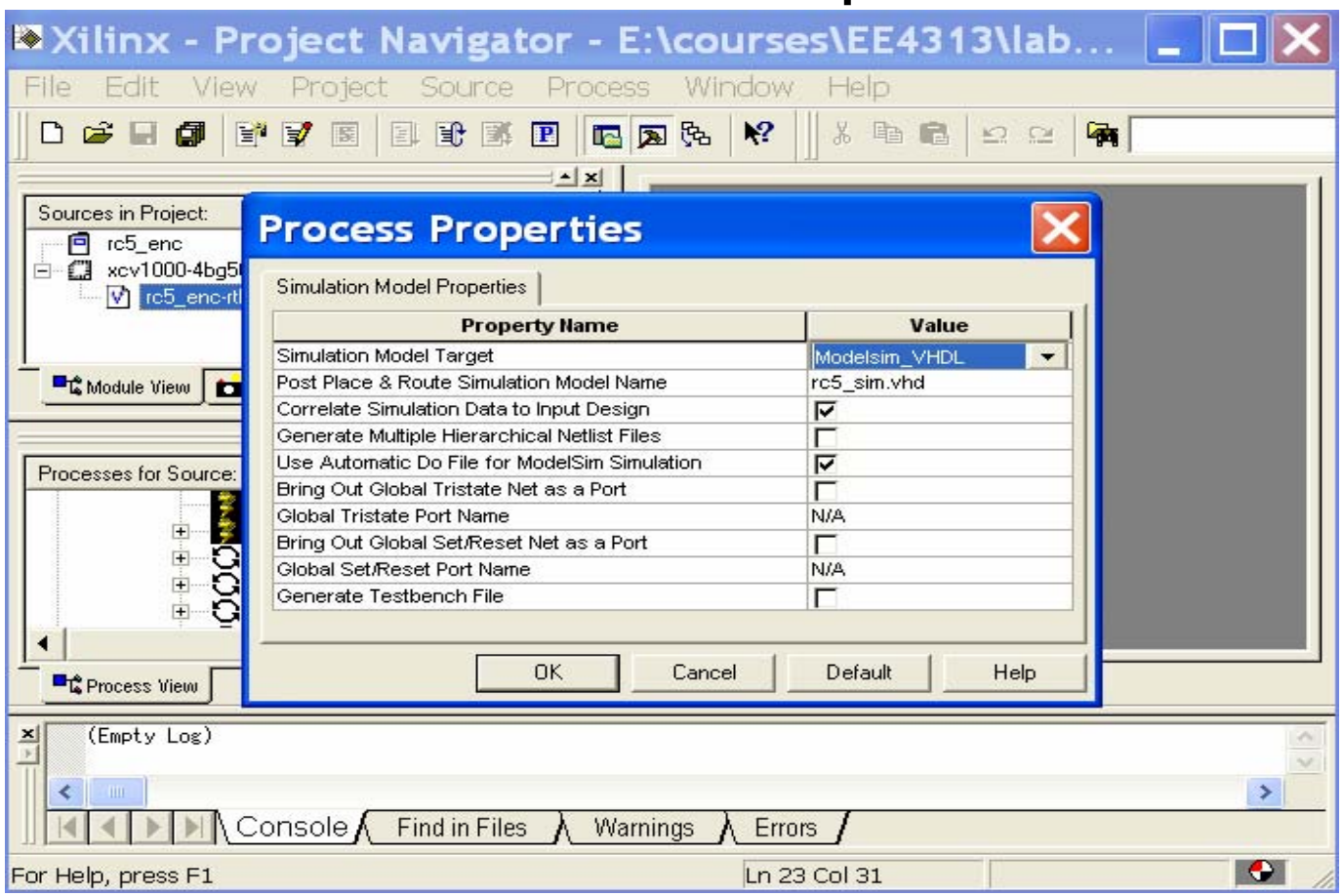


FPGA Synthesis and Implementation

- Start Xilinx Project Navigator
- Create new project
  - Project name: rc5_enc
  - Device: Virtex->xcv1000->bg560->-4
- Create timing simulation model
  - rc5_sim.vhd

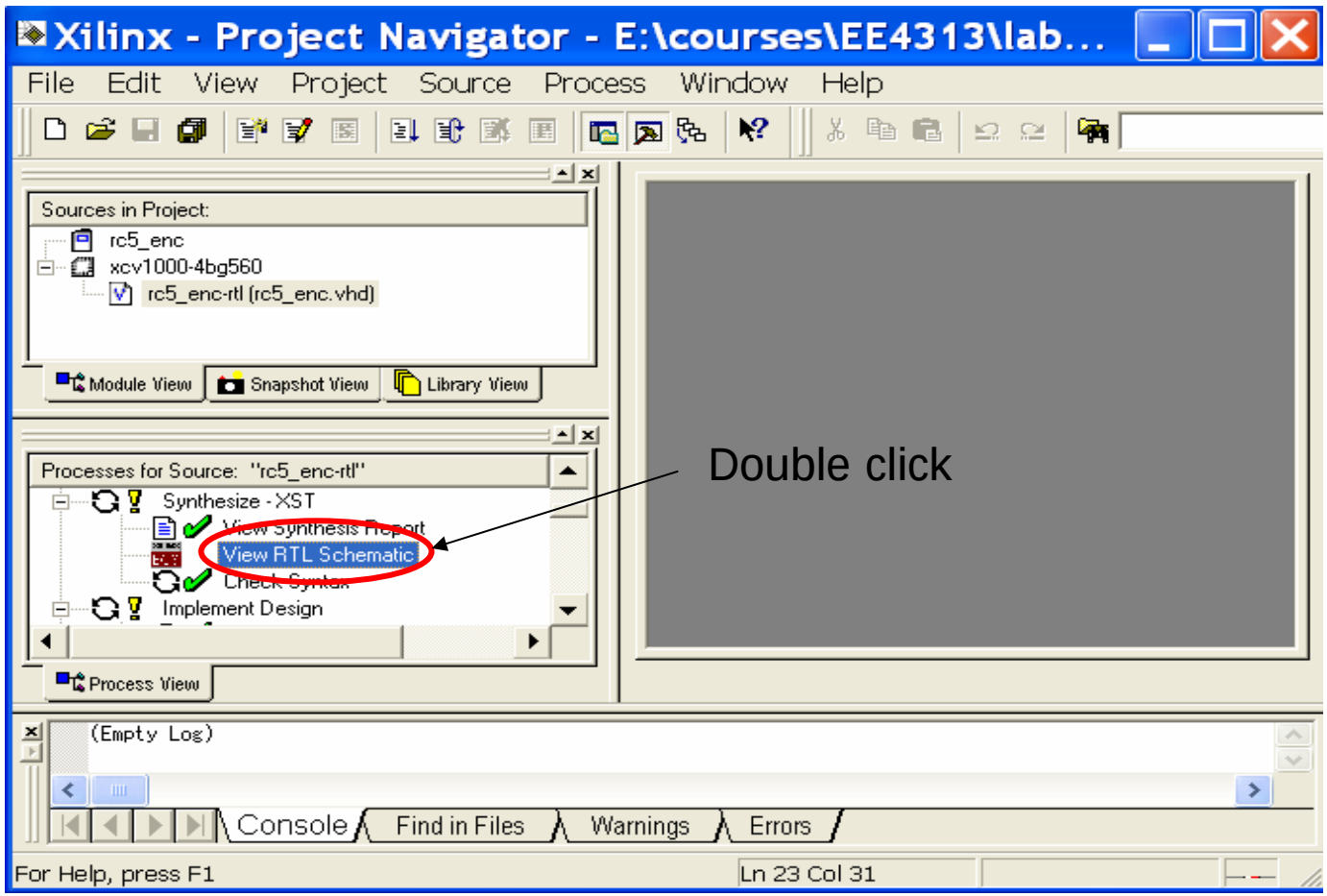
FPGA Synthesis

- Next>Simulation Model Properties



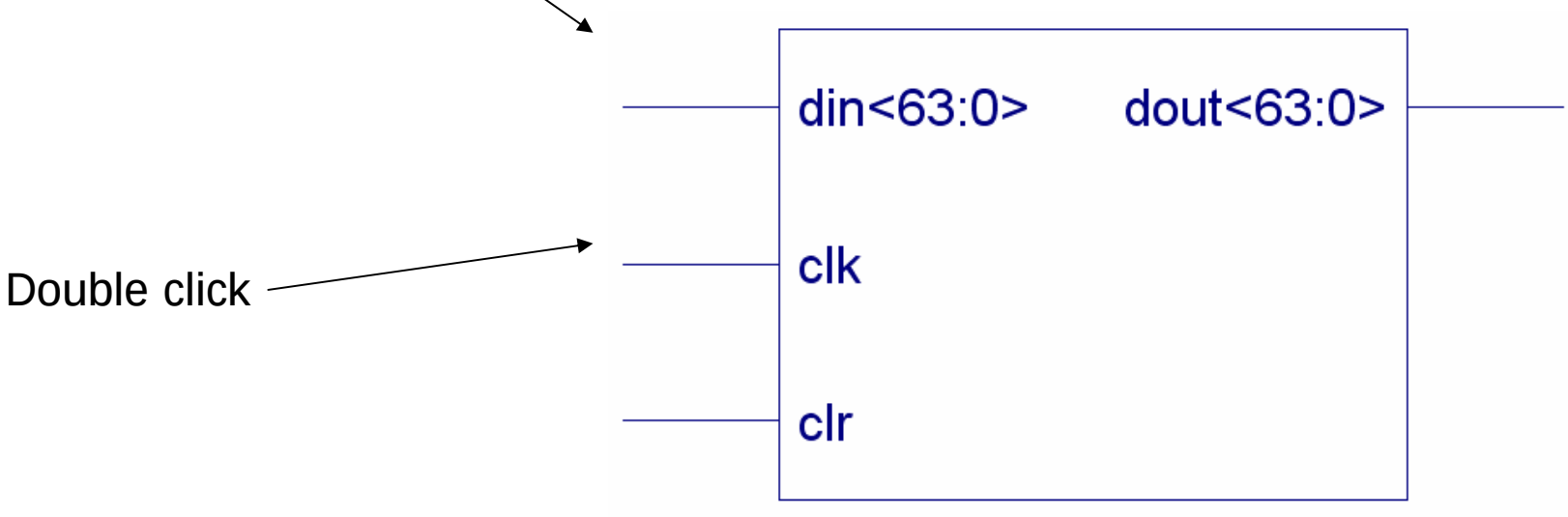
FPGA Synthesis

- View RTL Schematic

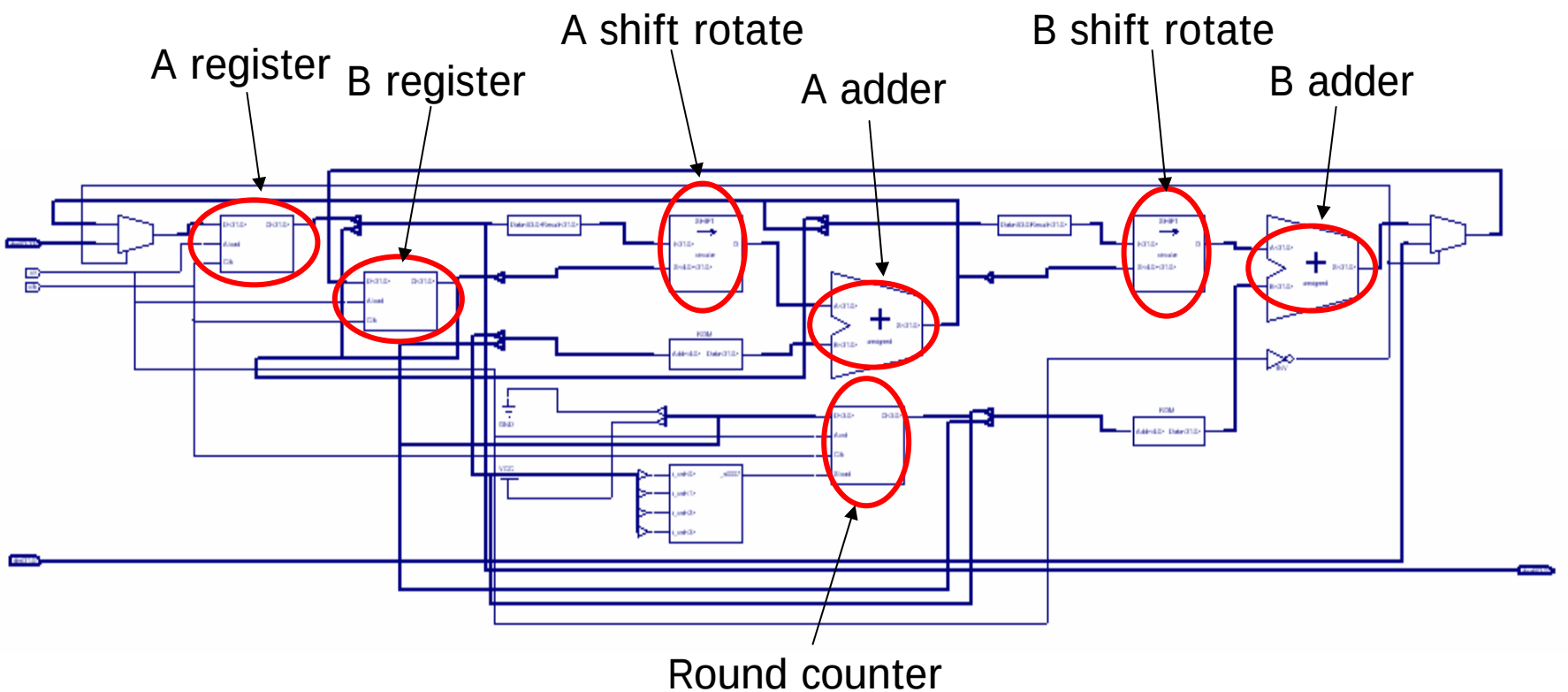


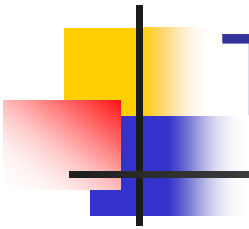
FPGA Synthesis

```
ENTITY rc5_enc IS
  PORT(
    clr, clk    : IN STD_LOGIC;
    din         : IN STD_LOGIC_VECTOR(63 DOWNTO 0);
    dout        : OUT STD_LOGIC_VECTOR(63 DOWNTO 0));
END rc5_enc;
```



FPGA Synthesis





Timing Simulation

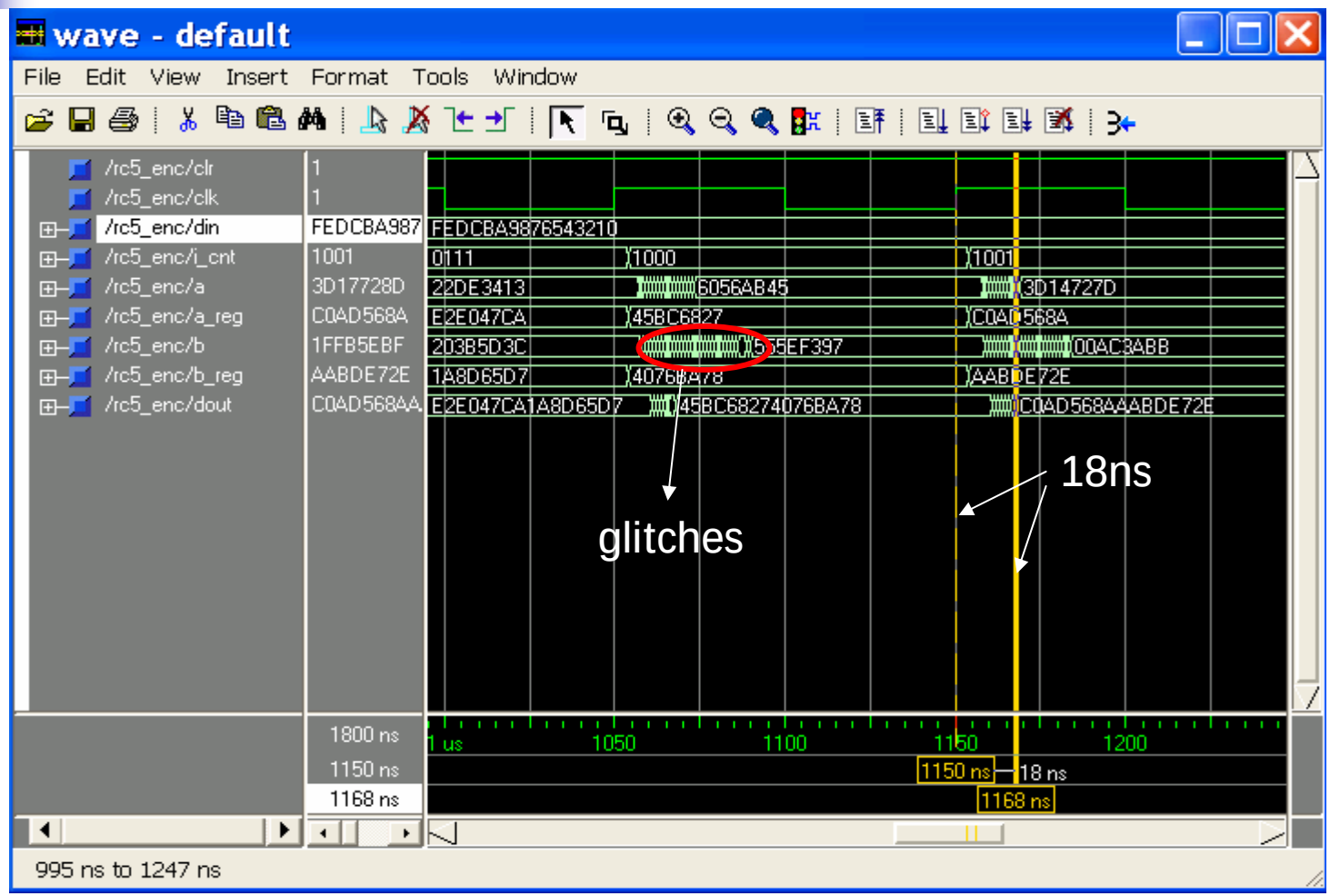
- Make sure timing simulation models are successfully generated
  - rc5_sim.vhd and rc5_sim.sdf
  - sdf stands for “standard delay format”
- Start ModelSim

Timing Simulation

- In ModelSim console window, type in following commands in sequence

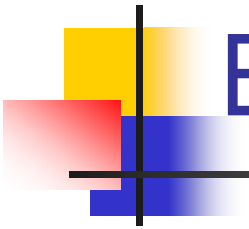
```
cd {c:/ee4313/proj2}
vlib work
vmap simprim c:/modeltech_5.6d/xilinx/simprim
vcom rc5_sim.vhd
vsim -sdftyp /=c:/ee4313/proj2/rc5_sim.sdf work.rc5_enc
view wave
add wave *
force clr 0 0, 1 200
force clk 0 0, 1 50 -repeat 100
force din 16#FEDCBA9876543210 0
run 1800
```

Timing Simulation



Results

- Timing – 18ns delay! (from ModelSim)
- Area
  - Logic Utilization: Number of 4 input LUTs: 723 out of 24,576 2%
  - Logic Distribution: Number of occupied Slices: 381 out of 12,288 1%
  - Number of Slices containing only related logic: 381 out of 381 100%
  - Number of Slices containing unrelated logic: 0 out of 381 0%
  - Number of 4 input LUTs: 726 out of 24,576 1%
  - Number of bonded IOBs: 129 out of 404 23%
  - Total equivalent gate count for design: 5,830



Exercise 2

- Modify the previous model to implement decryption operation
  - Simulate the Model
  - Synthesize the Design
  - Draw and Explain the Synthesized Design
- Analyze the timing simulation result, why is the delay less compared to lab1?