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

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	1 0	0/X	X	1	X
LDB	0/0	0/0	0/0	0/1	0/0	0/0	XXX	1 0	X/0	X	1	X
STA	0/0	0/0	0/0	0/0	0/0	0/0	XXX	1 1	X	0	X	X
JMP	0/0	0/0	0/0	0/0	0/0	0/0	XXX	1 1	X	1	X	X
STAB	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	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	1 0	0/0	0/0	0/0	XXX	X	X	X	X	X
CLRZ	0/0	0/0	0/0	1 0	0/0	0/0	XXX	X	X	X	X	X
PC <= PC + 4	0/0	1 0	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	1 0	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

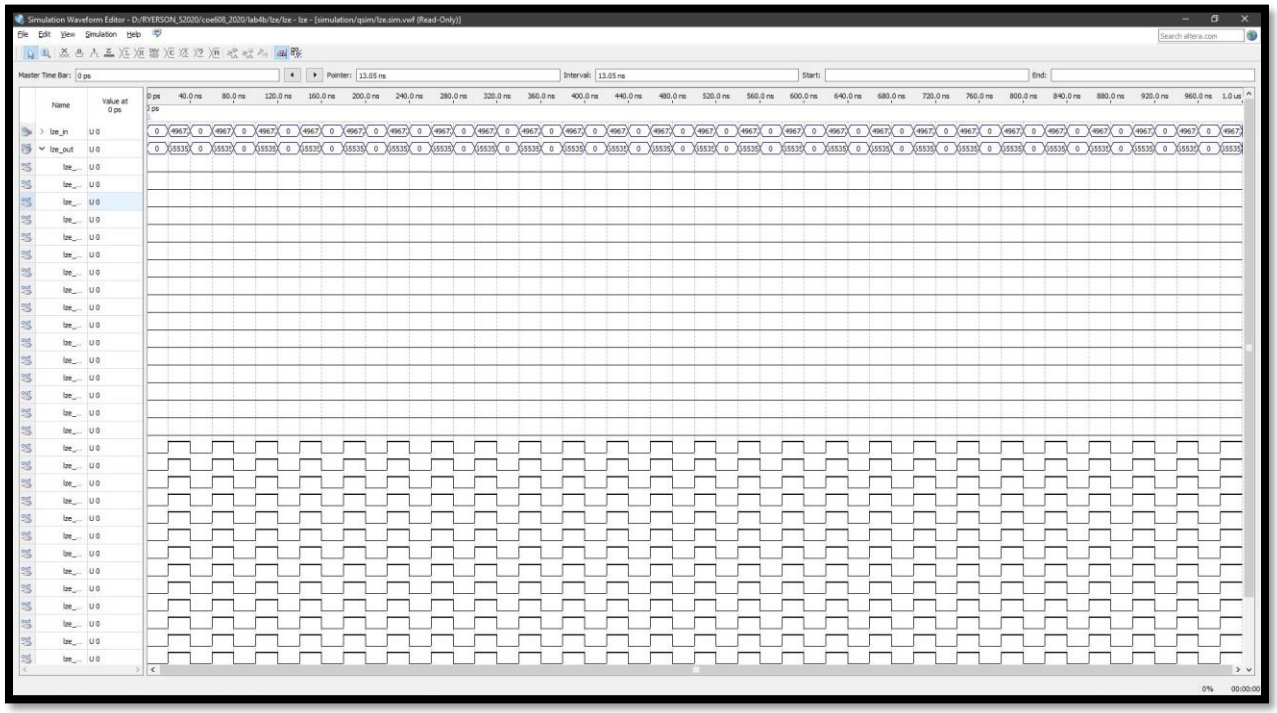


Figure 5: LZE Functional Waveform

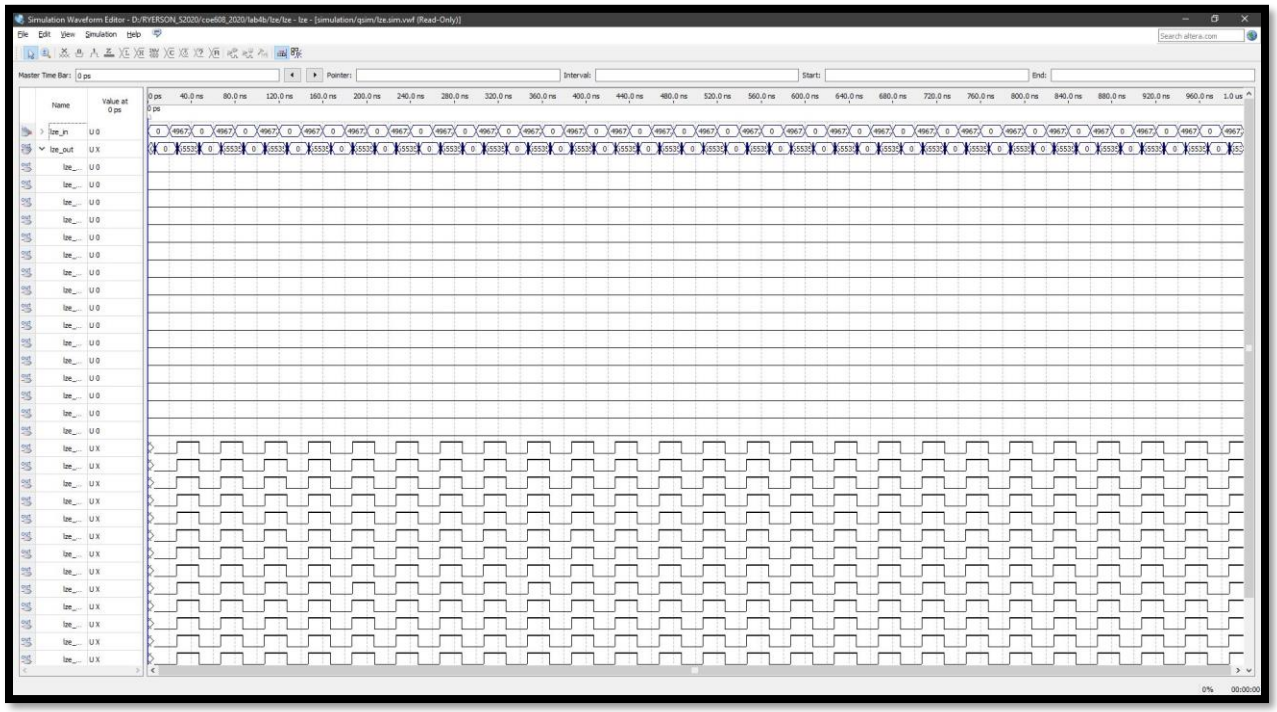


Figure 6: LZE Timing Waveform

UZE: Functional and Timing

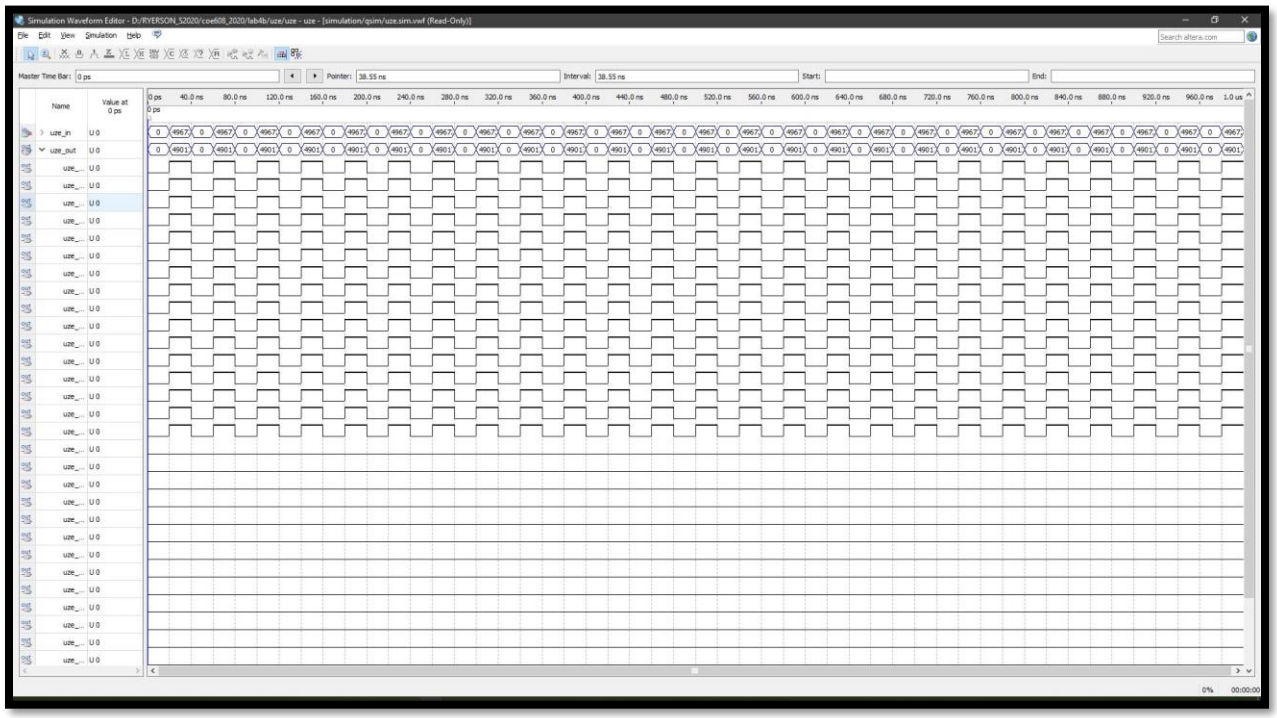


Figure 7: UZE Functional Waveform

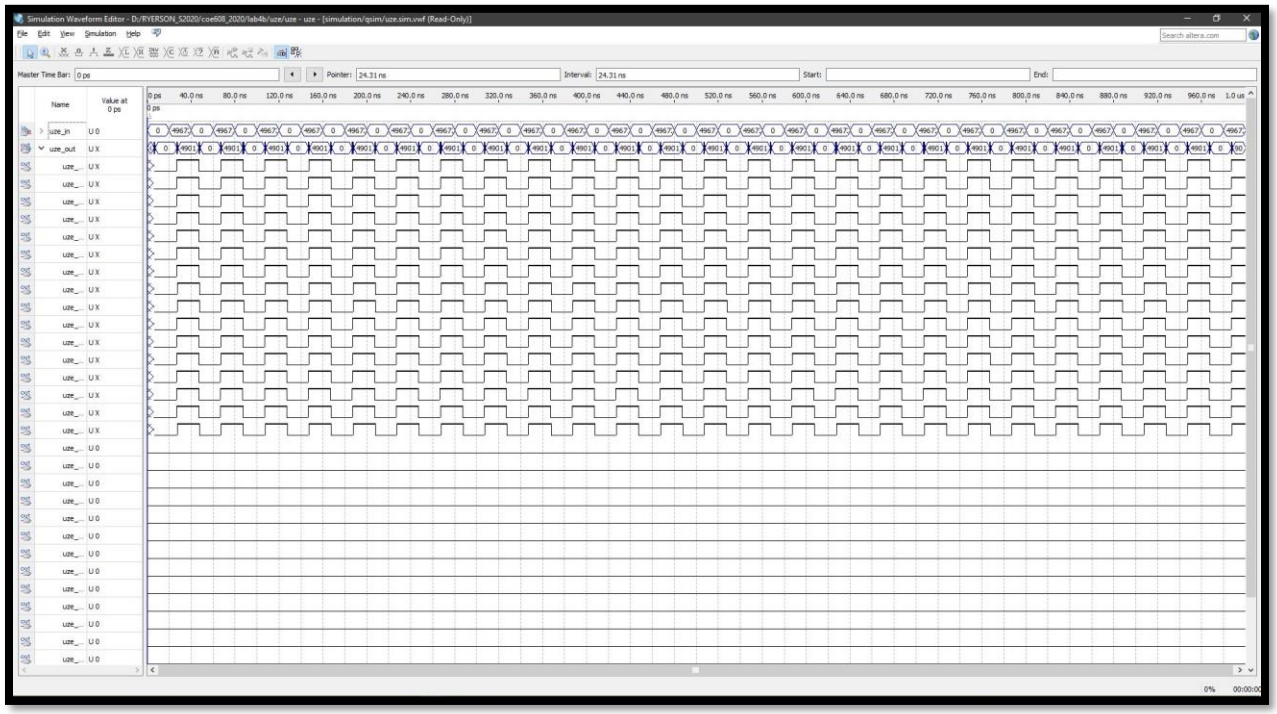


Figure 8: UZE Timing Waveform

Reducer: Functional and Timing

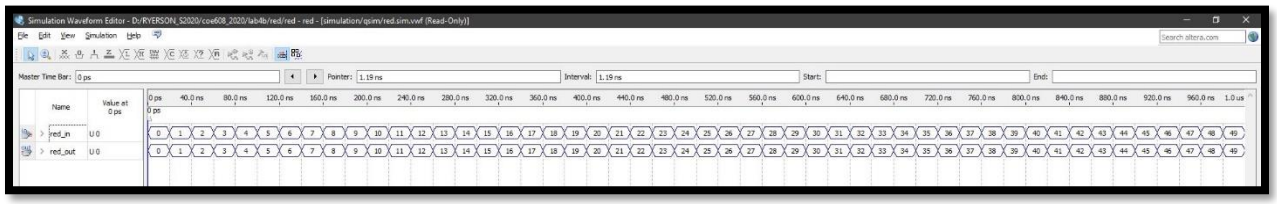


Figure 9:Reducer Functional Waveform

Figure 10: Reducer Timing Waveform

Data-path: Functional and Timing

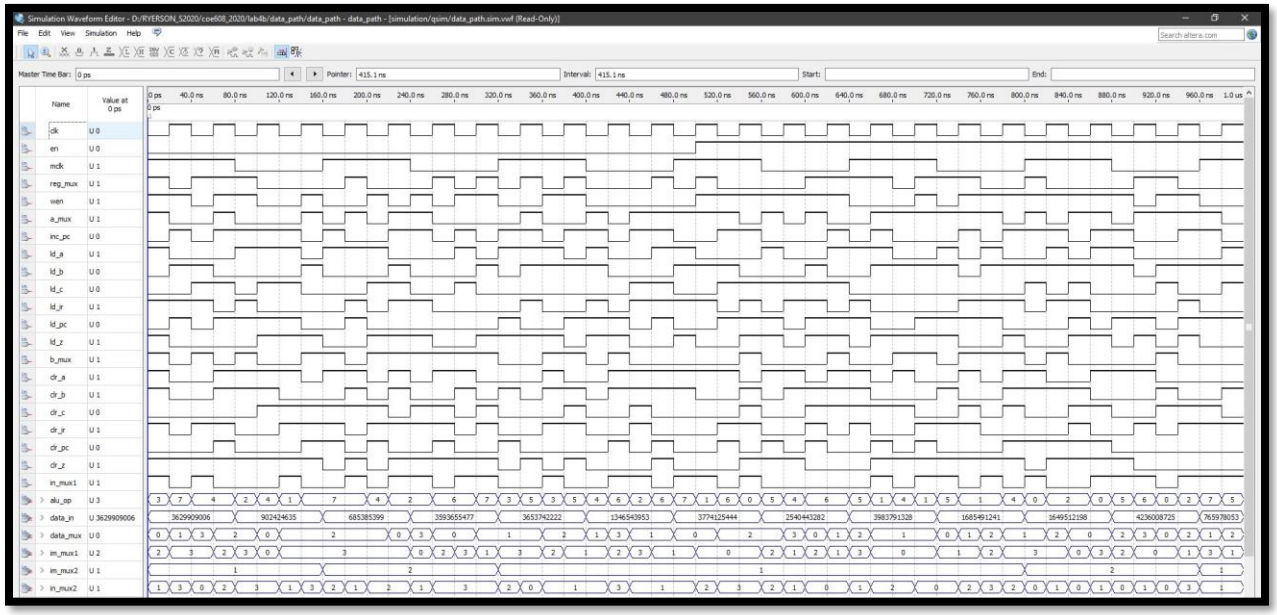


Figure 11: Data-path Random Input Waveform

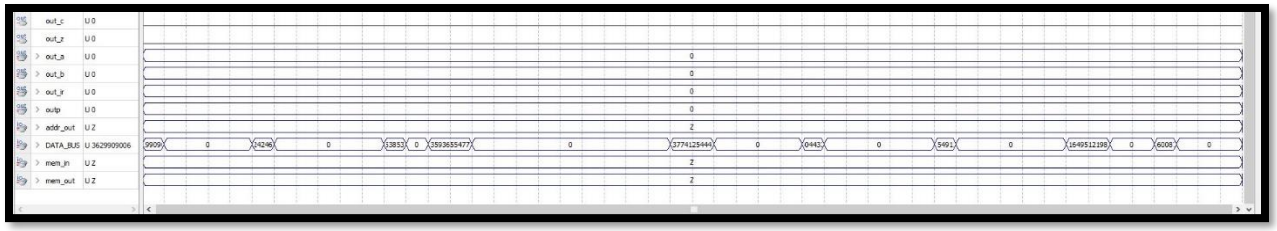


Figure 12: Data-path Output Functional Waveform

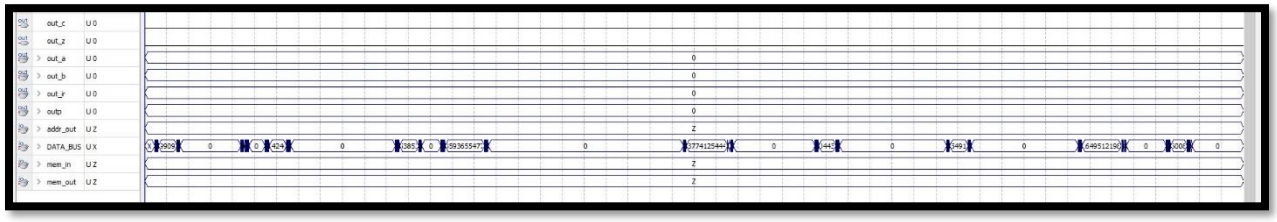


Figure 13: Data-path Output Timing Waveform

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.

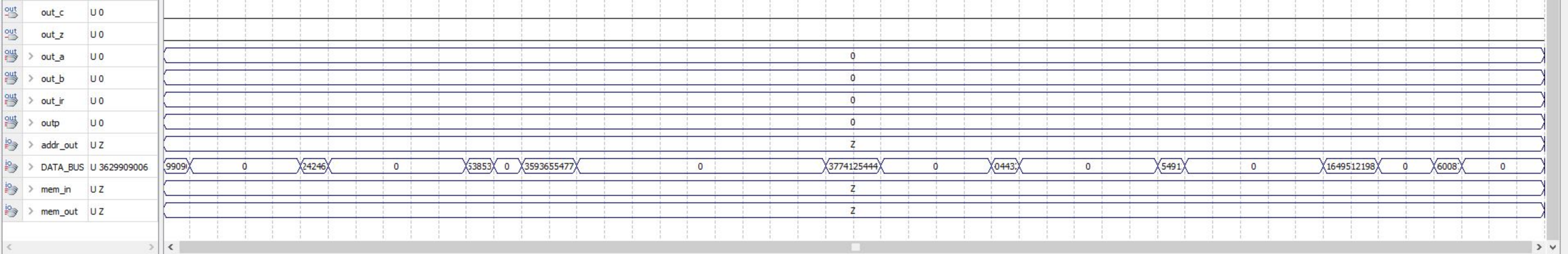
Appendix: VHDL Codes and Screenshots of Waveforms

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

```
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));
```

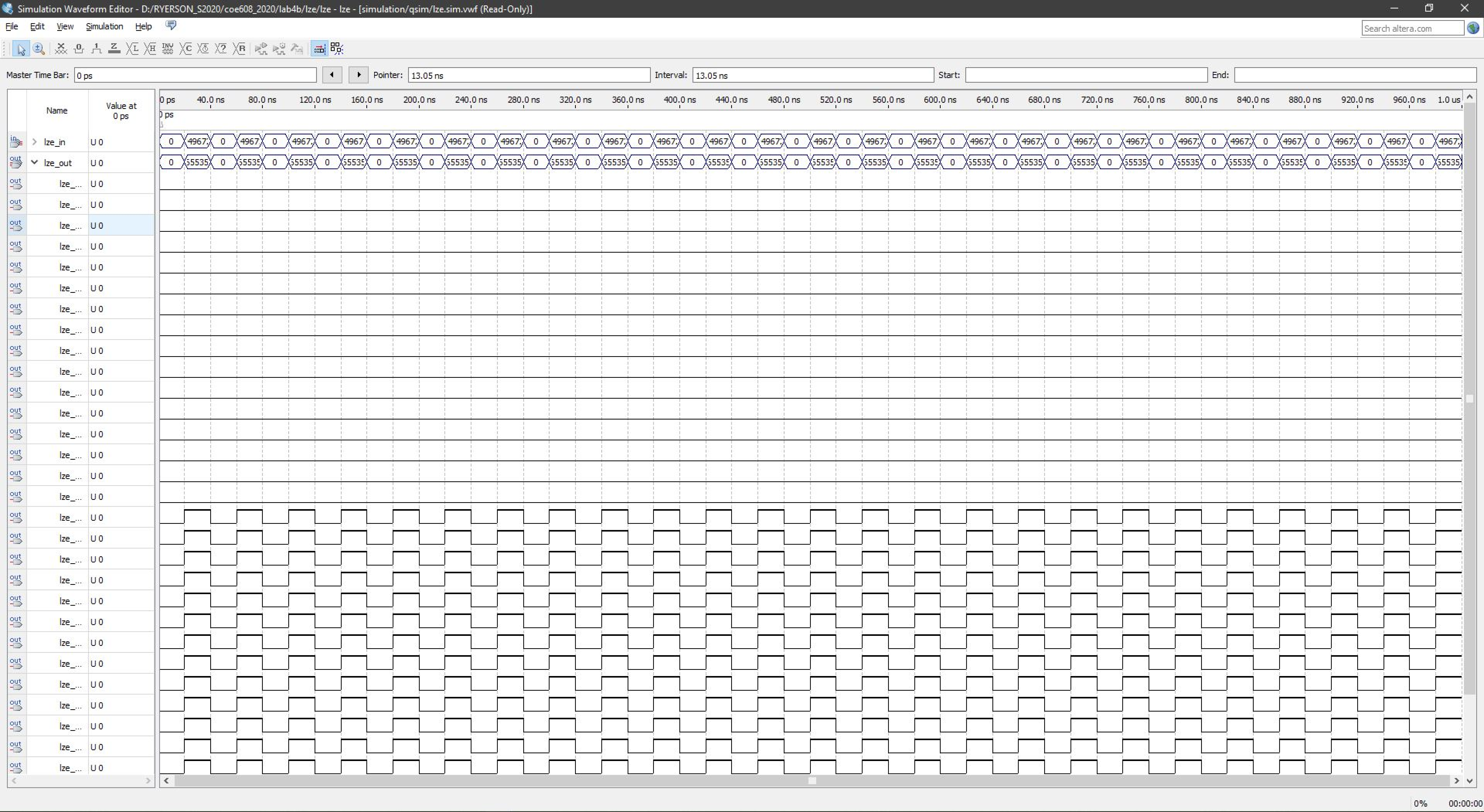
```
63      end component register32;
64
65      component pc
66         port(
67            clr, clk, ld, inc: in std_logic;
68            d: in std_logic_vector(31 downto 0);
69            q: inout std_logic_vector(31 downto 0));
70      end component pc;
71
72      component memoryModule
73         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
0 ps
lze_in U 0
0 4967 0 4967 0 4967 0 4967 0 4967 0 4967 0 4967
lze_out U 0
0 65535 0 65535 0 65535 0 65535 0 65535 0 65535 0 65535
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

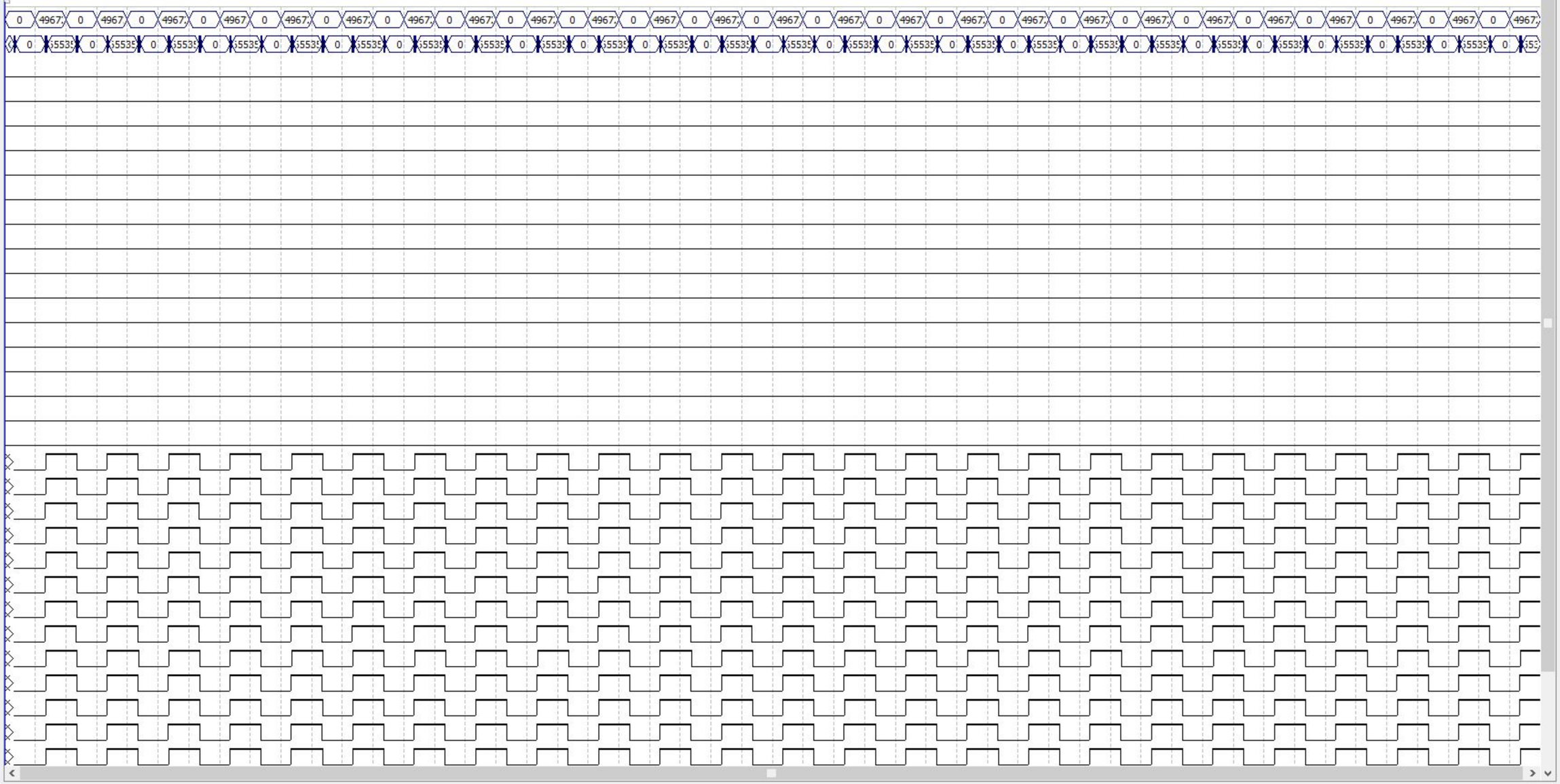
Search altera.com

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

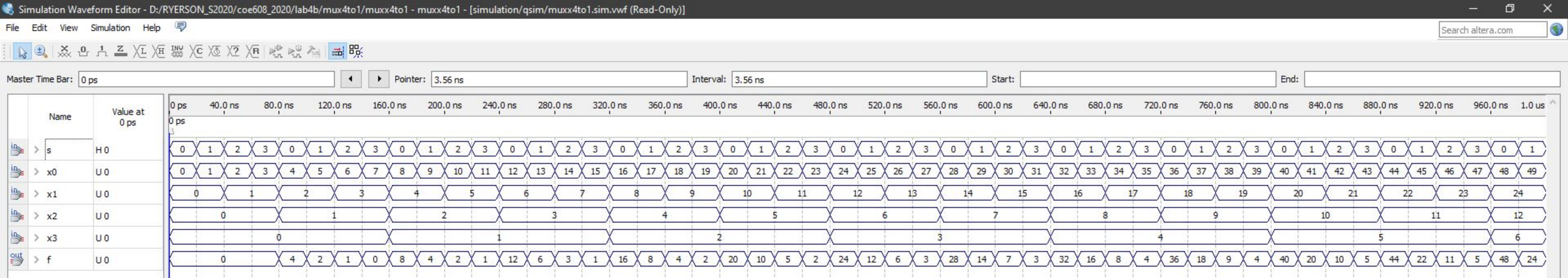
Name	Value at 0 ps
lze_in	U 0
lze_out	U X
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U 0
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...	U X
lze_...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

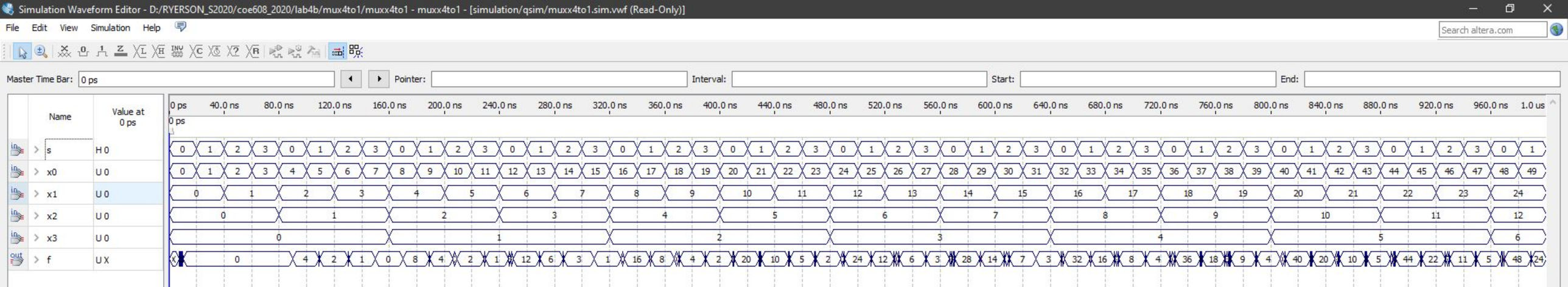
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us

0 ps



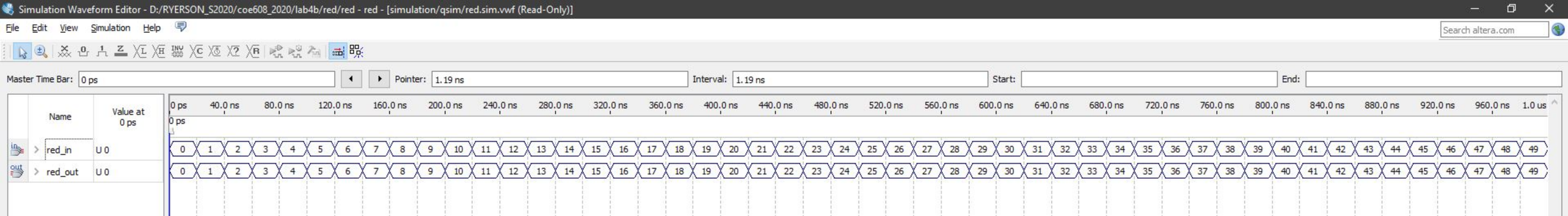
0% 00:00:00

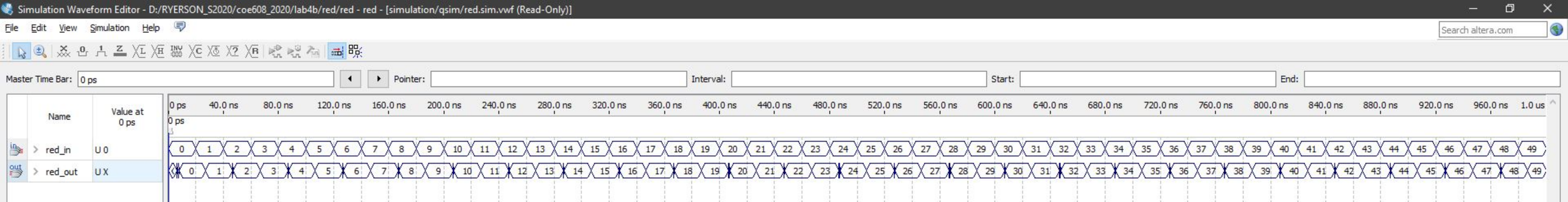

Simulation Waveform Editor - D:/RYERSON_S2020/coe608_2020/lab4b/mux4to1/muxx4to1 - muxx4to1 - [simulation/qsim/muxx4to1.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 3.56 ns
Interval: 3.56 ns
Start:
End:
Name
Value at 0 ps
s H 0
x0 U 0
x1 U 0
x2 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
```

Master Time Bar: 0 ps

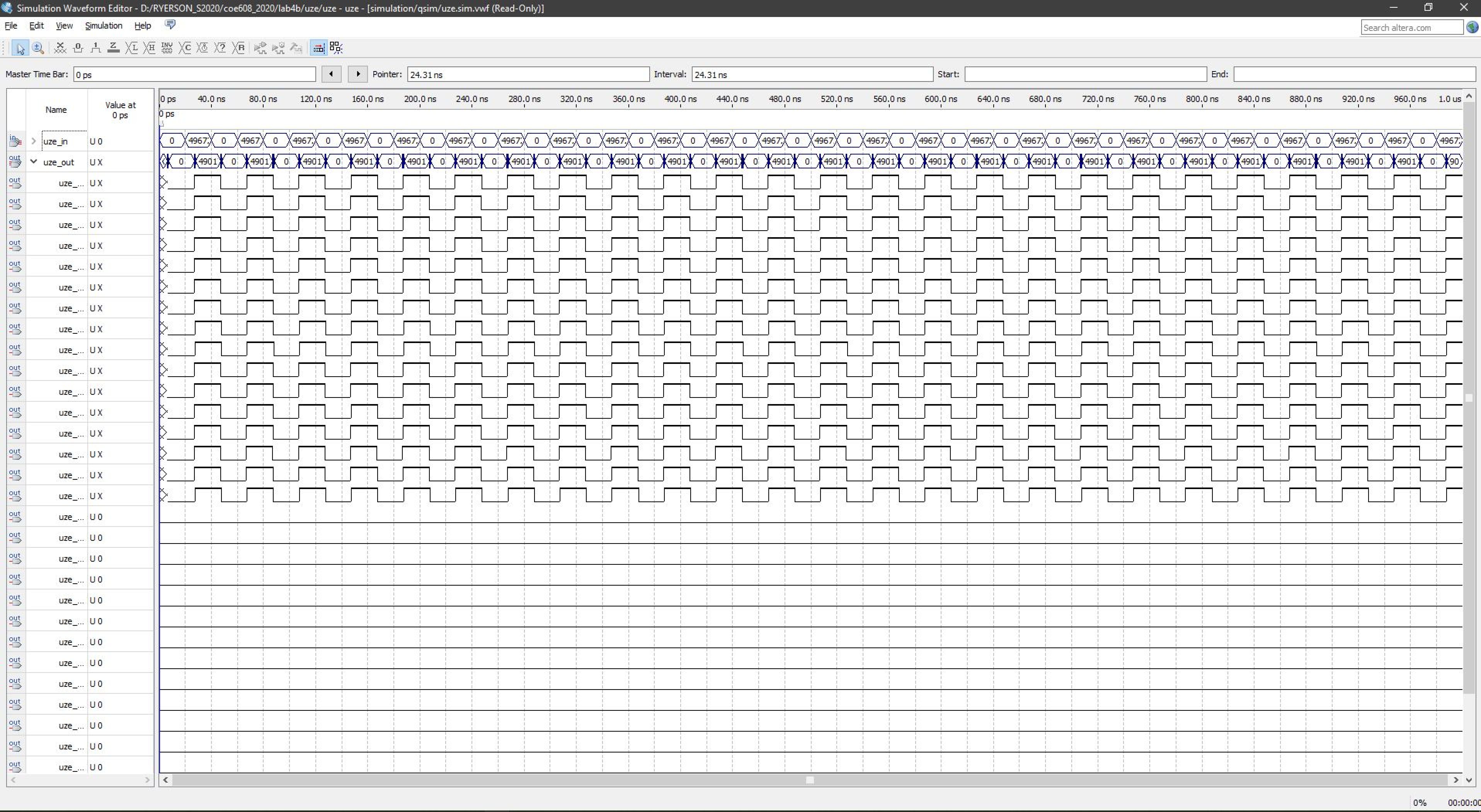
Pointer: 38.55 ns

Interval: 38.55 ns

Start:

End:

Name	Value at 0 ps
uze_in	U 0
uze_out	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0
uze_...	U 0


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
uze_... U X
uze_... U 0
0
4967
4901
0%
00:00:00