COEN 313 Lab 4 (UR-X)

by

Mamadou Kaba (27070179)

Performed on

March 21st, 2023

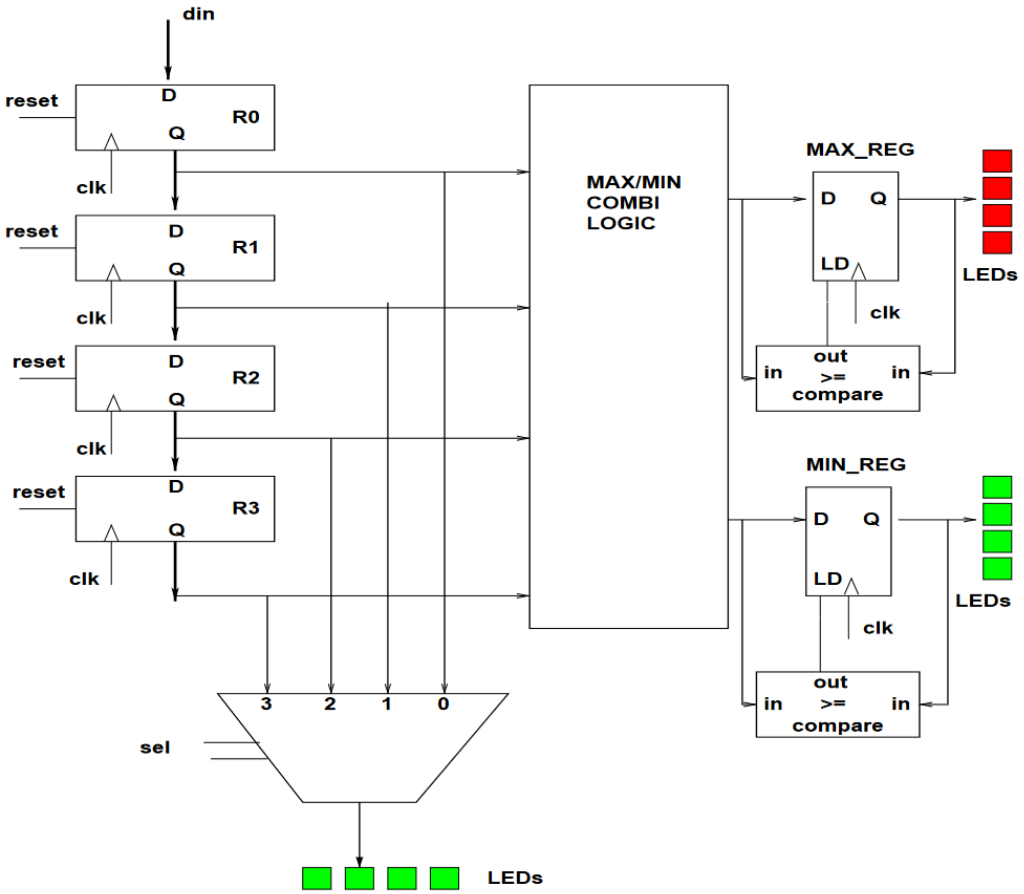
"I certify that this submission is my original work and meets the Faculty's Expectations of Originality"

Objective

The main objective of the fourth lab is to become acquainted with clocked processes and registers in VHDL.

Introduction

A shift register file together with logic to determine the maximum and minimum values contained within the 4 registers comprising the shift register will be designed using clocked processes in VHDL. Two output registers will be used to store the maximum and minimum values which have been entered into the shift register file via the data input port.



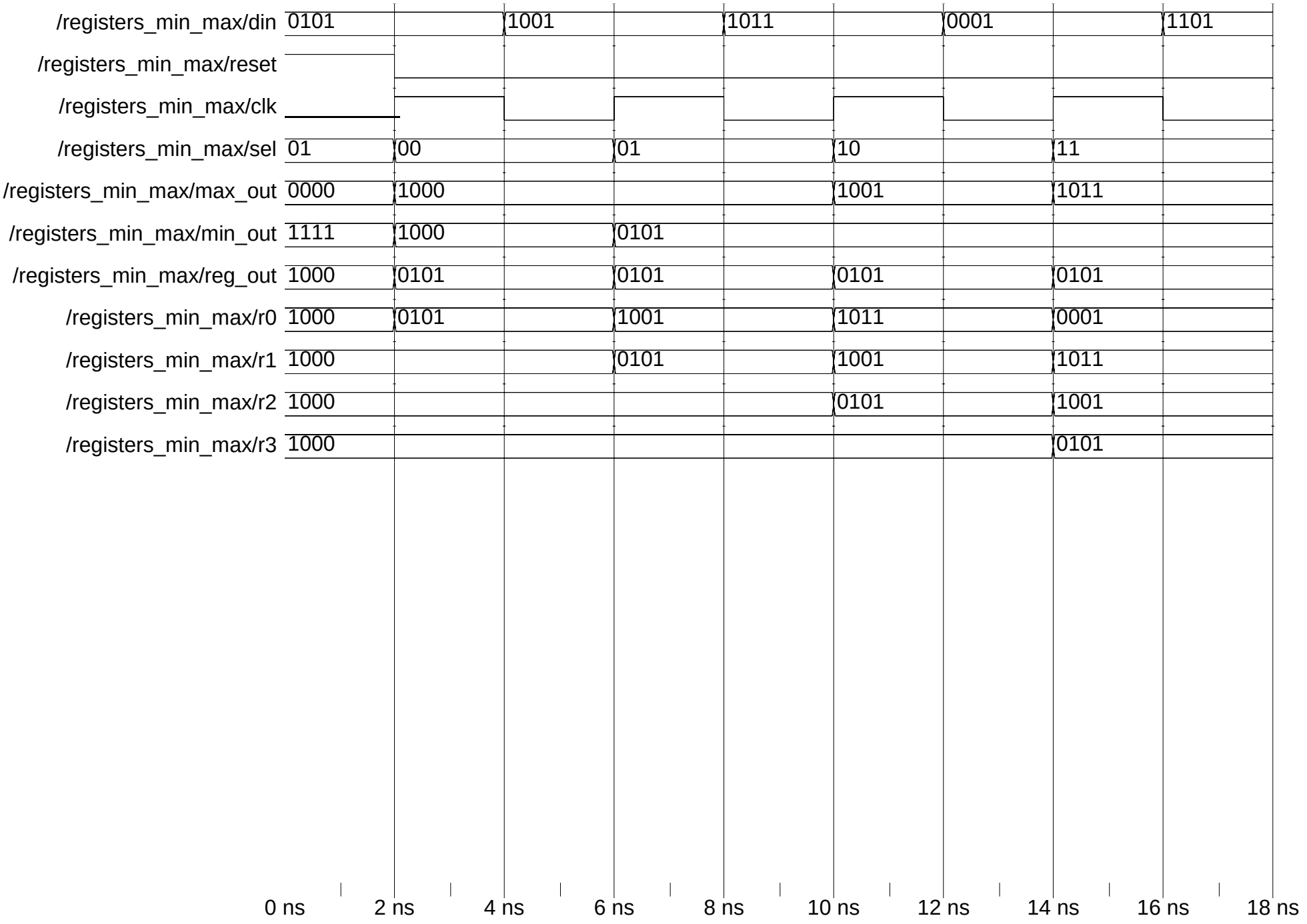
Results

Registers Min Max VHDL

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
entity registers_min_max is
port( din: in std_logic_vector(3 downto 0);
reset : in std_logic;
clk: in std_logic;
sel: in std_logic_vector(1 downto 0);
max_out : out std_logic_vector(3 downto 0);
min_out : out std_logic_vector(3 downto 0);
reg_out : out std_logic_vector(3 downto 0));
end registers_min_max ;

architecture lab4 of registers_min_max is

signal R0, R1, R2, R3: std_logic_vector(3 downto 0);
signal upr0, upr1, upr2, upr3: std_logic_vector(3 downto 0);
signal mintmp, maxtmp, MAX_REG, MIN_REG, maxup, minup: std_logic_vector(3 downto 0);

begin
process(clk, reset)
begin
if (reset='1')then
   R0<="1000";
   R1<="1000";
   R2<="1000";
   R3<="1000";
elsif(clk'event and clk='1') then
```

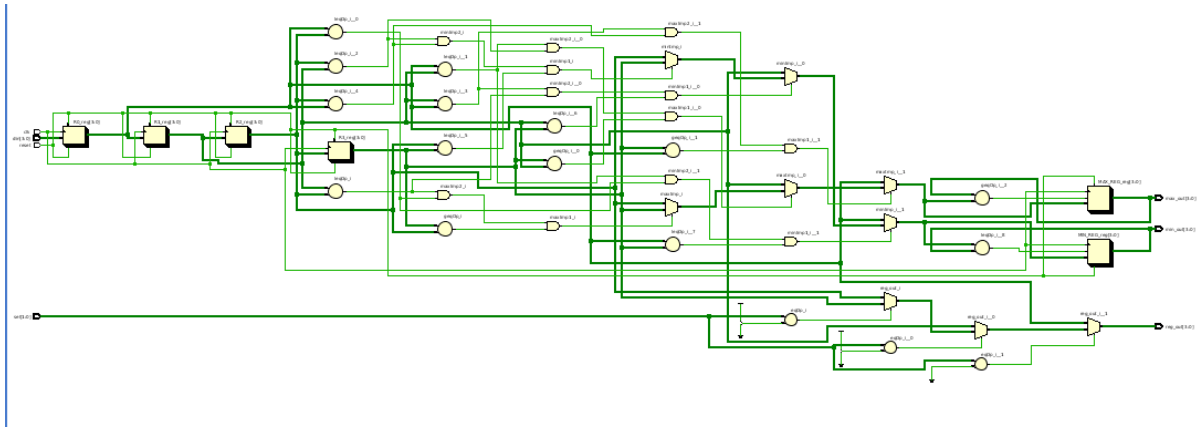
```
        R0 <= upr0;
        R1 <= upr1;
        R2 <= upr2;
        R3 <= upr3;
end if;
end process;
upr0 <= din;
upr1 <= R0;
upr2 <= R1;
upr3 <= R2;
reg_out <=R0 when sel="00" else
        R1 when sel="01" else
        R2 when sel="10" else
        R3 ;

process (clk, reset)
begin
if (reset='1') then
        max_reg <= "0000";
        min_reg <= "1111";

elsif (clk'event and clk='1') then
        MAX_REG <= maxup;
        MIN_REG <= minup;

end if;
end process;


process (R0,R1,R2,R3)
begin
```

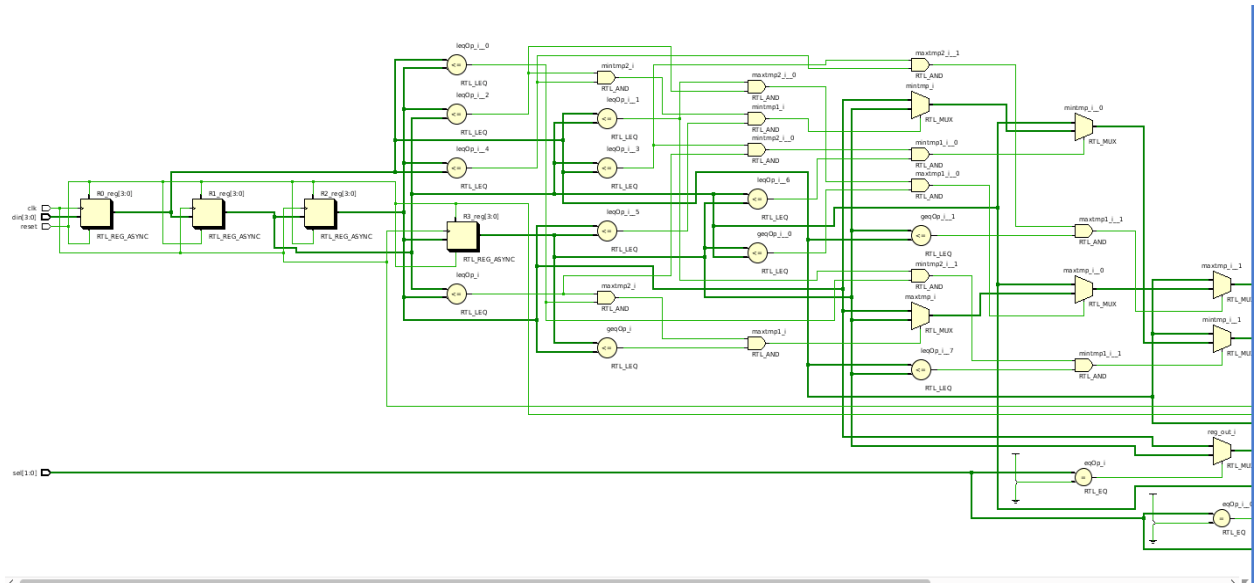
```
if ((R0<=R1) and (R0<=R2) and (R0<=R3))  then
mintmp<=R0;
elsif ((R1<=R0) and (R1<=R2) and (R1<=R3))  then
mintmp<=R1;
elsif ((R2<=R1) and (R2<=R0) and (R2<=R3))  then
mintmp<=R2;
else mintmp<=R3;
end if;
if ((R0>=R1) and (R0>=R2) and (R0>=R3))  then
maxtmp<=R0;
elsif ((R1>=R0) and (R1>=R2) and (R1>=R3))  then
maxtmp<=R1;
elsif ((R2>=R1) and (R2>=R0) and (R2>=R3))  then
maxtmp<=R2;
else maxtmp<=R3;
end if;
end process;
minup <= mintmp when (mintmp <= MIN_REG) else
                MIN_REG;
maxup <= maxtmp when (maxtmp >= MAX_REG) else
                MAX_REG;
max_out <= MAX_REG;
min_out <= MIN_REG;


end lab4;
```

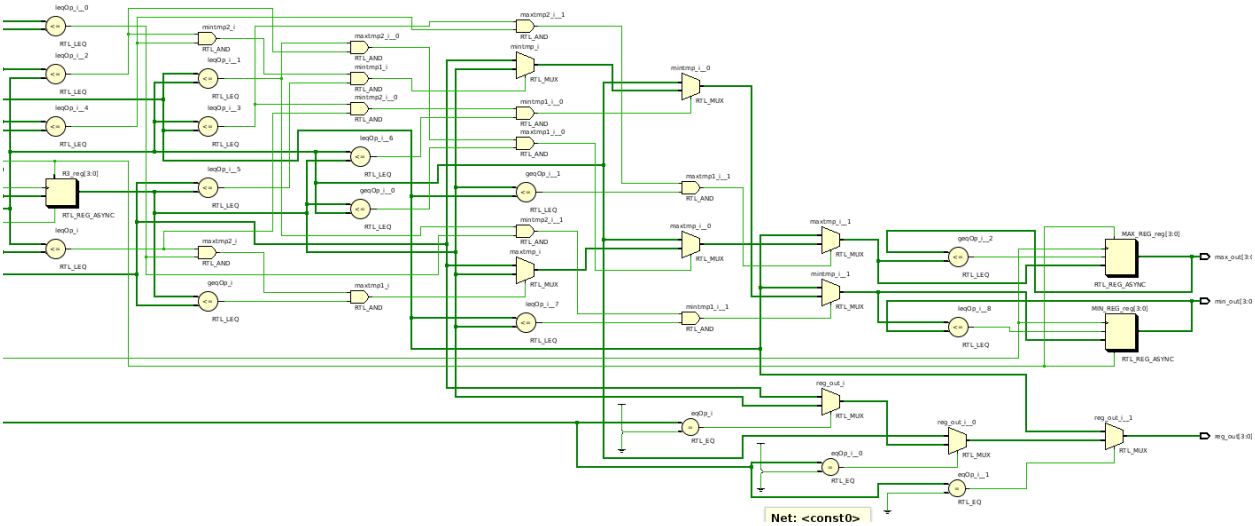


Entity:registers_min_max Architecture:lab4 Date: Tue Mar 21 08:12:44 PM EDT 2023 Row: 1 Page: 1

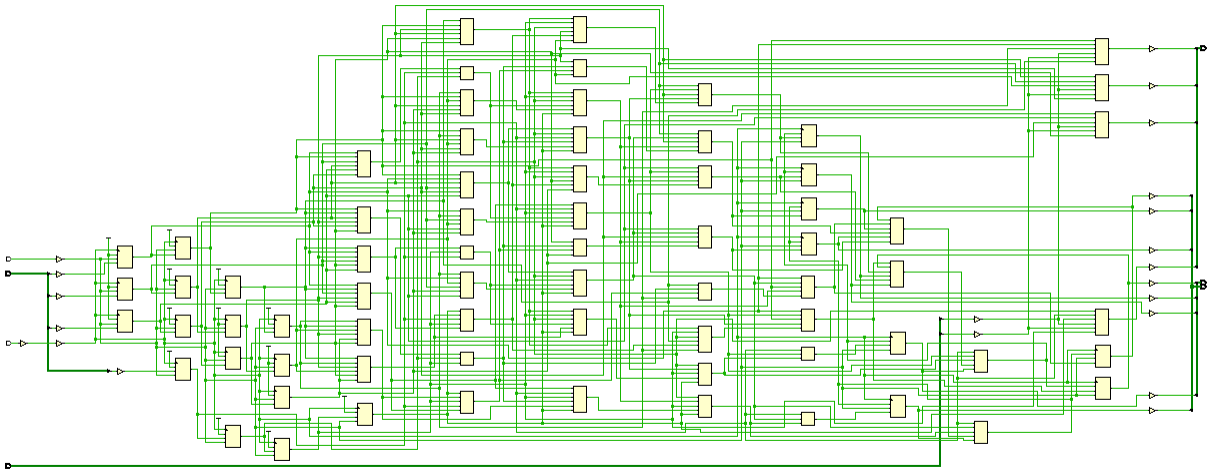
Elaborated Schematic

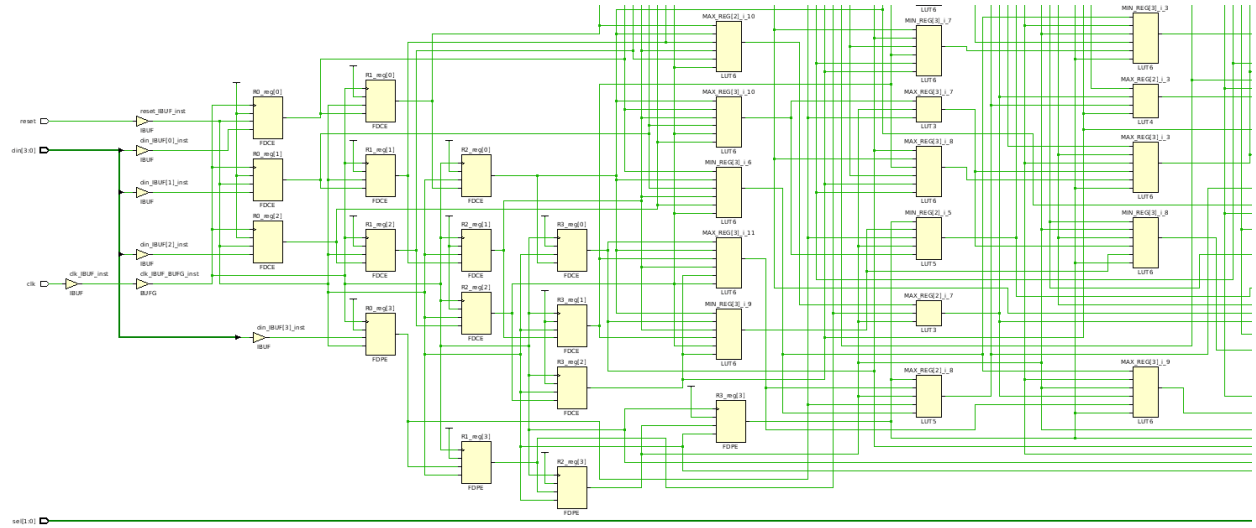


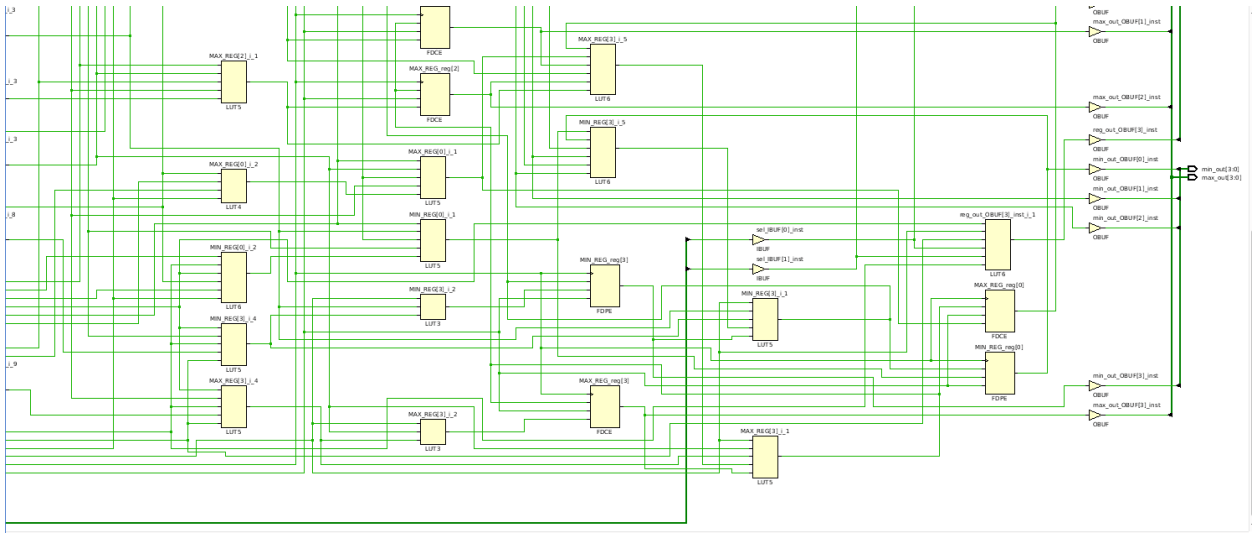


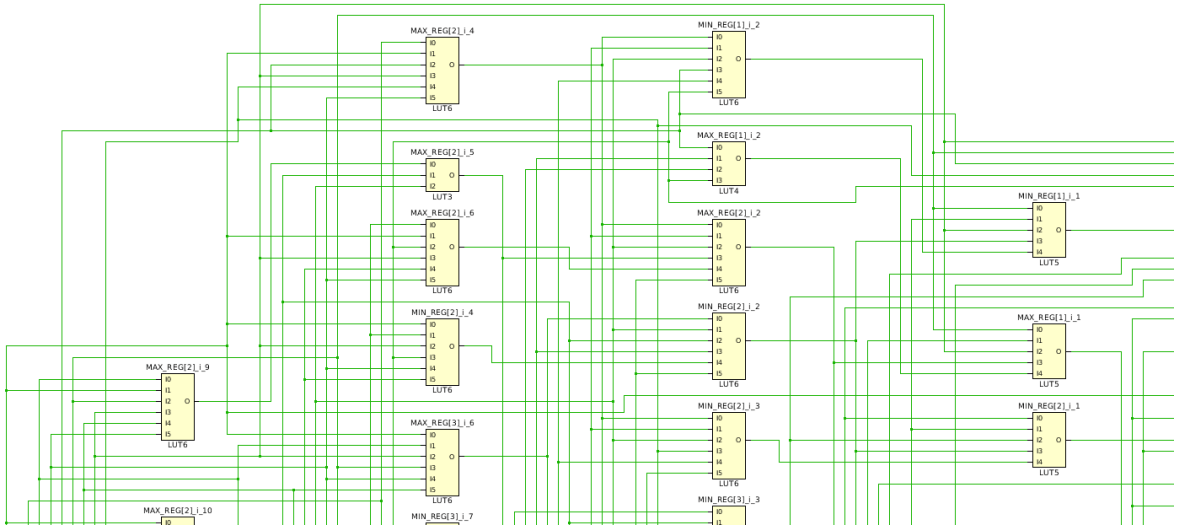


Implemented Design




reset
din[3:0]
clk
sel[1:0]
reset_IBUF_inst
din_IBUF[0]_inst
din_IBUF[1]_inst
din_IBUF[2]_inst
din_IBUF[3]_inst
clk_IBUF_inst
clk_IBUF_BUFG_inst
R0_reg[0]
R0_reg[1]
R0_reg[2]
R0_reg[3]
R1_reg[0]
R1_reg[1]
R1_reg[2]
R1_reg[3]
R2_reg[0]
R2_reg[1]
R2_reg[2]
R2_reg[3]
R3_reg[0]
R3_reg[1]
R3_reg[2]
R3_reg[3]
MAX_REG[2]_i_10
MAX_REG[3]_i_10
MIN_REG[3]_i_6
MAX_REG[3]_i_11
MIN_REG[3]_i_9
MIN_REG[3]_i_3
MAX_REG[2]_i_3
MAX_REG[3]_i_3
MIN_REG[3]_i_8
MIN_REG[3]_i_7
MAX_REG[3]_i_7
MAX_REG[3]_i_8
MIN_REG[2]_i_5
MAX_REG[2]_i_7
MAX_REG[2]_i_8
MAX_REG[3]_i_9


MAX_REG[2]_i_1
MAX_REG[0]_i_2
MIN_REG[0]_i_2
MIN_REG[3]_i_4
MAX_REG[3]_i_4
MAX_REG[3]_i_5
MAX_REG_reg[2]
MIN_REG[3]_i_5
MAX_REG[0]_i_1
MIN_REG[0]_i_1
MIN_REG_reg[3]
MIN_REG[3]_i_2
MAX_REG[3]_i_2
sel_IBUF[0]_inst
sel_IBUF[1]_inst
MIN_REG[3]_i_1
MAX_REG_reg[0]
MIN_REG_reg[0]
MAX_REG[3]_i_1
max_out_OBUF[1]_inst
max_out_OBUF[2]_inst
reg_out_OBUF[3]_inst
min_out_OBUF[0]_inst
min_out_OBUF[1]_inst
min_out_OBUF[2]_inst
min_out_OBUF[3]_inst
max_out_OBUF[3]_inst
min_out[3:0]
max_ou[3:0]


MAX_REG[2]_i_4
MAX_REG[2]_i_5
MAX_REG[2]_i_6
MIN_REG[2]_i_4
MAX_REG[2]_i_9
MAX_REG[3]_i_6
MAX_REG[2]_i_10
MIN_REG[3]_i_7
MIN_REG[1]_i_2
MAX_REG[1]_i_2
MAX_REG[2]_i_2
MIN_REG[2]_i_2
MIN_REG[2]_i_3
MIN_REG[3]_i_3
MIN_REG[1]_i_1
MAX_REG[1]_i_1
MIN_REG[2]_i_1

Simulation

The results attained from the Modelsim simulation below match the anticipated results based on my chosen numbers.

Question

1. The clocked process should be sensitive to the clock and to reset.

Conclusion

By the end of this lab experiment I got a better understanding on the implementation of VHDL and the use clocks and resets in latches, this lab has been tougher and took so much more time to complete than the previous three.

Appendix

Registers Min Max XDC

```
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [ get_ports { din[0] } ];
set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [ get_ports { din[1] } ];
set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [ get_ports { din[2] } ];
set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [ get_ports { din[3] } ];

set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { sel[0] }];
set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { sel[1] }];

set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { reset }];
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk];
set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { clk }];

set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { max_out[0] }];
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { max_out[1] }];
set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { max_out[2] }];
set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { max_out[3] }];

set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { min_out[0] }];
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { min_out[1] }];
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { min_out[2] }];
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { min_out[3] }];

set_property -dict { PACKAGE_PIN V16 IOSTANDARD LVCMOS33 } [get_ports { reg_out[0] }];
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { reg_out[1] }];
set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { reg_out[2] }];
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { reg_out[3] }];
```

Registers Min Max DO file

```
add wave din
add wave reset
add wave clk
add wave sel
add wave max_out
add wave min_out
add wave reg_out
add wave R0
add wave R1
add wave R2
add wave R3

force din 0101
force reset 1
force clk 0
force sel 01
run 2

force reset 0
force clk 1
force sel 00
run 2

force din 1001
force clk 0
run 2

force clk 1
force sel 01
run 2
```

```
force din 1011
force clk 0
run 2

force clk 1
force sel 10
run 2

force din 0001
force clk 0
run 2

force clk 1
force sel 11
run 2

force din 1101
force clk 0
run 2

force clk 1
force sel 10
run 2
```