**Lab 3 (Task B)**

VHDL Code

**RAM behavior Code**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**use** IEEE**.**NUMERIC\_STD**.ALL;**

-- Synchronous write / asynchrounous read 32x16 single-port RAM

**entity** RAM **is**

**Port** **(** clk **:** **in** STD\_LOGIC**;**

write\_en **:** **in** STD\_LOGIC**;** -- Write enable

Data\_In **:** **in** UNSIGNED **(**15 **downto** 0**);** -- 16-bit data input

Address **:** **in** UNSIGNED **(**4 **downto** 0**);** -- 5-bit address

Data\_Out **:** **out** UNSIGNED **(**15 **downto** 0**));** -- 16-bit data output

**end** RAM**;**

**architecture** Behavioral **of** RAM **is**

**type** ram\_type **is** **array** **(**0 **to** 31**)** **of** UNSIGNED**(**15 **downto** 0**);**

**signal** ram\_inst**:** ram\_type**;**

**begin**

-- Asynchronous read

Data\_Out **<=** ram\_inst**(to\_integer(**Address**));**

-- Synchronous write (write enable signal)

**process** **(**clk**)**

**begin**

**if** **(rising\_edge(**clk**))** **then**

**if** **(**write\_en**=**'1'**)** **then**

ram\_inst**(to\_integer(**Address**))** **<=** Data\_In**;**

**end** **if;**

**end** **if;**

**end** **process;**

**end** Behavioral**;**

**Control Behaviour**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**use** IEEE**.**NUMERIC\_STD**.ALL;**

**entity** Control **is**

**Port** **(** clk **:** **in** STD\_LOGIC**;**

rst **:** **in** STD\_LOGIC**;**

nxt **:** **in** STD\_LOGIC**;**

mem\_wr **:** **out** STD\_LOGIC**;**

Mem\_Addr **:** **out** UNSIGNED **(**4 **downto** 0**);**

r1\_en **:** **out** STD\_LOGIC**;**

r2\_en **:** **out** STD\_LOGIC**;**

out\_en **:** **out** STD\_LOGIC**;**

Mux\_Sel **:** **out** STD\_LOGIC\_VECTOR **(**1 **downto** 0**));**

**end** Control**;**

**architecture** Behavioral **of** Control **is**

**type** fsm\_states **is** **(**STORE\_0**,** STORE\_1**,** LOAD\_R1**,** LOAD\_R2**,** STORE\_N**,** WAIT\_S**);**

**signal** state**:** fsm\_states**;**

**signal** next\_state**:** fsm\_states**;**

**signal** done **:** STD\_LOGIC**;**

**signal** cnt\_en **:** STD\_LOGIC**;**

**signal** count **:** UNSIGNED**(**4 **downto** 0**);**

**begin**

counter**:** **process** **(**clk**)** **is**

**begin**

**if** **rising\_edge(**clk**)** **then**

**if** rst **=** '1' **then**

count **<=** "00000"**;**

**else**

**if** cnt\_en **=** '1' **then**

count **<=** count **+**1**;**

**end** **if;**

**end** **if;**

**end** **if;**

**end** **process** counter**;**

state\_reg **:** **process** **(**clk**)** **is**

**begin**

**if** **rising\_edge(**clk**)** **then**

**if** **(**rst **=** '1'**)** **then**

state **<=** STORE\_0**;**

**else**

state **<=** next\_state**;**

**end** **if;**

**end** **if;**

**end** **process** state\_reg**;**

next\_states**:** **process(**state**,**rst**,**nxt**,**done**)** **is**

**begin**

**case** state **is**

**when** STORE\_0 **=>**

**if** nxt **=** '1' **then**

next\_state **<=** STORE\_1**;**

**else**

next\_state **<=** state**;**

**end** **if;**

**when** STORE\_1 **=>**

**if** nxt **=** '1' **then**

next\_state **<=** LOAD\_R1**;**

**else**

next\_state **<=** state**;**

**end** **if;**

**when** LOAD\_R1 **=>**

next\_state **<=** LOAD\_R2**;**

**when** LOAD\_R2 **=>**

next\_state **<=** STORE\_N**;**

**when** STORE\_N **=>**

next\_state **<=** WAIT\_S**;**

**when** WAIT\_S **=>**

**if** nxt **=**'1' and done **=** '0' **then**

next\_state **<=** LOAD\_R1**;**

**else**

next\_state **<=** state**;**

**end** **if;**

**when** **others** **=>**

next\_state **<=** state**;**

**end** **case;**

**end** **process** next\_states**;**

done **<=** '1' **when** **(**count **=** "11000"**)** **else** '0'**;**

cnt\_en **<=** nxt **when** **(**state **=** STORE\_0**)** or **(**state **=** STORE\_1**)**

**else** **(**nxt and **(**not done**))** **when** state **=** WAIT\_S

**else** '0'**;**

mem\_wr **<=** '1' **when** **(**state **=** STORE\_0**)**

or **(**state **=** STORE\_1**)**

or **(**state **=** STORE\_N**)**

**else** '0'**;**

Mem\_Addr **<=** count **when** **(**state **=** STORE\_0**)**

or **(**state **=** STORE\_1**)**

or **(**state **=** STORE\_N**)**

**else** **(**count **-** 1**)** **when** **(**state **=** LOAD\_R2**)**

**else** **(**count **-** 2**)** **when** **(**state **=** LOAD\_R1**)**

**else** "00000"**;**

r1\_en **<=** '1' **when** **(**state **=** LOAD\_R1**)**

**else** '0'**;**

r2\_en **<=** '1' **when** **(**state **=** LOAD\_R2**)**

**else** '0'**;**

out\_en **<=** '1' **when** **(**state **=** STORE\_0**)**

or **(**state **=** STORE\_1**)**

or **(**state **=** STORE\_N**)**

**else** '0'**;**

Mux\_Sel **<=** "10" **when** **(**state **=** STORE\_0**)**

**else** "11" **when** **(**state **=** STORE\_1**)**

**else** "00" **when** **(**state **=** STORE\_N**)**

**else** "00"**;**

**end** Behavioral**;**

VHDL Testbench Code

**LIBRARY** ieee**;**

**USE** ieee**.**std\_logic\_1164**.ALL;**

**USE** ieee**.**numeric\_std**.ALL;**

**ENTITY** Fibonacci\_tb **IS**

**END** Fibonacci\_tb**;**

**ARCHITECTURE** behavior **OF** Fibonacci\_tb **IS**

-- Component Declaration for the Unit Under Test (UUT)

**COMPONENT** Fibonacci

**PORT(**

clk **:** **IN** std\_logic**;**

rst **:** **IN** std\_logic**;**

nxt **:** **IN** std\_logic**;**

Fib\_Out **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

**signal** clk **:** std\_logic **:=** '0'**;**

**signal** rst **:** std\_logic **:=** '0'**;**

**signal** nxt **:** std\_logic **:=** '0'**;**

--Outputs

**signal** Fib\_Out **:** std\_logic\_vector**(**15 **downto** 0**);**

-- Clock period definitions

**constant** clk\_period **:** time **:=** 10 ns**;**

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** Fibonacci **PORT** **MAP** **(**

clk **=>** clk**,**

rst **=>** rst**,**

nxt **=>** nxt**,**

Fib\_Out **=>** Fib\_Out

**);**

-- Clock process definitions

clk\_process **:process**

**begin**

clk **<=** '0'**;**

**wait** **for** clk\_period**/**2**;**

clk **<=** '1'**;**

**wait** **for** clk\_period**/**2**;**

**end** **process;**

-- Stimulus process

stim\_proc**:** **process**

**begin**

-- hold reset state for 100 ns.

**wait** **for** 100 ns**;**

nxt **<=** '0'**;**

rst **<=** '1'**;**

**wait** **for** clk\_period**\***2**;**

next\_loop**:** **LOOP** --loop for next button to toggle with 3 clock periods

nxt **<=** '1'**;**

**wait** **for** clk\_period**\***3**;**

nxt **<=** '0'**;**

**wait** **for** clk\_period**\***3**;**

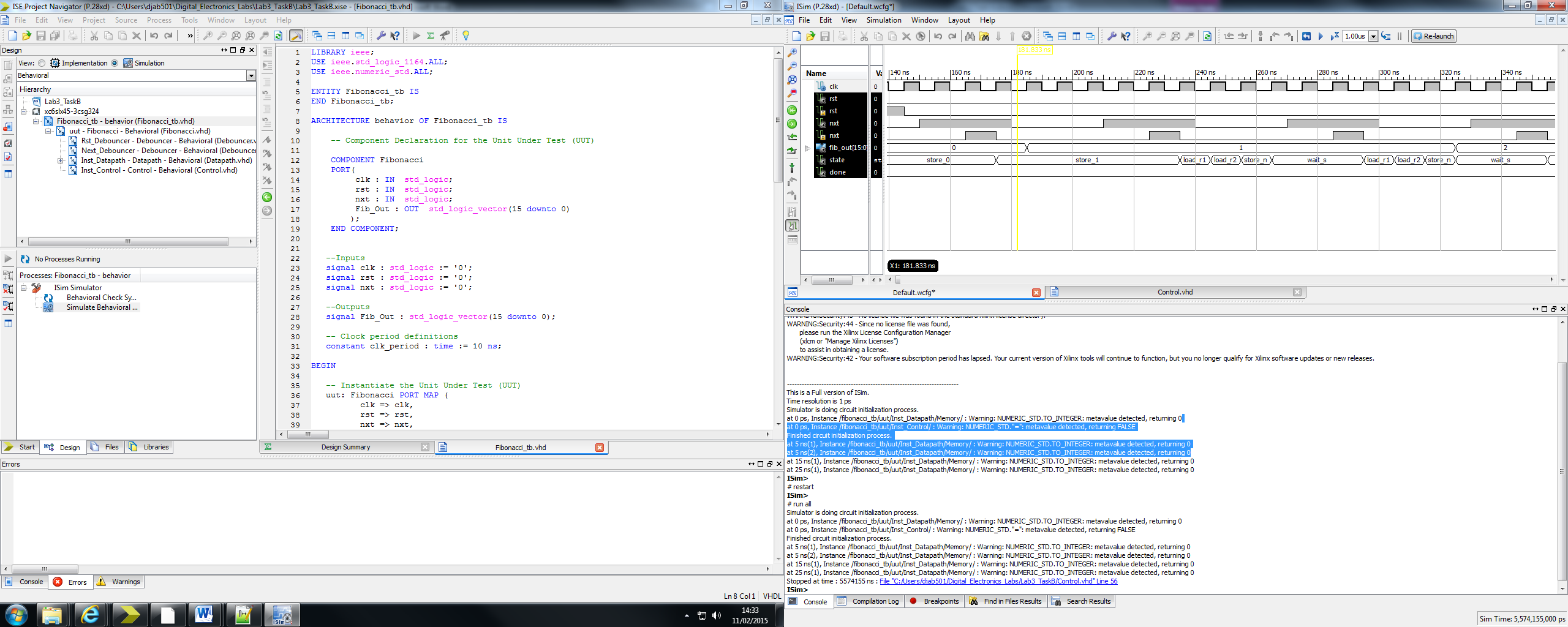
**END** **LOOP** next\_loop**;**

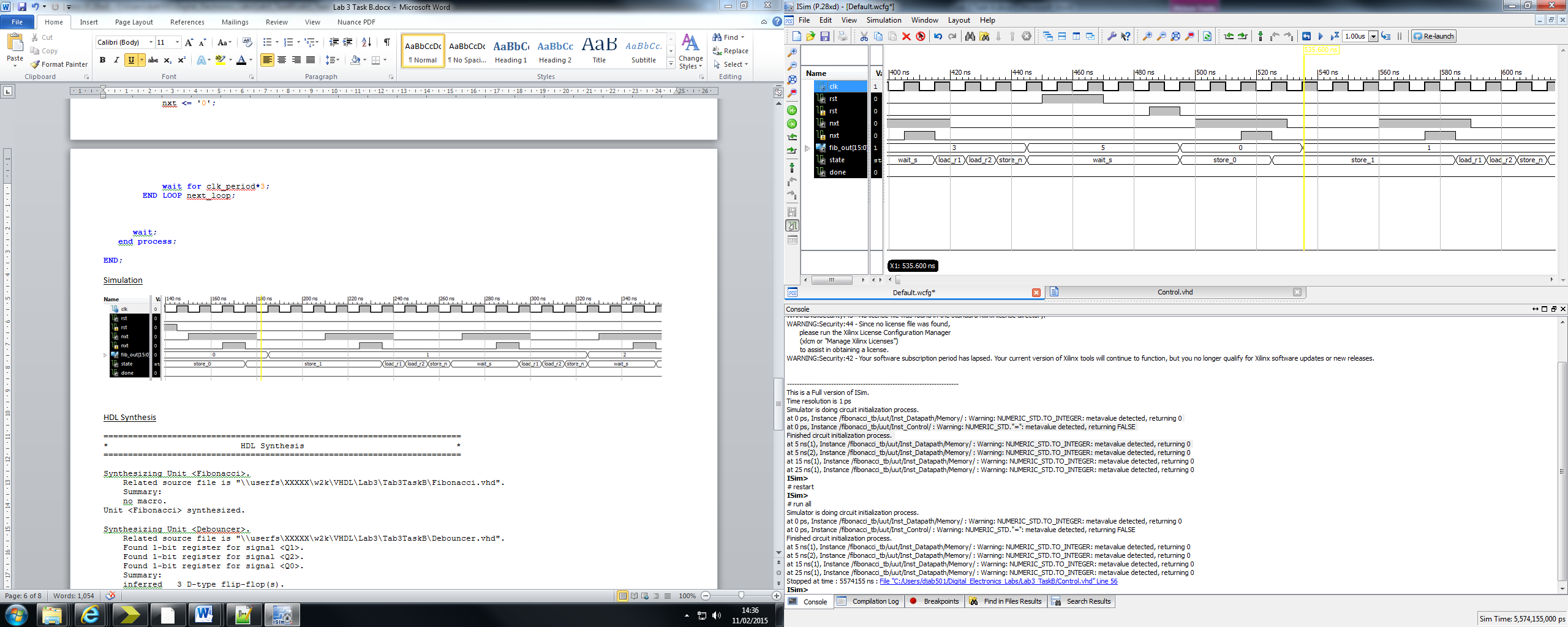
**wait;**

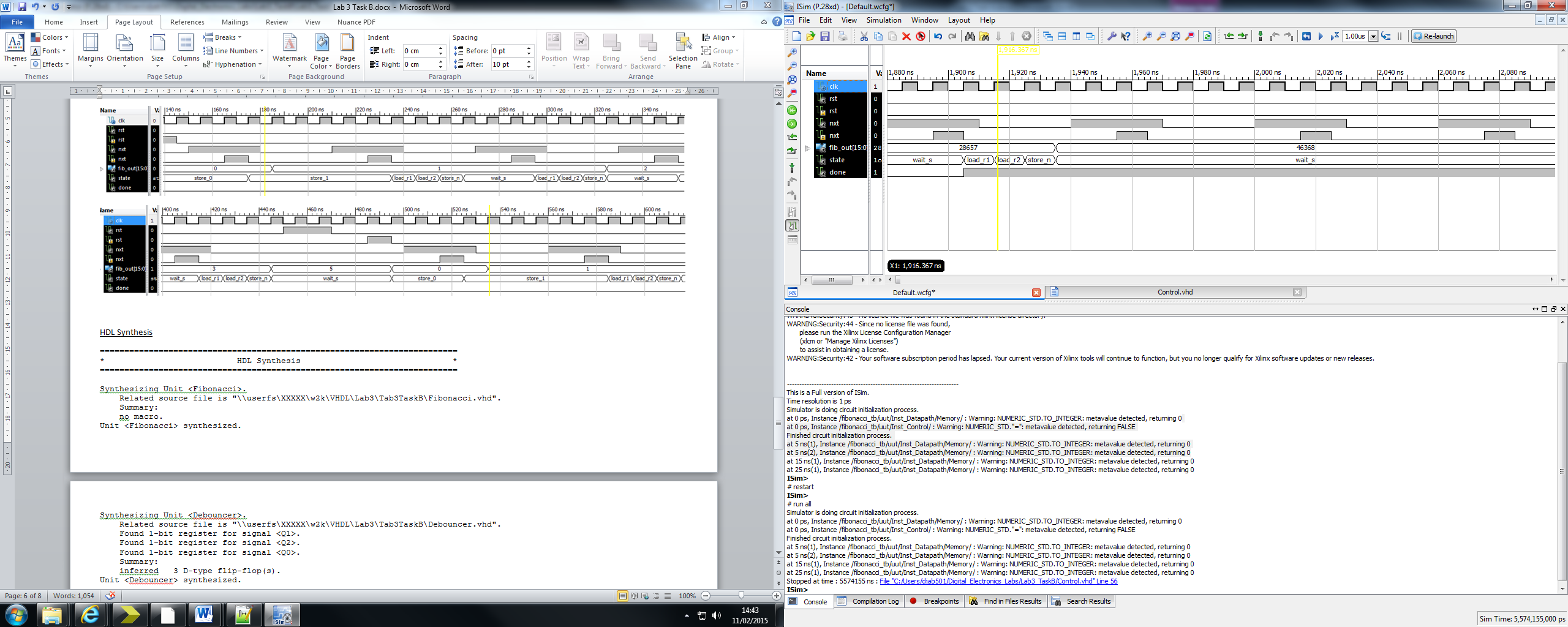
**end** **process;**

**END;**

Simulation







HDL Synthesis

=========================================================================

\* HDL Synthesis \*

=========================================================================

Synthesizing Unit <Fibonacci>.

Related source file is "\\userfs\XXXXX\w2k\VHDL\Lab3\Tab3TaskB\Fibonacci.vhd".

Summary:

no macro.

Unit <Fibonacci> synthesized.

Synthesizing Unit <Debouncer>.

Related source file is "\\userfs\XXXXX\w2k\VHDL\Lab3\Tab3TaskB\Debouncer.vhd".

Found 1-bit register for signal <Q1>.

Found 1-bit register for signal <Q2>.

Found 1-bit register for signal <Q0>.

Summary:

inferred 3 D-type flip-flop(s).

Unit <Debouncer> synthesized.

Synthesizing Unit <Datapath>.

Related source file is "\\userfs\XXXXX\w2k\VHDL\Lab3\Tab3TaskB\Datapath.vhd".

Found 16-bit register for signal <R2\_Out>.

Found 16-bit register for signal <Fib\_Out>.

Found 16-bit register for signal <R1\_Out>.

Found 16-bit adder for signal <R1\_Out[15]\_R2\_Out[15]\_add\_3\_OUT> created at line 32.

Summary:

inferred 1 Adder/Subtractor(s).

inferred 48 D-type flip-flop(s).

Unit <Datapath> synthesized.

Synthesizing Unit <RAM>.

Related source file is "\\userfs\XXXXX\w2k\VHDL\Lab3\Tab3TaskB\RAM.vhd".

Found 32x16-bit single-port RAM <Mram\_ram\_inst> for signal <ram\_inst>.

Summary:

inferred 1 RAM(s).

Unit <RAM> synthesized.

Synthesizing Unit <Control>.

Related source file is "\\userfs\XXXXX\w2k\VHDL\Lab3\Tab3TaskB\Control.vhd".

Found 3-bit register for signal <state>.

Found 5-bit register for signal <count>.

Found finite state machine <FSM\_0> for signal <state>.

-----------------------------------------------------------------------

| States | 6 |

| Transitions | 10 |

| Inputs | 2 |

| Outputs | 11 |

| Clock | clk (rising\_edge) |

| Reset | rst (positive) |

| Reset type | synchronous |

| Reset State | store\_0 |

| Power Up State | store\_0 |

| Encoding | auto |

| Implementation | LUT |

-----------------------------------------------------------------------

Found 5-bit adder for signal <count[4]\_GND\_10\_o\_add\_0\_OUT> created at line 1241.

Found 5-bit subtractor for signal <GND\_10\_o\_GND\_10\_o\_sub\_23\_OUT<4:0>> created at line 1308.

Found 5-bit subtractor for signal <GND\_10\_o\_GND\_10\_o\_sub\_25\_OUT<4:0>> created at line 1308.

Found 5-bit 3-to-1 multiplexer for signal <\_n0053> created at line 100.

Summary:

inferred 2 Adder/Subtractor(s).

inferred 5 D-type flip-flop(s).

inferred 6 Multiplexer(s).

inferred 1 Finite State Machine(s).

Unit <Control> synthesized.

=========================================================================

HDL Synthesis Report

Macro Statistics

# RAMs : 1

32x16-bit single-port RAM : 1

# Adders/Subtractors : 3

16-bit adder : 1

5-bit adder : 1

5-bit subtractor : 1

# Registers : 10

1-bit register : 6

16-bit register : 3

5-bit register : 1

# Multiplexers : 6

1-bit 2-to-1 multiplexer : 2

2-bit 2-to-1 multiplexer : 1

5-bit 2-to-1 multiplexer : 3

# FSMs : 1