Ryerson University

Department of Electrical, Computer & Biomedical Engineering
Faculty of Engineering & Architectural Science

Programs: Computer Engineering

Course Number	**COE608**
Course Title	**Computer Organization and Architecture**
Semester/Year	**Summer 2020**
Instructor	**Patrick Siddavaatam**

Lab Report No. 4a

Lab Title	**Data Memory Module**

Section No.	**011**
Group No.	
Submission Date	**21 July 2020**
Due Date	**23 July 2020**

Name	Student ID	Signature*
Duanwei Zhang	*500824903*	DZ
Xinyu Hadrian Hu	500194233	XHH

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*
www.ryerson.ca/senate/current/pol60.pdf.

Name: Xinyu Hadrian Hu, and Duan Wei Zhang

Student Number: 500194233, and 500824903

TA: Jasminder Singh

Date: June 28, 2020

COE 608: Computer Architecture and Design

COE 608: Lab 4, Part 1 Report

Objective

The purpose of this lab is to create a memory module for the 32-bit ALU, and later the instruction set architecture (ISA) for the million instructions per second (MIPS) ISA.

Design and Implementation

Provided below is the truth table for the memory module. The address is an 8-bit data, and the data-in and data-outs are 32-bit data. See Figure 1: Memory Module Truth Table.

Memory Module (sample from waveform)				
clk	**en**	**wen (unsigne**	**data_in (unsigne**	**data_out(unsigne**
0	1	0	0	0
1	0	1	0	0
0	1	2	2	0
1	1	3	2	0
0	1	4	4	0
1	0	5	4	0
0	1	6	6	0
1	1	7	6	0
0	0	8	8	0
1	0	9	8	0
0	1	10	10	0
1	1	11	10	0
0	1	12	12	4
1	1	13	12	4

Figure 1: Memory Module Truth Table

Observations and Results

Below are the observations from functional and timing waveforms: See Figure 2: Memory Module Functional Waveform and Figure 3: Memory Module Timing Waveform.

Memory Module: Functional and Timing

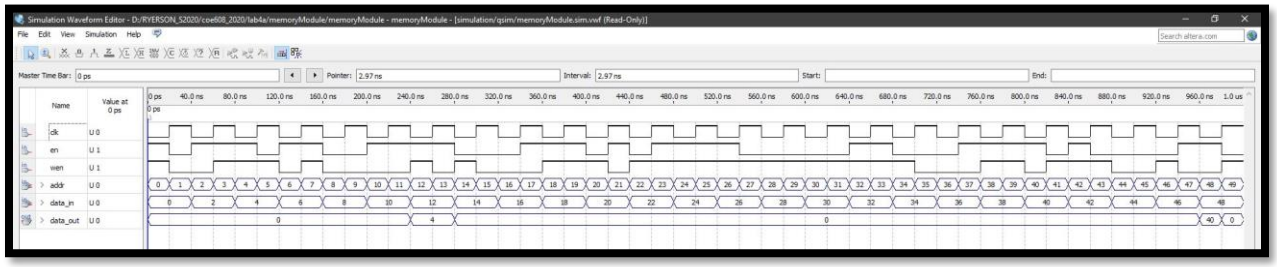


Figure 2: Memory Module Functional Waveform

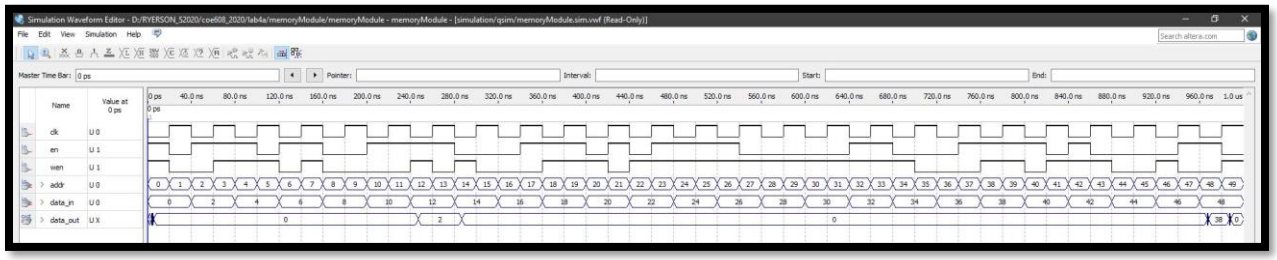


Figure 3: Memory Module Timing Waveform

Discussions and Conclusions

The memory module works as intended. The worst case delays for the memory module is 5 ns, as can be observed from the timing simulation waveform.

Appendix: VHDL Codes and Screenshots of Waveforms

I have provided the VHDL and screenshots of the waveforms for easier reading.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity memoryModule is
6     port (
7        clk: in std_logic;
8        addr: in unsigned (7 downto 0);
9        data_in : in std_logic_vector(31 downto 0);
10       wen, en: in std_logic;
11       data_out: out std_logic_vector(31 downto 0));
12 end memoryModule;
13
14 architecture description of memoryModule is
15    type memory2D is array (7 downto 0) of std_logic_vector(31 downto 0);
16    signal memoryModuleArray : memory2D;
17       begin
18          process(clk,en,wen)
19             begin
20                if falling_edge(clk) then
21                   if (en = '1' and wen = '0') then
22                      data_out <= memoryModuleArray(to_integer(unsigned(addr)));
23                   elsif (en = '1' and wen = '1') then
24                      memoryModuleArray(to_integer(unsigned(addr))) <= data_in;
25                      data_out <= (others => '0');
26                   end if;
27                end if;
28                if en = '0' then
29                   data_out <= (others => '0');
30                end if;
31          end process;
32 end description;
33
34
```

Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4a/memoryModule/memoryModule - memoryModule - [simulation/qsim/memoryModule.sim.vwf (Read-Only)]

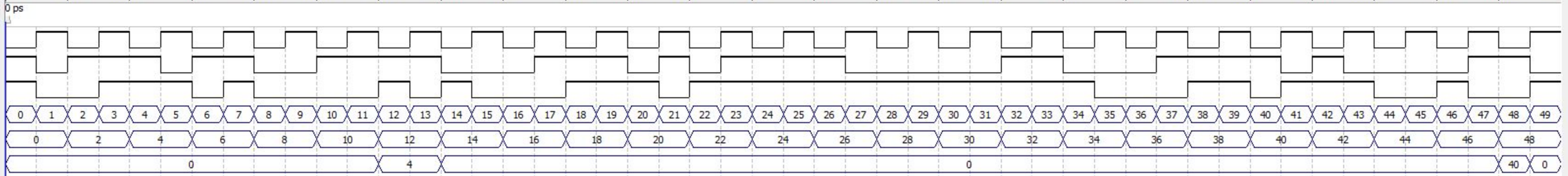
File Edit View Simulation Help

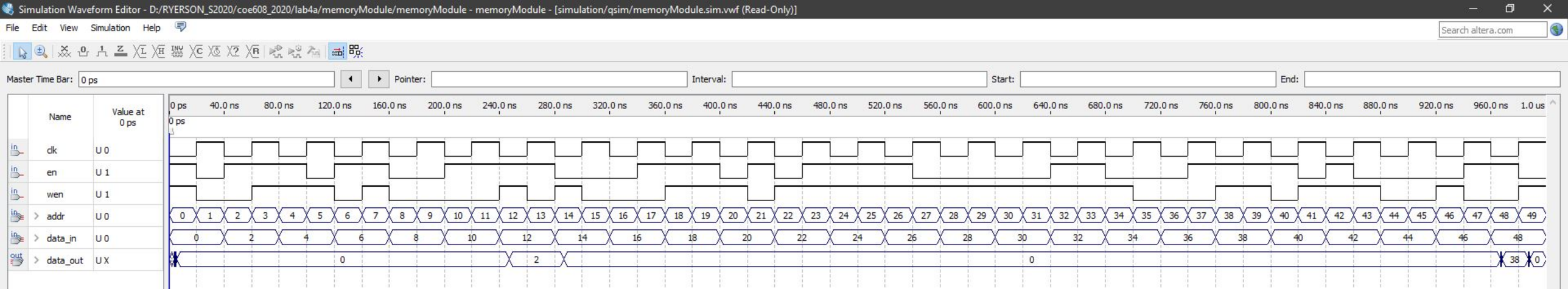
Search altera.com

Master Time Bar: 0 ps | Pointer: 2.97 ns | Interval: 2.97 ns | Start: | End:

Name	Value at 0 ps
clk	U 0
en	U 1
wen	U 1
addr	U 0
data_in	U 0
data_out	U 0

0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us




Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4a/memoryModule/memoryModule - memoryModule - [simulation/qsim/memoryModule.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer:
Interval:
Start:
End:
Name
Value at 0 ps
clk U 0
en U 1
wen U 1
addr U 0
data_in U 0
data_out U X
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48
0 2 0 38 0

Ryerson University

Department of Electrical, Computer & Biomedical Engineering
Faculty of Engineering & Architectural Science

Programs: Computer Engineering

Course Number	**COE608**
Course Title	**Computer Organization and Architecture**
Semester/Year	**Summer 2020**
Instructor	**Patrick Siddavaatam**

Lab Report No. 4b

Lab Title	**CPU Data Path Design**

Section No.	**011**
Group No.	
Submission Date	**21 July 2020**
Due Date	**23 July 2020**

Name	Student ID	Signature*
Duanwei Zhang	*500824903*	DZ
Xinyu Hadrian Hu	500194233	XHH

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*
www.ryerson.ca/senate/current/pol60.pdf.

Name: Xinyu Hadrian Hu, and Duan Wei Zhang

Student Number: 500194233, and 500824903

TA: Jasminder Singh

Date: June 28, 2020

COE 608: Computer Architecture and Design

COE 608: Lab 4, Part 2 Report

Table of Figures

Objective

The purpose of this lab is to put together everything we have done so far, with the addition of the upper zero extender, lower zero extender, and reducer with the program counter, arithmetic logical unit, multiplexers, and operational inverters into a functioning data-path for the MIPS instructional set architecture.

Design and Implementation

I first started with the reducers, LZE and UZE implementations. I did the data-path last. The following truth-tables provide examples of some of the purposes of the LZE, UZE, and reducer. The LZE extends zeroes by padding the bits from 15 down to 0. The UZE does the inverse of the LZE operation by padding zeroes from 31 down to 16. The reducer helps to reduce a 32-bit data into 8-bit data. The data-path is programmed in VHDL using the structural behavior, by utilizing port mapping and signals to denote the wires and buses between the many devices used throughout the course.

The truth tables for each of the new devices are illustrated in the following pages.

See the following figures below: Figure 1: UZE Truth Table, Figure 2: LZE Truth Table, Figure 3: Reducer Truth Table and Figure 4: Data-path signals truth table.

Upper Zero Extender	
In (hex)	**Out (hex)**
00000001	00000000
0000000A	00000000
000F9EDF	000F0000
ABC85FG1	ABC80000

Figure 1: UZE Truth Table

Next, is the Lower Zero Extender Truth Table:

Lower Zero Extender	
In (hex)	**Out (hex)**
00000001	000000001
0000000A	000000000A
000F9EDF	000009EDF
ABC85FG1	000005FG1

Figure 2: LZE Truth Table

Next, is the Reducer Truth Table:

Reducer	
In (hex)	**Out (hex)**
00000001	01
0000000A	0A
000F9EDF	9EDF
ABC85FG1	5FG1

Figure 3: Reducer Truth Table

Finally, the Data-path Truth Table:

Name: Xinyu Hadrian Hu		Student #: 500194233		Section: 01								
INST	CLR_IR LD_IR	LD_PC INC_PC	CLR_A LD_A	CLR_B LD_B	CLR_C LD_C	CLR_Z LD_Z	ALU OP	EN WEN	A/B MUX	REG MUX	Data MUX	IM_MUX1 IM_MUX2
LDA	0/0	0/0	0/1	0/0	0/0	0/0	XXX	10	0/X	X	1	X
LDB	0/0	0/0	0/0	0/1	0/0	0/0	XXX	10	X/0	X	1	X
STA	0/0	0/0	0/0	0/0	0/0	0/0	XXX	11	X	0	X	X
JMP	0/0	0/0	0/0	0/0	0/0	0/0	XXX	11	X	1	X	X
STAB	0/0	10	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X
LDAI	0/0	0/0	0/1	0/0	0/0	0/0	XXX	X	1/X	X	X	X
LDBI	0/0	0/0	0/0	0/1	0/0	0/0	XXX	X	X/1	X	X	X
LUI	0/0	0/0	0/1	10	0/0	0/0	1	X	0/X	X	10	1/X
ANDI	0/0	0/0	0/1	0/0	0/1	0/1	0	X	0/X	X	10	0/01
DECA	0/0	0/0	0/1	0/0	0/1	0/1	110	X	0/X	X	10	0/10
AND	0/0	0/0	0/1	0/0	0/1	0/1	10	X	0/X	X	10	0/00
ADDI	0/0	0/0	0/1	0/0	0/1	0/1	110	X	0/X	X	10	0/00
SUB	0/0	0/0	0/1	0/0	0/1	0/1	10	X	0/X	X	10	0/10
INCA	0/0	0/0	0/1	0/0	0/1	0/1	0	X	0/X	X	10	0/00
ORI	0/0	0/0	0/1	0/0	0/1	0/1	10	X	0/X	X	10	0/01
ROL	0/0	0/0	0/1	0/0	0/1	0/1	1	X	0/X	X	10	0/01
ROR	0/0	0/0	0/1	0/0	0/1	0/1	100	X	0/X	X	10	0/X
CLRA	0/0	0/0	0/1	0/0	0/1	0/1	101	X	0/X	X	10	0/X
CLRC	0/0	0/0	10	0/0	0/0	0/0	XXX	X	X	X	X	X
CLRZ	0/0	0/0	0/0	10	0/0	0/0	XXX	X	X	X	X	X
PC <= PC + 4	0/0	10	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X
IR <= M[INT]	0/1	0/0	0/0	0/0	0/0	0/0	XXX	X	X	X	0	X
PC IR [15..0]	0/0	10	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X

Figure 4: Data-path signals truth table

Observations and Results

Below are the functional and timing waveforms derived for each of the new components listed for this lab.

The following figures are the observations of this project:

Figure 5: LZE Functional Waveform, Figure 6: LZE Timing Waveform, Figure 7: UZE Functional Waveform, Figure 8: UZE Timing Waveform, Figure 9:Reducer Functional Waveform, Figure 10: Reducer Timing Waveform, Figure 11: Data-path Random Input Waveform, Figure 12: Data-path Output Functional Waveform, and Figure 13: Data-path Output Timing Waveform are the figures of interest.

LZE: Functional and Timing

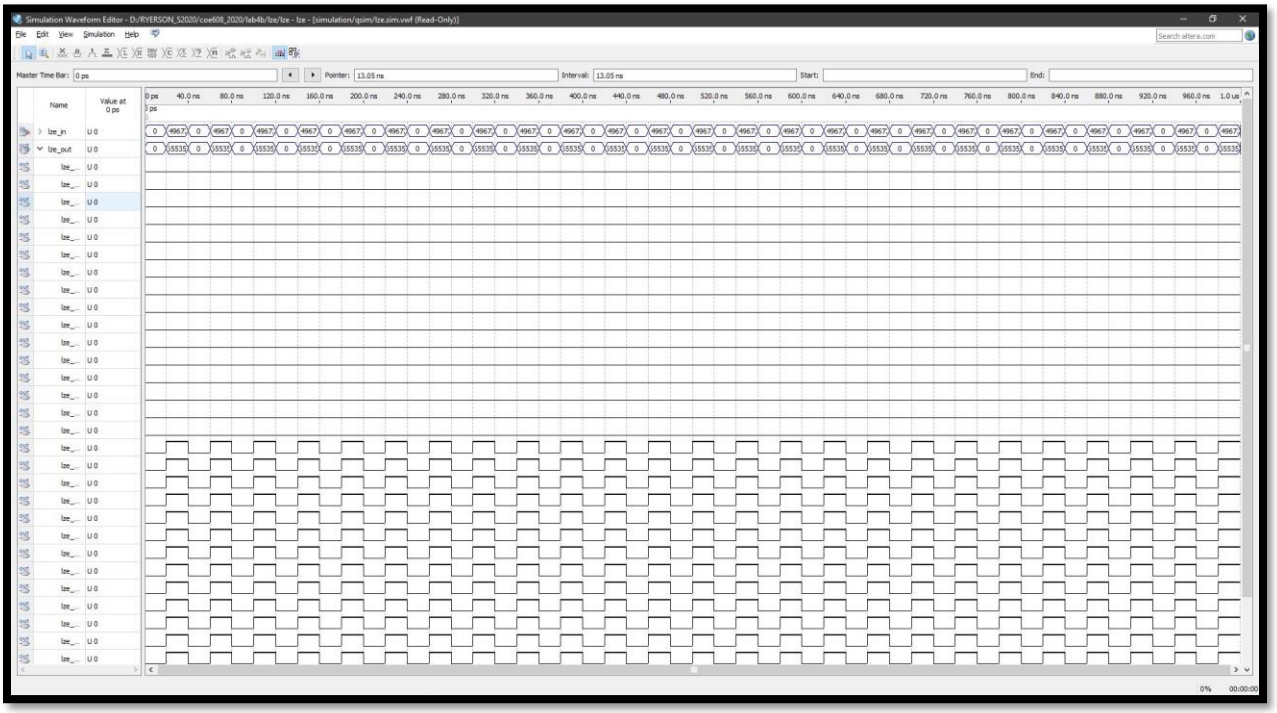


Figure 5: LZE Functional Waveform

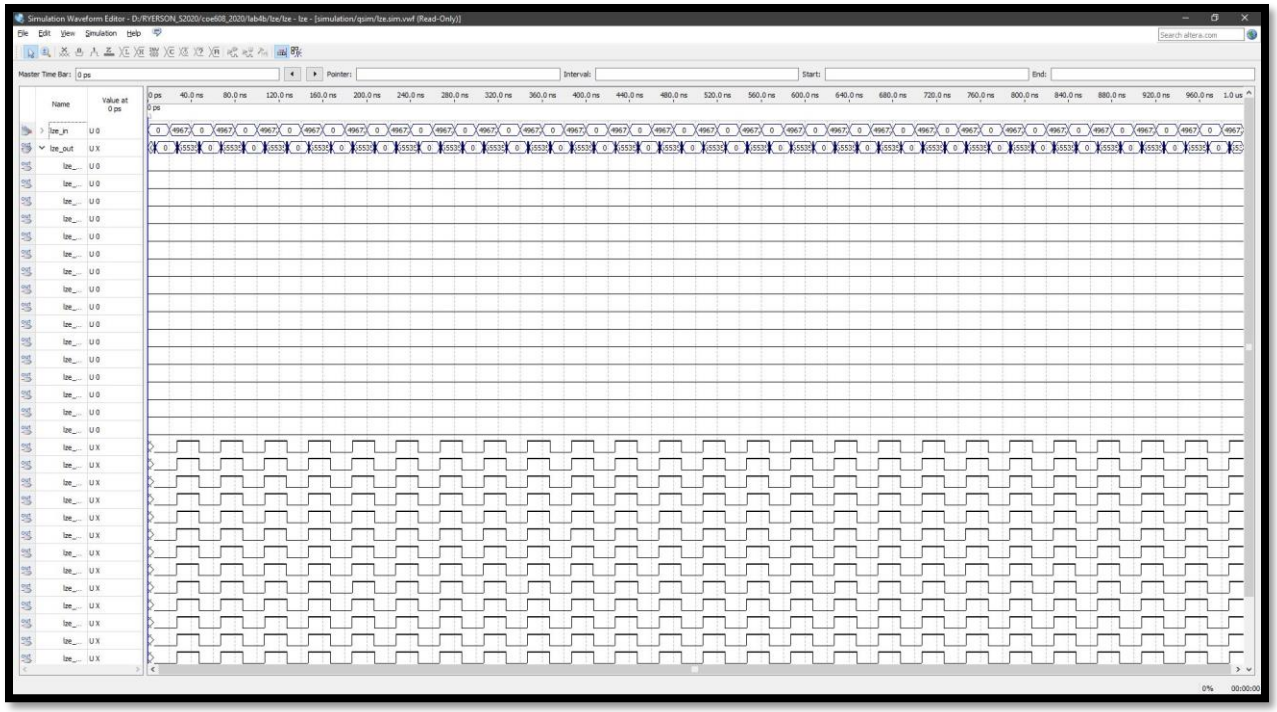


Figure 6: LZE Timing Waveform

UZE: Functional and Timing

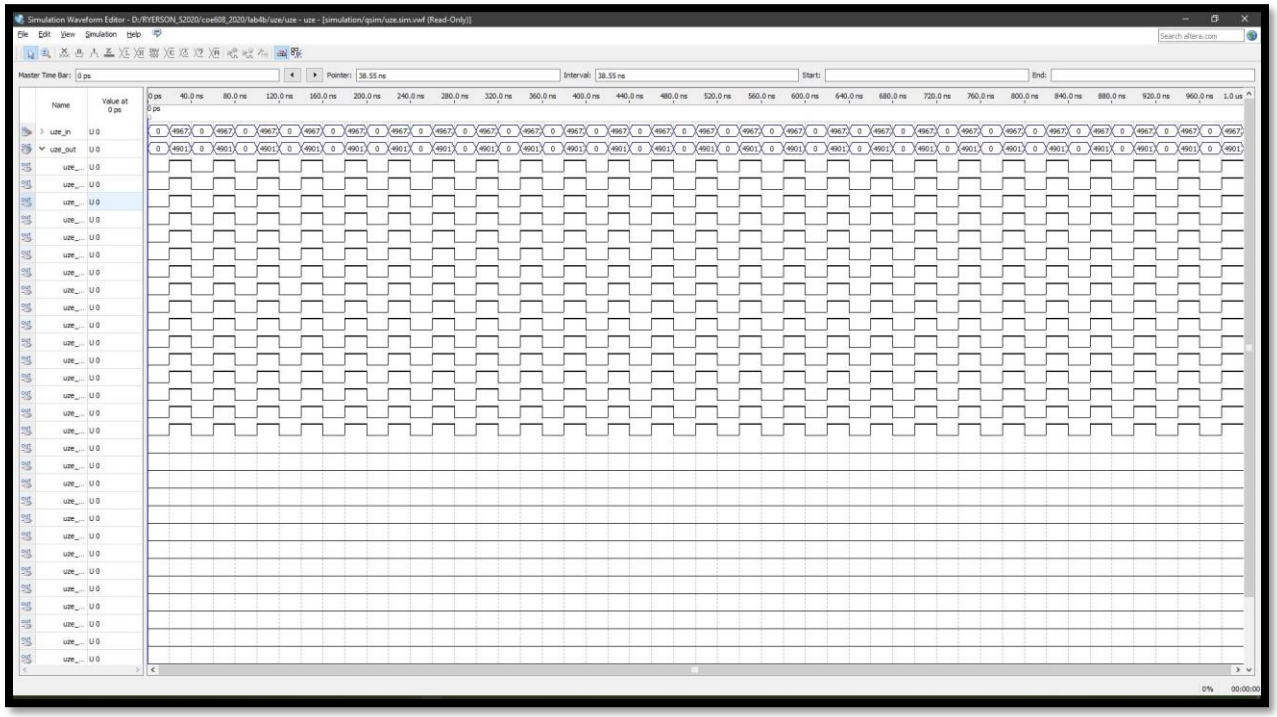


Figure 7: UZE Functional Waveform

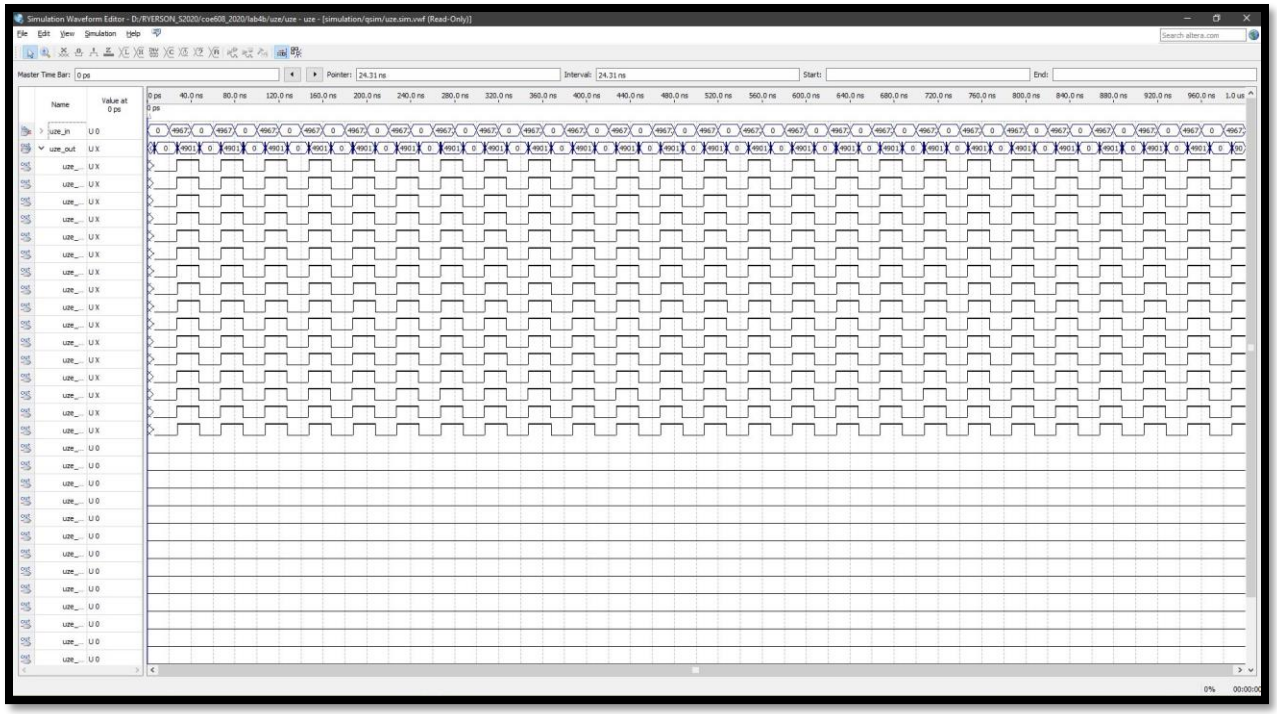


Figure 8: UZE Timing Waveform

Reducer: Functional and Timing

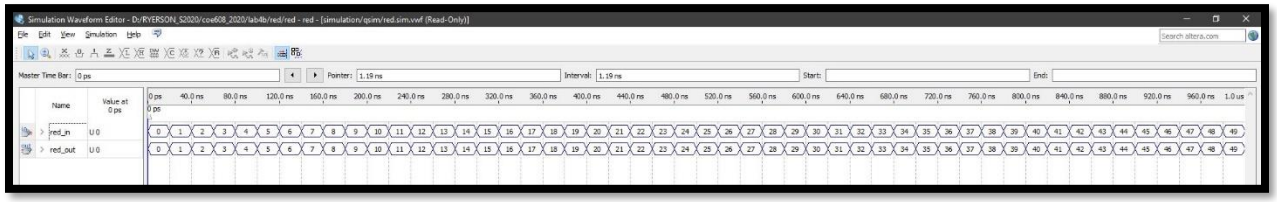


Figure 9:Reducer Functional Waveform

Figure 10: Reducer Timing Waveform

Data-path: Functional and Timing

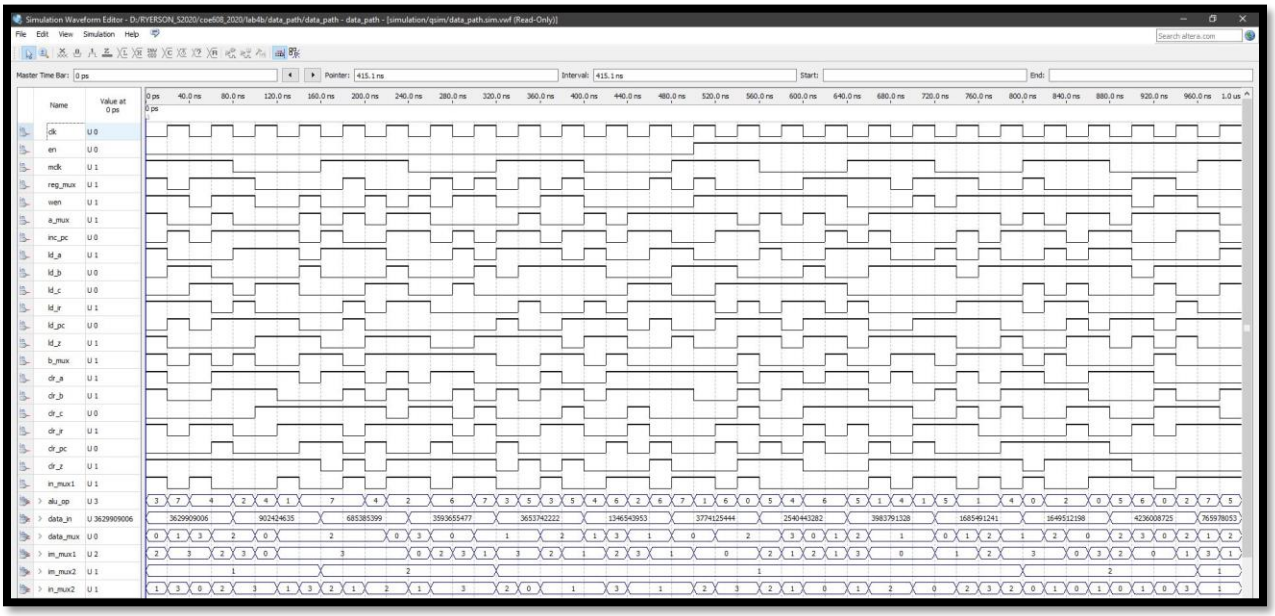


Figure 11: Data-path Random Input Waveform

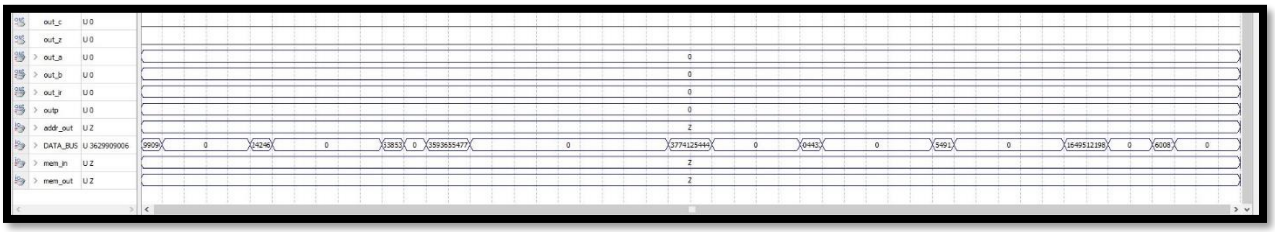


Figure 12: Data-path Output Functional Waveform

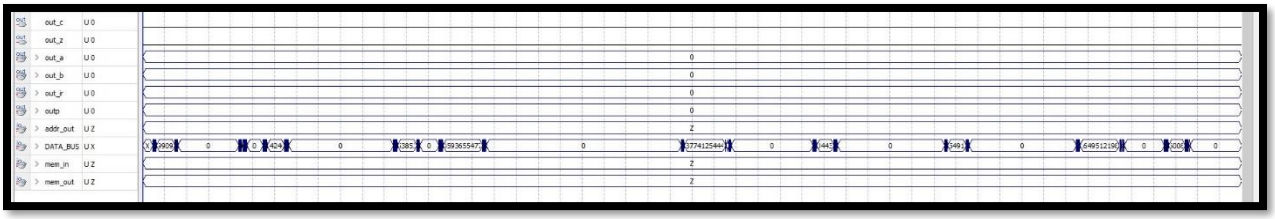


Figure 13: Data-path Output Timing Waveform

For the purposes of understand the operations of MIPS data-path, we have also included the results of specific operations. The random operations waveforms above are not very helpful in demonstrating whether a specific operation works or not.

The following waveform screenshots are courtesy of Duanwei Zhang:

The following figures illustrate the most important operations for the MIPS data-path:

Figure 14: LDAI Operation Waveform, *Figure 15: LDBI Operation Waveform*, *Figure 16: STA Operation Waveform*, *Figure 17: STB Operation Waveform*, *Figure 18: LDA Operation Waveform*, *Figure 19: LDB Operation Waveform*, *Figure 20: LUI Operation Waveform*, *Figure 21: JMP Operation Waveform*, *Figure 22: ADD Operation Waveform*, *Figure 23: ADDI Operation Waveform*, *Figure 24: SUB Operation Waveform*, *Figure 25: INCA Operation Waveform*, *Figure 26: ROL Operation Waveform*, *Figure 27: CLRA Operation Waveform*, *Figure 28: ANDI Operation Waveform*, *Figure 29: ORI Operation Waveform*, *Figure 30: DECA Operation Waveform*, and *Figure 31: ROR Operation Waveform*.

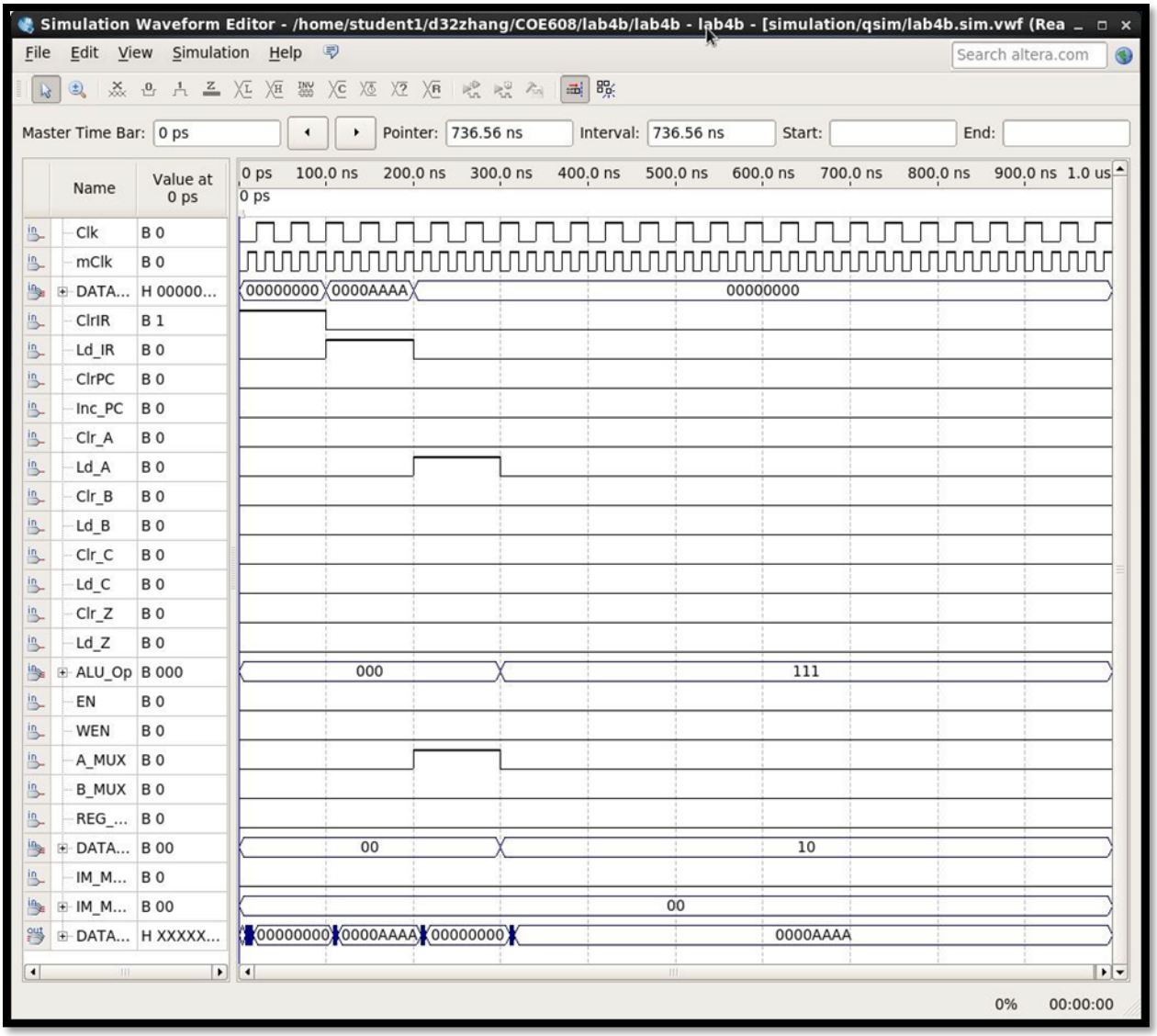


Figure 14: LDAI Operation Waveform

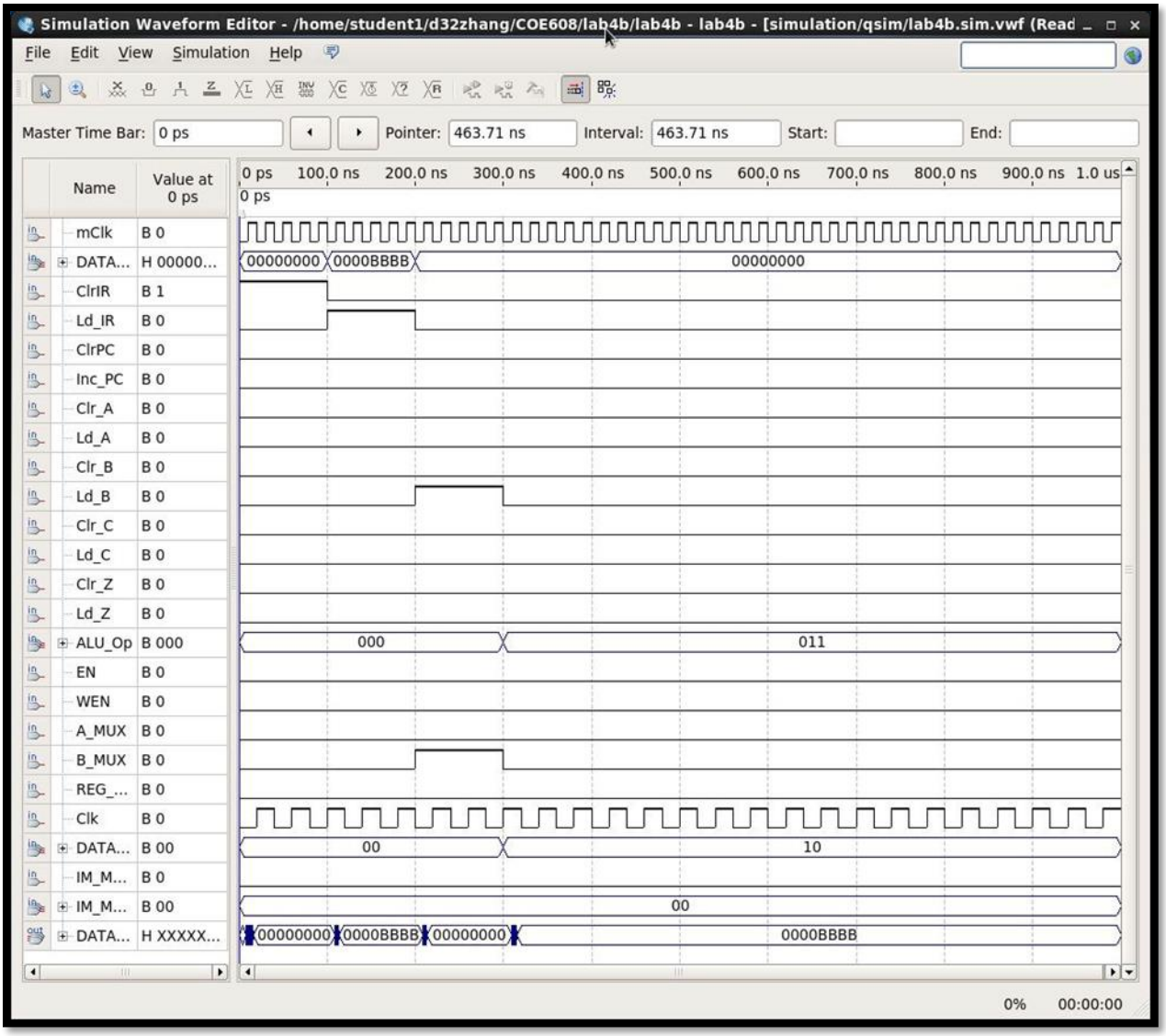


Figure 15: LDBI Operation Waveform

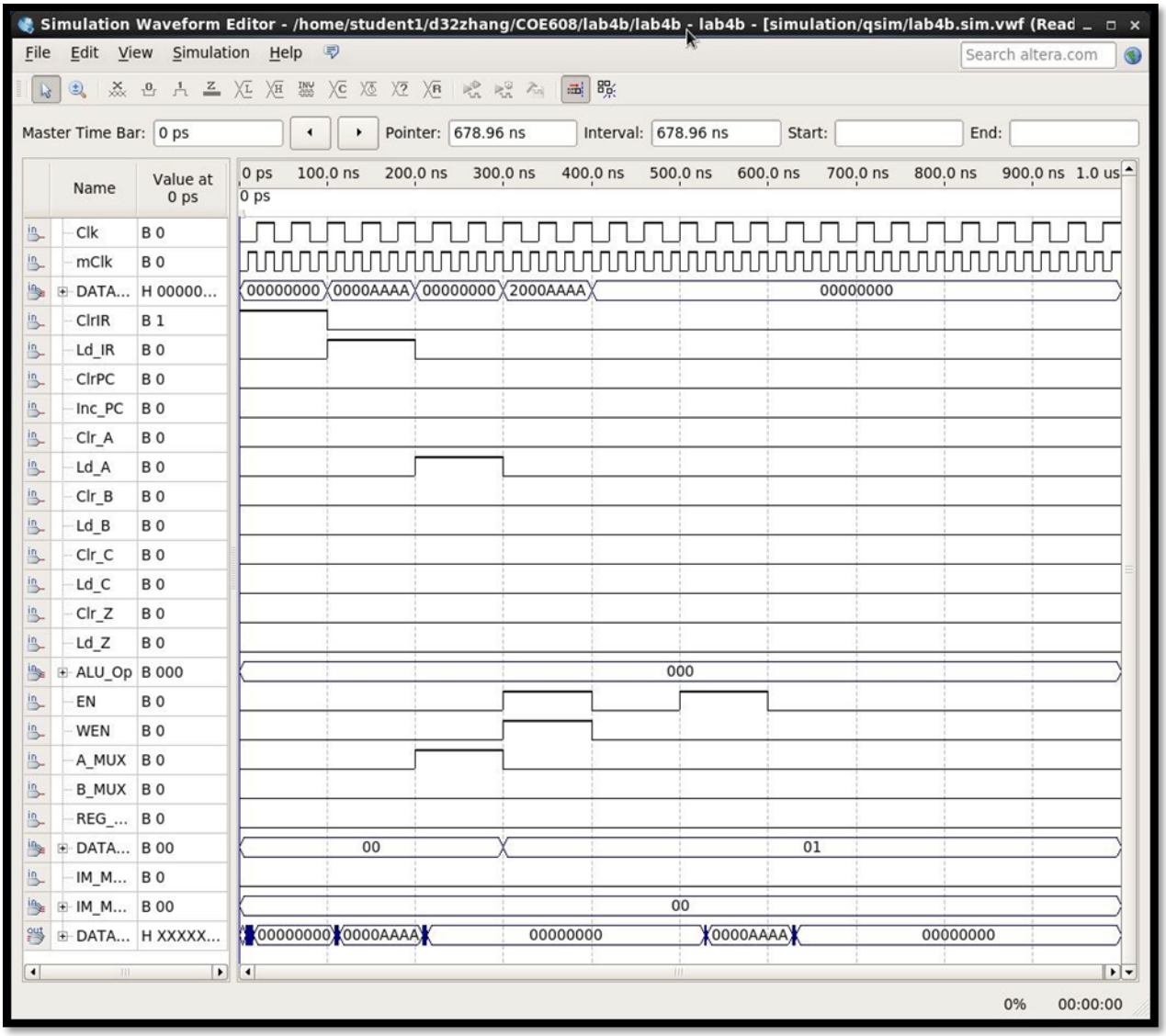


Figure 16: STA Operation Waveform

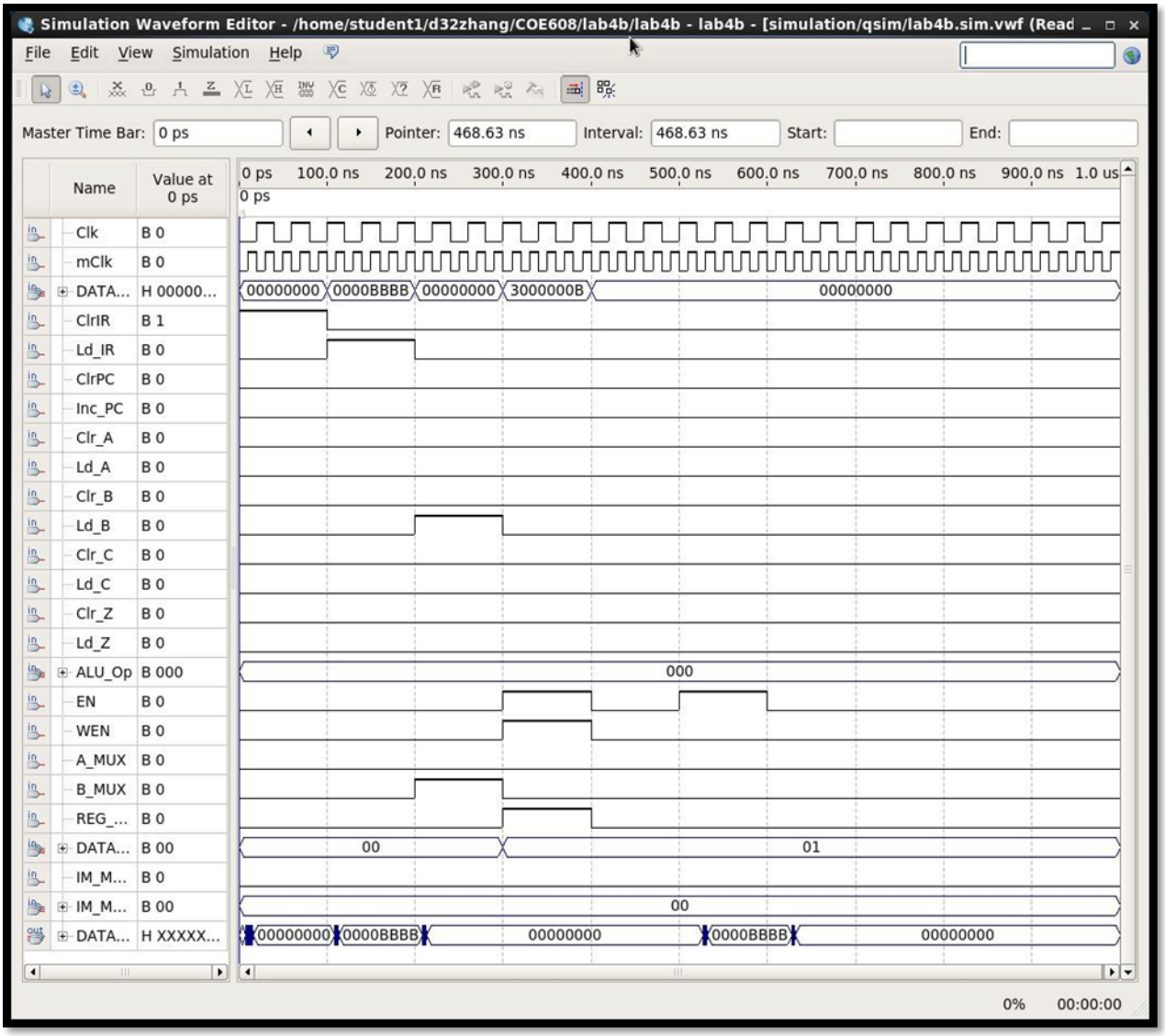


Figure 17: STB Operation Waveform

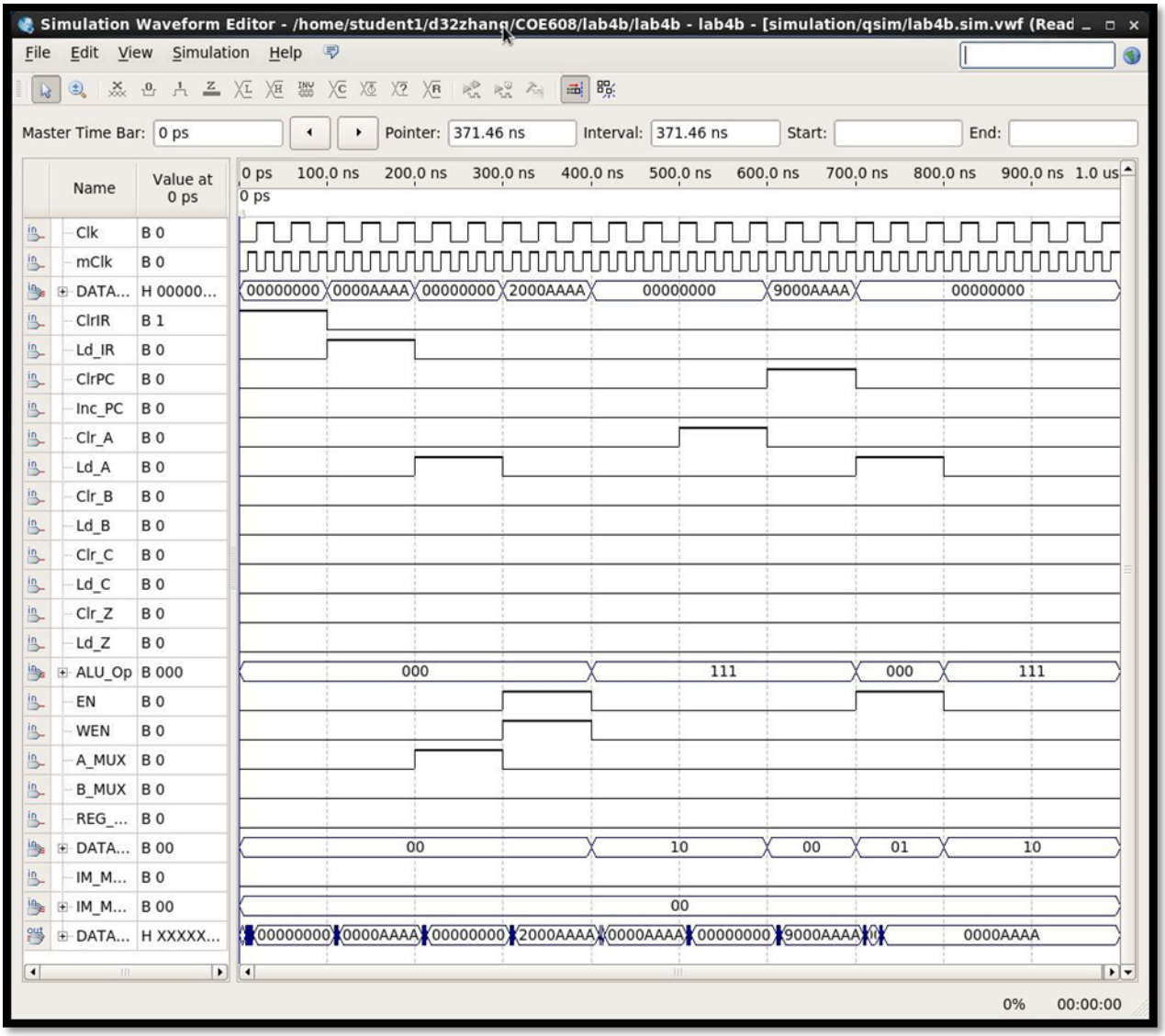


Figure 18: LDA Operation Waveform

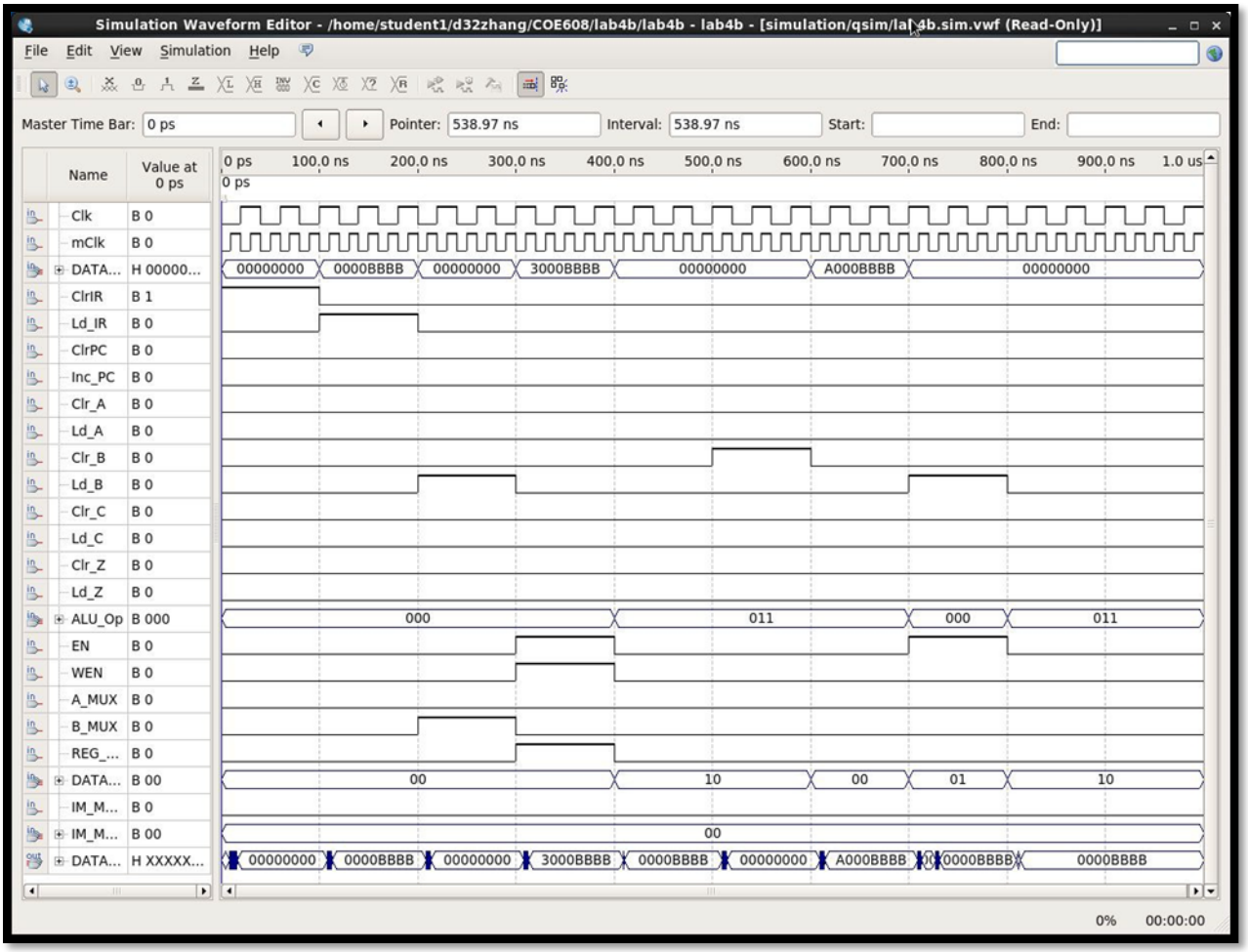


Figure 19: LDB Operation Waveform

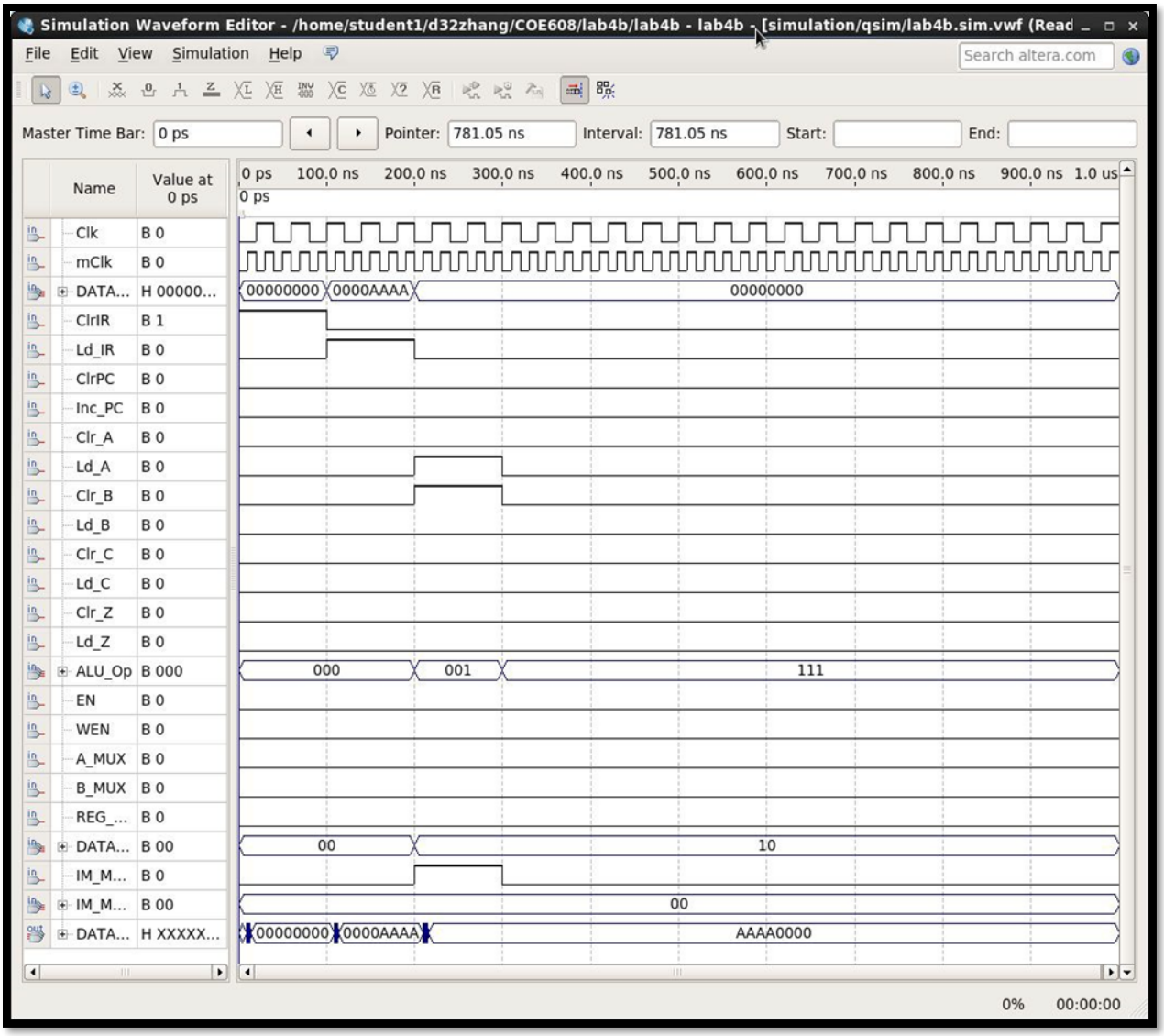


Figure 20: LUI Operation Waveform

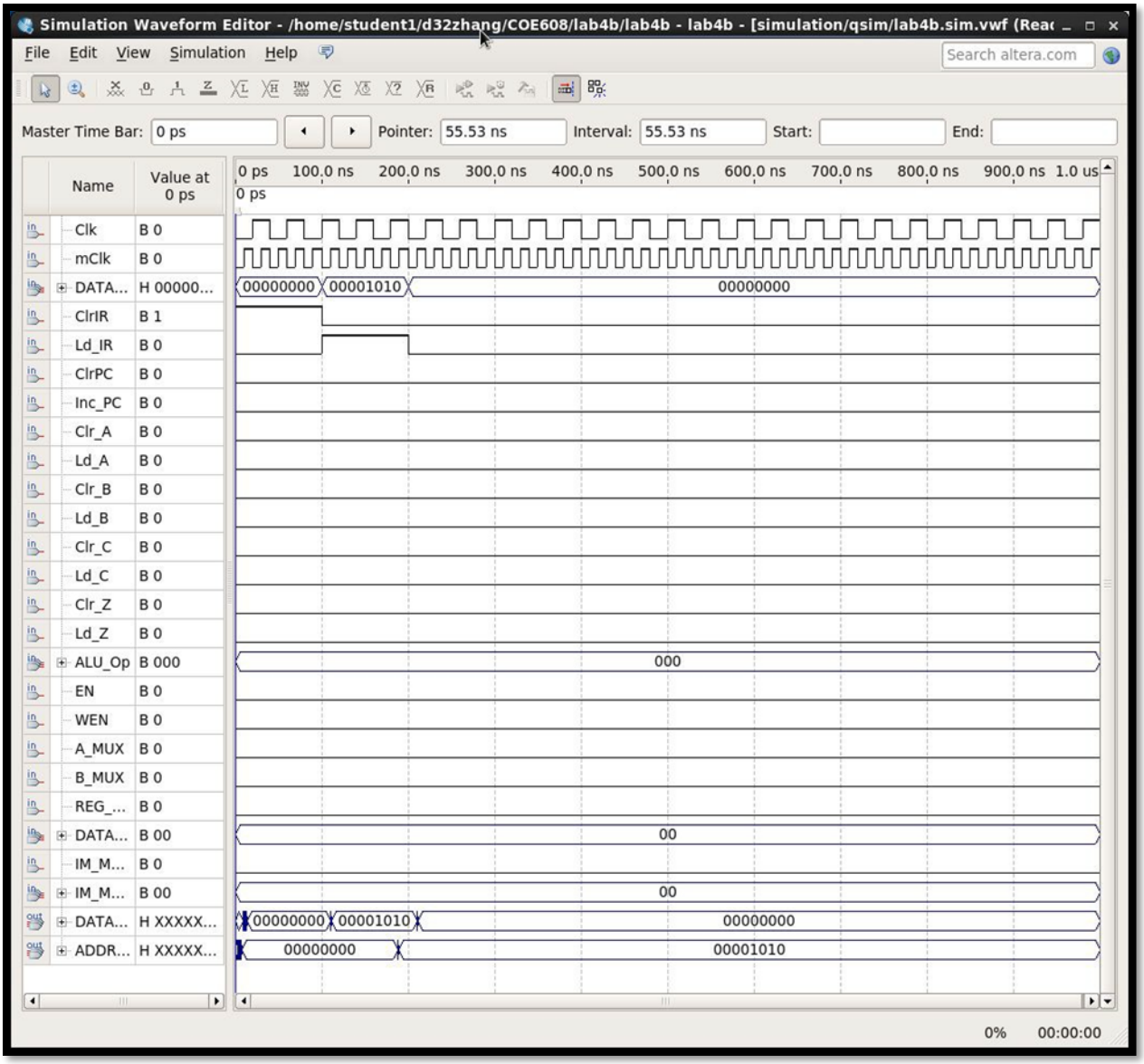


Figure 21: JMP Operation Waveform

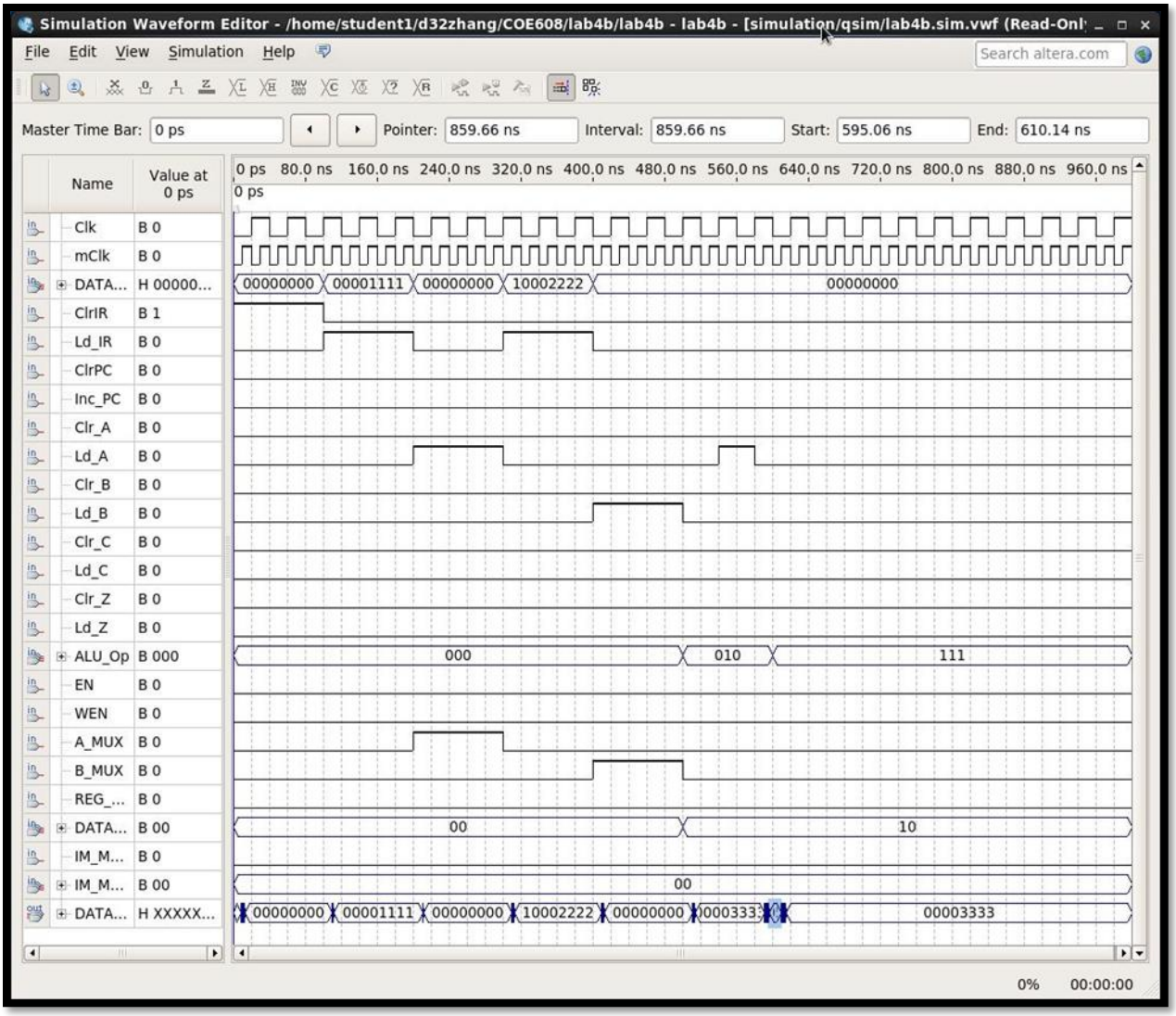


Figure 22: ADD Operation Waveform

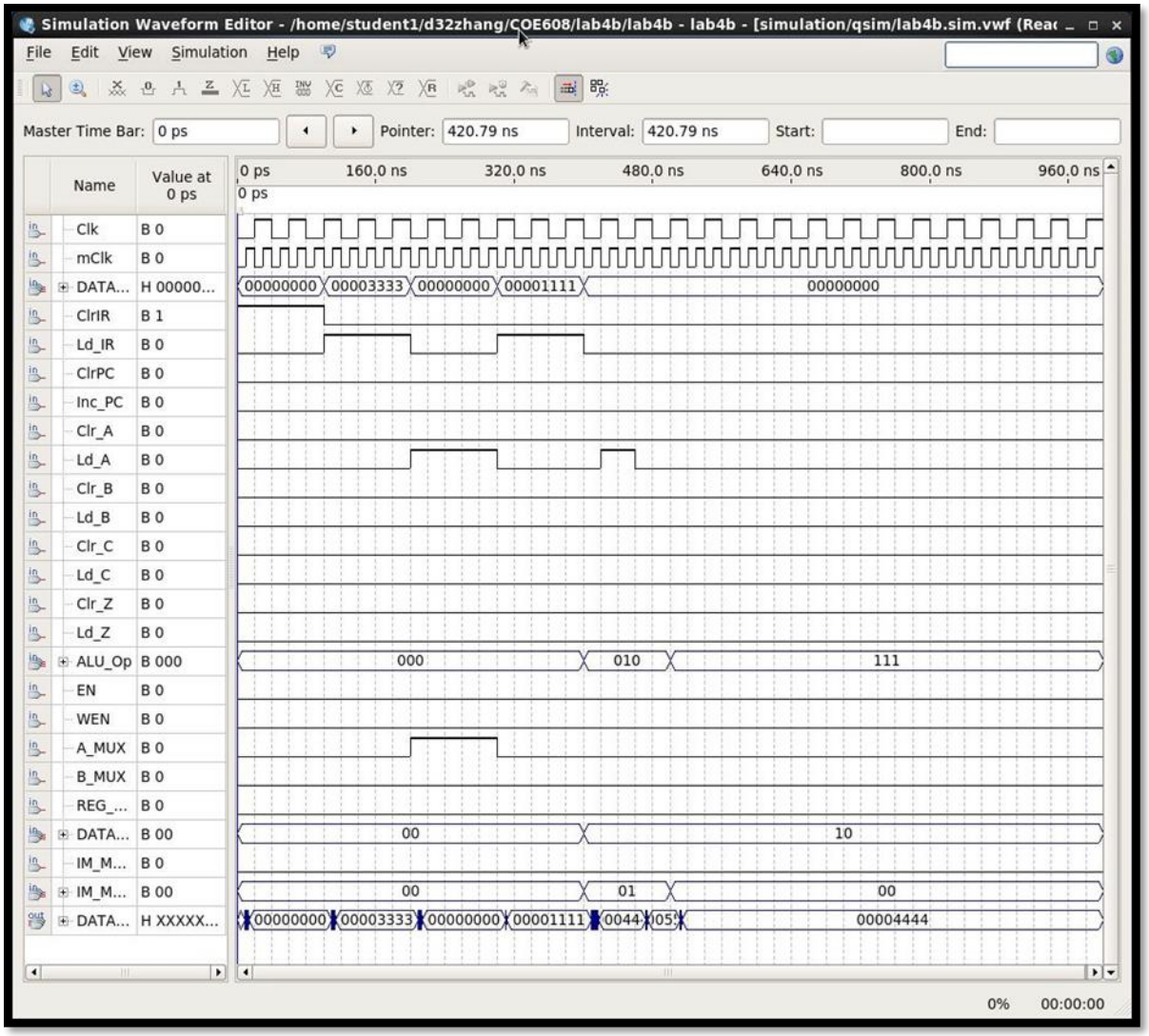


Figure 23: ADDI Operation Waveform

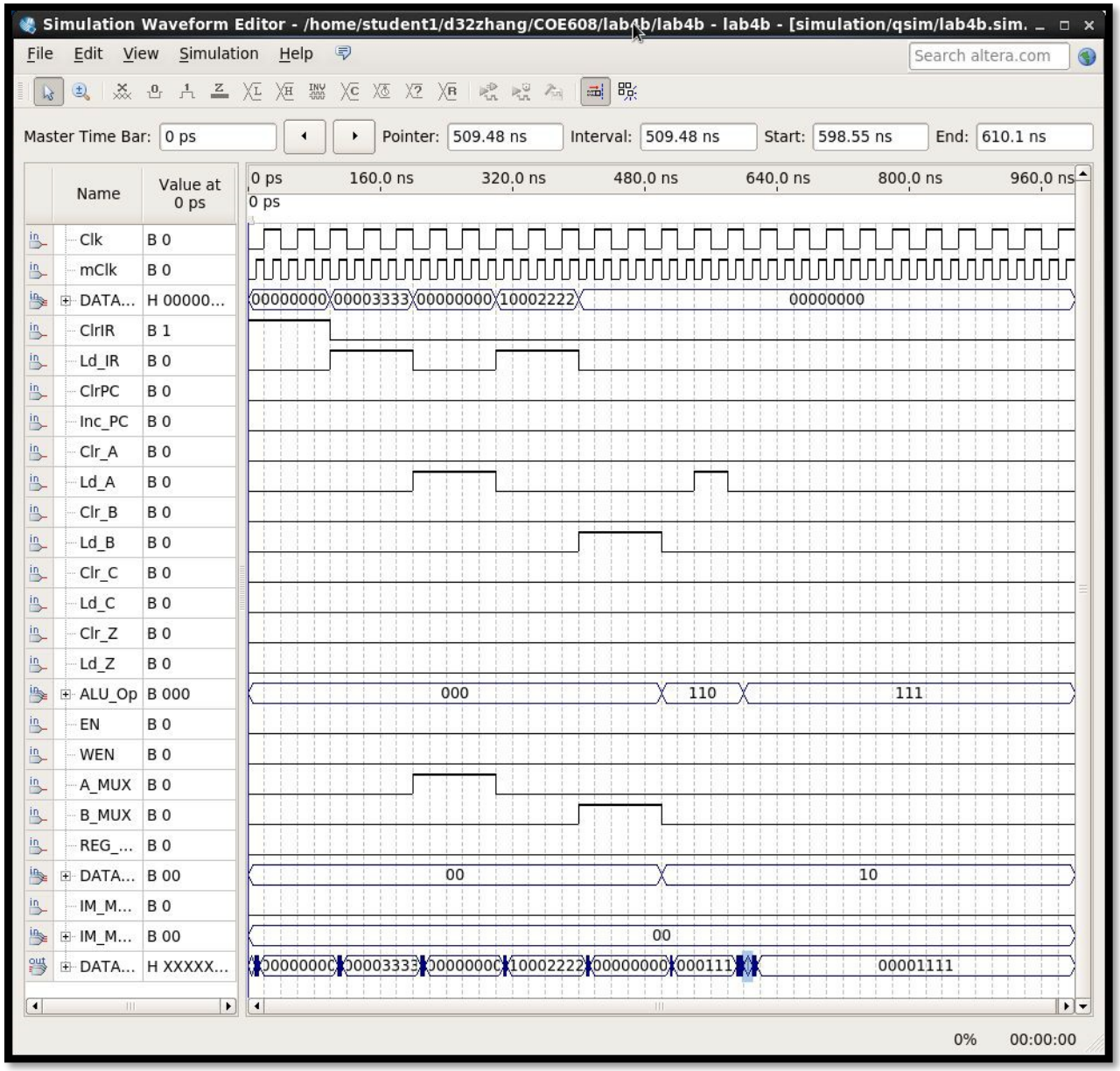


Figure 24: SUB Operation Waveform

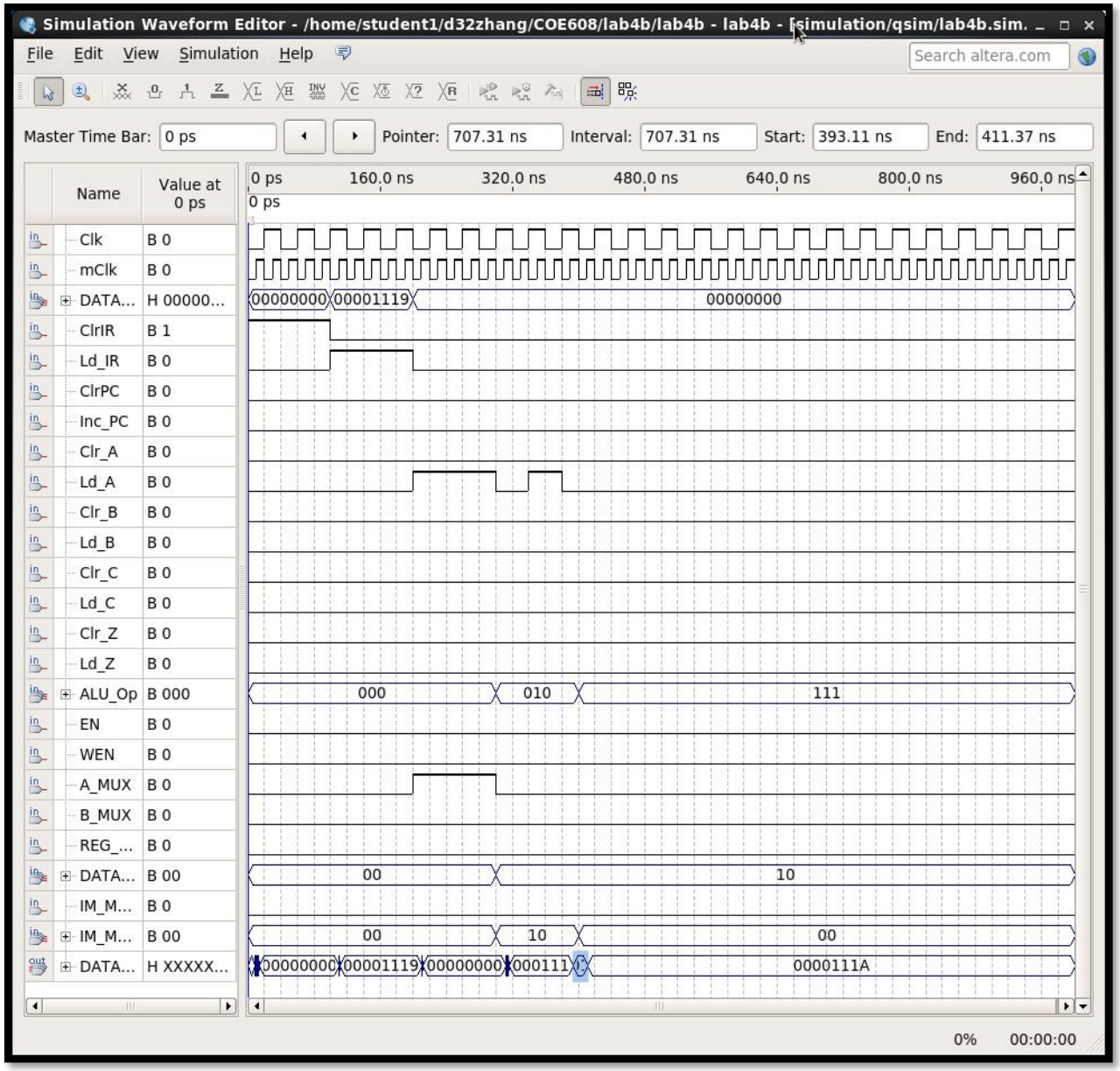


Figure 25: INCA Operation Waveform

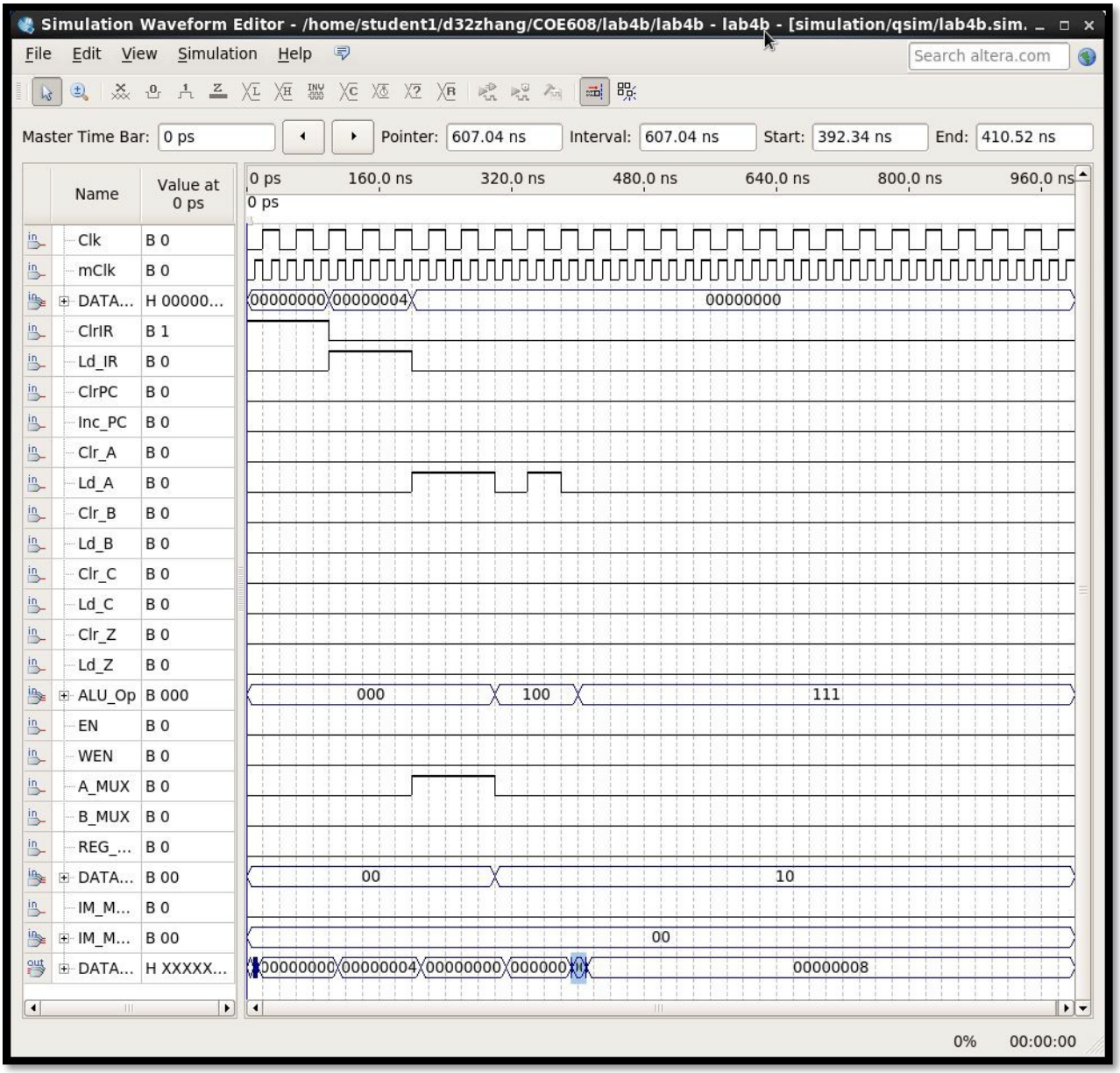


Figure 26: ROL Operation Waveform

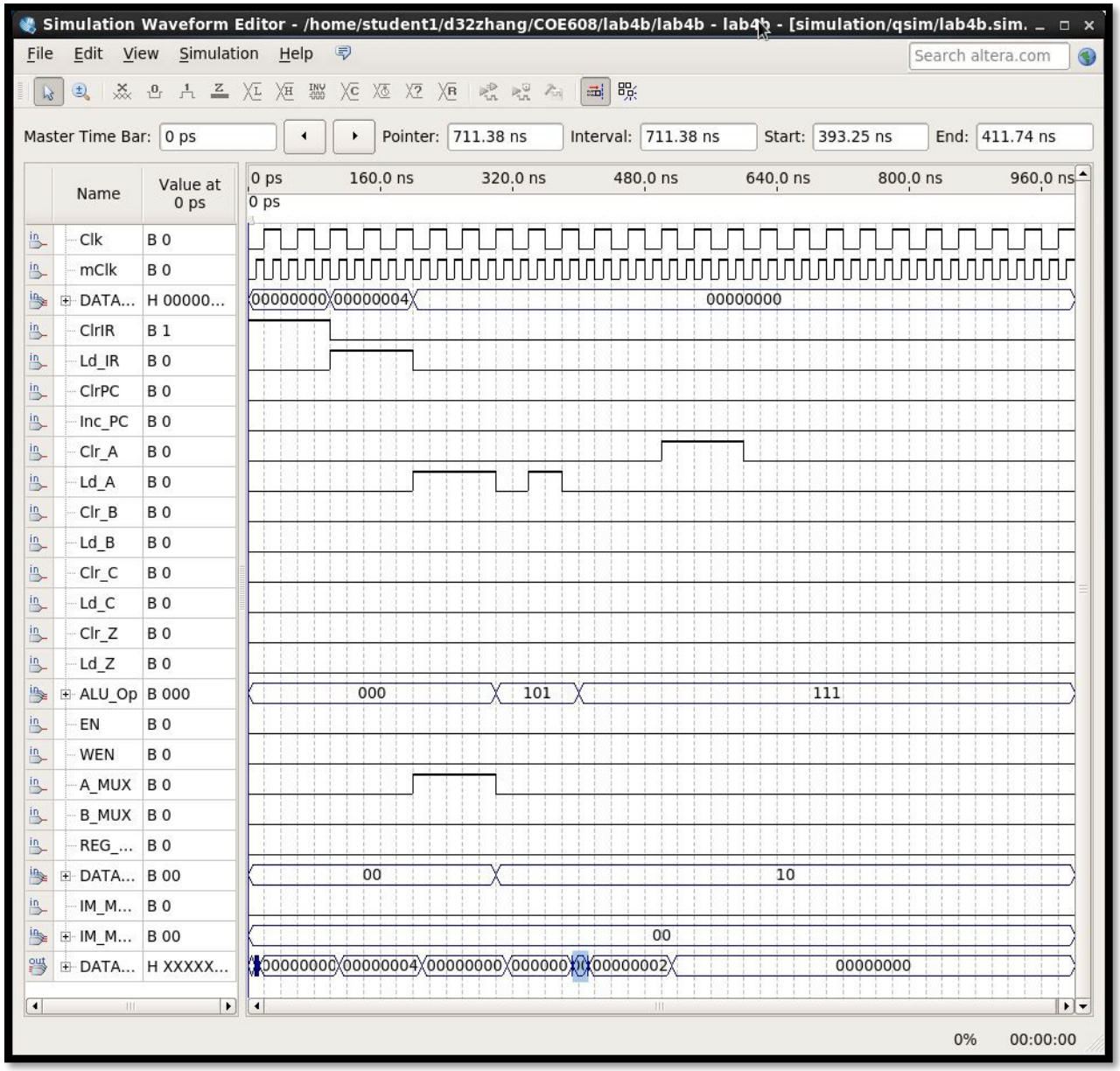


Figure 27: CLRA Operation Waveform

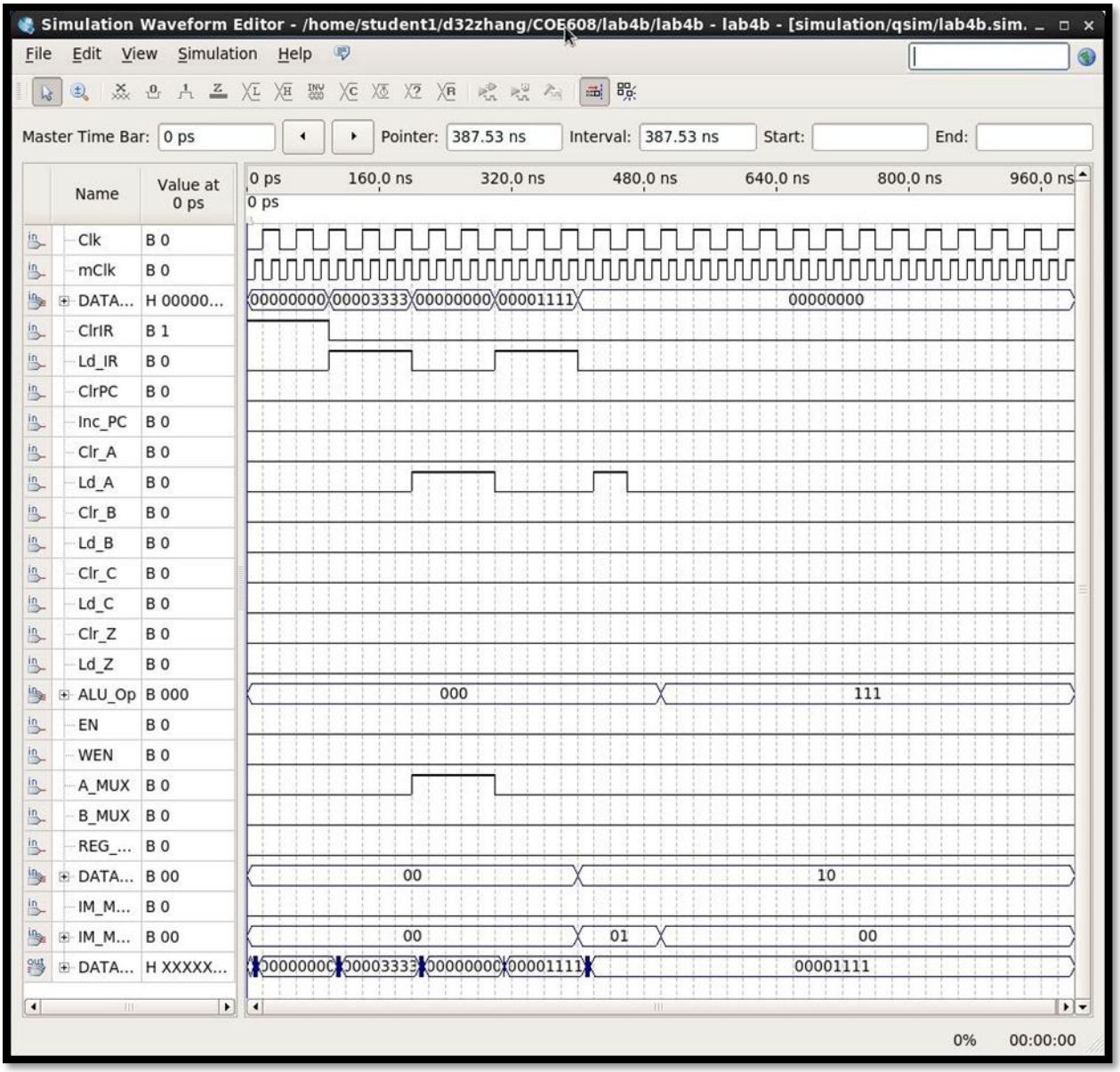


Figure 28: ANDI Operation Waveform

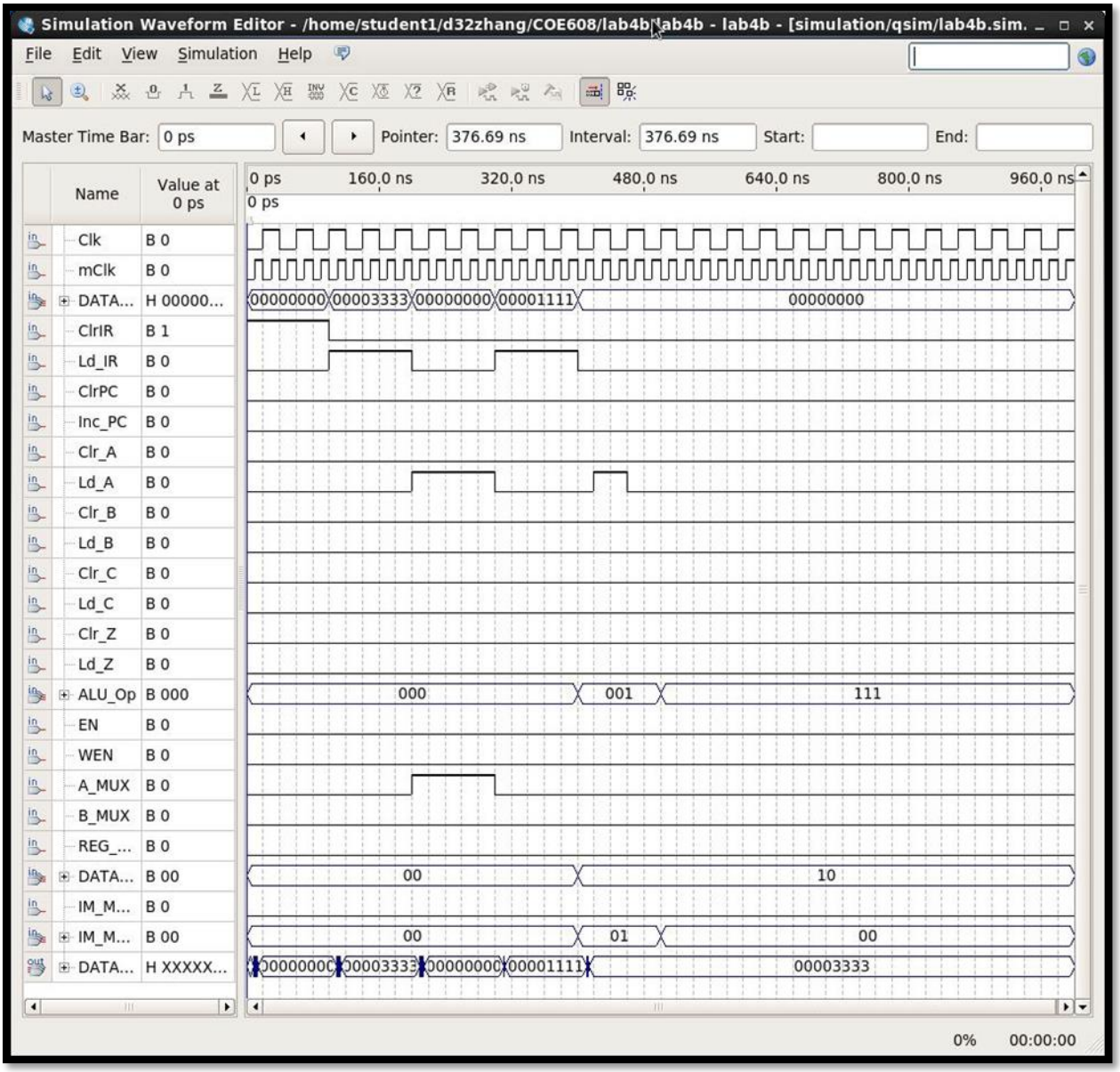


Figure 29: ORI Operation Waveform

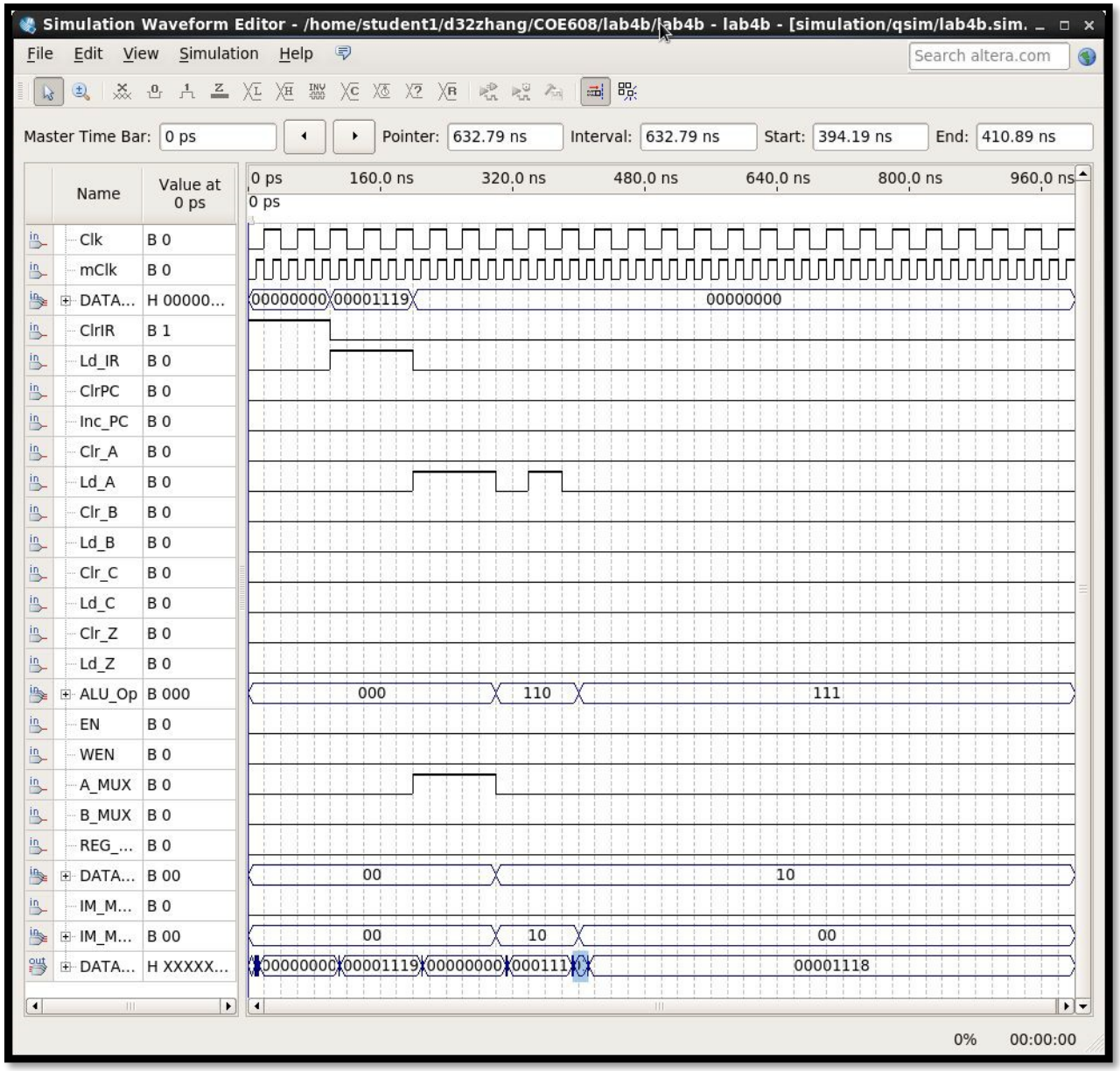


Figure 30: DECA Operation Waveform

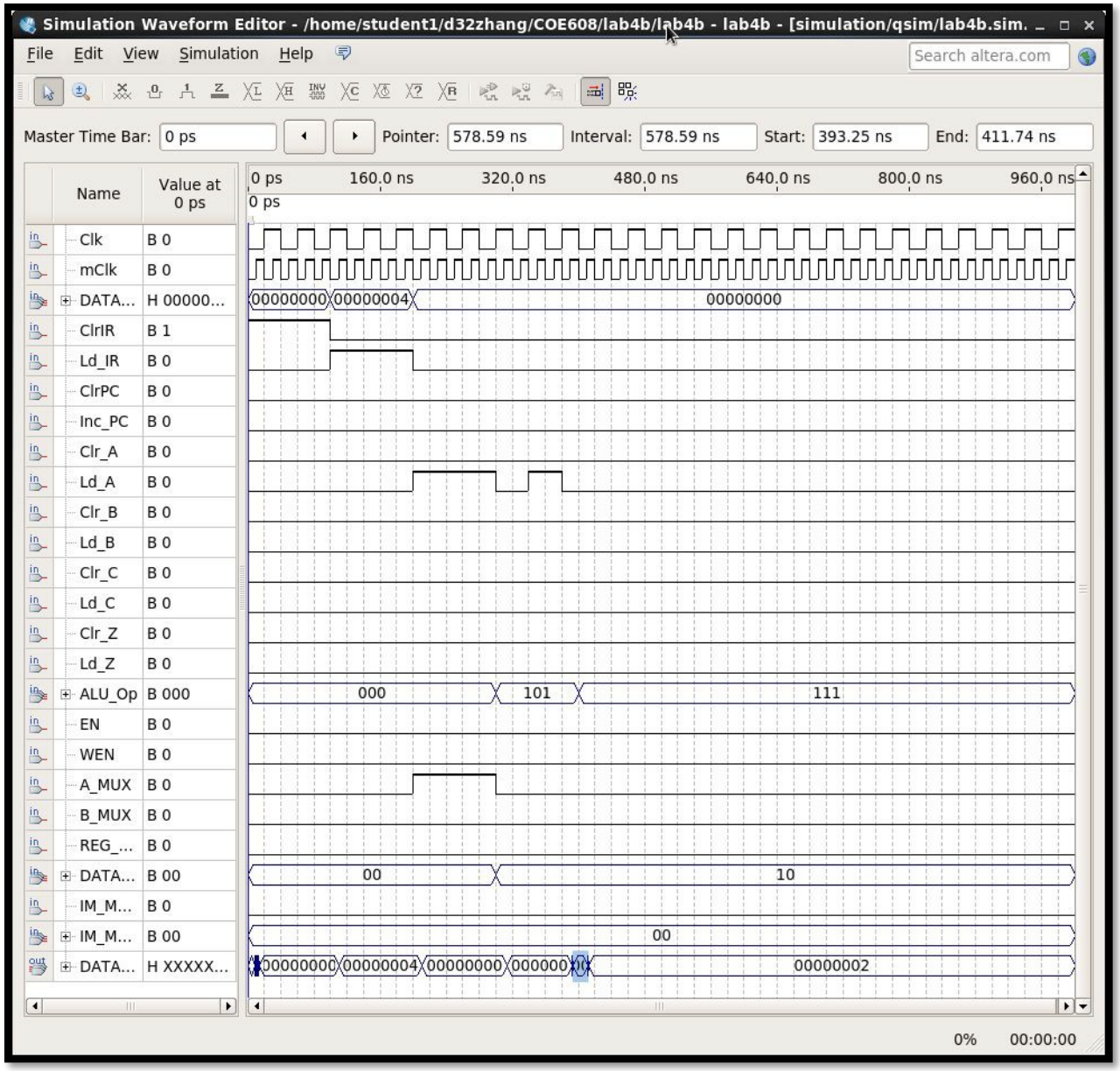


Figure 31: ROR Operation Waveform

Discussions and Conclusions

The data-path implements INCA, ADDI, LDBI and LDA operations by using a finite state machine to select these advanced operations.

The main determinant of the data-path speed is the longest operation performed. In this case, the worst-case timing depends on the operation that is most time-consuming. From Figure 13, it appears the worst-case delay is 10 ns.

The most reliable limit for the data-path clock is 10 ns.

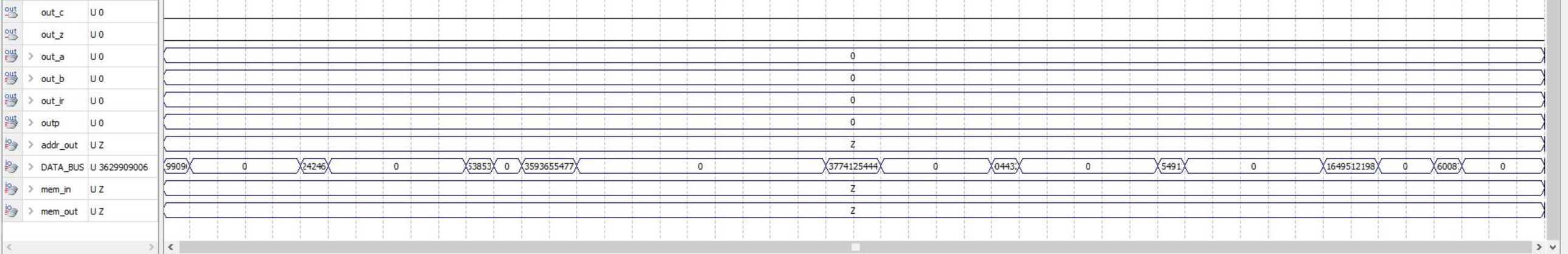
Appendix: VHDL Codes and Screenshots of Waveforms

I have provided the VHDLD and screenshots of the waveforms for easier reading. I also included Duan Wei Zhang's draft report to include the actual data that was captured in each operation waveform listed from Figure 14 to Figure 31.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity data_path is
   port(
      --clock and memory clock
      clk, mclk : in std_logic;
      --memory signals
      wen, en: in std_logic;
      --register control signals
      clr_a, ld_a: in std_logic;
      clr_b, ld_b: in std_logic;
      clr_c, ld_c: in std_logic;
      clr_z, ld_z: in std_logic;
      clr_pc, ld_pc: in std_logic;
      clr_ir, ld_ir: in std_logic;
      --register outputs
      out_a, out_b: out std_logic_vector(31 downto 0);
      out_c, out_z: out std_logic;
      outp, out_ir: out std_logic_vector(31 downto 0);
      --pc special input
      inc_pc: in std_logic;
      --address and data bus signal debug
      addr_out : inout std_logic_vector(31 downto 0);
      data_in: in std_logic_vector(31 downto 0);
      DATA_BUS, mem_out, mem_in: inout std_logic_vector(31 downto 0);
      in_mux1: in std_logic;
      in_mux2: in std_logic_vector(1 downto 0);
      --alu op
      alu_op: in std_logic_vector(2 downto 0);
      --mux controllers
      data_mux : in std_logic_vector(1 downto 0);
      reg_mux: in std_logic;
      a_mux, b_mux: in std_logic;
      im_mux1, im_mux2: in std_logic_vector(1 downto 0));
end data_path;

architecture description of data_path is
   --components

   component alu
      port(
         a, b: in std_logic_vector(31 downto 0);
         op: in std_logic_vector(2 downto 0);
         result: inout std_logic_vector(31 downto 0);
         cout: out std_logic;
         zero: out std_logic);
   end component alu;

   component register1
      port(
         d,ld,clr,clk: in std_logic;
         Q: out std_logic);
   end component register1;

   component register32
      port(
         d: in std_logic_vector(31 downto 0);
         ld,clr,clk: in std_logic;
         Q: out std_logic_vector(31 downto 0));
```

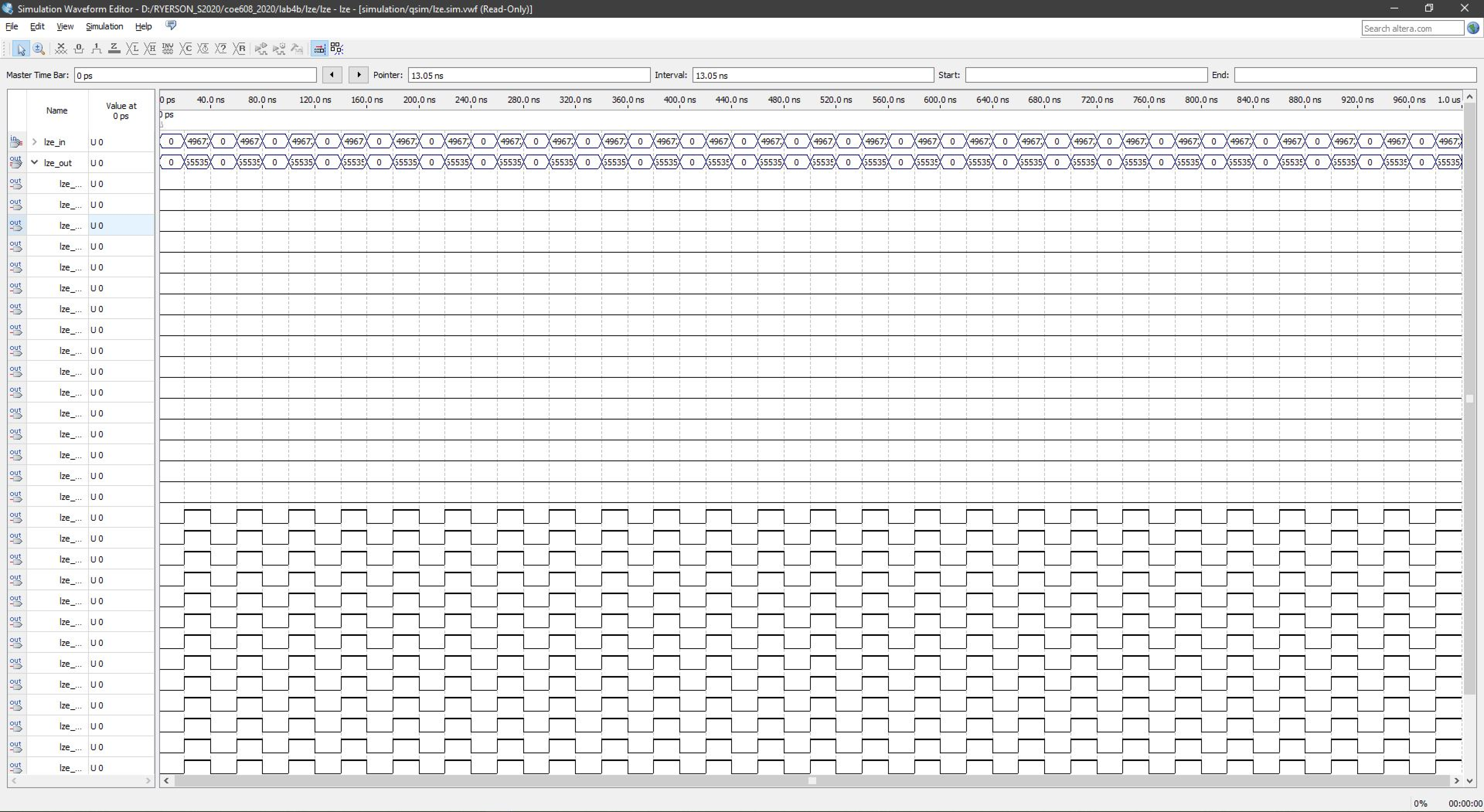
```
63      end component register32;
64
65      component pc
66         port(
67            clr, clk, ld, inc: in std_logic;
68            d: in std_logic_vector(31 downto 0);
69            q: inout std_logic_vector(31 downto 0));
70      end component pc;
71
72      component memoryModule
73         port (
74            clk: in std_logic;
75            addr: in unsigned (7 downto 0);
76            data_in : in std_logic_vector(31 downto 0);
77            wen, en: in std_logic;
78            data_out: out std_logic_vector(31 downto 0));
79      end component memoryModule;
80
81      component lze
82         port(
83            lze_in: in std_logic_vector(31 downto 0);
84            lze_out: out std_logic_vector(31 downto 0));
85      end component lze;
86
87      component uze
88         port(
89            uze_in : in std_logic_vector(31 downto 0);
90            uze_out: out std_logic_vector(31 downto 0));
91      end component uze;
92
93      component red
94         port(
95            red_in : in std_logic_vector(31 downto 0);
96            red_out : out unsigned(7 downto 0));
97      end component red;
98
99      component mux2to1
100        port (
101           s : in std_logic;
102           w0, w1 : in std_logic_vector(31 downto 0);
103           f: out std_logic_vector(31 downto 0));
104     end component mux2to1;
105
106     component mux4to1
107        port (
108           s : in std_logic_vector(1 downto 0);
109           x0, x1, x2, x3 : in std_logic_vector(31 downto 0);
110           f : out std_logic_vector(31 downto 0));
111     end component mux4to1;
112
113     --signals
114
115     signal IR_out,data_bus1,lze_pc_out,lze_Amux_out,lze_Bmux_out,Amux_out, Bmux_out,
    registerA_out, registerB_out, registerMux_out, memory_out : std_logic_vector(31 downto 0);
116     signal uze_imMUX1_out, imMUX1_out, lze_imMUX2_out, imMUX2_out, alu_out : std_logic_vector
    (31 downto 0);
117     signal red_DataMem_out: unsigned (7 downto 0);
118     signal zero_flag, carry_flag : std_logic;
119     signal temp : std_logic_vector(30 downto 0) := (others => '0');
120        begin
121           ir_register : register32 port map (data_bus, ld_ir, clr_ir, clk, IR_out);
122           a_register: register32 port map (Amux_out, ld_a, clr_a, clk, registerA_out);
```

```
123           b_register: register32 port map(Bmux_out, ld_b, clr_b, clk, registerB_out);
124           alu_component: alu port map(imMuX1_out, imMUX2_out, alu_op, alu_out, carry_flag,
      zero_flag);
125           memory_component: memoryModule port map (mclk, RED_DataMem_out, registerMux_out,
      wen, en, memory_out);
126           DATA_BUS <= data_bus1;
127           lze_pc : lze port map (IR_out, lze_pc_out);
128           lze_A_mux : lze port map (IR_out, lze_Amux_out);
129           A_mux0 : mux2to1 port map (a_mux, data_bus, lze_Amux_out, Amux_out);
130           lze_B_mux : lze port map (IR_out, lze_Bmux_out);
131           B_mux0: mux2to1 port map (b_mux, data_bus, lze_Bmux_out, Bmux_out);
132           reg_mux0: mux2to1 port map (reg_mux, registerA_out, registerB_out, registerMux_out
      );
133           RED_DATA_MEM: red port map (IR_out, red_DataMem_out);
134           UZE_IM_MUX1: uze port map (IR_out, UZE_imMUX1_out);
135           IM_MUX1a : mux2to1 port map (in_mux1,registerA_out, UZE_imMUX1_out, imMUX1_out);

136           LZE_IM_MUX2 : lze port map (IR_out, LZE_imMUX2_out);
137           IM_MUX2a : mux4to1 port map (in_mux2, registerB_out, LZE_imMUX2_out, (temp & '1'),
      (others => '0'), imMUX2_out);
138           DATA_MUX0: mux4to1 port map (data_mux, data_in, memory_out, alu_out, (others => '0'
      ), data_bus1);
139      end description;
140
141
142
143
144
145
146
```


out_c U 0
out_z U 0
out_a U 0
out_b U 0
out_jr U 0
outp U 0
addr_out U Z
DATA_BUS U 3629909006
mem_in U Z
mem_out U Z
0
0
0
0
Z
9909
0
24246
0
53853
0
3593655477
0
3774125444
0
0443
0
5491
0
1649512198
0
6008
0
Z
Z


out_c U 0
out_z U 0
out_a U 0
out_b U 0
out_ir U 0
outp U 0
addr_out U Z
DATA_BUS U X
mem_in U Z
mem_out U Z
0
0
0
0
Z
X
3909
0
0
424
0
385
0
59365547
0
3774125444
0
443
0
5491
0
649512198
0
008
0
Z
Z

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_unsigned.all;
4
5   entity lze is
6      port(
7         lze_in: in std_logic_vector(31 downto 0);
8         lze_out: out std_logic_vector(31 downto 0));
9   end lze;
10
11  architecture behavior of lze is
12     signal zeros : std_logic_vector(31 downto 16) := (others => '0');
13        begin
14           lze_out <= zeros & lze_in(15 downto 0);
15  end behavior;
16
17
```


Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/lze/lze - lze - [simulation/qsim/lze.sim.vwf (Read-Only)]
File
Edit
View
Simulation
Help
Search altera.com
Master Time Bar:
0 ps
Pointer:
13.05 ns
Interval:
13.05 ns
Start:
End:
Name
Value at 0 ps
0 ps
40.0 ns
80.0 ns
120.0 ns
160.0 ns
200.0 ns
240.0 ns
280.0 ns
320.0 ns
360.0 ns
400.0 ns
440.0 ns
480.0 ns
520.0 ns
560.0 ns
600.0 ns
640.0 ns
680.0 ns
720.0 ns
760.0 ns
800.0 ns
840.0 ns
880.0 ns
920.0 ns
960.0 ns
1.0 us
lze_in
U 0
lze_out
U 0
lze_...
U 0
0%
00:00:00

Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/lze/lze - lze - [simulation/qsim/lze.sim.vwf (Read-Only)]

File Edit View Simulation Help

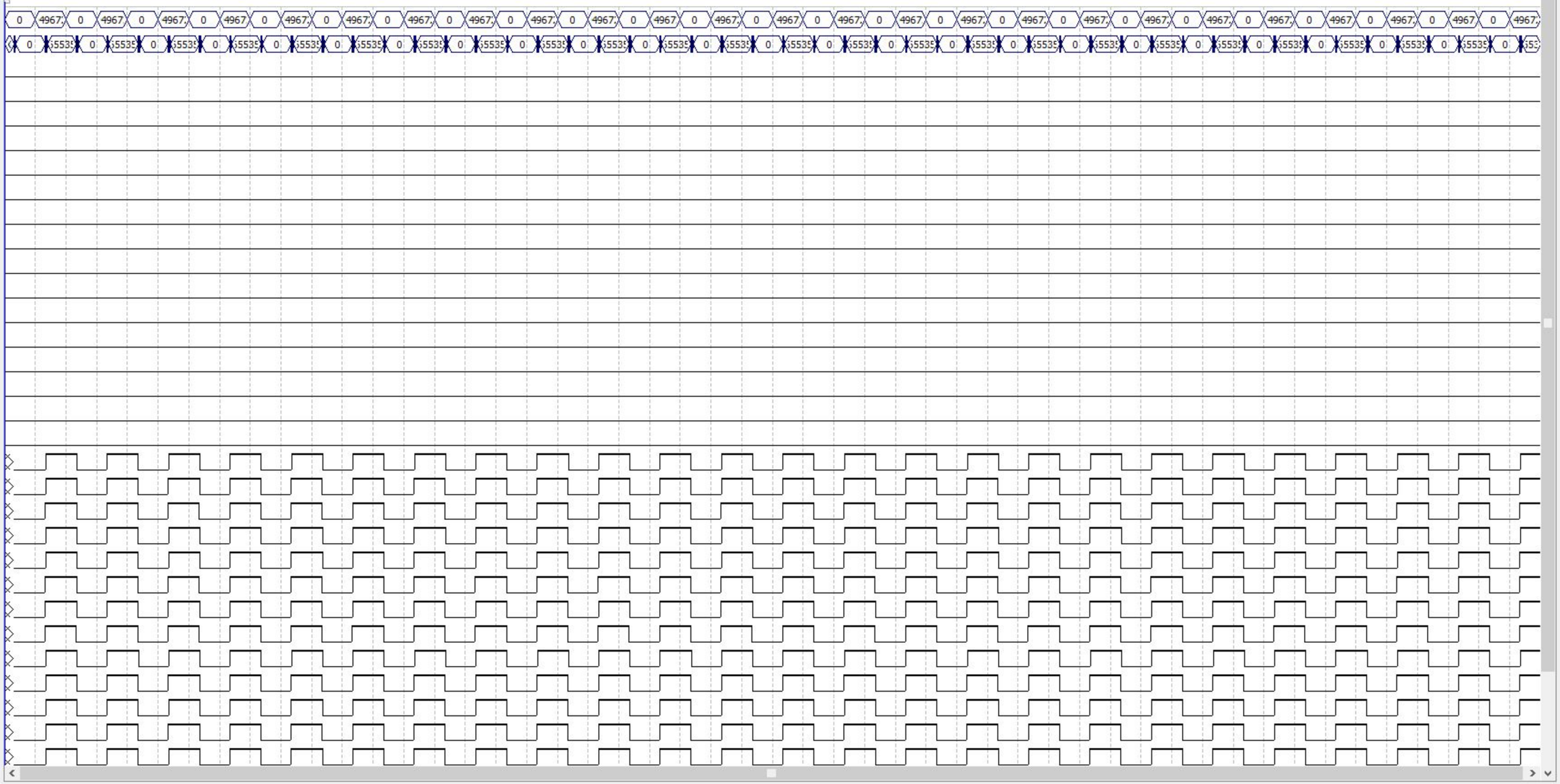
Search altera.com

Master Time Bar: 0 ps Pointer: Interval: Start: End:

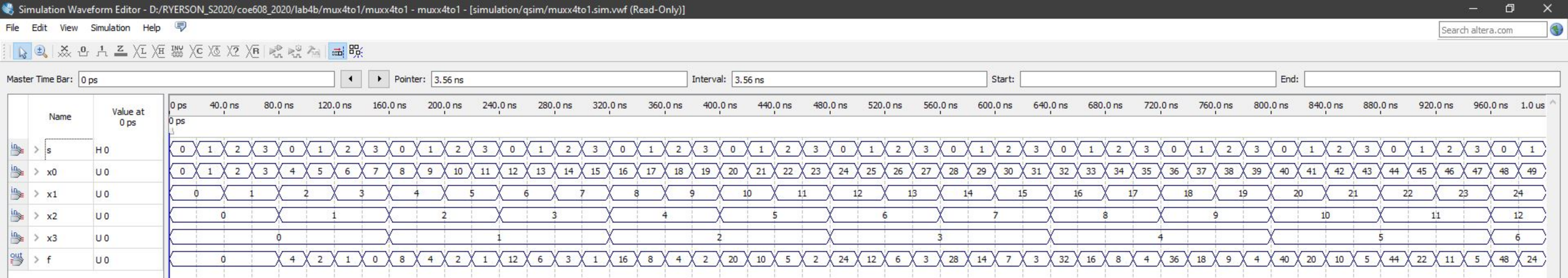
Name	Value at 0 ps
lze_in	U 0
lze_out	U X
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X

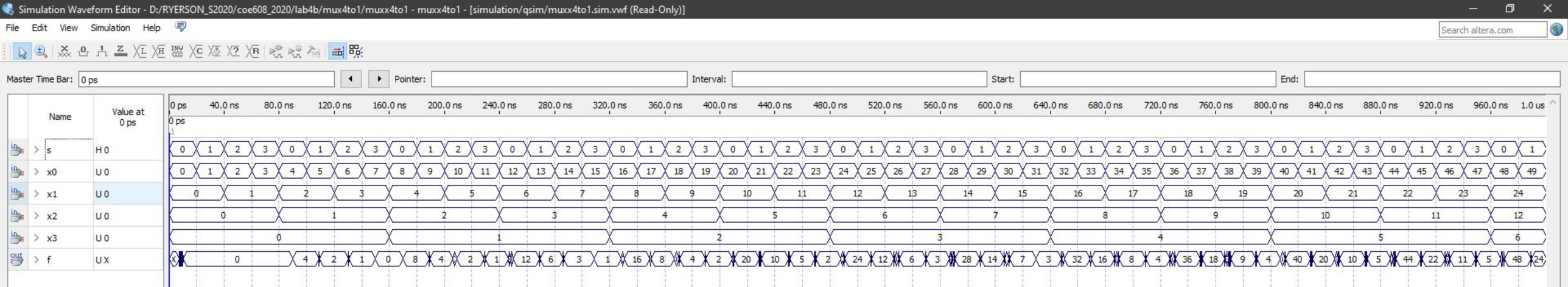
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us

0 ps



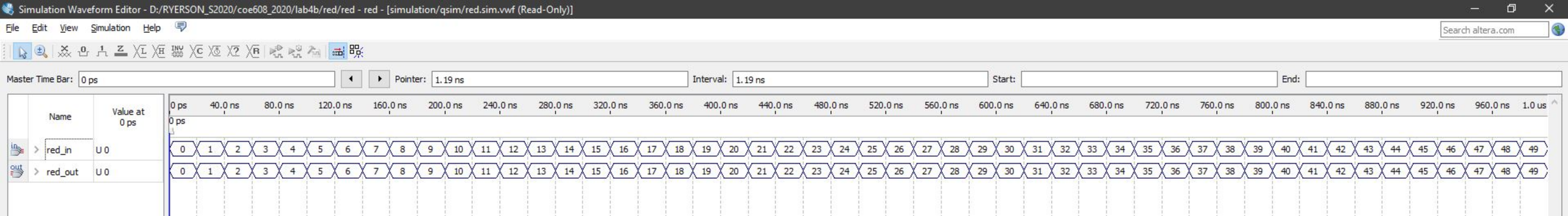
0% 00:00:00

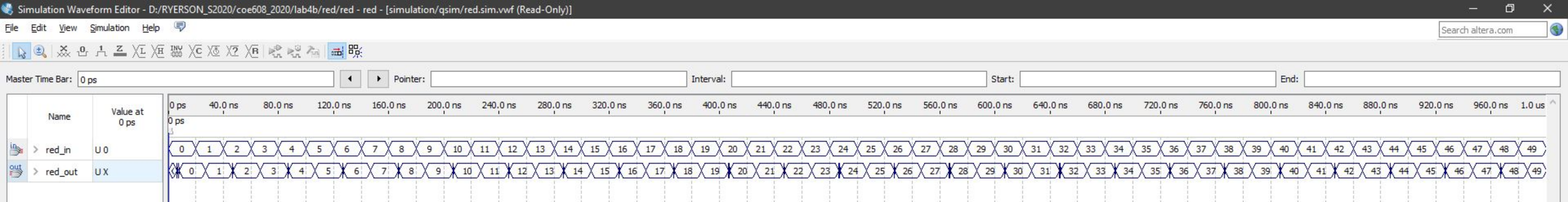

Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/mux4to1/muxx4to1 - muxx4to1 - [simulation/qsim/muxx4to1.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 3.56 ns
Interval: 3.56 ns
Start:
End:
Name
Value at 0 ps
s H 0
x0 U 0
x1 U 0
x2 U 0
x3 U 0
f U 0
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us


Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/mux4to1/muxx4to1 - muxx4to1 - [simulation/qsim/muxx4to1.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer:
Interval:
Start:
End:
Name
Value at 0 ps
s H 0
x0 U 0
x1 U 0
x2 U 0
x3 U 0
f U X
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us

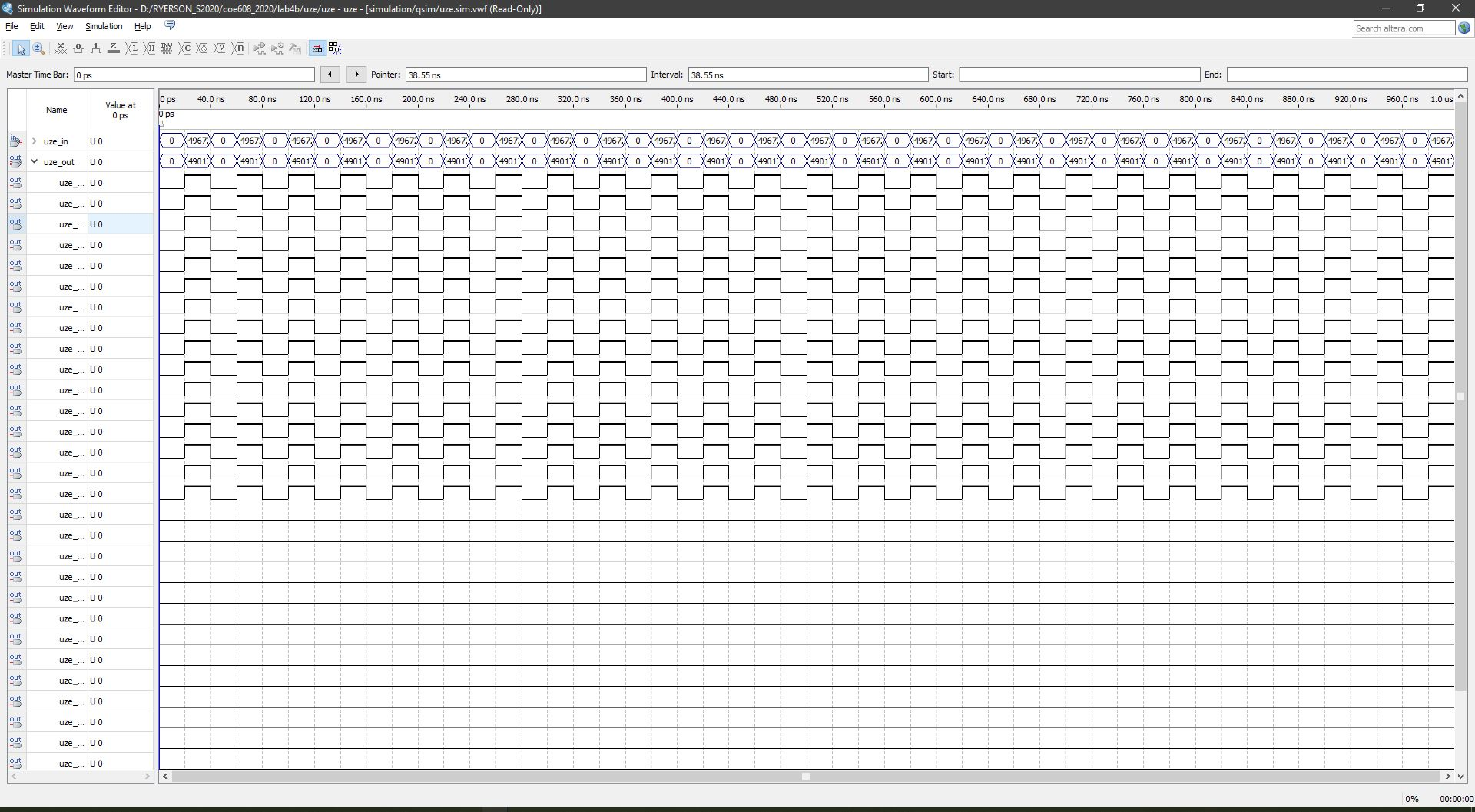
```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_unsigned.all;
4
5   entity mux4to1 is
6      port
7         (s : in std_logic_vector(1 downto 0);
8         x0, x1, x2, x3 : in std_logic_vector(31 downto 0);
9         f : out std_logic_vector(31 downto 0));
10  end mux4to1;
11
12  architecture behavior of mux4to1 is
13     begin
14        with s select
15           f <= x0 when "00",
16                x1 when "01",
17                x2 when "10",
18                x3 when "11";
19  end behavior;
20
21
22
```

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.numeric_std.all;
4
5   entity red is
6      port(
7         red_in : in std_logic_vector(31 downto 0);
8         red_out : out unsigned(7 downto 0));
9   end red;
10
11  architecture behavior of red is
12     begin
13        red_out <= unsigned(red_in(7 downto 0));
14  end behavior;
15
16
```

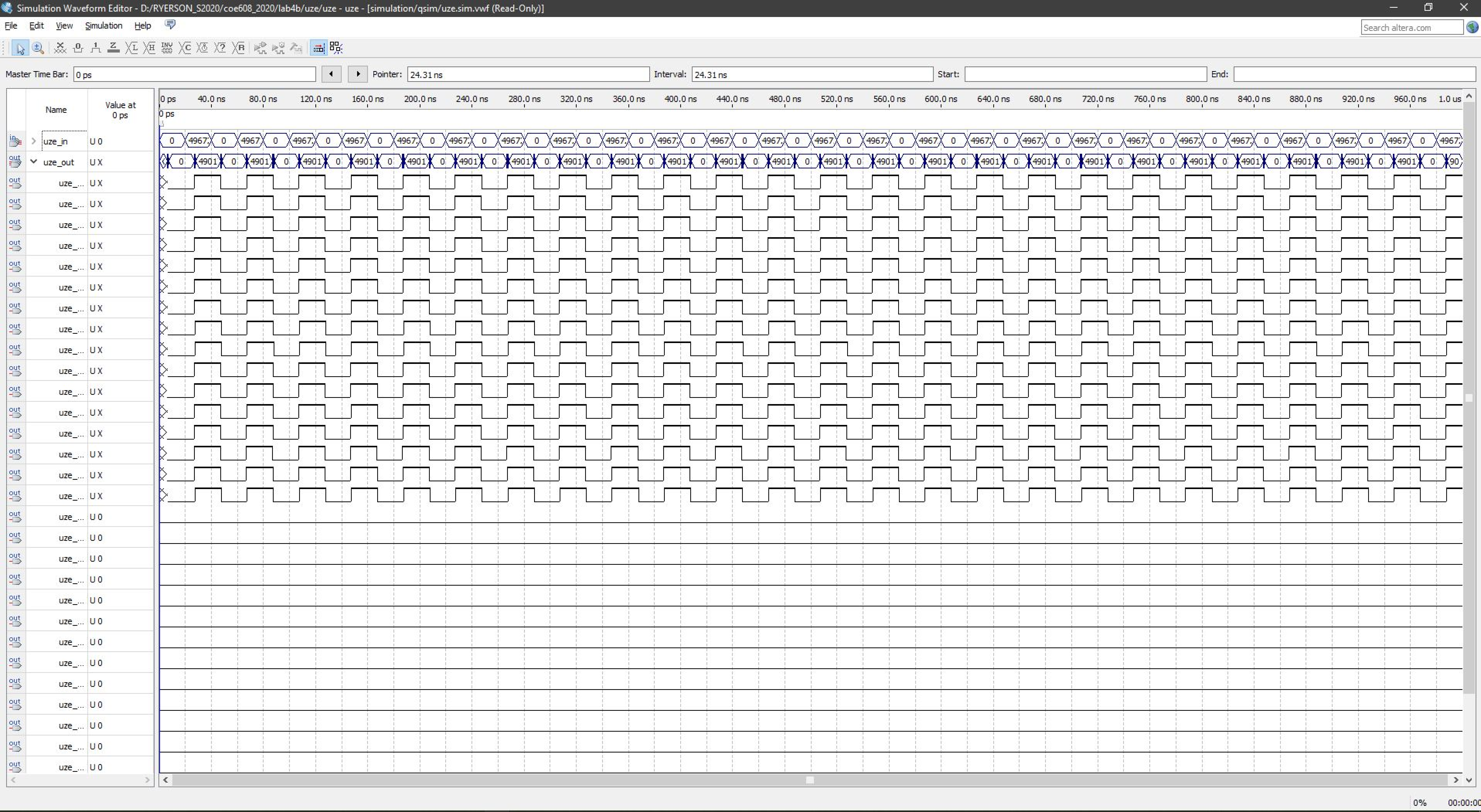

Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/red/red - red - [simulation/qsim/red.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 1.19 ns
Interval: 1.19 ns
Start:
End:
Name
Value at 0 ps
red_in
U 0
red_out
U 0
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us
0 ps
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49


Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/red/red - red - [simulation/qsim/red.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer:
Interval:
Start:
End:
Name
Value at 0 ps
red_in
U 0
red_out
U X
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us
0 ps
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
X 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_unsigned.all;
4
5   entity uze is
6      port(
7         uze_in : in std_logic_vector(31 downto 0);
8         uze_out: out std_logic_vector(31 downto 0));
9   end uze;
10
11  architecture behavior of uze is
12     signal zeros : std_logic_vector(15 downto 0) := (others => '0');
13        begin
14           uze_out <= uze_in(31 downto 16) & zeros;
15  end behavior;
16
17
```



Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/uze/uze - uze - [simulation/qsim/uze.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 38.55 ns
Interval: 38.55 ns
Start:
End:
Name
Value at 0 ps
uze_in U 0
uze_out U 0
uze_... U 0
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us
0
4967
4901
0%
00:00:00


Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/uze/uze - uze - [simulation/qsim/uze.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 24.31 ns
Interval: 24.31 ns
Start:
End:
Name
Value at 0 ps
0 ps
40.0 ns
80.0 ns
120.0 ns
160.0 ns
200.0 ns
240.0 ns
280.0 ns
320.0 ns
360.0 ns
400.0 ns
440.0 ns
480.0 ns
520.0 ns
560.0 ns
600.0 ns
640.0 ns
680.0 ns
720.0 ns
760.0 ns
800.0 ns
840.0 ns
880.0 ns
920.0 ns
960.0 ns
1.0 us
uze_in U 0
uze_out U X
0
4967
4901
0%
00:00:00

Department of Electrical, Computer
& Biomedical Engineering
Faculty of Engineering & Architectural Science

Programs: Computer Engineering

Course Number	**COE608**
Course Title	**Computer Organization and Architecture**
Semester/Year	**Winter 2020**
Instructor	**Nagi Mekhiel**

Lab Report No. 4b

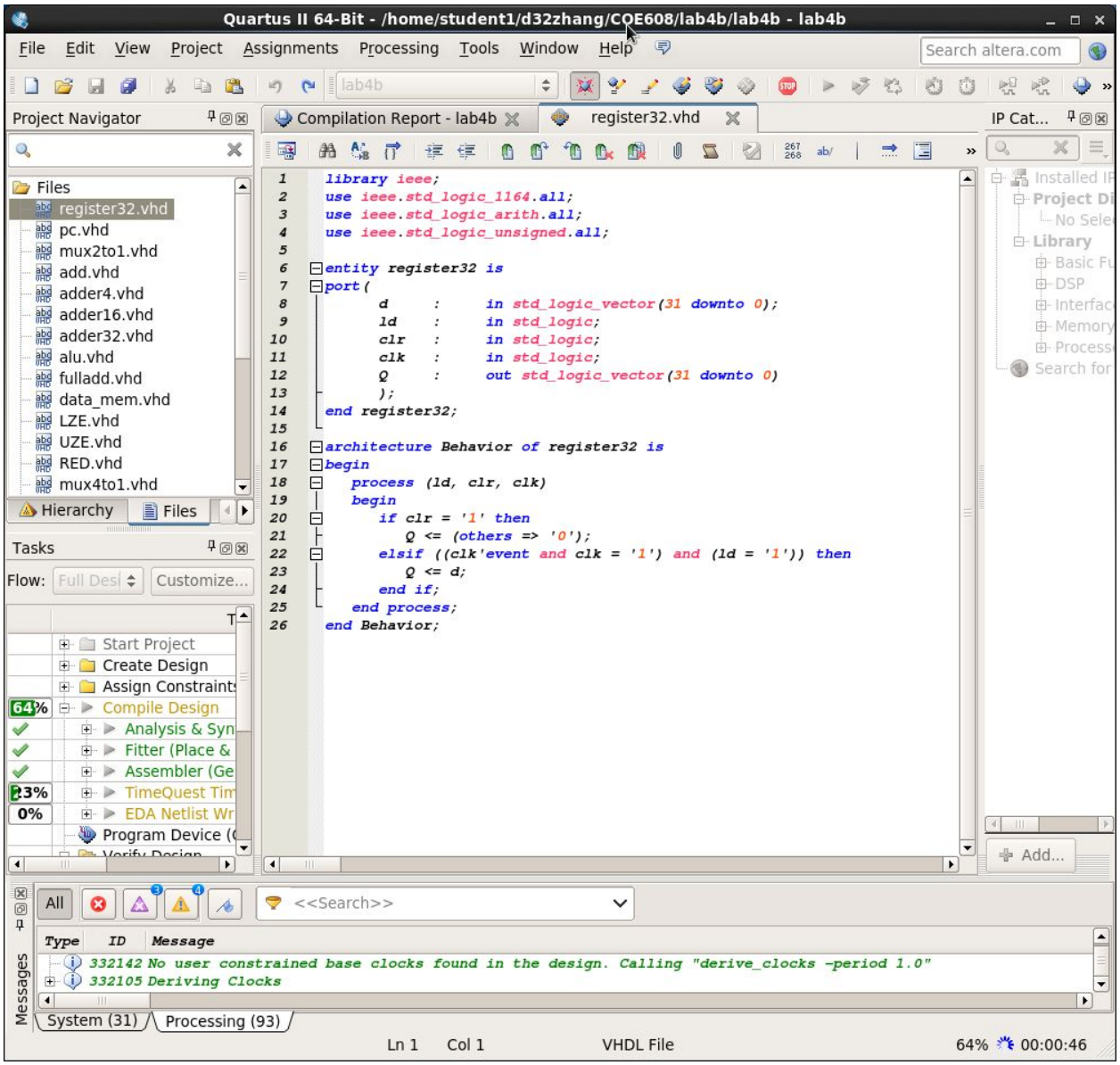
Lab Title	**Data-Path Design**

Section No.	**04**
Group No.	
Submission Date	**03/10/2020**
Due Date	**03/10/2020**

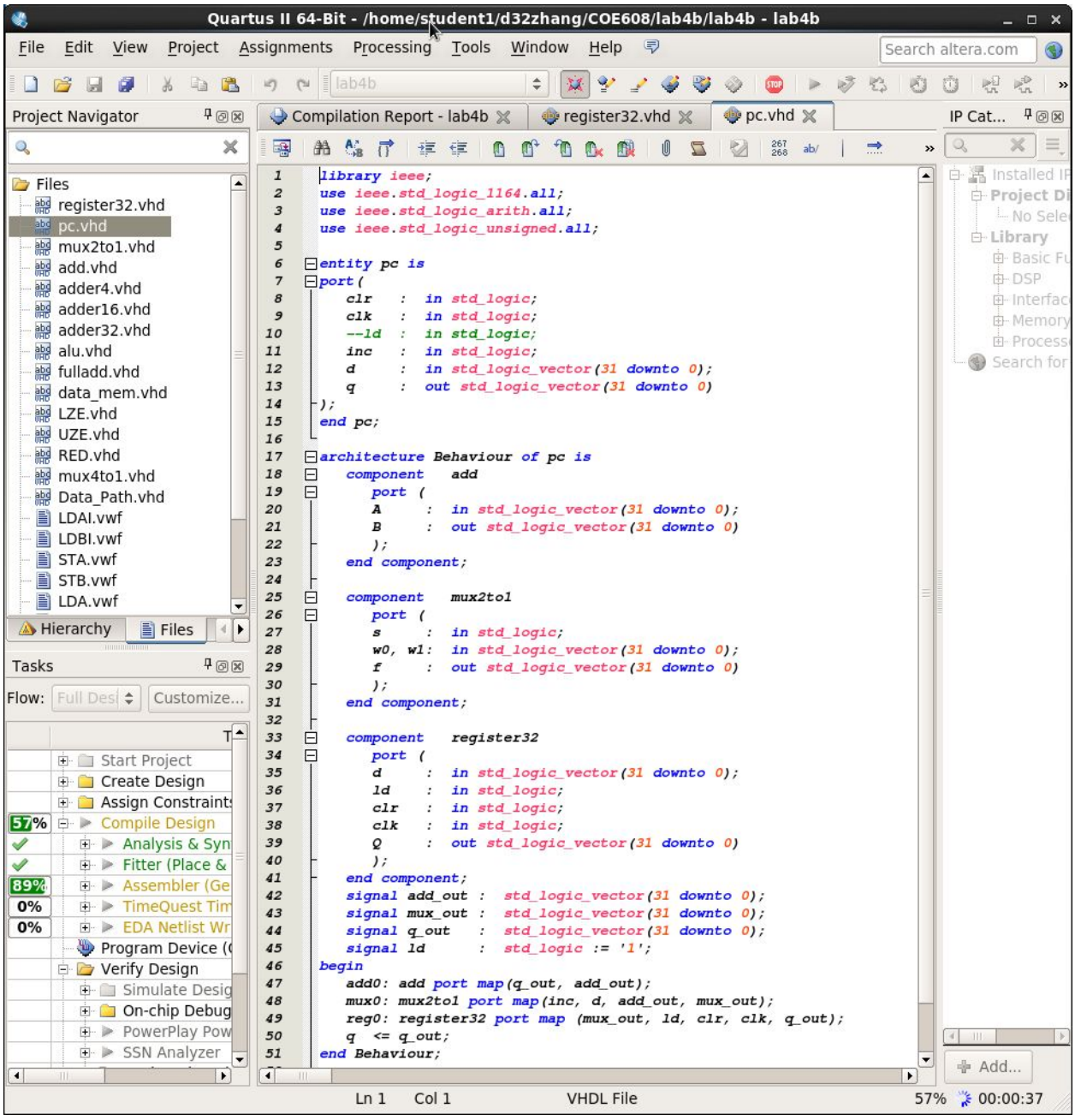
Name	Student ID	Signature*
Duanwei Zhang	*500824903*	

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*
www.ryerson.ca/senate/current/pol60.pdf.

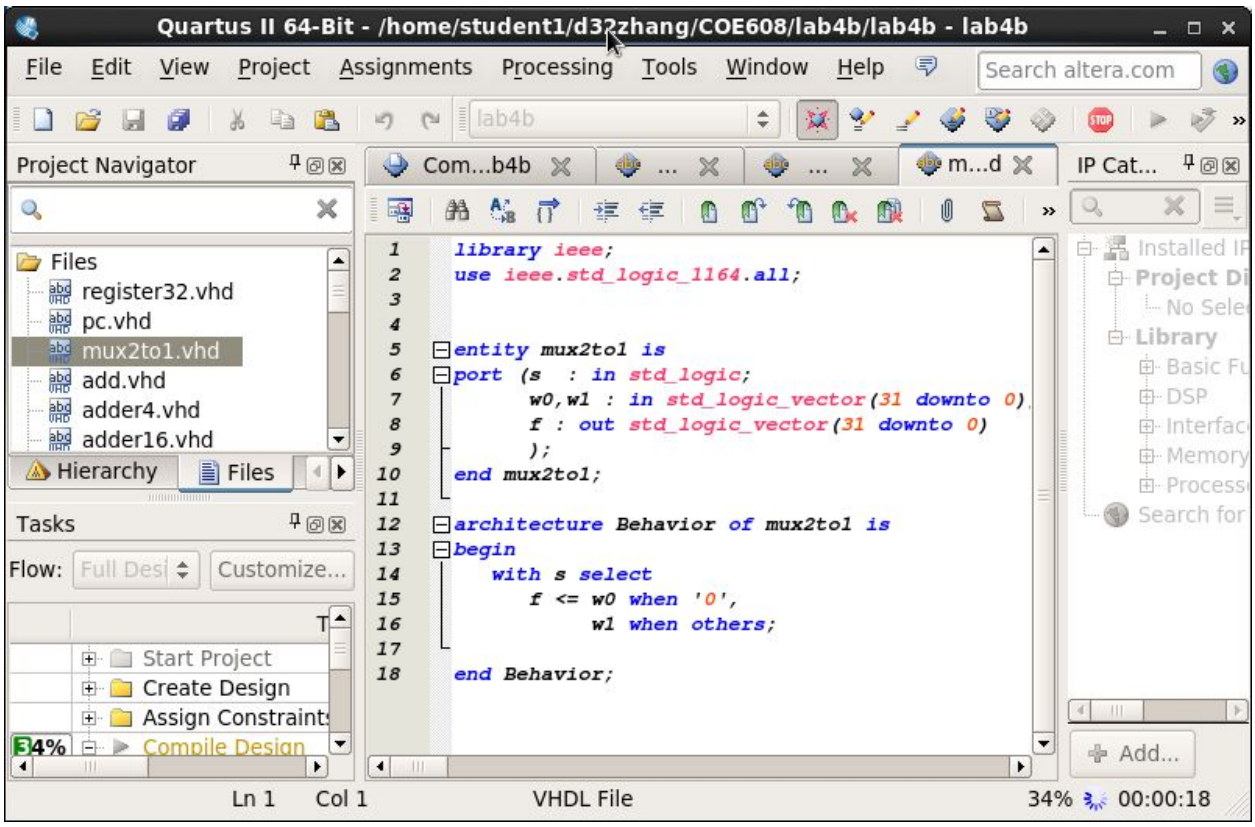
1. Register32.vhd



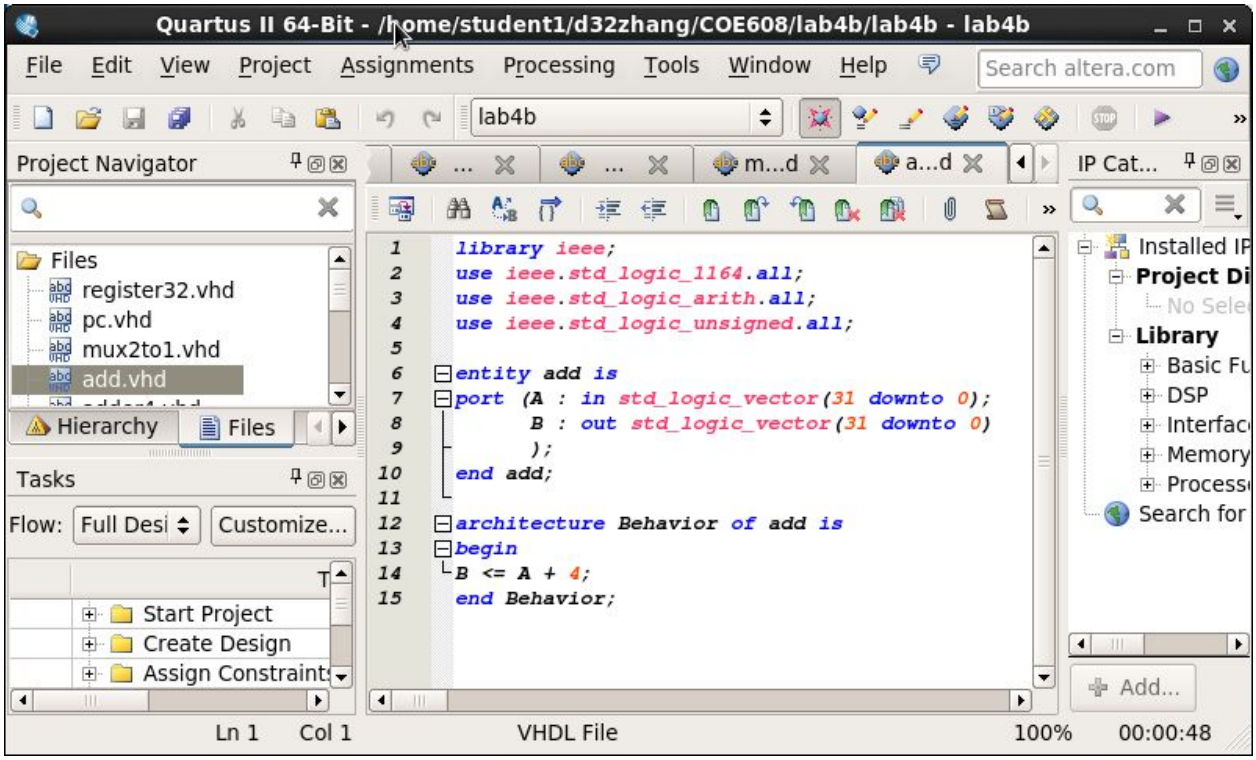
2. Pc.vhd


Quartus II 64-Bit - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Project Navigator
Compilation Report - lab4b
register32.vhd
pc.vhd
IP Cat...
Files
register32.vhd
pc.vhd
mux2to1.vhd
add.vhd
adder4.vhd
adder16.vhd
adder32.vhd
alu.vhd
fulladd.vhd
data_mem.vhd
LZE.vhd
UZE.vhd
RED.vhd
mux4to1.vhd
Data_Path.vhd
LDAI.vwf
LDBI.vwf
STA.vwf
STB.vwf
LDA.vwf
Hierarchy
Files
Tasks
Flow:
Customize...
Start Project
Create Design
Assign Constraints
57%
Compile Design
Analysis & Syn
Fitter (Place &
89%
Assembler (Ge
0%
TimeQuest Tim
0%
EDA Netlist Wr
Program Device
Verify Design
Simulate Desig
On-chip Debug
PowerPlay Pow
SSN Analyzer
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.std_logic_arith.all;
4 use ieee.std_logic_unsigned.all;
5
6 entity pc is
7 port(
8 clr : in std_logic;
9 clk : in std_logic;
10 --ld : in std_logic;
11 inc : in std_logic;
12 d : in std_logic_vector(31 downto 0);
13 q : out std_logic_vector(31 downto 0)
14);
15 end pc;
16
17 architecture Behaviour of pc is
18 component add
19 port (
20 A : in std_logic_vector(31 downto 0);
21 B : out std_logic_vector(31 downto 0)
22);
23 end component;
24
25 component mux2to1
26 port (
27 s : in std_logic;
28 w0, w1: in std_logic_vector(31 downto 0);
29 f : out std_logic_vector(31 downto 0)
30);
31 end component;
32
33 component register32
34 port (
35 d : in std_logic_vector(31 downto 0);
36 ld : in std_logic;
37 clr : in std_logic;
38 clk : in std_logic;
39 Q : out std_logic_vector(31 downto 0)
40);
41 end component;
42 signal add_out : std_logic_vector(31 downto 0);
43 signal mux_out : std_logic_vector(31 downto 0);
44 signal q_out : std_logic_vector(31 downto 0);
45 signal ld : std_logic := '1';
46 begin
47 add0: add port map(q_out, add_out);
48 mux0: mux2to1 port map(inc, d, add_out, mux_out);
49 reg0: register32 port map (mux_out, ld, clr, clk, q_out);
50 q <= q_out;
51 end Behaviour;
Add...
Ln 1 Col 1 VHDL File 57% 00:00:37

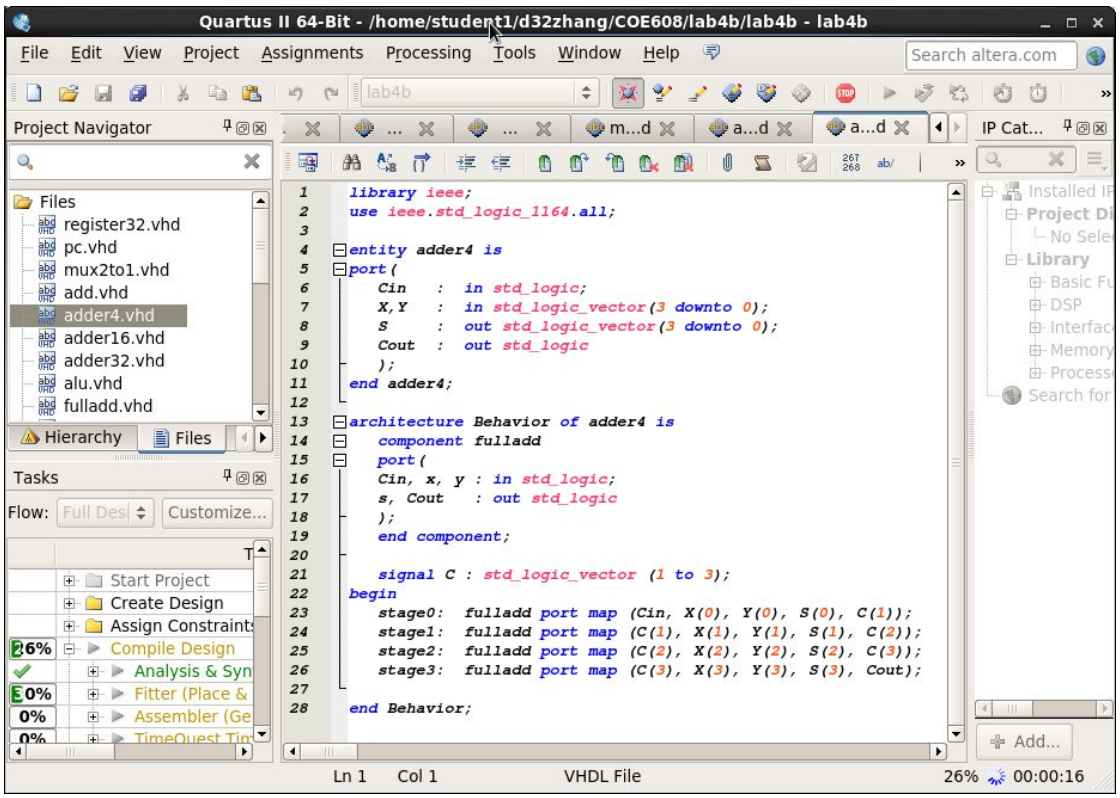
3. Mux2to1.vhd



4. Add.vhd

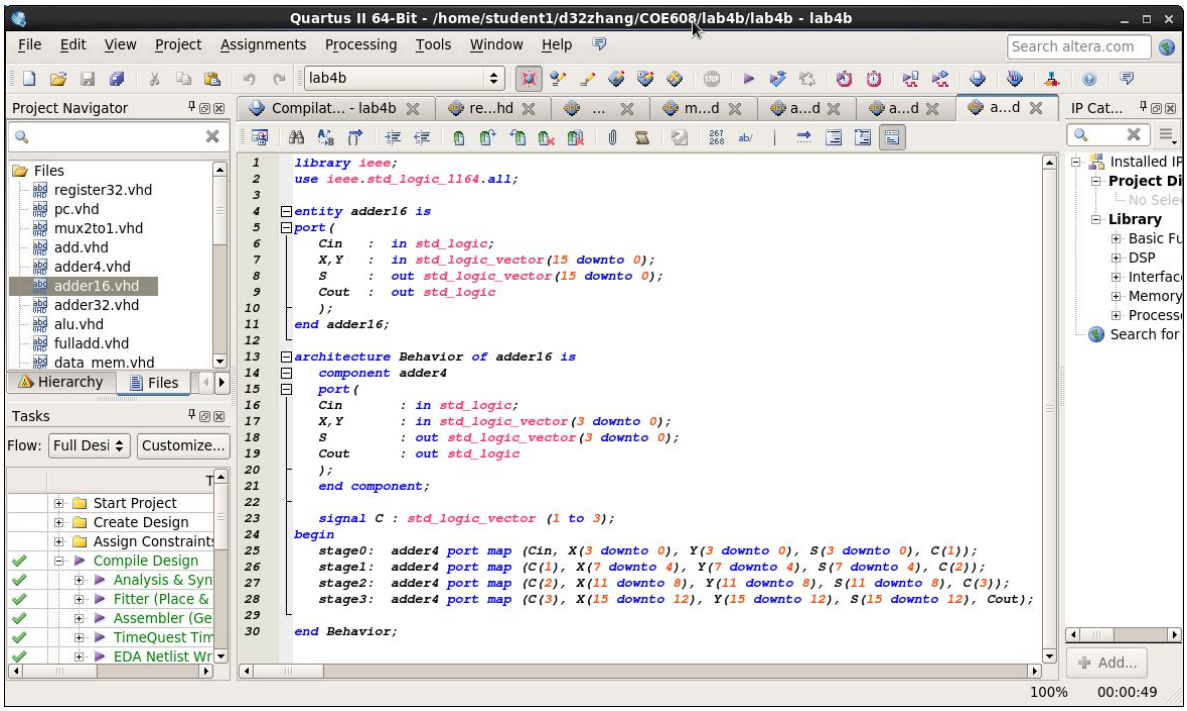


5. Adder4.vhd



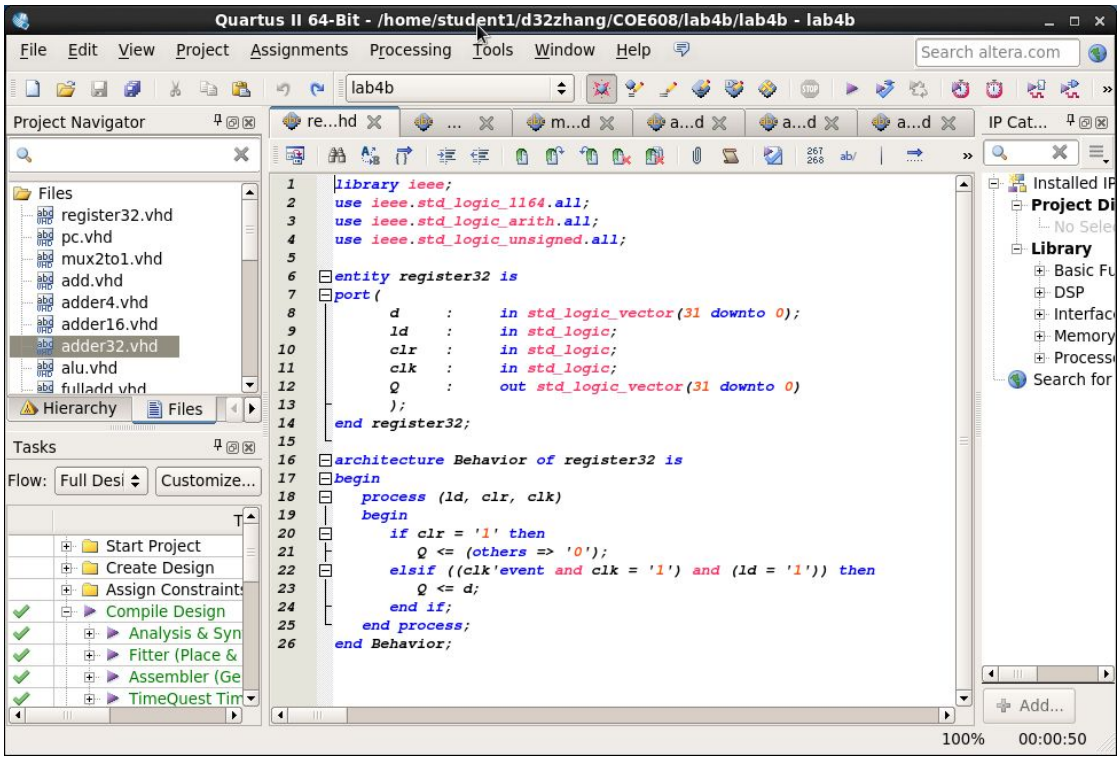

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity adder4 is
5  port(
6      Cin   :  in std_logic;
7      X,Y   :  in std_logic_vector(3 downto 0);
8      S     :  out std_logic_vector(3 downto 0);
9      Cout  :  out std_logic
10     );
11 end adder4;
12
13 architecture Behavior of adder4 is
14     component fulladd
15     port(
16     Cin, x, y : in std_logic;
17     s, Cout   : out std_logic
18     );
19     end component;
20
21     signal C : std_logic_vector (1 to 3);
22 begin
23     stage0:  fulladd port map (Cin, X(0), Y(0), S(0), C(1));
24     stage1:  fulladd port map (C(1), X(1), Y(1), S(1), C(2));
25     stage2:  fulladd port map (C(2), X(2), Y(2), S(2), C(3));
26     stage3:  fulladd port map (C(3), X(3), Y(3), S(3), Cout);
27
28 end Behavior;
```

6. Adder16.vhd



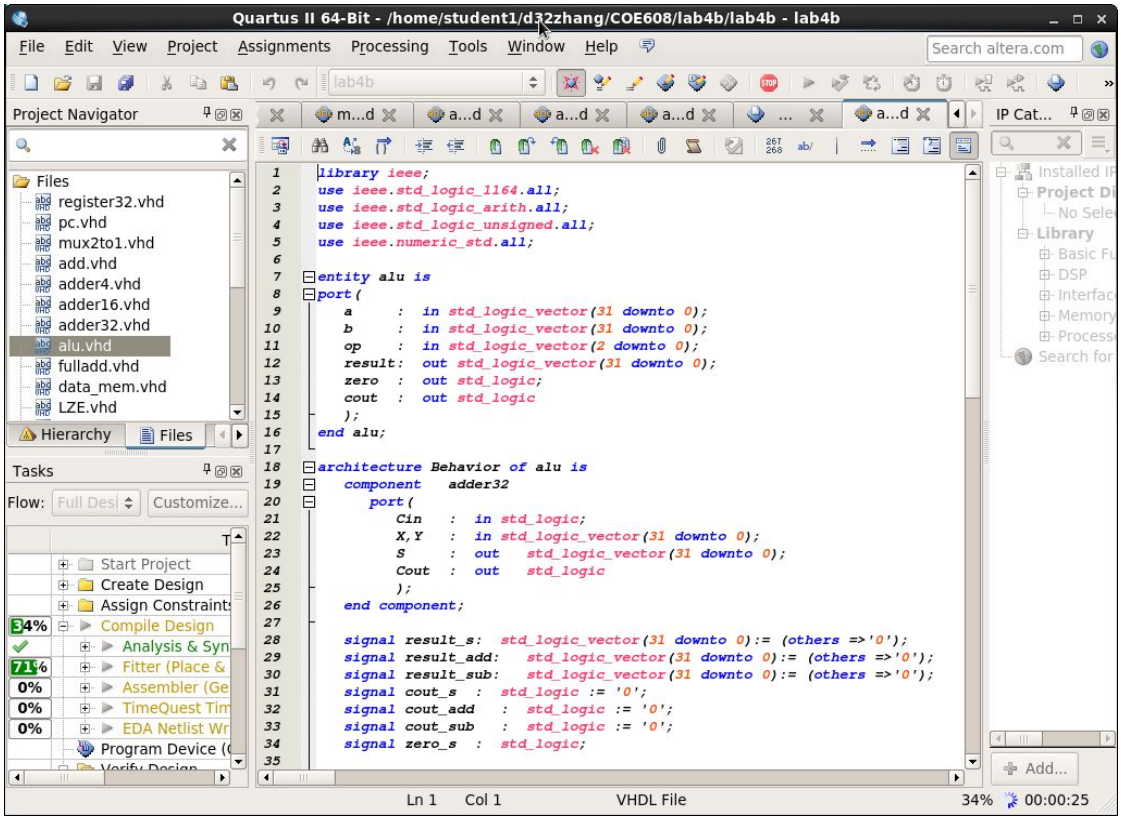

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity adder16 is
5  port(
6      Cin   :  in std_logic;
7      X,Y   :  in std_logic_vector(15 downto 0);
8      S     :  out std_logic_vector(15 downto 0);
9      Cout  :  out std_logic
10     );
11 end adder16;
12
13 architecture Behavior of adder16 is
14     component adder4
15     port(
16     Cin       : in std_logic;
17     X,Y       : in std_logic_vector(3 downto 0);
18     S         : out std_logic_vector(3 downto 0);
19     Cout      : out std_logic
20     );
21     end component;
22
23     signal C : std_logic_vector (1 to 3);
24 begin
25     stage0:  adder4 port map (Cin, X(3 downto 0), Y(3 downto 0), S(3 downto 0), C(1));
26     stage1:  adder4 port map (C(1), X(7 downto 4), Y(7 downto 4), S(7 downto 4), C(2));
27     stage2:  adder4 port map (C(2), X(11 downto 8), Y(11 downto 8), S(11 downto 8), C(3));
28     stage3:  adder4 port map (C(3), X(15 downto 12), Y(15 downto 12), S(15 downto 12), Cout);
29
30 end Behavior;
```

7. Adder32.vhd

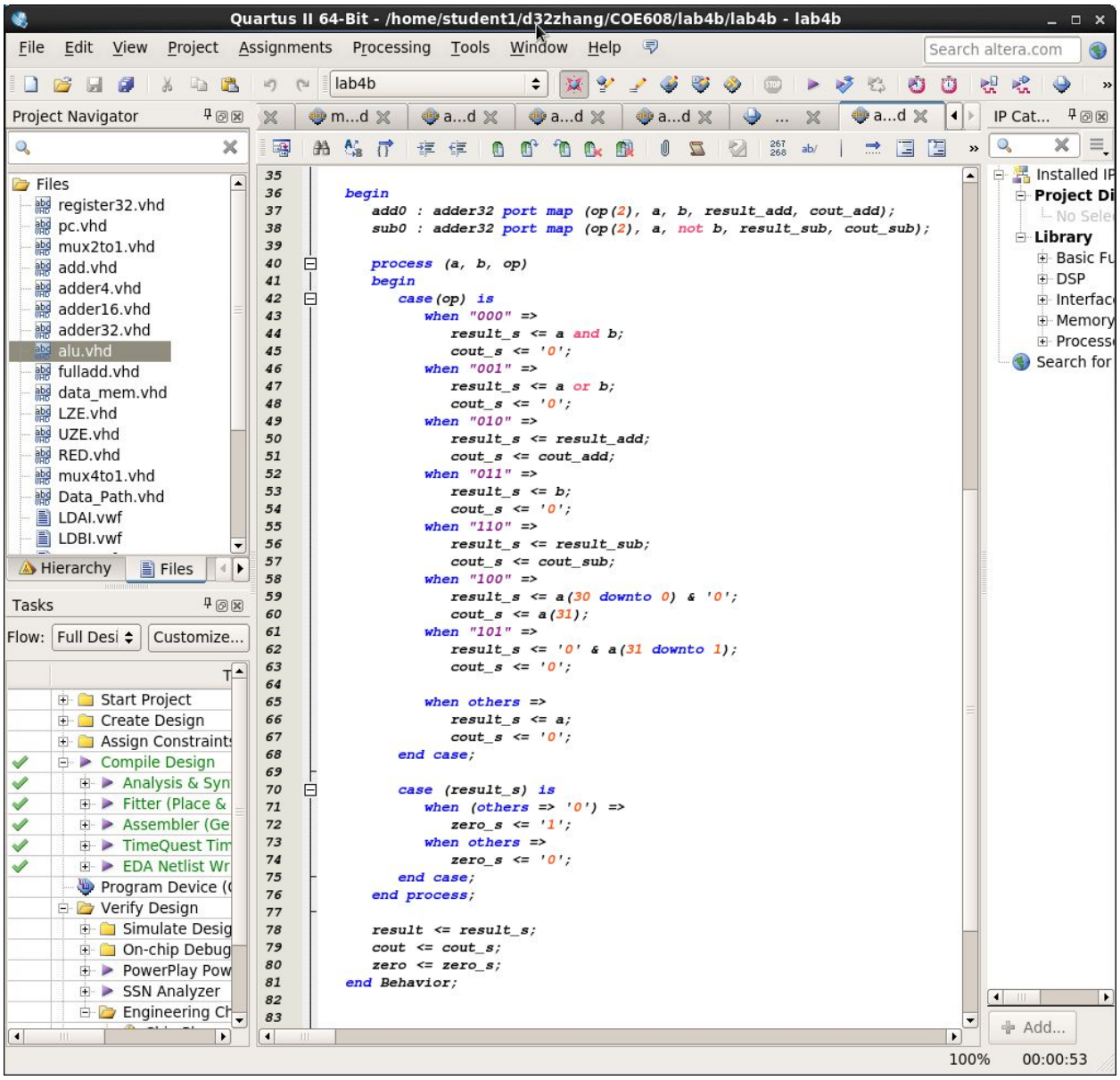



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity register32 is
port(
      d     :     in std_logic_vector(31 downto 0);
      ld    :     in std_logic;
      clr   :     in std_logic;
      clk   :     in std_logic;
      Q     :     out std_logic_vector(31 downto 0)
      );
end register32;

architecture Behavior of register32 is
begin
   process (ld, clr, clk)
   begin
      if clr = '1' then
         Q <= (others => '0');
      elsif ((clk'event and clk = '1') and (ld = '1')) then
         Q <= d;
      end if;
   end process;
end Behavior;
```

8. Alu.vhd

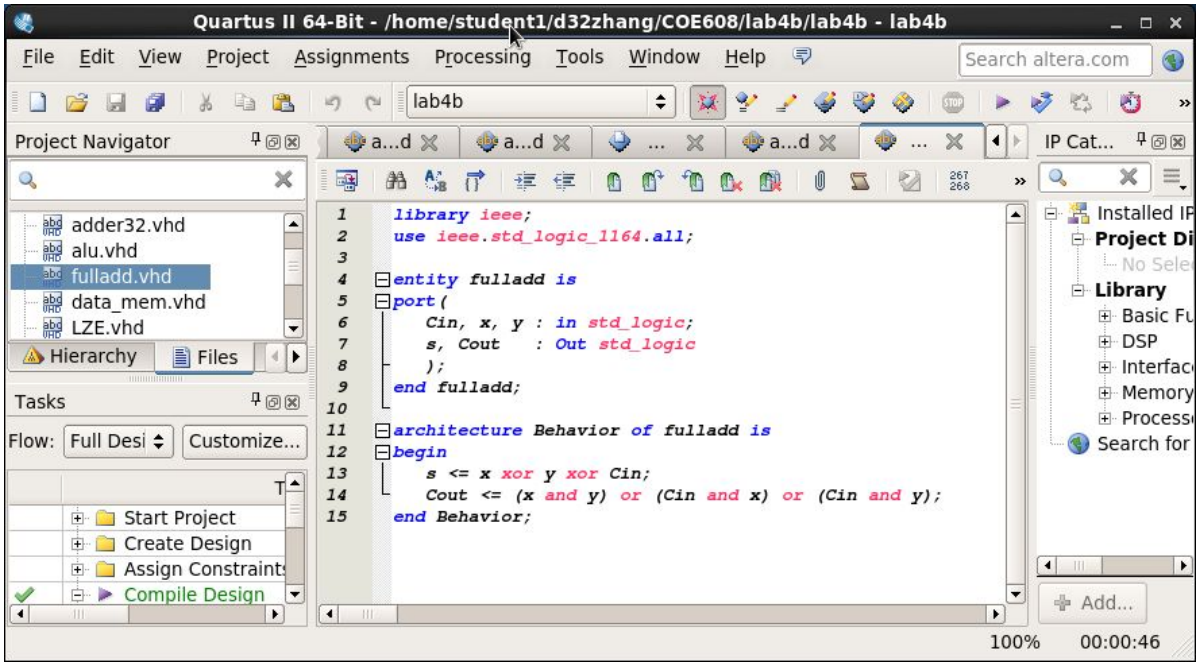



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
use ieee.numeric_std.all;

entity alu is
port(
   a      :  in std_logic_vector(31 downto 0);
   b      :  in std_logic_vector(31 downto 0);
   op     :  in std_logic_vector(2 downto 0);
   result:  out std_logic_vector(31 downto 0);
   zero   :  out std_logic;
   cout   :  out std_logic
   );
end alu;

architecture Behavior of alu is
   component   adder32
      port(
         Cin   :  in std_logic;
         X,Y   :  in std_logic_vector(31 downto 0);
         S     :  out   std_logic_vector(31 downto 0);
         Cout  :  out   std_logic
         );
   end component;

   signal result_s:  std_logic_vector(31 downto 0):= (others =>'0');
   signal result_add:   std_logic_vector(31 downto 0):= (others =>'0');
   signal result_sub:   std_logic_vector(31 downto 0):= (others =>'0');
   signal cout_s  :  std_logic := '0';
   signal cout_add   :  std_logic := '0';
   signal cout_sub   :  std_logic := '0';
   signal zero_s  :  std_logic;
```



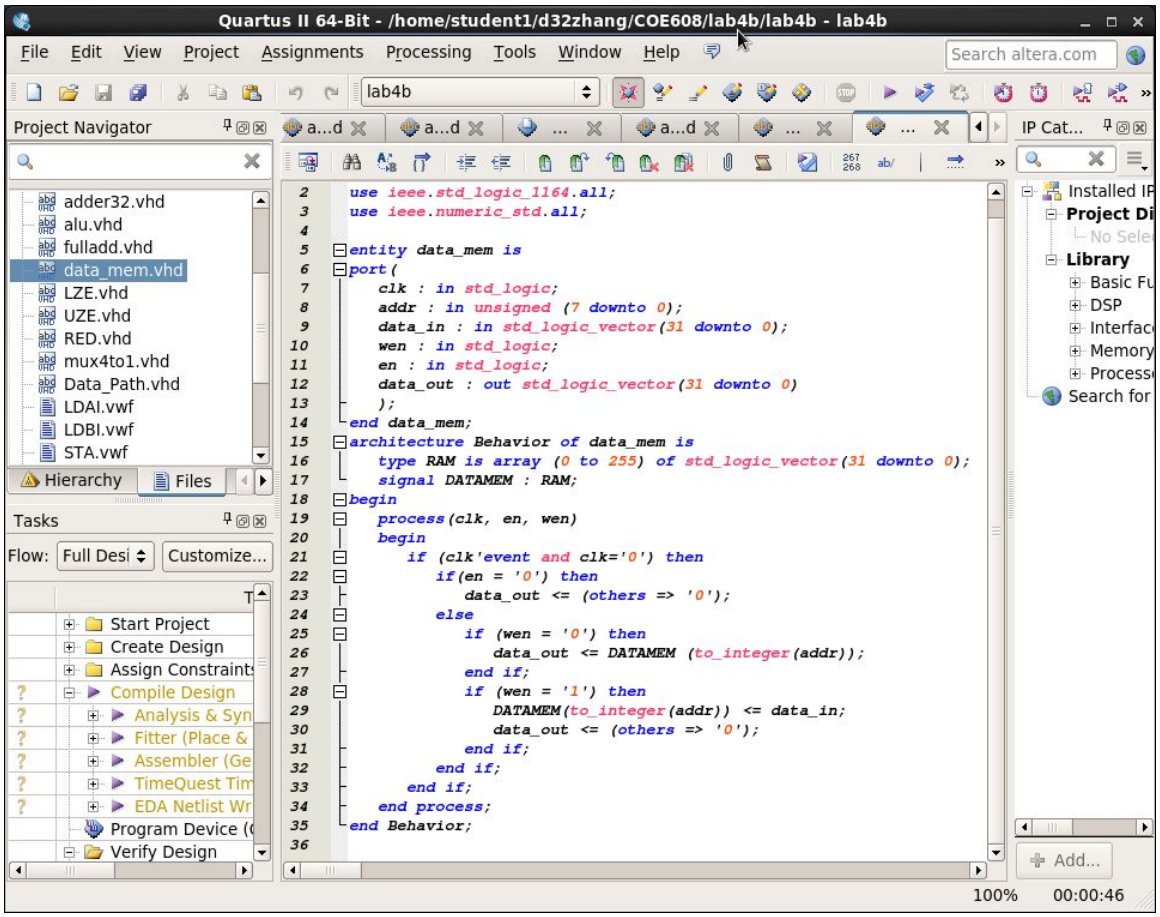

```
35
36      begin
37         add0 : adder32 port map (op(2), a, b, result_add, cout_add);
38         sub0 : adder32 port map (op(2), a, not b, result_sub, cout_sub);
39
40         process (a, b, op)
41         begin
42            case(op) is
43               when "000" =>
44                  result_s <= a and b;
45                  cout_s <= '0';
46               when "001" =>
47                  result_s <= a or b;
48                  cout_s <= '0';
49               when "010" =>
50                  result_s <= result_add;
51                  cout_s <= cout_add;
52               when "011" =>
53                  result_s <= b;
54                  cout_s <= '0';
55               when "110" =>
56                  result_s <= result_sub;
57                  cout_s <= cout_sub;
58               when "100" =>
59                  result_s <= a(30 downto 0) & '0';
60                  cout_s <= a(31);
61               when "101" =>
62                  result_s <= '0' & a(31 downto 1);
63                  cout_s <= '0';
64
65               when others =>
66                  result_s <= a;
67                  cout_s <= '0';
68            end case;
69
70            case (result_s) is
71               when (others => '0') =>
72                  zero_s <= '1';
73               when others =>
74                  zero_s <= '0';
75            end case;
76         end process;
77
78         result <= result_s;
79         cout <= cout_s;
80         zero <= zero_s;
81      end Behavior;
82
83
```

9. Fulladd.vhd



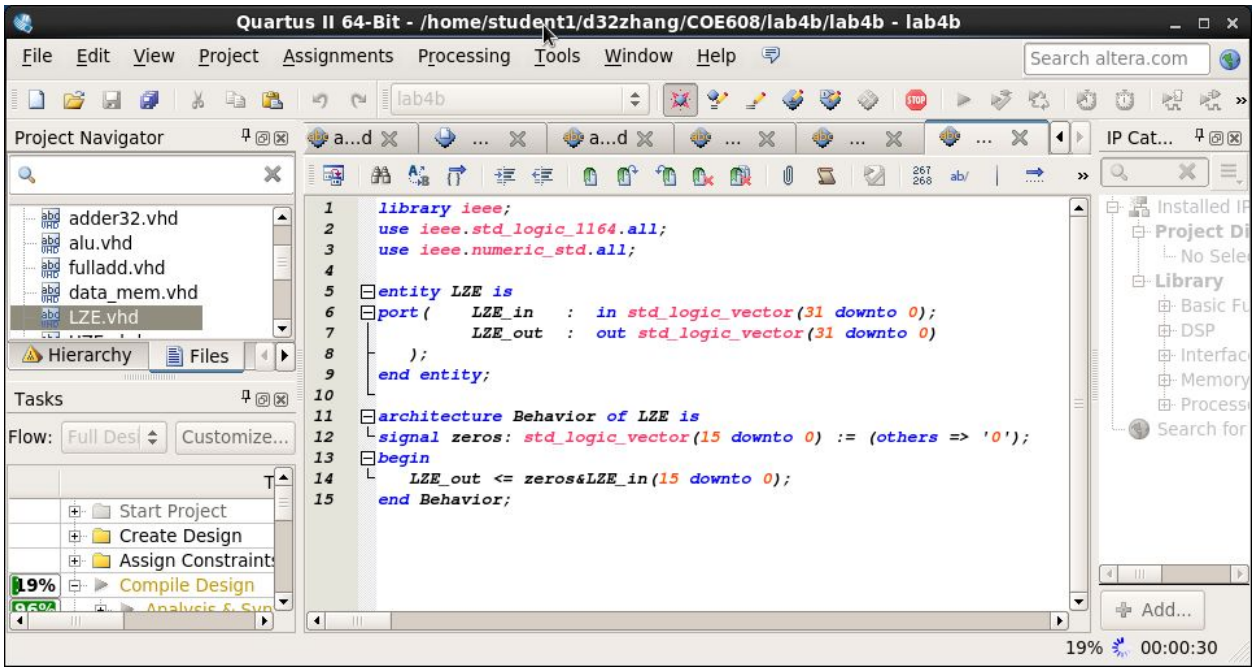

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity fulladd is
5  port(
6      Cin, x, y : in std_logic;
7      s, Cout   : Out std_logic
8      );
9  end fulladd;
10
11 architecture Behavior of fulladd is
12 begin
13     s <= x xor y xor Cin;
14     Cout <= (x and y) or (Cin and x) or (Cin and y);
15 end Behavior;
```

10. Data_mem.vhd

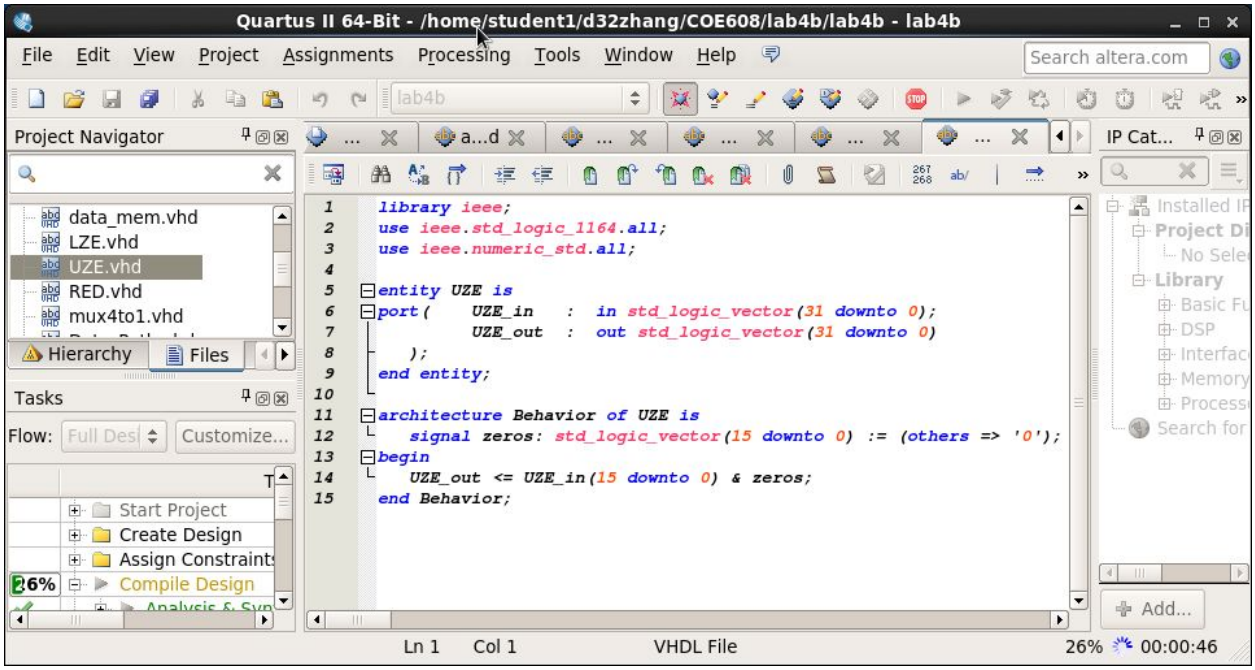



```
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity data_mem is
6  port(
7      clk : in std_logic;
8      addr : in unsigned (7 downto 0);
9      data_in : in std_logic_vector(31 downto 0);
10     wen : in std_logic;
11     en : in std_logic;
12     data_out : out std_logic_vector(31 downto 0)
13     );
14 end data_mem;
15 architecture Behavior of data_mem is
16     type RAM is array (0 to 255) of std_logic_vector(31 downto 0);
17     signal DATAMEM : RAM;
18 begin
19     process(clk, en, wen)
20     begin
21         if (clk'event and clk='0') then
22             if(en = '0') then
23                 data_out <= (others => '0');
24             else
25                 if (wen = '0') then
26                     data_out <= DATAMEM (to_integer(addr));
27                 end if;
28                 if (wen = '1') then
29                     DATAMEM(to_integer(addr)) <= data_in;
30                     data_out <= (others => '0');
31                 end if;
32             end if;
33         end if;
34     end process;
35 end Behavior;
36
```

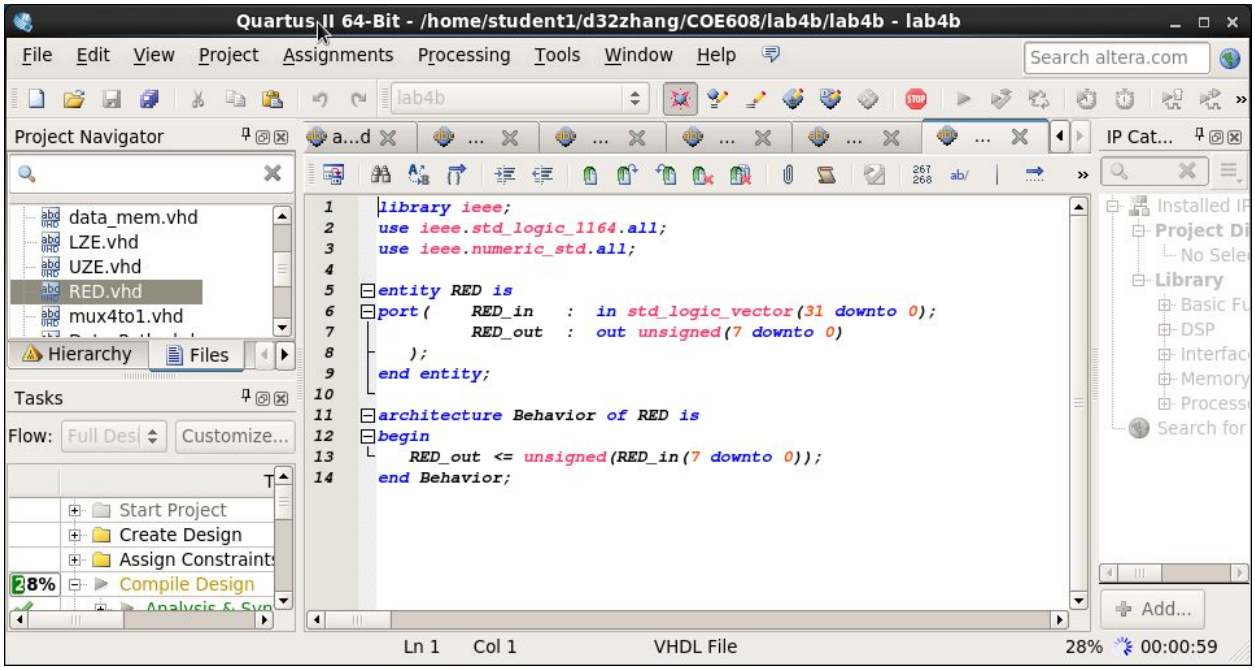
11. LZE.vhd



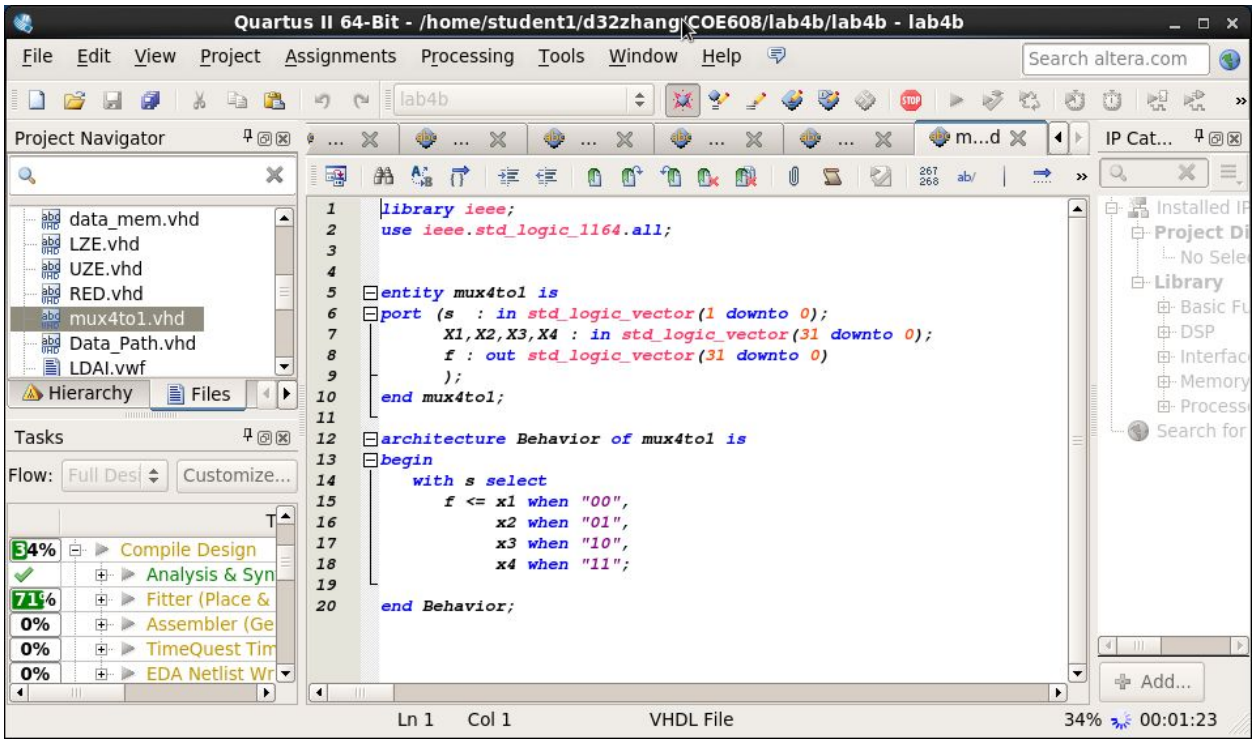
12. UZE.vhd



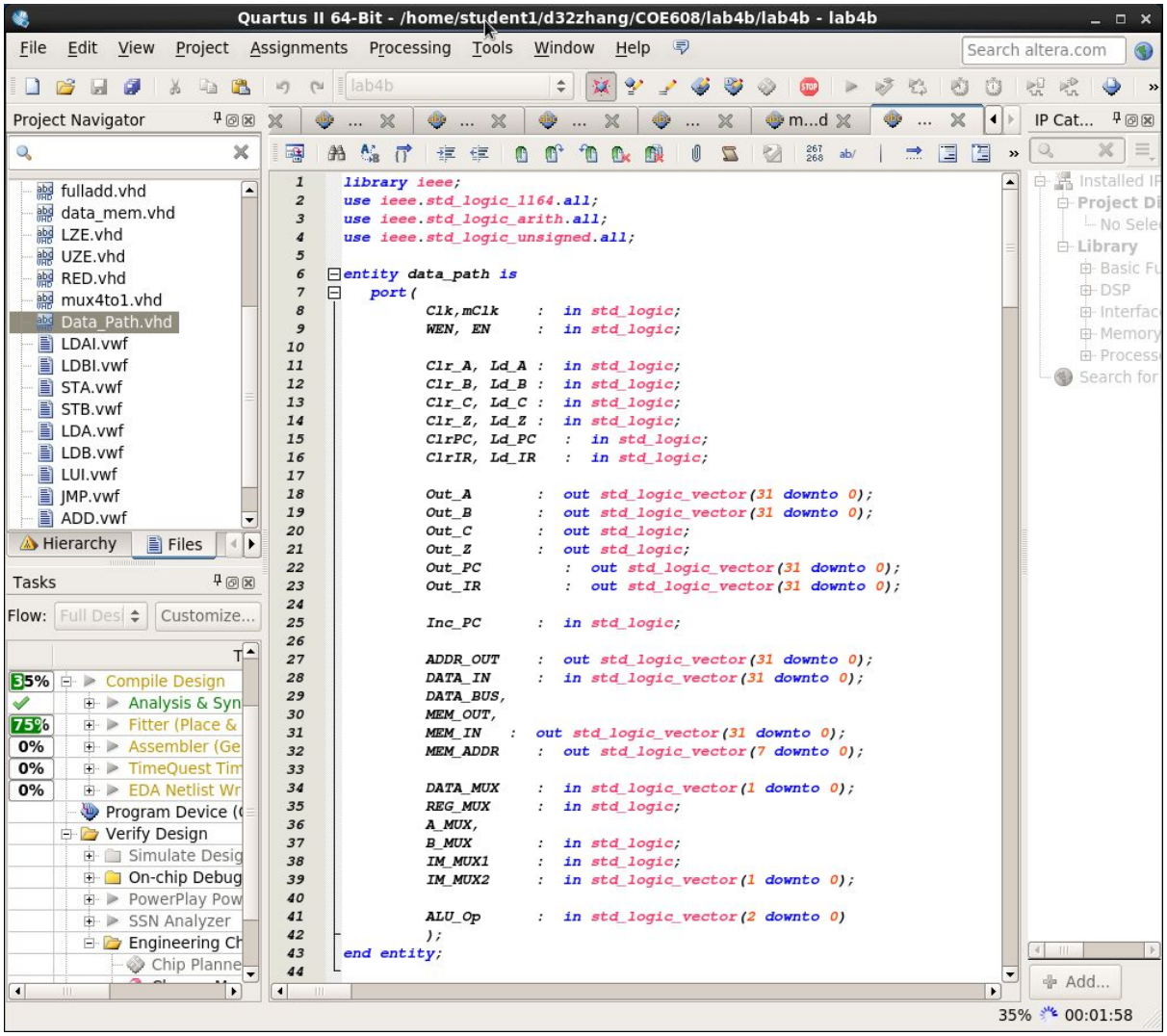
13. RED.vhd



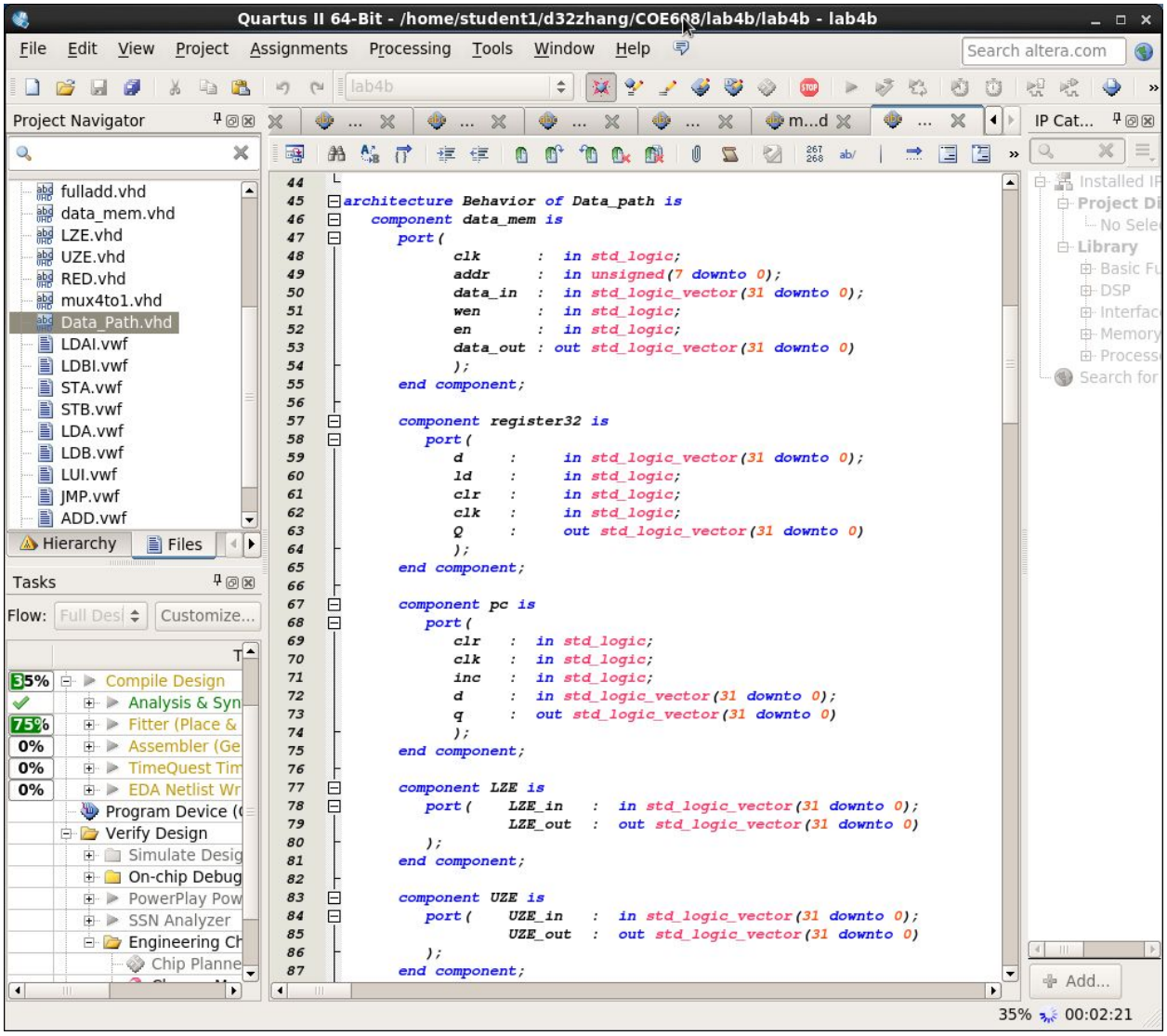
14. Mux4to1.vhd



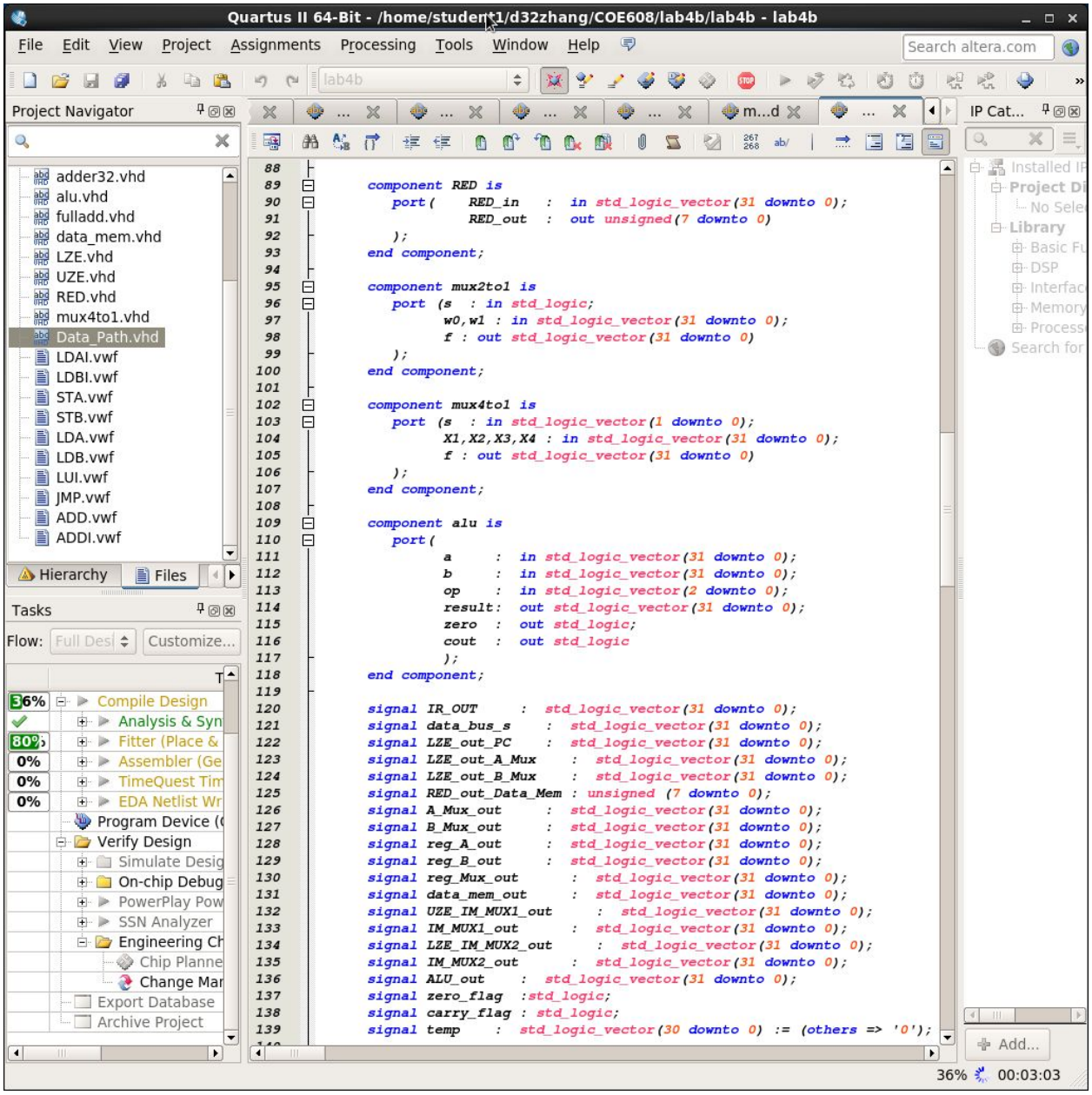
15. Data_Path.vhd



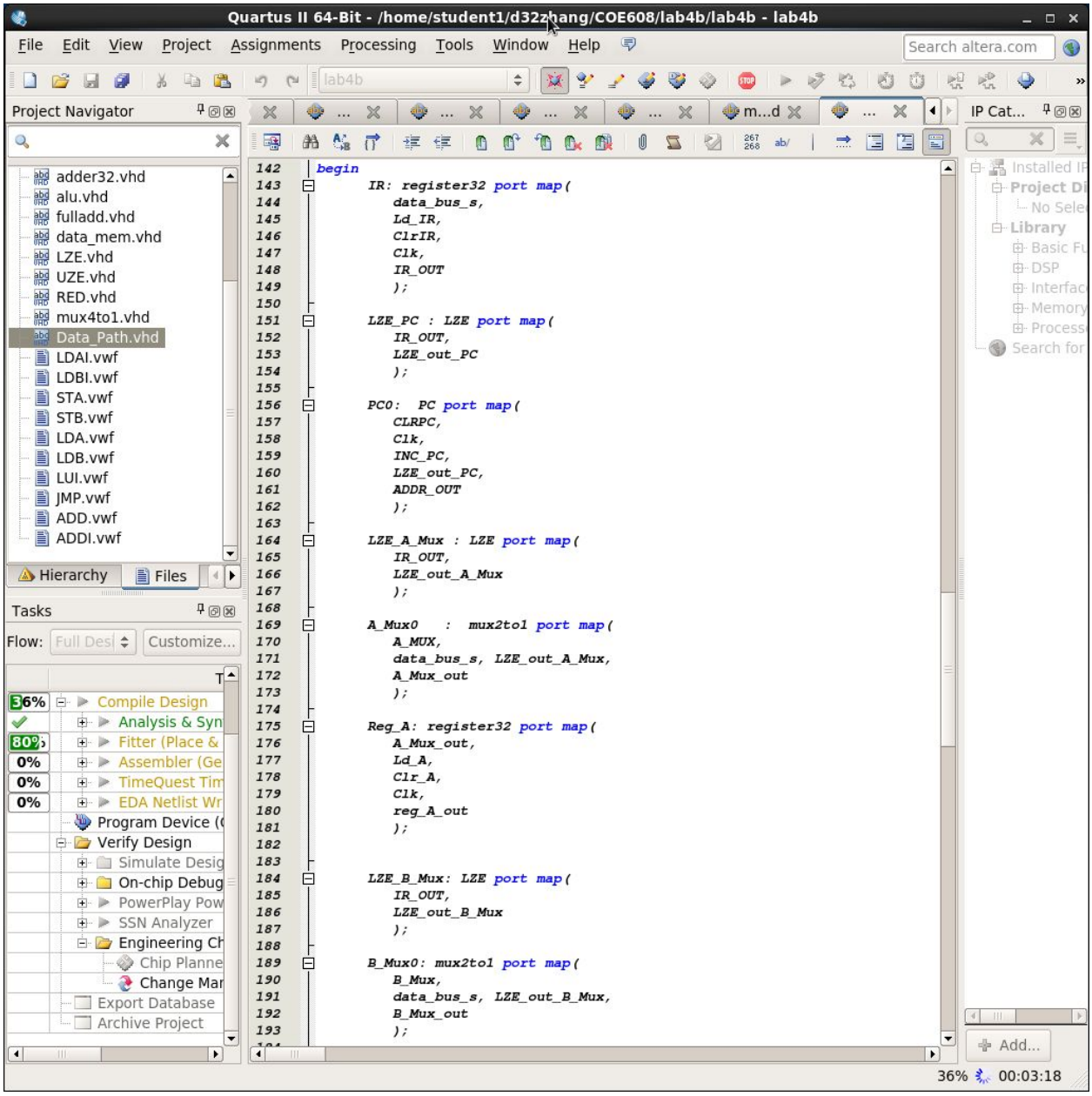

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_arith.all;
4   use ieee.std_logic_unsigned.all;
5
6   entity data_path is
7      port(
8            Clk,mClk    :  in std_logic;
9            WEN, EN     :  in std_logic;
10
11           Clr_A, Ld_A :  in std_logic;
12           Clr_B, Ld_B :  in std_logic;
13           Clr_C, Ld_C :  in std_logic;
14           Clr_Z, Ld_Z :  in std_logic;
15           ClrPC, Ld_PC   :  in std_logic;
16           ClrIR, Ld_IR   :  in std_logic;
17
18           Out_A       :  out std_logic_vector(31 downto 0);
19           Out_B       :  out std_logic_vector(31 downto 0);
20           Out_C       :  out std_logic;
21           Out_Z       :  out std_logic;
22           Out_PC         :  out std_logic_vector(31 downto 0);
23           Out_IR         :  out std_logic_vector(31 downto 0);
24
25           Inc_PC      :  in std_logic;
26
27           ADDR_OUT    :  out std_logic_vector(31 downto 0);
28           DATA_IN     :  in std_logic_vector(31 downto 0);
29           DATA_BUS,
30           MEM_OUT,
31           MEM_IN   :  out std_logic_vector(31 downto 0);
32           MEM_ADDR    :  out std_logic_vector(7 downto 0);
33
34           DATA_MUX    :  in std_logic_vector(1 downto 0);
35           REG_MUX     :  in std_logic;
36           A_MUX,
37           B_MUX       :  in std_logic;
38           IM_MUX1     :  in std_logic;
39           IM_MUX2     :  in std_logic_vector(1 downto 0);
40
41           ALU_Op      :  in std_logic_vector(2 downto 0)
42           );
43  end entity;
44
```



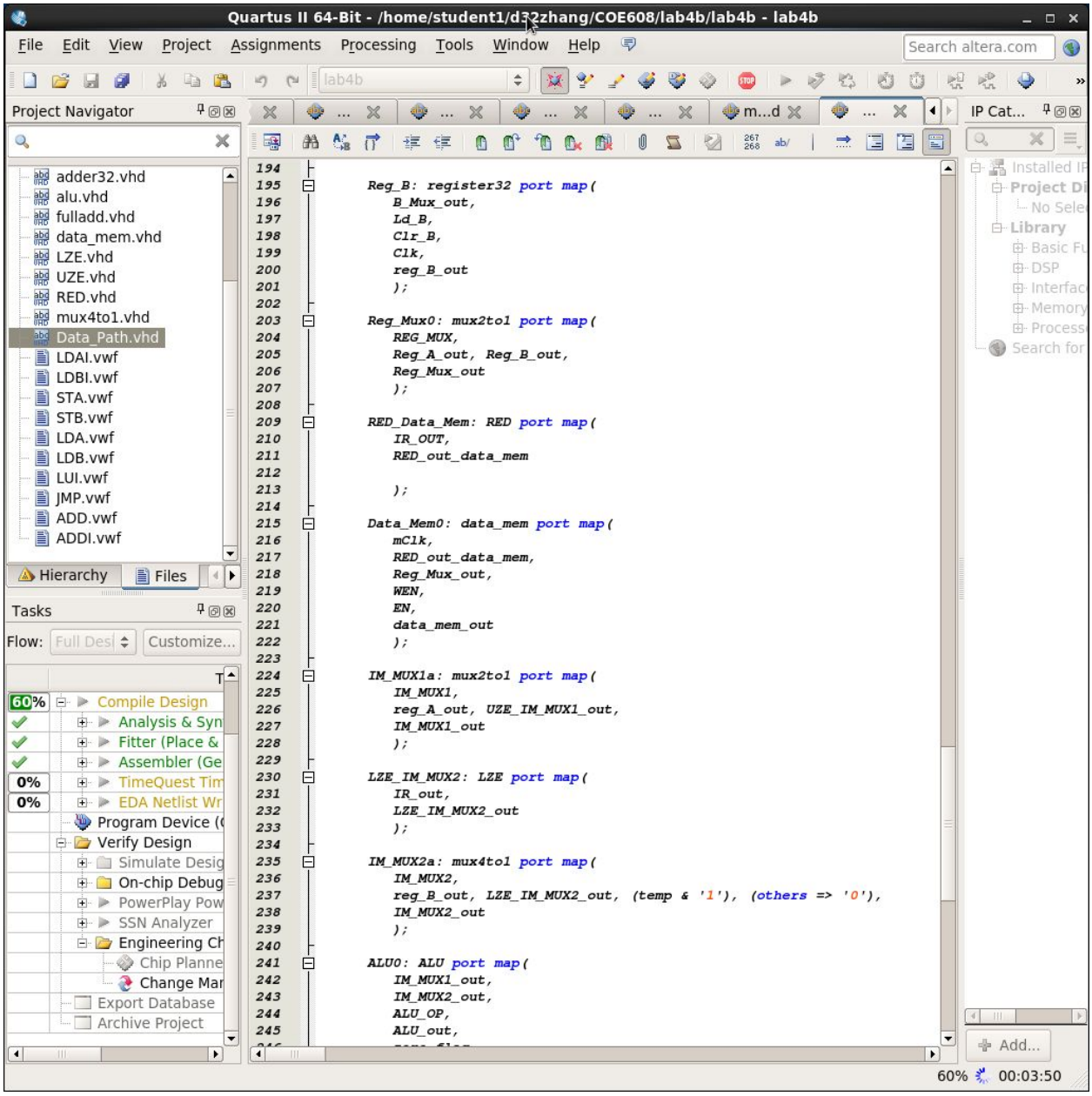

```
44
45  architecture Behavior of Data_path is
46     component data_mem is
47        port(
48              clk      :  in std_logic;
49              addr     :  in unsigned(7 downto 0);
50              data_in  :  in std_logic_vector(31 downto 0);
51              wen      :  in std_logic;
52              en       :  in std_logic;
53              data_out : out std_logic_vector(31 downto 0)
54              );
55        end component;
56
57        component register32 is
58           port(
59              d     :     in std_logic_vector(31 downto 0);
60              ld    :     in std_logic;
61              clr   :     in std_logic;
62              clk   :     in std_logic;
63              Q     :     out std_logic_vector(31 downto 0)
64              );
65        end component;
66
67        component pc is
68           port(
69              clr   :  in std_logic;
70              clk   :  in std_logic;
71              inc   :  in std_logic;
72              d     :  in std_logic_vector(31 downto 0);
73              q     :  out std_logic_vector(31 downto 0)
74              );
75        end component;
76
77        component LZE is
78           port(    LZE_in   :  in std_logic_vector(31 downto 0);
79                    LZE_out  :  out std_logic_vector(31 downto 0)
80           );
81        end component;
82
83        component UZE is
84           port(    UZE_in   :  in std_logic_vector(31 downto 0);
85                    UZE_out  :  out std_logic_vector(31 downto 0)
86           );
87        end component;
```



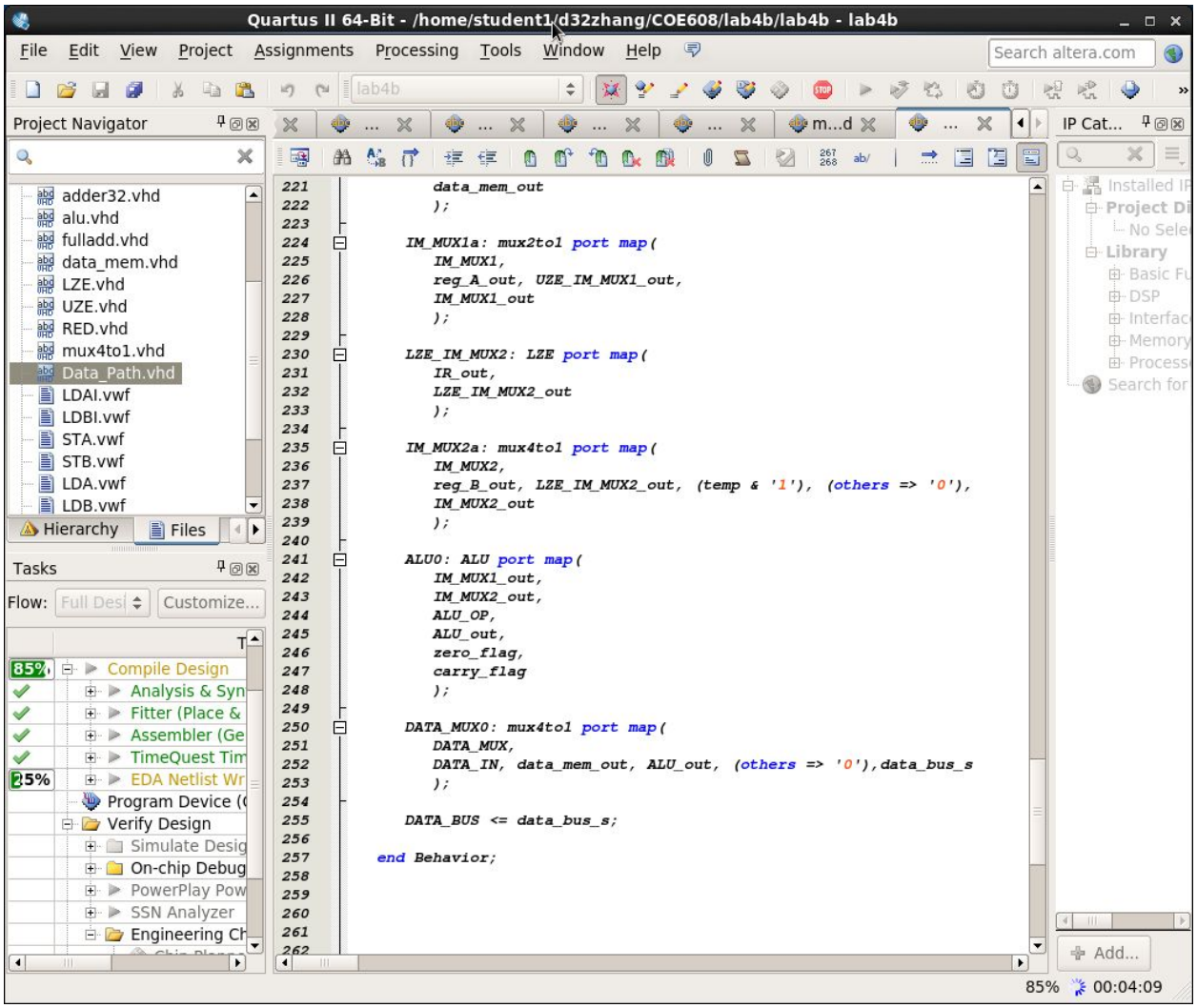

```
88
89      component RED is
90         port(    RED_in   :  in std_logic_vector(31 downto 0);
91                  RED_out  :  out unsigned(7 downto 0)
92         );
93      end component;
94
95      component mux2to1 is
96         port (s  : in std_logic;
97               w0,w1 : in std_logic_vector(31 downto 0);
98               f : out std_logic_vector(31 downto 0)
99         );
100     end component;
101
102     component mux4to1 is
103        port (s  : in std_logic_vector(1 downto 0);
104              X1,X2,X3,X4 : in std_logic_vector(31 downto 0);
105              f : out std_logic_vector(31 downto 0)
106        );
107     end component;
108
109     component alu is
110        port(
111              a      :  in std_logic_vector(31 downto 0);
112              b      :  in std_logic_vector(31 downto 0);
113              op     :  in std_logic_vector(2 downto 0);
114              result:   out std_logic_vector(31 downto 0);
115              zero   :  out std_logic;
116              cout   :  out std_logic
117              );
118     end component;
119
120     signal IR_OUT     :  std_logic_vector(31 downto 0);
121     signal data_bus_s     :  std_logic_vector(31 downto 0);
122     signal LZE_out_PC     :  std_logic_vector(31 downto 0);
123     signal LZE_out_A_Mux     :  std_logic_vector(31 downto 0);
124     signal LZE_out_B_Mux     :  std_logic_vector(31 downto 0);
125     signal RED_out_Data_Mem : unsigned (7 downto 0);
126     signal A_Mux_out      :  std_logic_vector(31 downto 0);
127     signal B_Mux_out      :  std_logic_vector(31 downto 0);
128     signal reg_A_out      :  std_logic_vector(31 downto 0);
129     signal reg_B_out      :  std_logic_vector(31 downto 0);
130     signal reg_Mux_out       :  std_logic_vector(31 downto 0);
131     signal data_mem_out      :  std_logic_vector(31 downto 0);
132     signal UZE_IM_MUX1_out      :  std_logic_vector(31 downto 0);
133     signal IM_MUX1_out       :  std_logic_vector(31 downto 0);
134     signal LZE_IM_MUX2_out      :  std_logic_vector(31 downto 0);
135     signal IM_MUX2_out       :  std_logic_vector(31 downto 0);
136     signal ALU_out     :  std_logic_vector(31 downto 0);
137     signal zero_flag  :std_logic;
138     signal carry_flag : std_logic;
139     signal temp     :  std_logic_vector(30 downto 0) := (others => '0');
```




```
142   begin
143         IR: register32 port map(
144            data_bus_s,
145            Ld_IR,
146            ClrIR,
147            Clk,
148            IR_OUT
149            );
150
151         LZE_PC : LZE port map(
152            IR_OUT,
153            LZE_out_PC
154            );
155
156         PC0:  PC port map(
157            CLRPC,
158            Clk,
159            INC_PC,
160            LZE_out_PC,
161            ADDR_OUT
162            );
163
164         LZE_A_Mux : LZE port map(
165            IR_OUT,
166            LZE_out_A_Mux
167            );
168
169         A_Mux0    :  mux2to1 port map(
170            A_MUX,
171            data_bus_s, LZE_out_A_Mux,
172            A_Mux_out
173            );
174
175         Reg_A: register32 port map(
176            A_Mux_out,
177            Ld_A,
178            Clr_A,
179            Clk,
180            reg_A_out
181            );
182
183
184         LZE_B_Mux: LZE port map(
185            IR_OUT,
186            LZE_out_B_Mux
187            );
188
189         B_Mux0: mux2to1 port map(
190            B_Mux,
191            data_bus_s, LZE_out_B_Mux,
192            B_Mux_out
193            );
```




```
194
195       Reg_B: register32 port map(
196          B_Mux_out,
197          Ld_B,
198          Clr_B,
199          Clk,
200          reg_B_out
201          );
202
203       Reg_Mux0: mux2to1 port map(
204          REG_MUX,
205          Reg_A_out, Reg_B_out,
206          Reg_Mux_out
207          );
208
209       RED_Data_Mem: RED port map(
210          IR_OUT,
211          RED_out_data_mem
212
213          );
214
215       Data_Mem0: data_mem port map(
216          mClk,
217          RED_out_data_mem,
218          Reg_Mux_out,
219          WEN,
220          EN,
221          data_mem_out
222          );
223
224       IM_MUX1a: mux2to1 port map(
225          IM_MUX1,
226          reg_A_out, UZE_IM_MUX1_out,
227          IM_MUX1_out
228          );
229
230       LZE_IM_MUX2: LZE port map(
231          IR_out,
232          LZE_IM_MUX2_out
233          );
234
235       IM_MUX2a: mux4to1 port map(
236          IM_MUX2,
237          reg_B_out, LZE_IM_MUX2_out, (temp & '1'), (others => '0'),
238          IM_MUX2_out
239          );
240
241       ALU0: ALU port map(
242          IM_MUX1_out,
243          IM_MUX2_out,
244          ALU_OP,
245          ALU_out,
```


Quartus II 64-Bit - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
lab4b
Project Navigator
adder32.vhd
alu.vhd
fulladd.vhd
data_mem.vhd
LZE.vhd
UZE.vhd
RED.vhd
mux4to1.vhd
Data_Path.vhd
LDAI.vwf
LDBI.vwf
STA.vwf
STB.vwf
LDA.vwf
LDB.vwf
Hierarchy
Files
Tasks
Flow:
Full Desi
Customize...
85%
Compile Design
Analysis & Syn
Fitter (Place &
Assembler (Ge
TimeQuest Tim
25%
EDA Netlist Wr
Program Device (
Verify Design
Simulate Desig
On-chip Debug
PowerPlay Pow
SSN Analyzer
Engineering Ch
IP Cat...
Installed IP
Project Di
No Sele
Library
Basic Fu
DSP
Interfac
Memory
Process
Search for
Add...
85%
00:04:09
221 data_mem_out
222);
223
224 IM_MUX1a: mux2to1 port map(
225 IM_MUX1,
226 reg_A_out, UZE_IM_MUX1_out,
227 IM_MUX1_out
228);
229
230 LZE_IM_MUX2: LZE port map(
231 IR_out,
232 LZE_IM_MUX2_out
233);
234
235 IM_MUX2a: mux4to1 port map(
236 IM_MUX2,
237 reg_B_out, LZE_IM_MUX2_out, (temp & '1'), (others => '0'),
238 IM_MUX2_out
239);
240
241 ALU0: ALU port map(
242 IM_MUX1_out,
243 IM_MUX2_out,
244 ALU_OP,
245 ALU_out,
246 zero_flag,
247 carry_flag
248);
249
250 DATA_MUX0: mux4to1 port map(
251 DATA_MUX,
252 DATA_IN, data_mem_out, ALU_out, (others => '0'),data_bus_s
253);
254
255 DATA_BUS <= data_bus_s;
256
257 end Behavior;
258
259
260
261
262

How does this data-path implement the INCA, ADDI, LDBI and LDA operations?

For INCA operation, we first load LD_A,LD_C,LD_Z, assign A MUX to 0, Data MUX to 10, IM_MUX1 TO 0, IM_MUX2 to 10, and then go to the operation code “010” in alu, alu calculates the assigned value to produce the result.

For ADDI operation, we first load LD_A,LD_C,LD_Z, assign A MUX to 0, Data MUX to 10, IM_MUX1 TO 0, IM_MUX2 to 01, and then go to the operation code “010” in alu, alu calculates the assigned value to produce the result.

For LDBI operation, we first load LD_B, assign B MUX to 1.

For ADDI operation, we first load LD_A, assign EN to 1, assign WEN to 0, assign A MUX to 0, and Data MUX to 01.

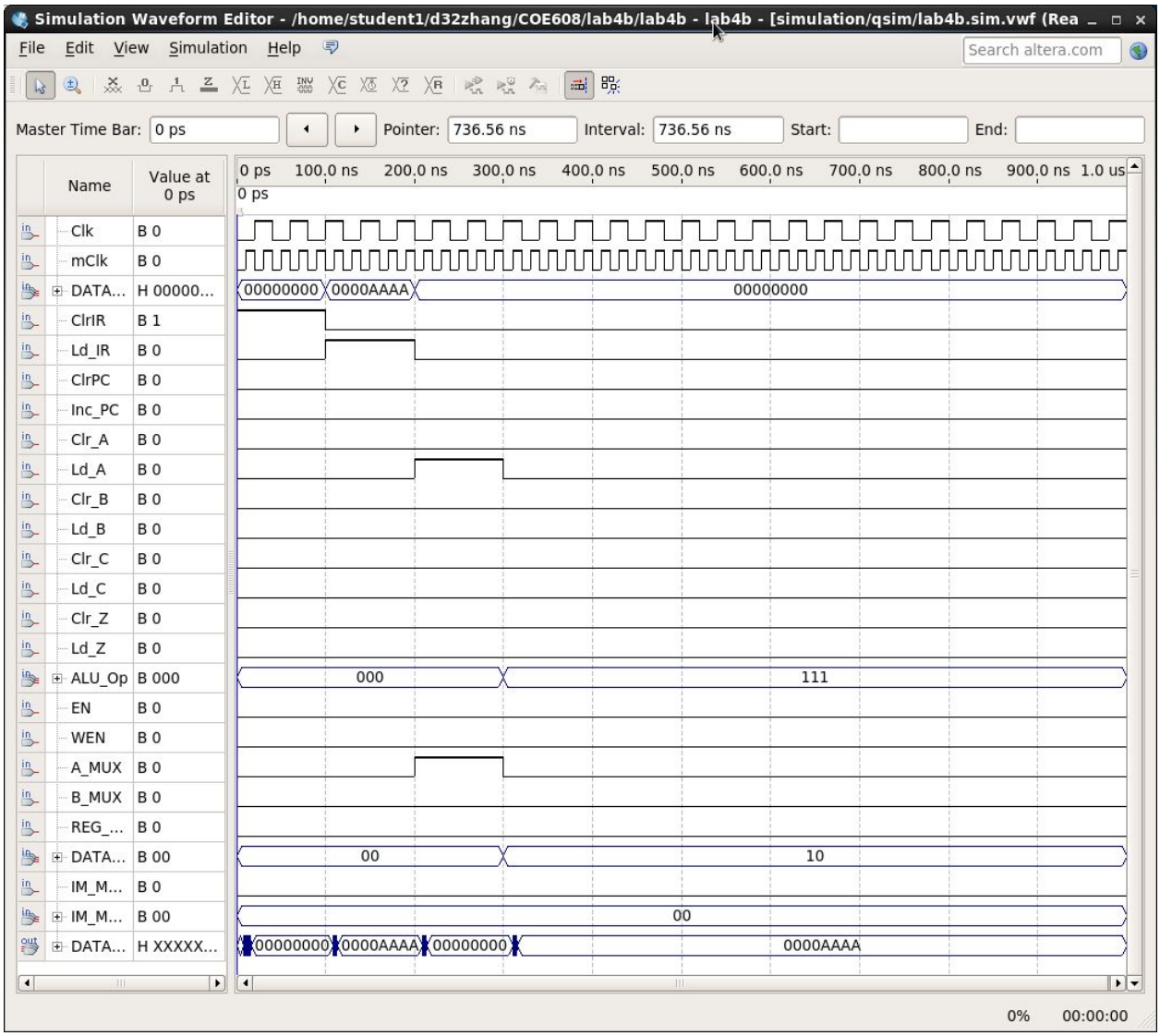
The data-path has a maximum reliable operating speed (Clk). What determines this speed? (i.e. how would you estimate the data-path circuit clock)?

Delays determine the maximum reliable operating speed, every time a different input was passed in, the corresponding result was generated with some delays, delays may be different than others.

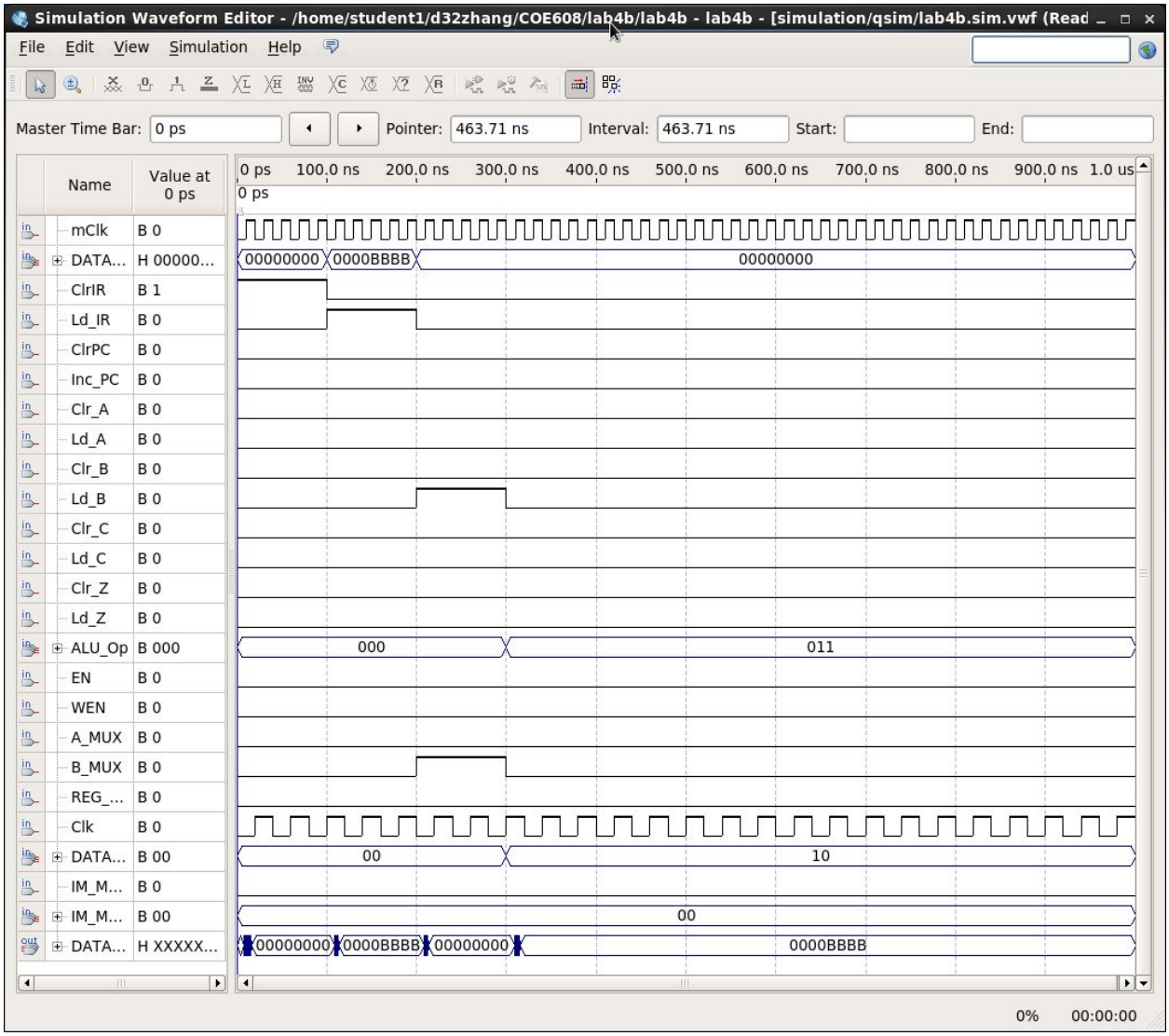
What is a reliable limit for you data-path clock?

The reliable limit for my data-path clock was 8.806.

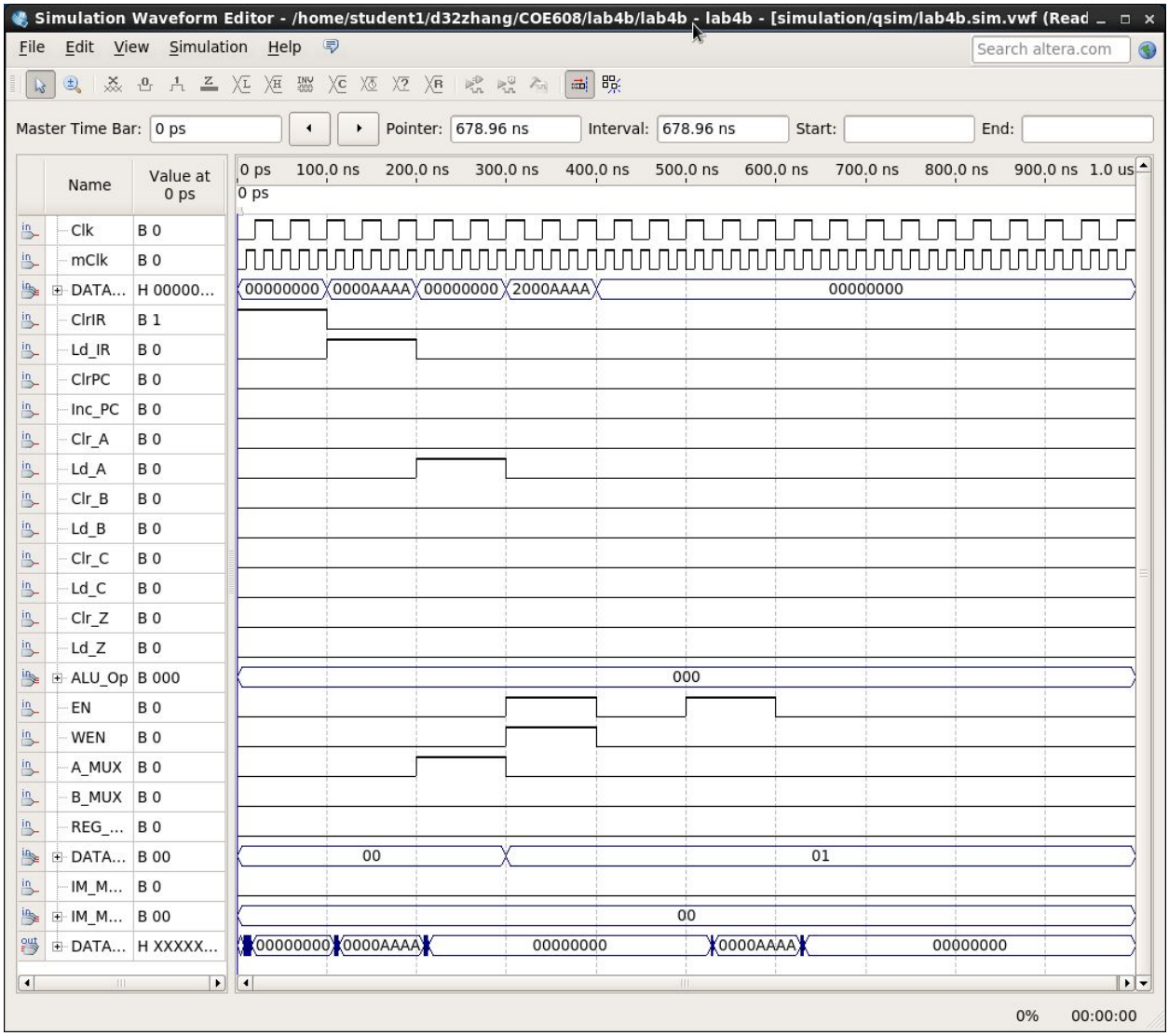
16. LDAI.vwf


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Rea
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 736.56 ns
Interval: 736.56 ns
Start:
End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
00000000 0000AAAA 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000 111
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00 10
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 0000AAAA 00000000 0000AAAA
0% 00:00:00

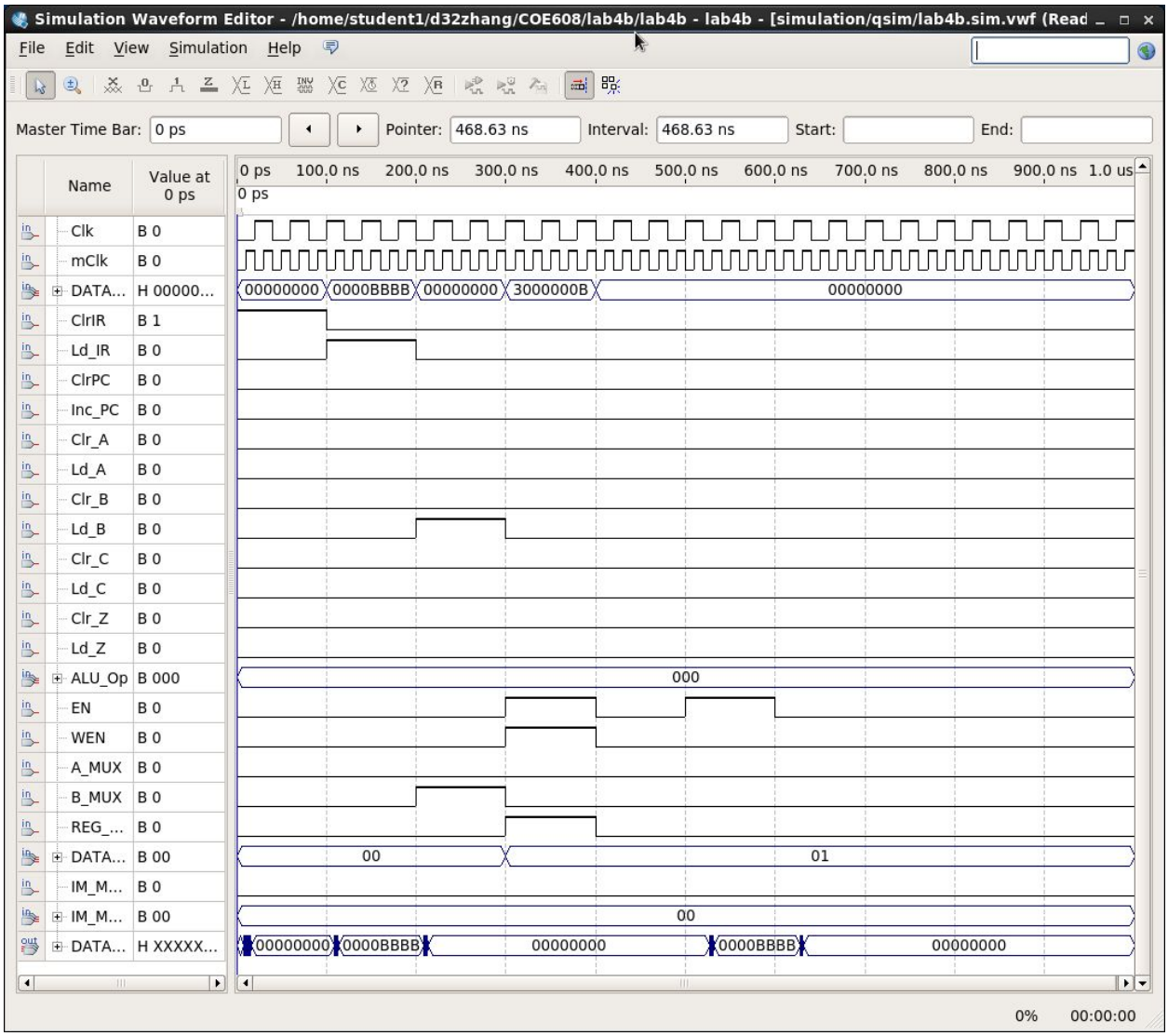
17. LDBI.vwf


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Master Time Bar: 0 ps Pointer: 463.71 ns Interval: 463.71 ns Start: End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
mClk B 0
DATA... H 00000...
00000000 0000BBBB 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000 011
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
Clk B 0
DATA... B 00
00 10
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 0000BBBB 00000000 0000BBBB
0% 00:00:00

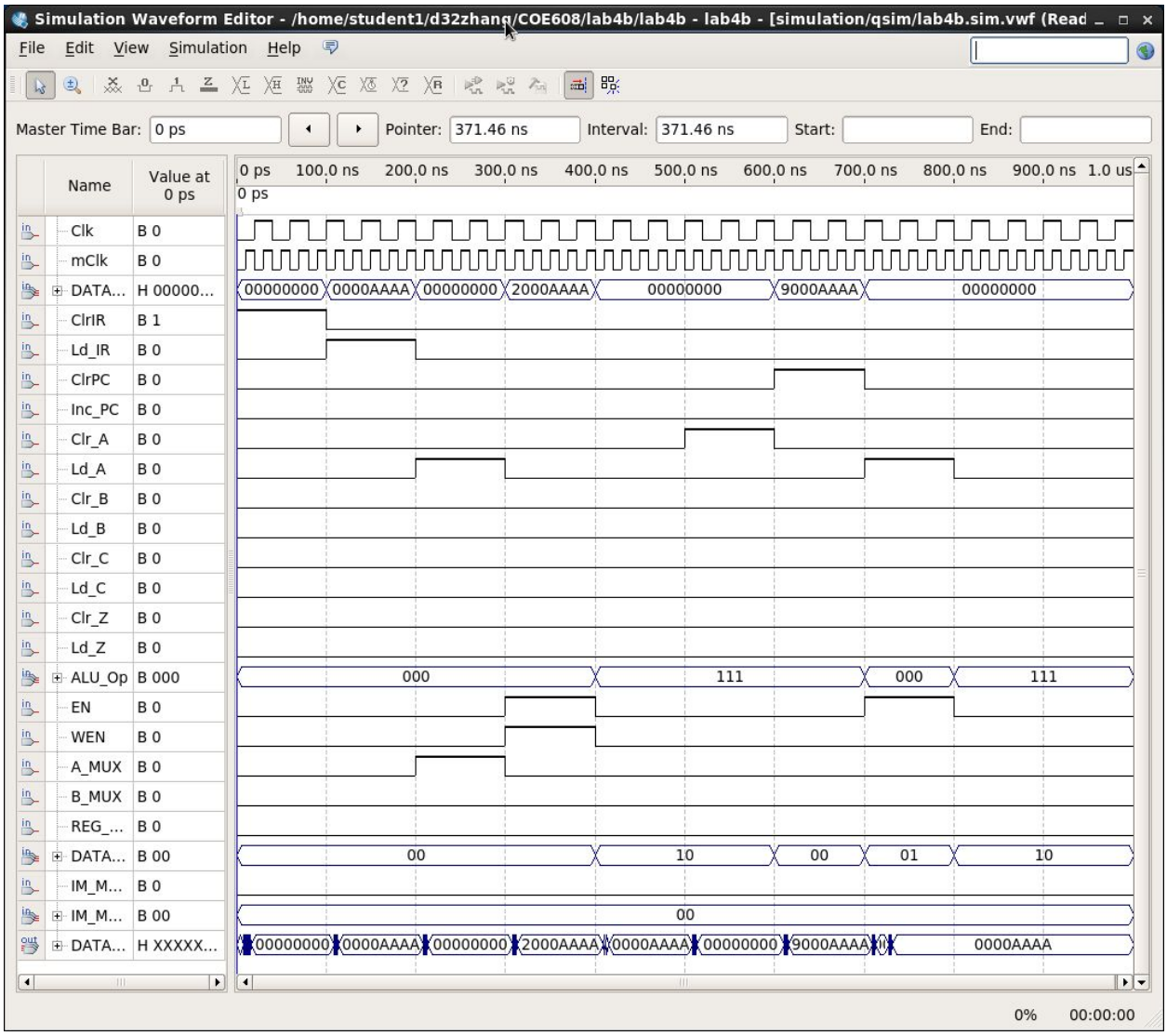
18. STA.vwf

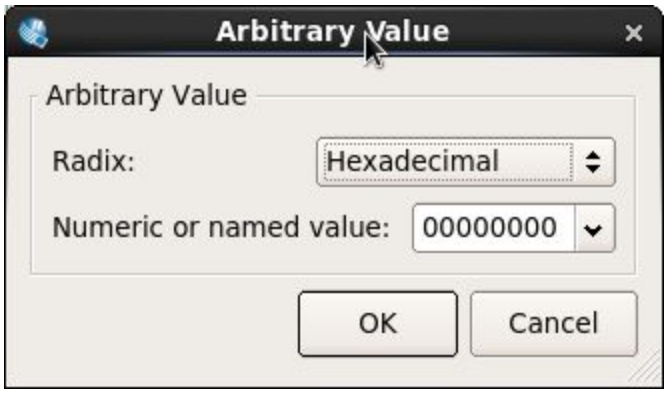

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 678.96 ns
Interval: 678.96 ns
Start:
End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
00000000 0000AAAA 00000000 2000AAAA 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00 01
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 0000AAAA 00000000 0000AAAA 00000000
0% 00:00:00

19. STB.vwf

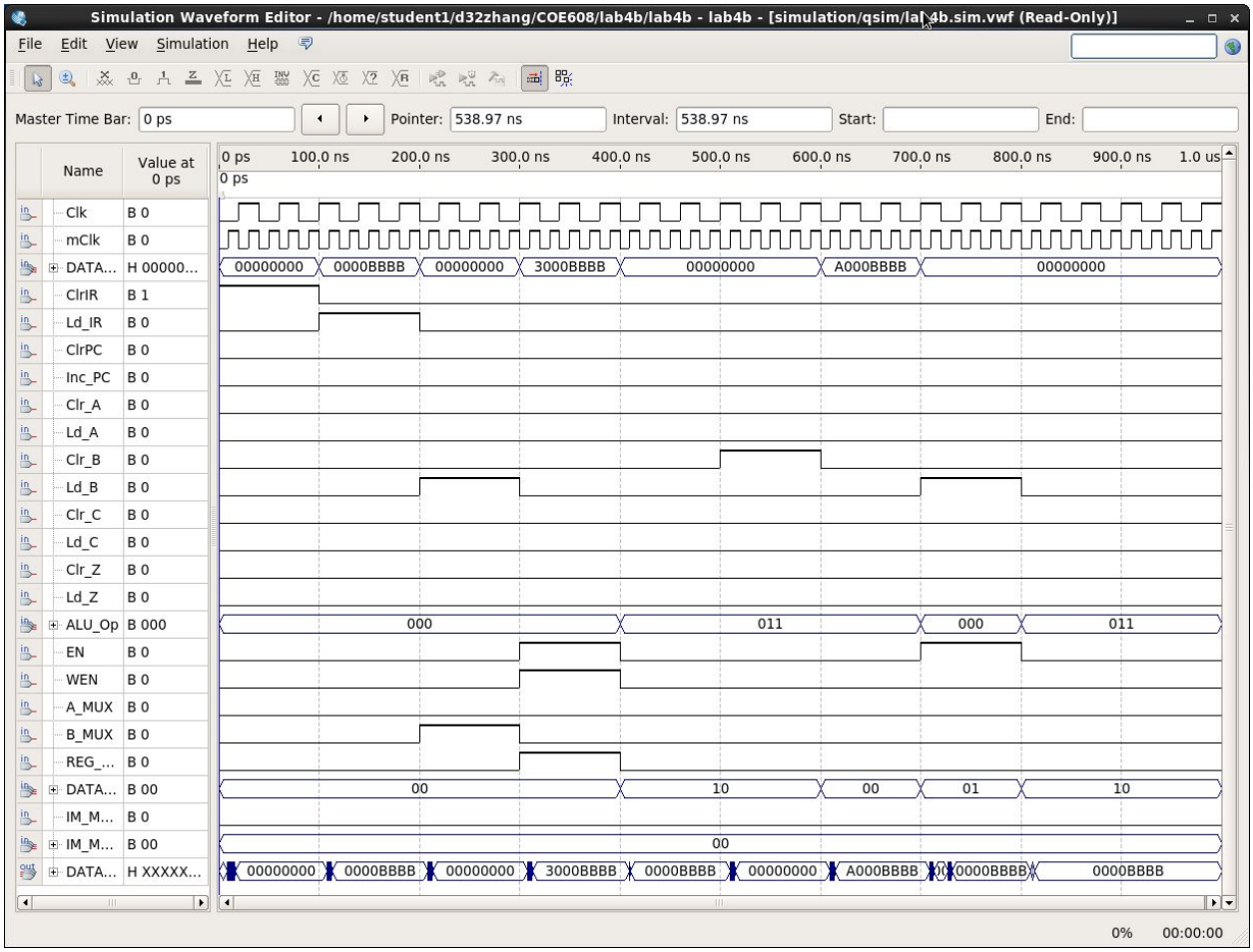

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Master Time Bar: 0 ps
Pointer: 468.63 ns
Interval: 468.63 ns
Start:
End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
00000000 0000BBBB 00000000 3000000B 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00 01
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 0000BBBB 00000000 0000BBBB 00000000
0% 00:00:00

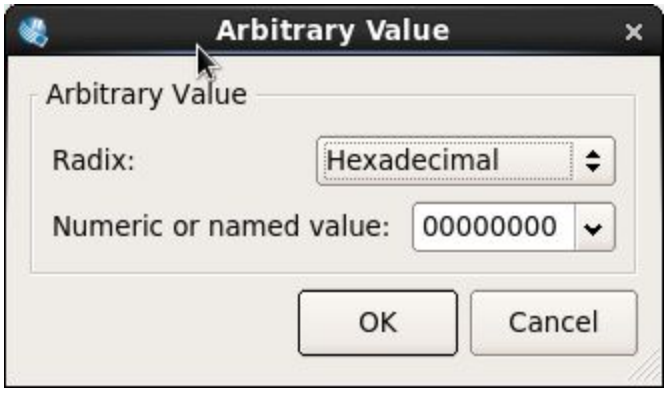
20. LDA.vwf


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Master Time Bar: 0 ps Pointer: 371.46 ns Interval: 371.46 ns Start: End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
00000000 0000AAAA 00000000 2000AAAA 00000000 9000AAAA 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000 111 000 111
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00 10 00 01 10
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 0000AAAA 00000000 2000AAAA 0000AAAA 00000000 9000AAAA 0000AAAA
0% 00:00:00

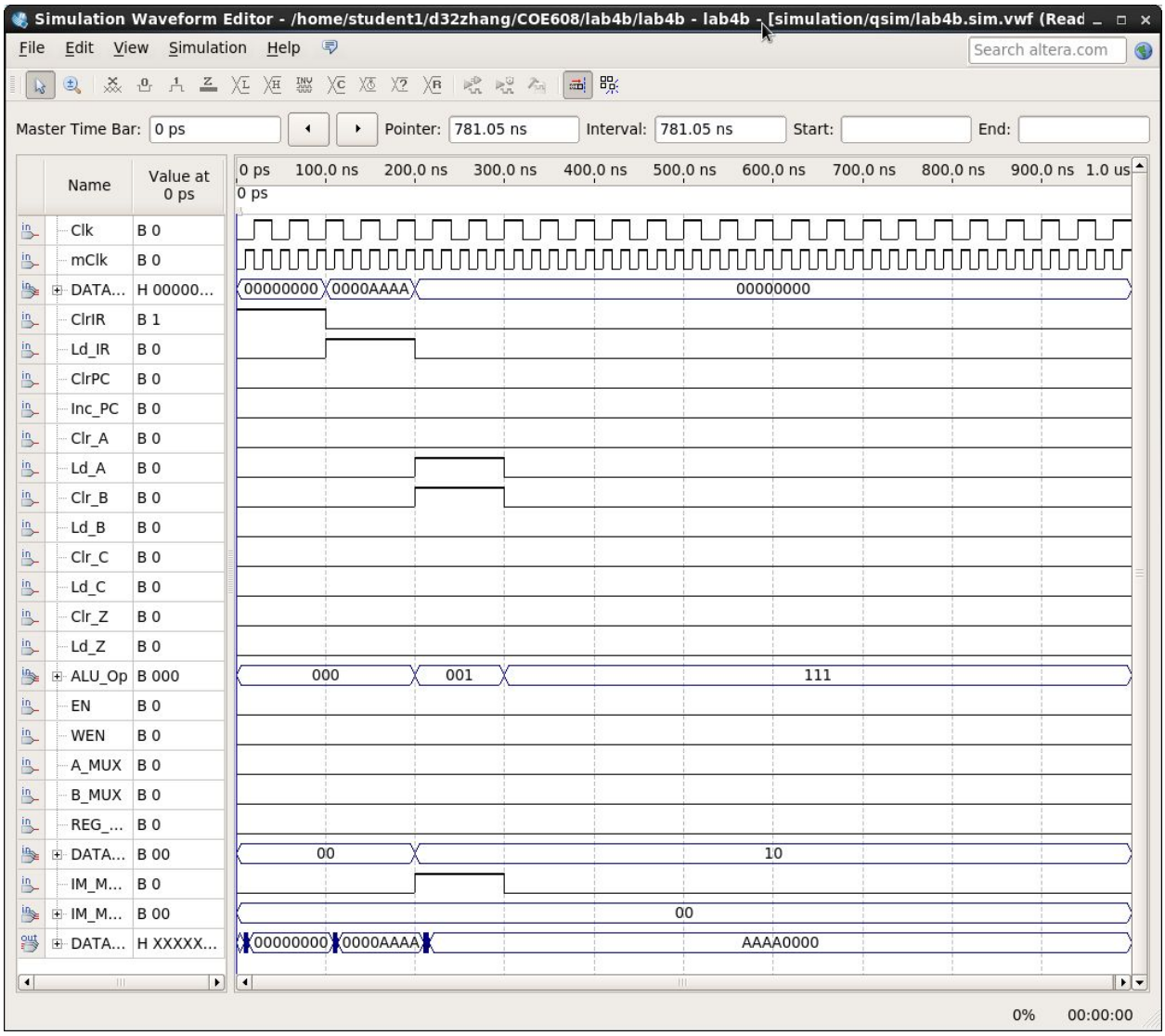

Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 00000000
OK
Cancel

21. LDB

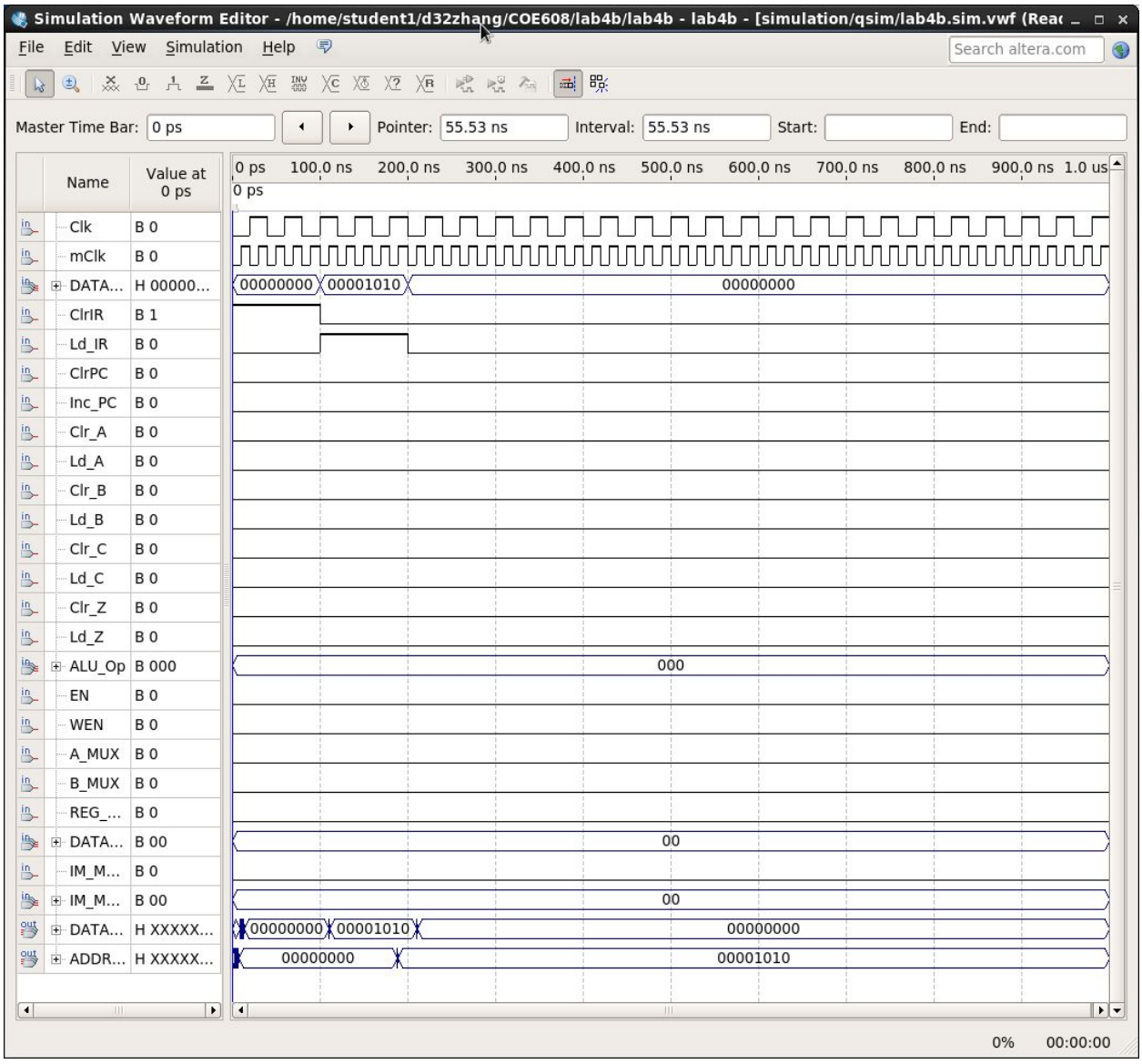

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read-Only)]
File Edit View Simulation Help
Master Time Bar: 0 ps Pointer: 538.97 ns Interval: 538.97 ns Start: End:
Name
Value at 0 ps
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
00000000 0000BBBB 00000000 3000BBBB 00000000 A000BBBB 00000000
000 011 000 011
00 10 00 01 10
00
0% 00:00:00


Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 00000000
OK
Cancel

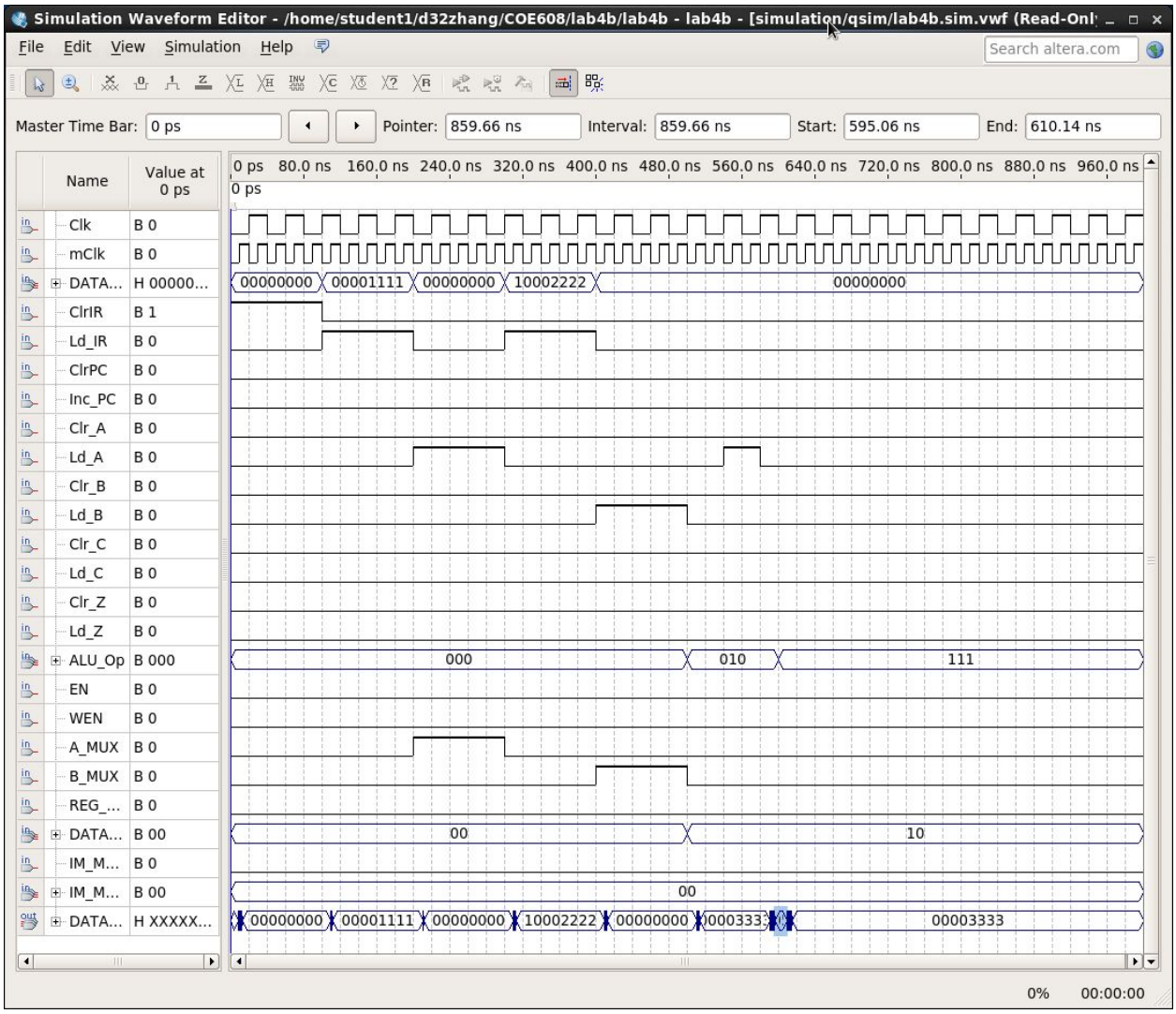
22. LUI

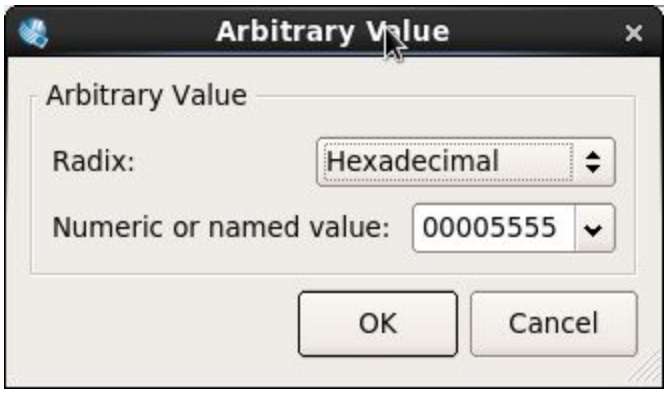

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 781.05 ns
Interval: 781.05 ns
Start:
End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
0000AAAA
00000000
000
001
111
00
10
00
00000000
0000AAAA
AAAA0000
0%
00:00:00

23. JMP

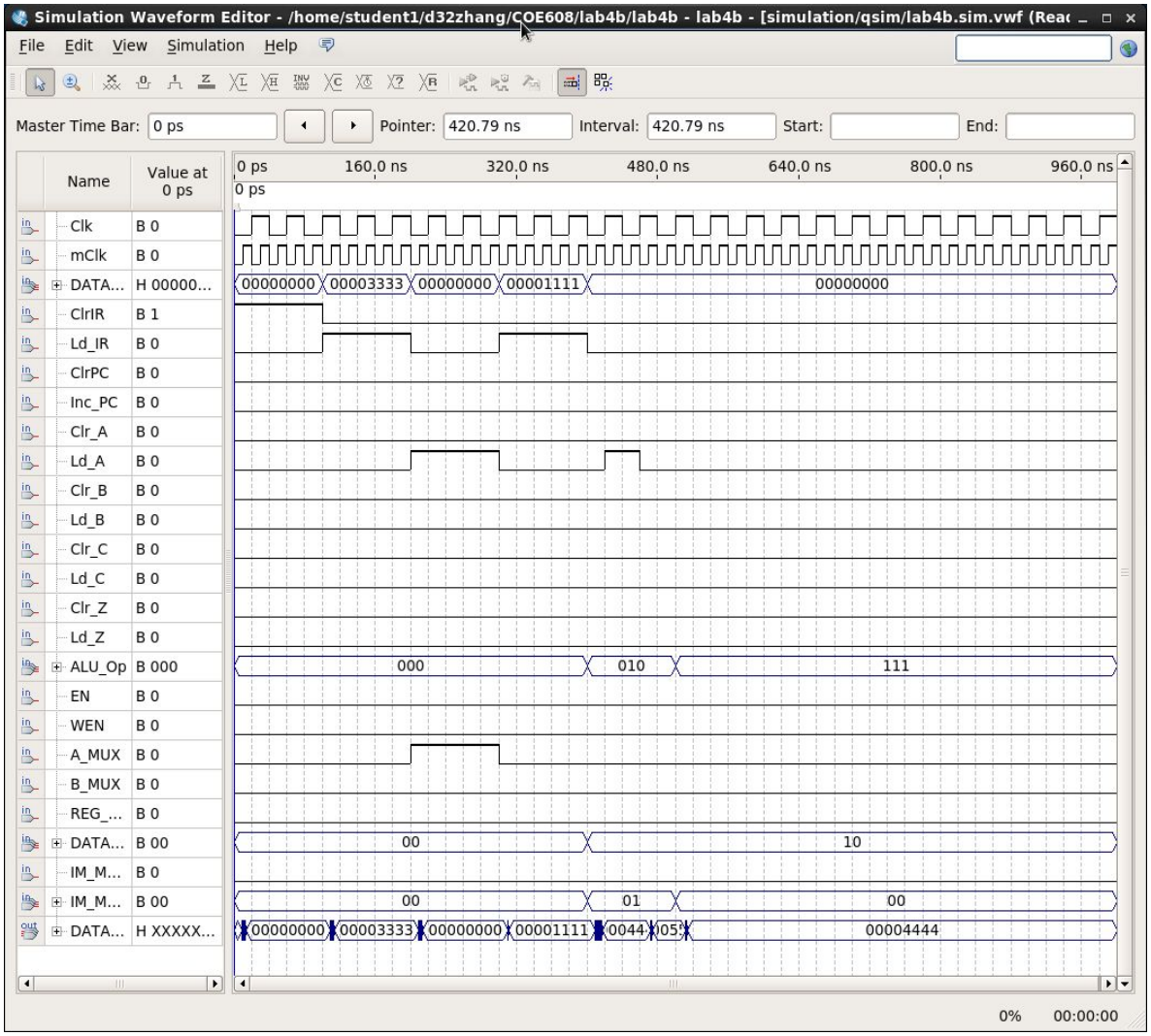

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 55.53 ns
Interval: 55.53 ns
Start:
End:
Name
Value at 0 ps
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us
Clk B 0
mClk B 0
DATA... H 00000...
00000000 00001010 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000 00001010 00000000
ADDR... H XXXXX...
00000000 00001010
0% 00:00:00

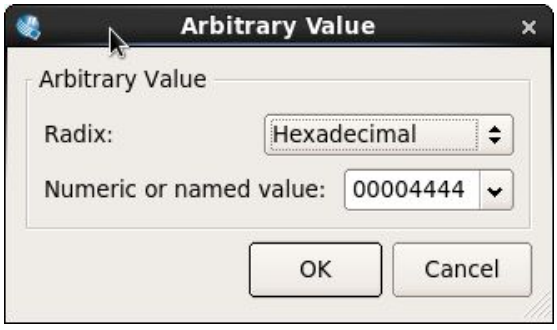
24. ADD

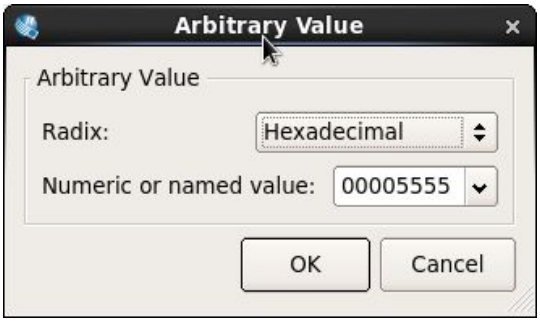

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Read-Onl
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 859.66 ns
Interval: 859.66 ns
Start: 595.06 ns
End: 610.14 ns
Name
Value at 0 ps
0 ps 80.0 ns 160.0 ns 240.0 ns 320.0 ns 400.0 ns 480.0 ns 560.0 ns 640.0 ns 720.0 ns 800.0 ns 880.0 ns 960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000 00001111 00000000 10002222 00000000
000 010 111
00 10
00
00000000 00001111 00000000 10002222 00000000 00003333 00003333
0% 00:00:00


Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 00005555
OK
Cancel

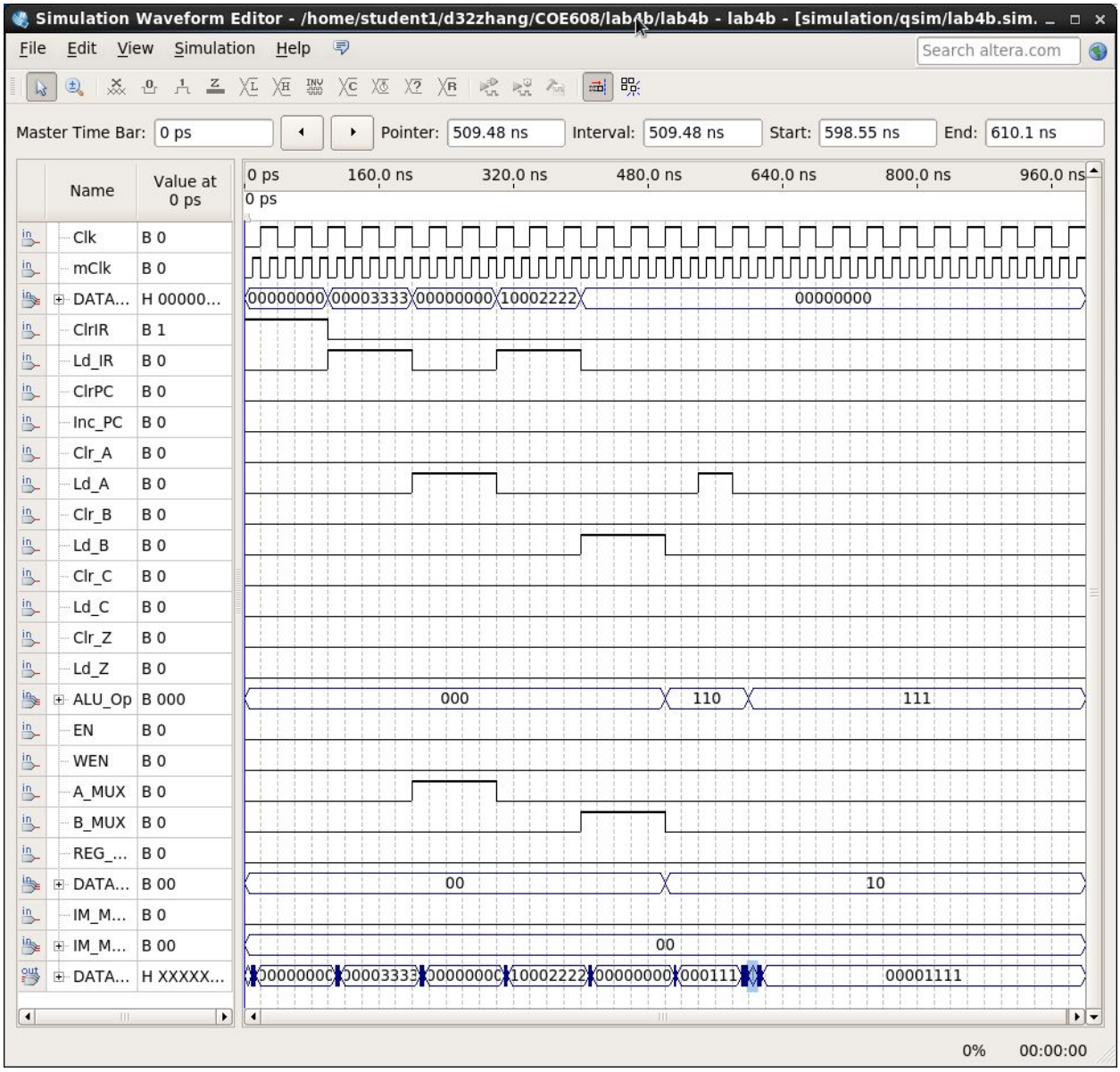
25. ADDI

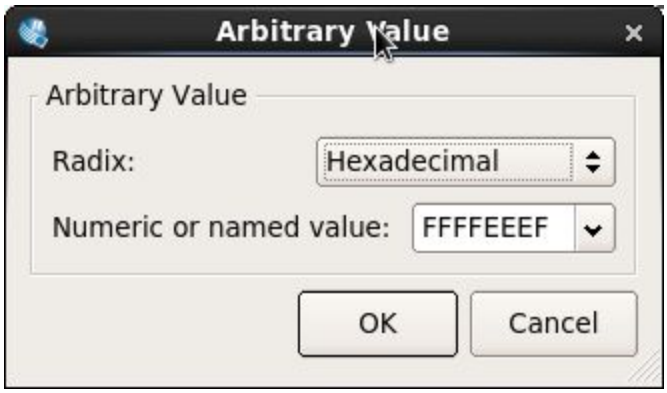

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.vwf (Reac
File Edit View Simulation Help
Master Time Bar: 0 ps Pointer: 420.79 ns Interval: 420.79 ns Start: End:
Name
Value at 0 ps
0 ps 160.0 ns 320.0 ns 480.0 ns 640.0 ns 800.0 ns 960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
00000000 00003333 00000000 00001111 00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000 010 111
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00 10
IM_M... B 0
IM_M... B 00
00 01 00
DATA... H XXXXX...
00000000 00003333 00000000 00001111 0044 005 00004444
0% 00:00:00


Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 00004444
OK Cancel

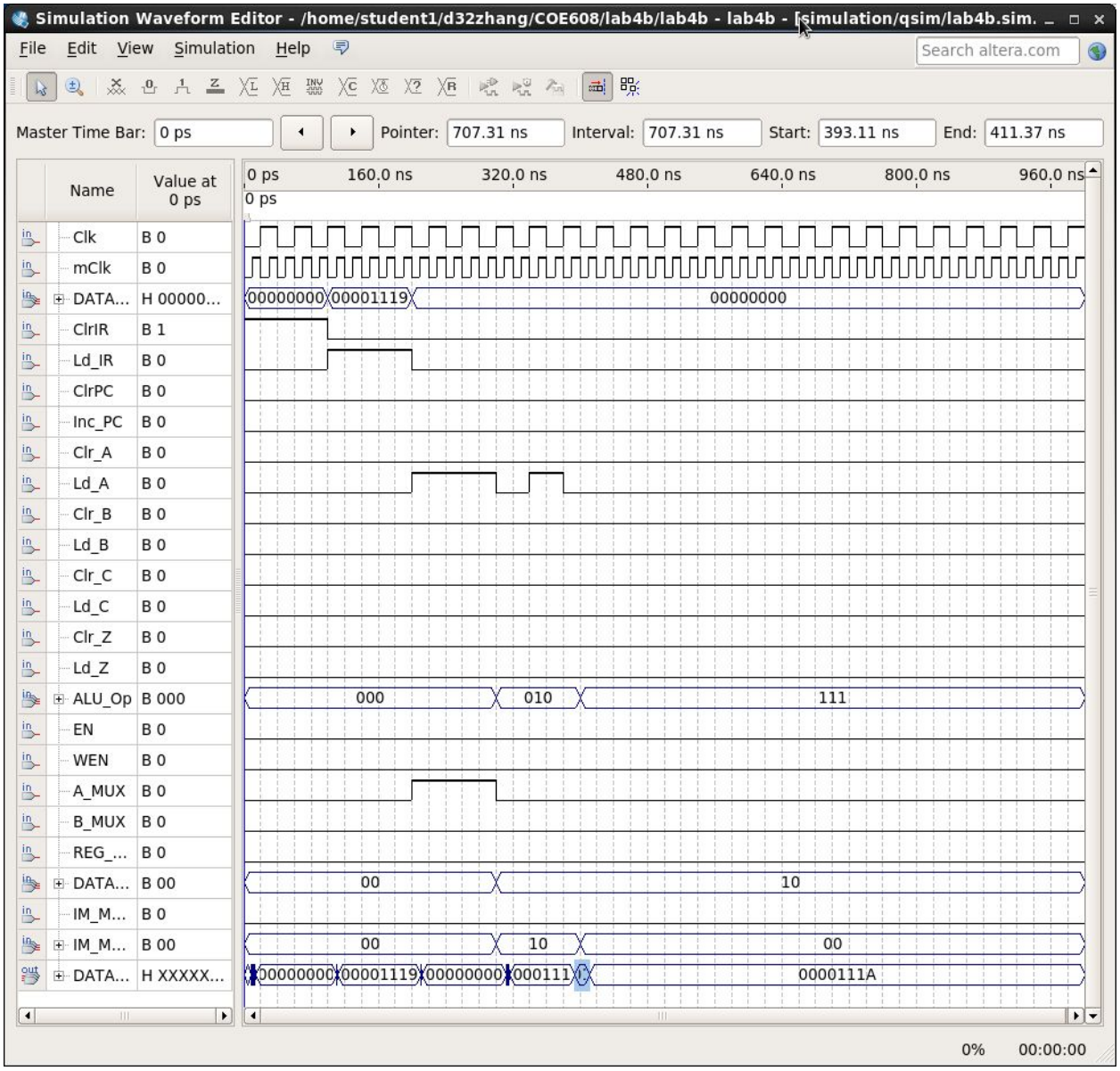

Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 00005555
OK Cancel

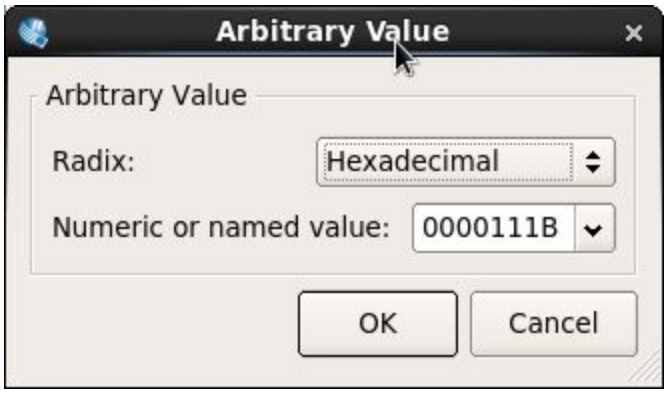
26. SUB


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 509.48 ns
Interval: 509.48 ns
Start: 598.55 ns
End: 610.1 ns
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
00000000
00003333
00000000
10002222
00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000
110
111
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00
10
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00001111
0%
00:00:00

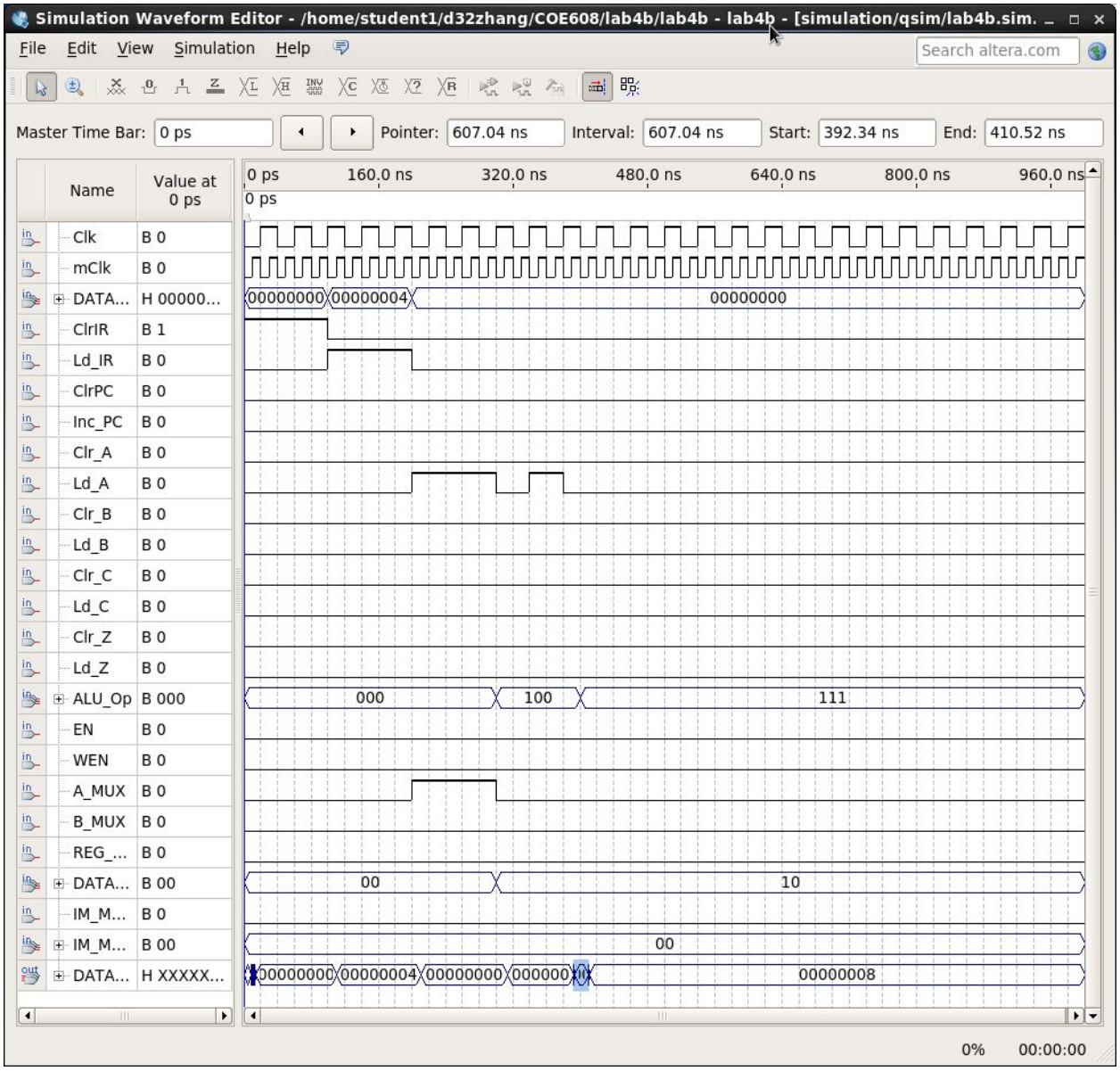

Arbitrary Value
Arbitrary Value
Radix:
Hexadecimal
Numeric or named value:
FFFFEEEF
OK
Cancel

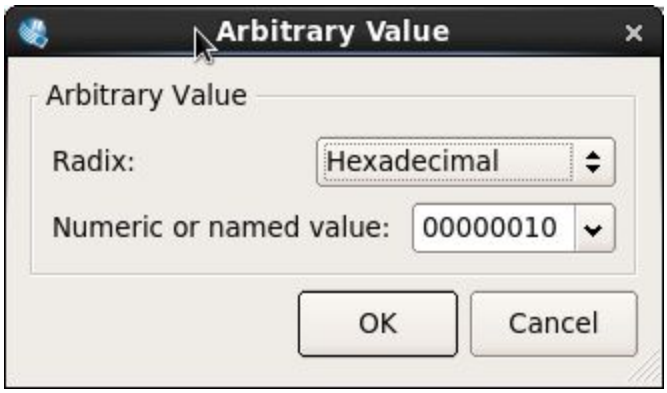
27. INCA


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 707.31 ns
Interval: 707.31 ns
Start: 393.11 ns
End: 411.37 ns
Name
Value at 0 ps
0 ps 160.0 ns 320.0 ns 480.0 ns 640.0 ns 800.0 ns 960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000 00001119 00000000
000 010 111
00 10
00 10 00
00000000 00001119 00000000 0000111B 0000111A
0% 00:00:00

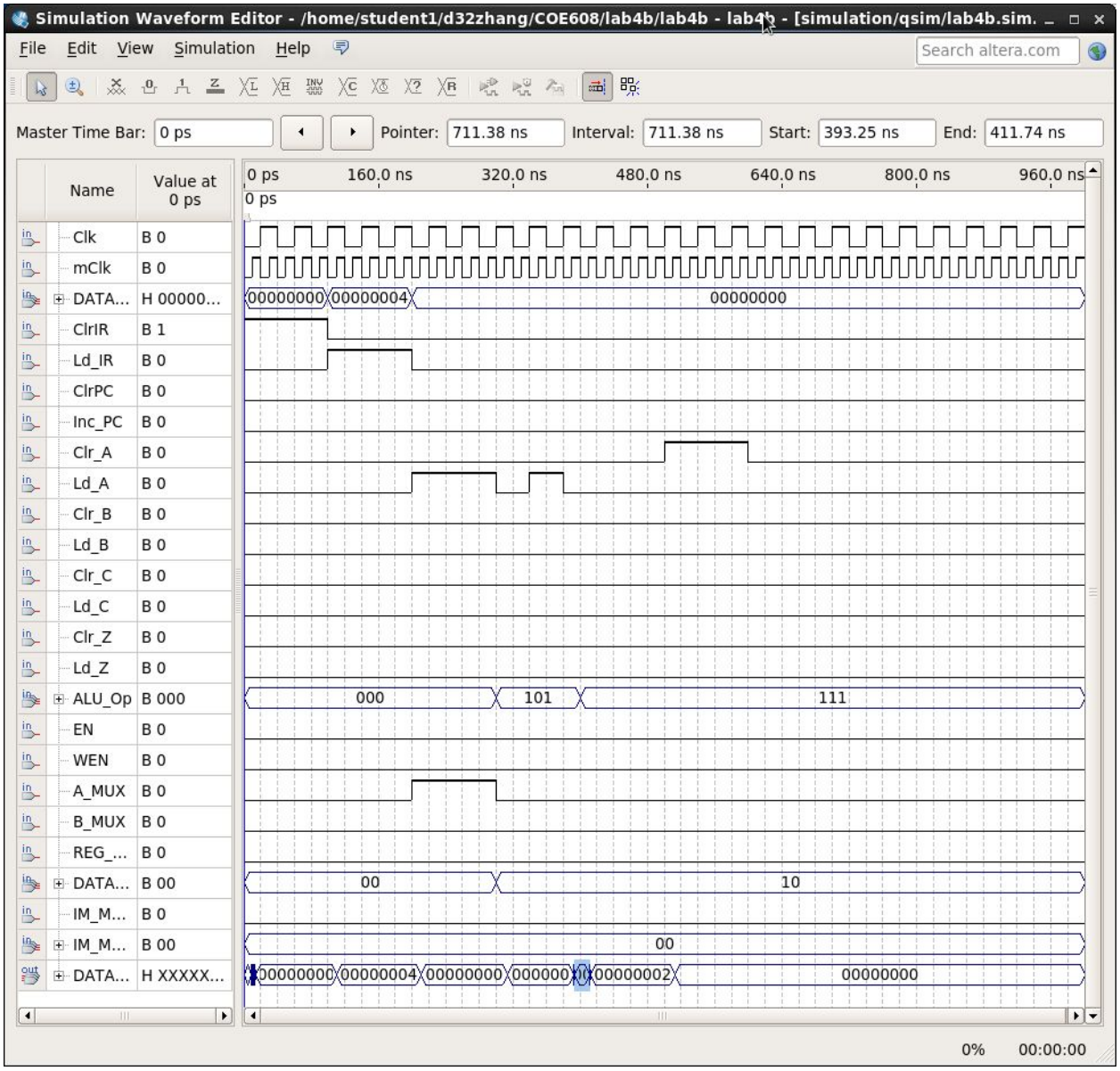

Arbitrary Value
Arbitrary Value
Radix: Hexadecimal
Numeric or named value: 0000111B
OK
Cancel

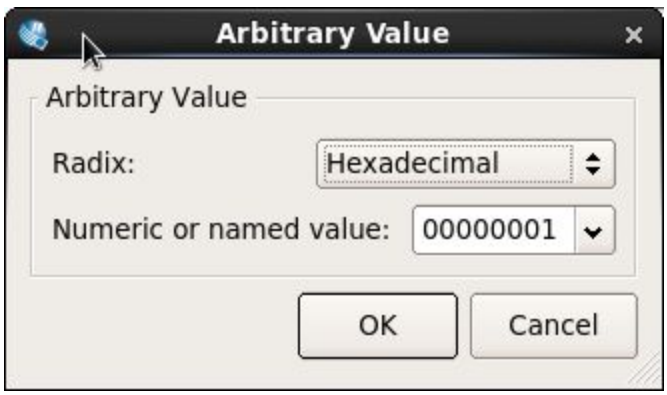
28. ROL


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 607.04 ns
Interval: 607.04 ns
Start: 392.34 ns
End: 410.52 ns
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
00000004
00000008
000
100
111
00
10
0%
00:00:00

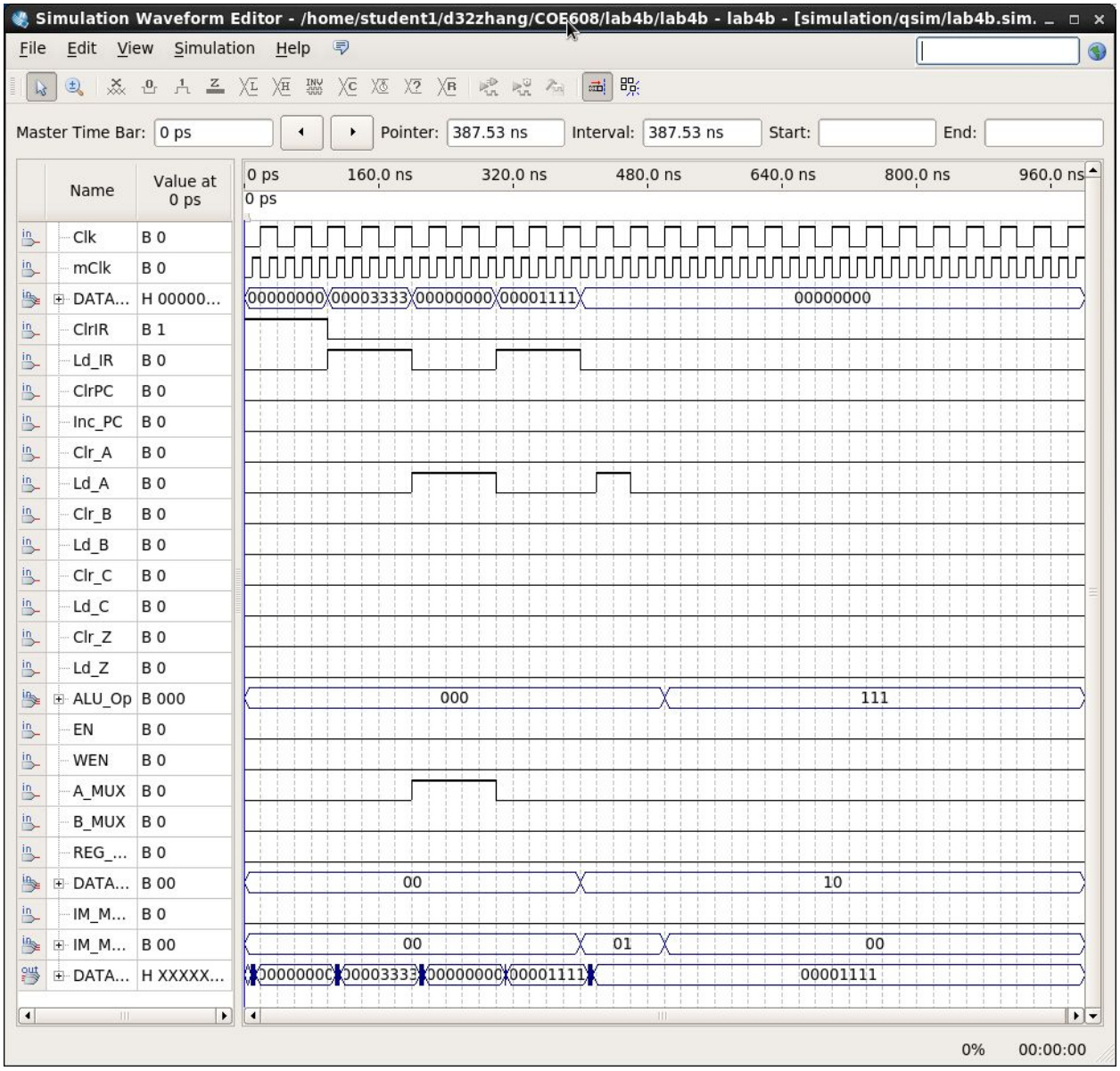

Arbitrary Value
Arbitrary Value
Radix:
Hexadecimal
Numeric or named value:
00000010
OK
Cancel

29. CLRA

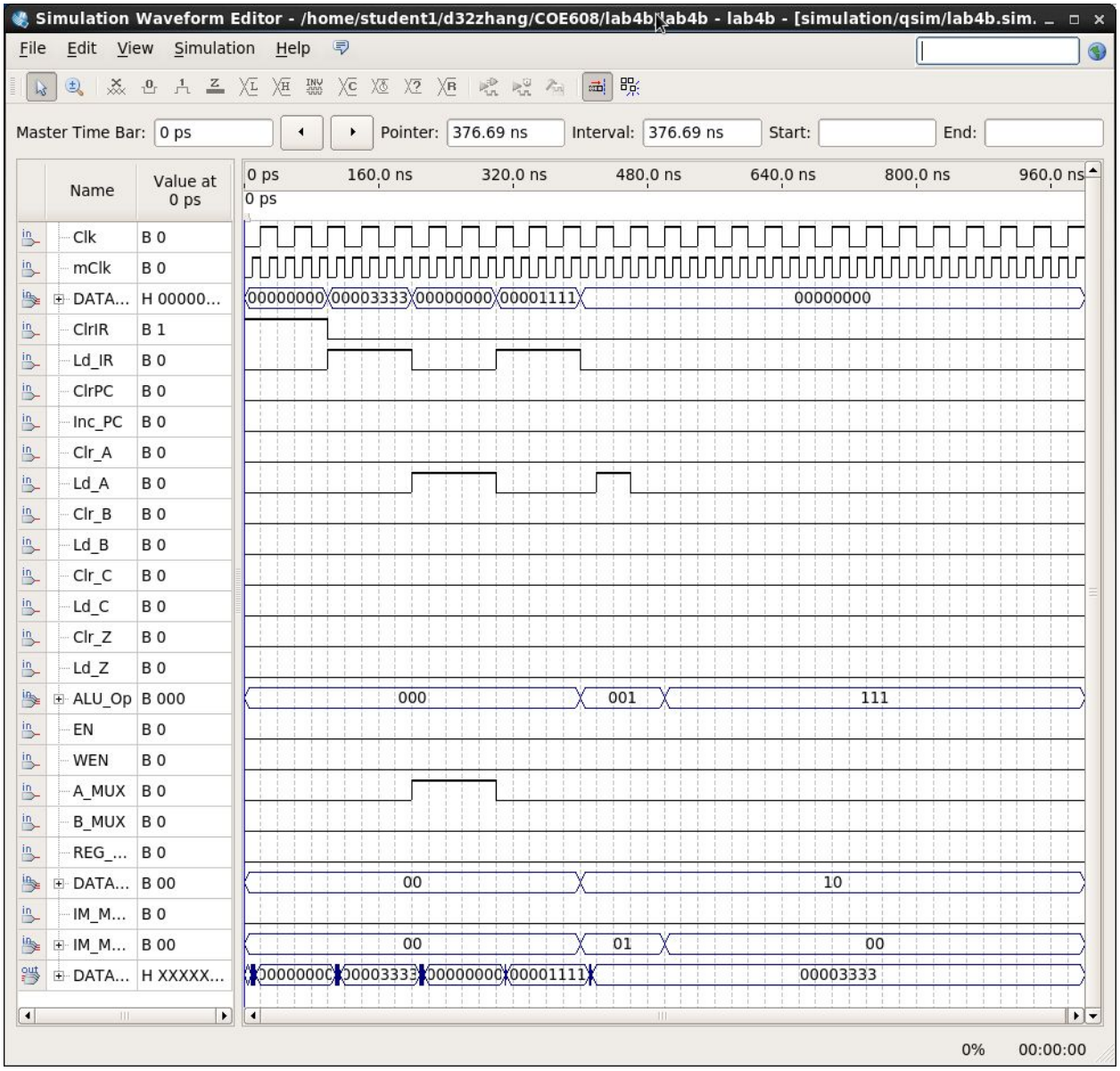

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 711.38 ns
Interval: 711.38 ns
Start: 393.25 ns
End: 411.74 ns
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
00000004
00000000
000
101
111
00
10
00
00000000
00000004
00000000
000000
00000002
00000000
0%
00:00:00


Arbitrary Value
Arbitrary Value
Radix:
Hexadecimal
Numeric or named value:
00000001
OK
Cancel

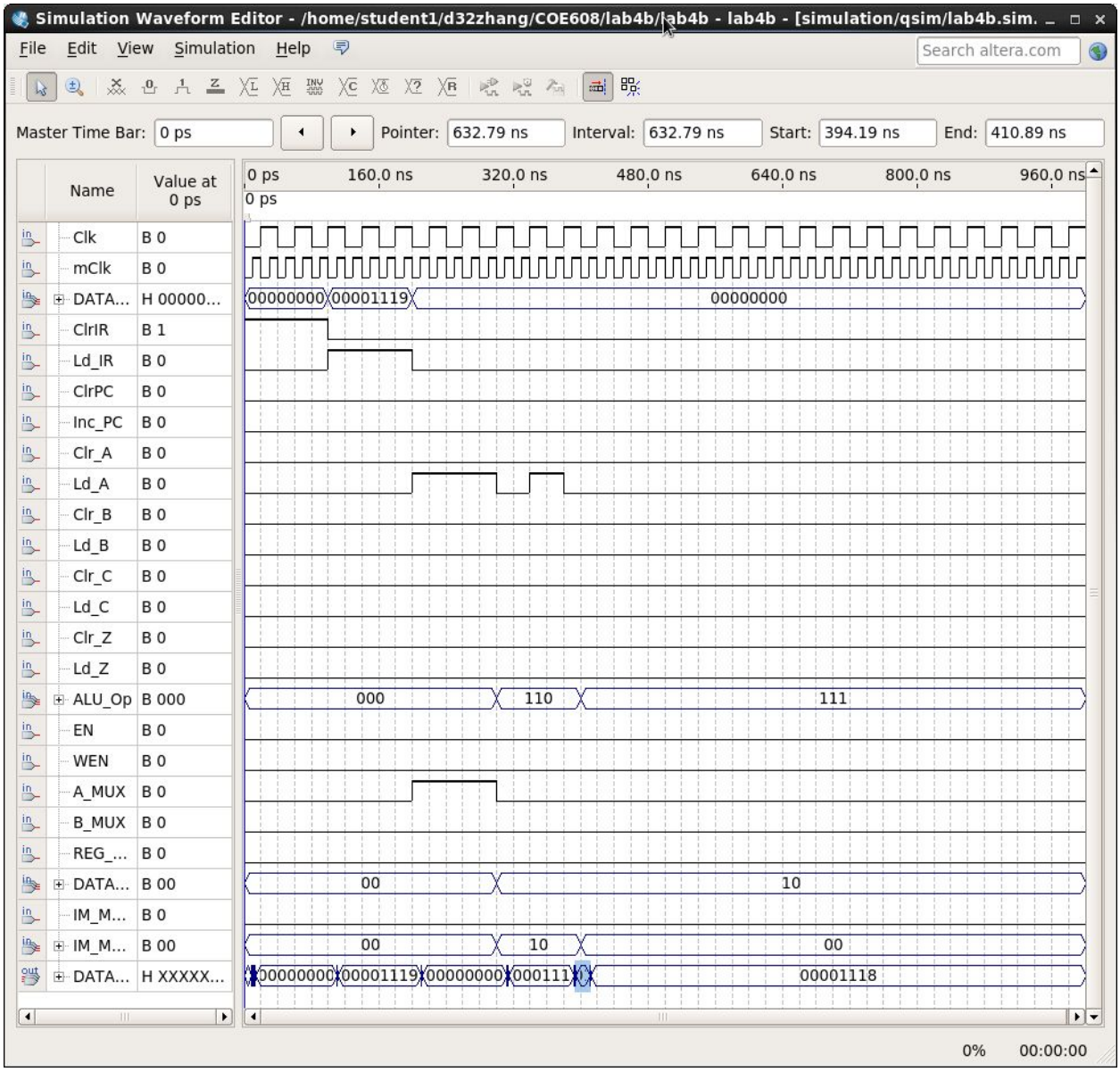
30. ANDI

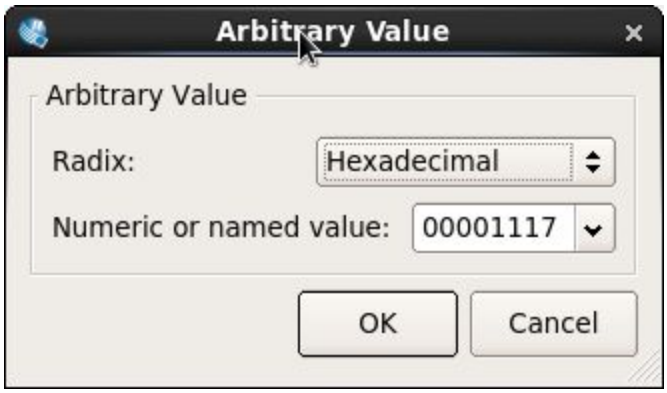

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Master Time Bar: 0 ps
Pointer: 387.53 ns
Interval: 387.53 ns
Start:
End:
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
00003333
00000000
00001111
00000000
000
111
00
10
00
01
00
00001111
0%
00:00:00

31. ORI

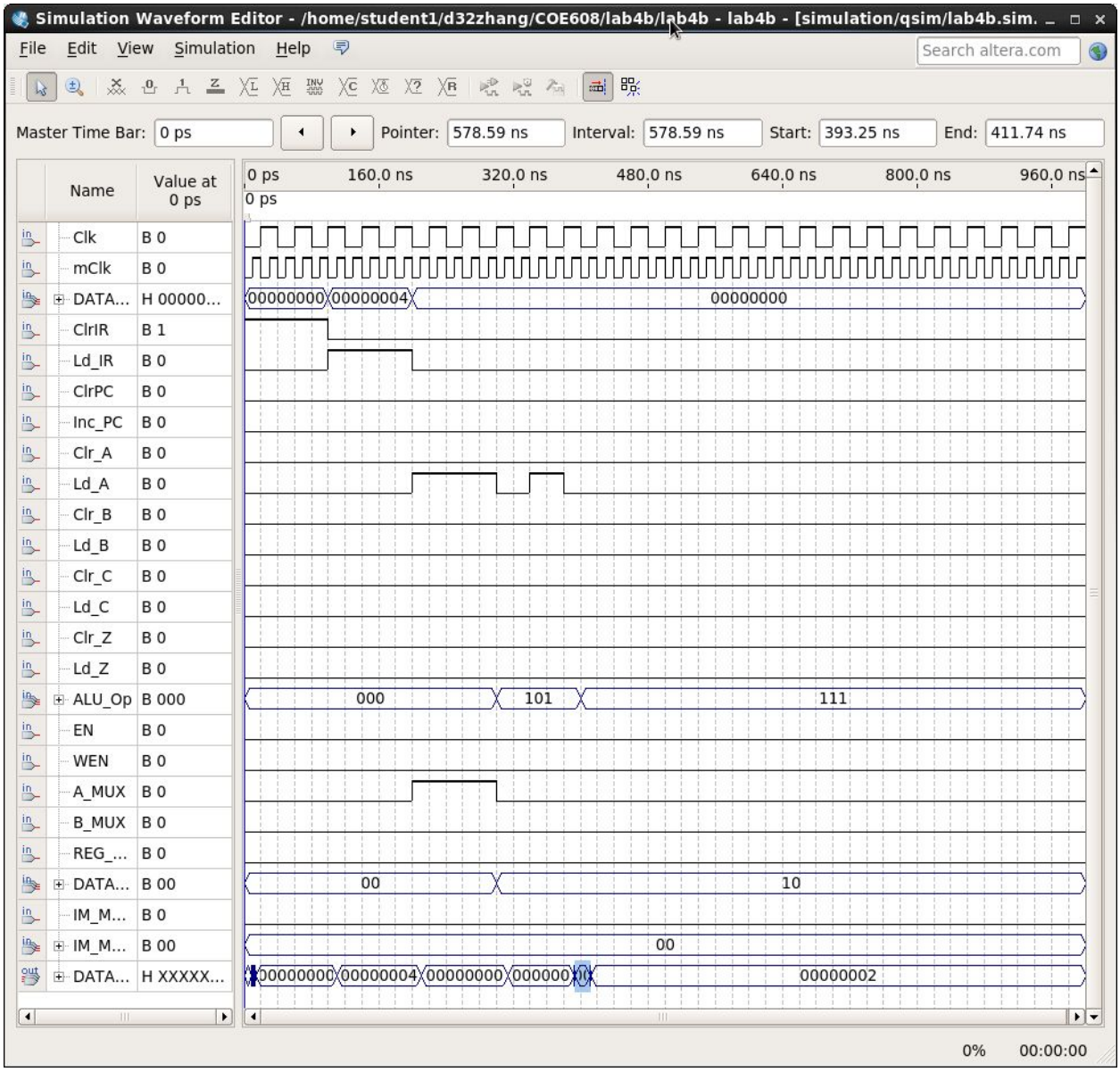

Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Master Time Bar: 0 ps
Pointer: 376.69 ns
Interval: 376.69 ns
Start:
End:
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
00003333
00000000
00001111
00000000
000
001
111
00
10
00
01
00
00003333
0%
00:00:00

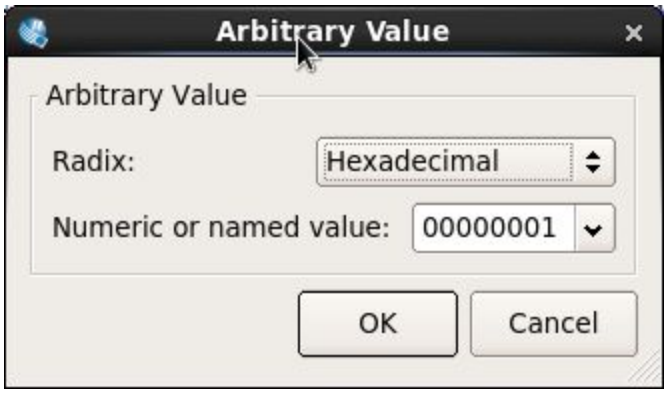
32. DECA


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 632.79 ns
Interval: 632.79 ns
Start: 394.19 ns
End: 410.89 ns
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
IM_M... B 0
IM_M... B 00
DATA... H XXXXX...
00000000
00001119
00000000
000
110
111
00
10
00
10
00
00000000
00001119
00000000
000111
00001118
0%
00:00:00


Arbitrary Value
Arbitrary Value
Radix:
Hexadecimal
Numeric or named value:
00001117
OK
Cancel

33. ROR


Simulation Waveform Editor - /home/student1/d32zhang/COE608/lab4b/lab4b - lab4b - [simulation/qsim/lab4b.sim.
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 578.59 ns
Interval: 578.59 ns
Start: 393.25 ns
End: 411.74 ns
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
Clk B 0
mClk B 0
DATA... H 00000...
00000000
00000004
00000000
ClrIR B 1
Ld_IR B 0
ClrPC B 0
Inc_PC B 0
Clr_A B 0
Ld_A B 0
Clr_B B 0
Ld_B B 0
Clr_C B 0
Ld_C B 0
Clr_Z B 0
Ld_Z B 0
ALU_Op B 000
000
101
111
EN B 0
WEN B 0
A_MUX B 0
B_MUX B 0
REG_... B 0
DATA... B 00
00
10
IM_M... B 0
IM_M... B 00
00
DATA... H XXXXX...
00000000
00000004
00000000
00000000
00000002
0%
00:00:00


Arbitrary Value
Arbitrary Value
Radix:
Hexadecimal
Numeric or named value:
00000001
OK
Cancel

Student Name: Duanwei Zhang **Student#:** 500824903 **Section:** 04

INST	CLR_IR LD_IR	LD_PC INC_PC	CLR_A LD_A	CLR_B LD_B	CLR_C LD_C	CLR_Z LD_Z	ALU OP	EN WEN	A/B MUX	REG MUX	Data MUX	IM_MUX1 IM_MUX2
LDA	0/0	0/0	0/1	0/0	0/0	0/0	XXX	1/0	0/X	X	01	X
LDB	0/0	0/0	0/0	0/1	0/0	0/0	XXX	1/0	X/0	X	01	X
STA	0/0	0/0	0/0	0/0	0/0	0/0	XXX	1/1	X	0	X	X
STB	0/0	0/0	0/0	0/0	0/0	0/0	XXX	1/1	X	1	X	X
JMP	0/0	1/0	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X
LDAI	0/0	0/0	0/1	0/0	0/0	0/0	XXX	X	1/X	X	X	X
LDBI	0/0	0/0	0/0	0/1	0/0	0/0	XXX	X	X/1	X	X	X
LUI	0/0	0/0	0/1	1/0	0/0	0/0	001	X	0/X	X	10	1/X
ANDI	0/0	0/0	0/1	0/0	0/1	0/1	000	X	0/X	X	10	0/01
DECA	0/0	0/0	0/1	0/0	0/1	0/1	110	X	0/X	X	10	0/10
ADD	0/0	0/0	0/1	0/0	0/1	0/1	010	X	0/X	X	10	0/00
SUB	0/0	0/0	0/1	0/0	0/1	0/1	110	X	0/X	X	10	0/00
INCA	0/0	0/0	0/1	0/0	0/1	0/1	010	X	0/X	X	10	0/10
AND	0/0	0/0	0/1	0/0	0/1	0/1	000	X	0/X	X	10	0/09
ADDI	0/0	0/0	0/1	0/0	0/1	0/1	010	X	0/X	X	10	0/01
ORI	0/0	0/0	0/1	0/0	0/1	0/1	001	X	0/X	X	10	0/01
ROL	0/0	0/0	0/1	0/0	0/1	0/1	100	X	0/X	X	10	0/X
ROR	0/0	0/0	0/1	0/0	0/1	0/1	101	X	0/X	X	10	0/X
CLRA	0/0	0/0	1/0	0/0	0/0	0/0	XXX	X	X	X	X	X
CLRB	0/0	0/0	0/0	1/0	0/0	0/0	XXX	X	X	X	X	X
CLRC	0/0	0/0	0/0	0/0	1/0	0/0	XXX	X	X	X	X	X
CLRZ	0/0	0/0	0/0	0/0	0/0	1/0	XXX	X	X	X	X	X
PC <= PC+4	0/0	1/1	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X
IR <= M[INST]	0/1	0/0	0/0	0/0	0/0	0/0	XXX	X	X	X	00	X
PC <= IR[15..0]	0/0	1/0	0/0	0/0	0/0	0/0	XXX	X	X	X	X	X

Table 1: Data-Path Control Signals