David Elliott

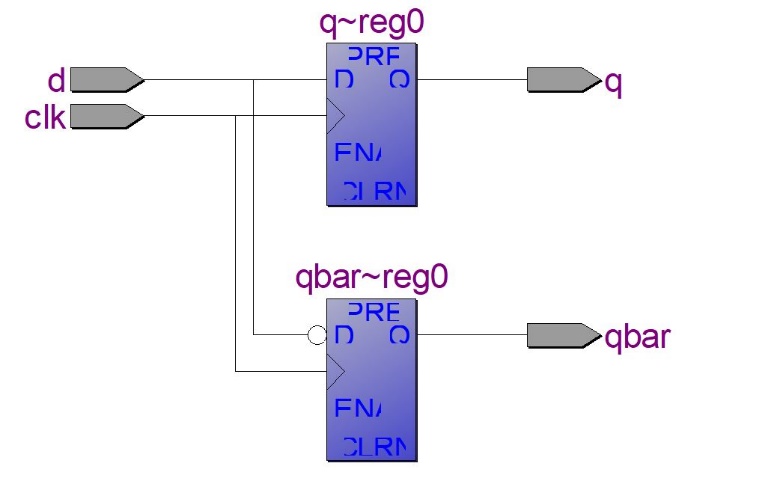
Evan Yeh

EE125 Homework #3

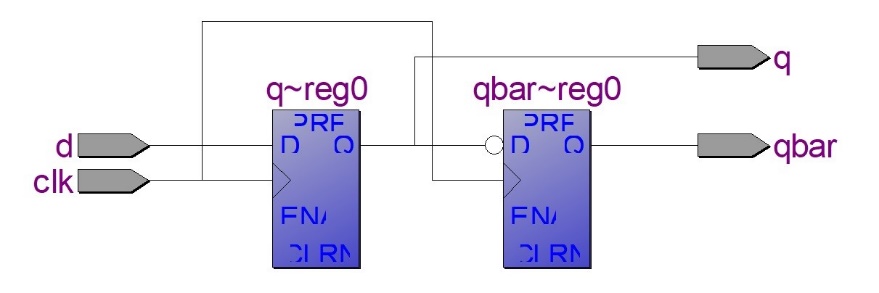
Exercise 1:

b.) arch1, arch2, arch3 are compiled and RTL views are shown below:

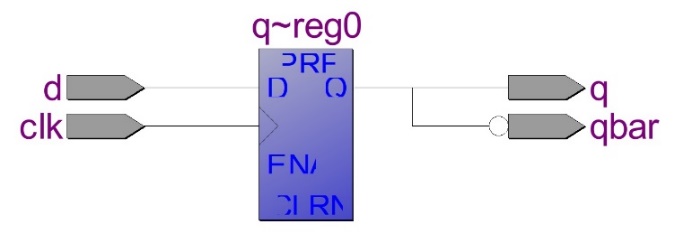
**arch1:**

****

**arch2:**

****

**arch3:**

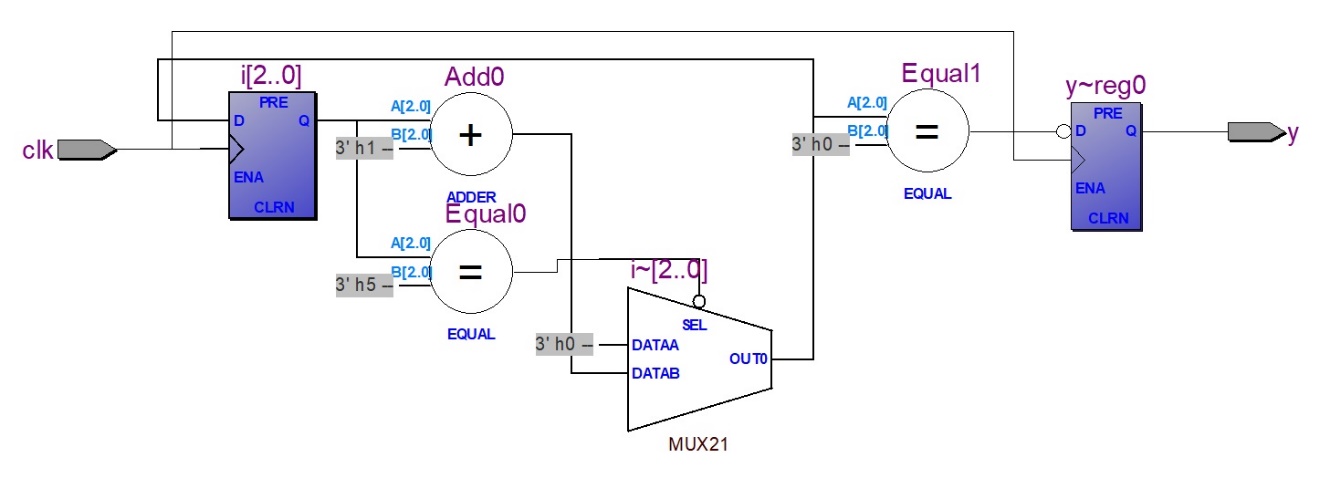


All circuits match with our predicted ones.

Exercise 2:

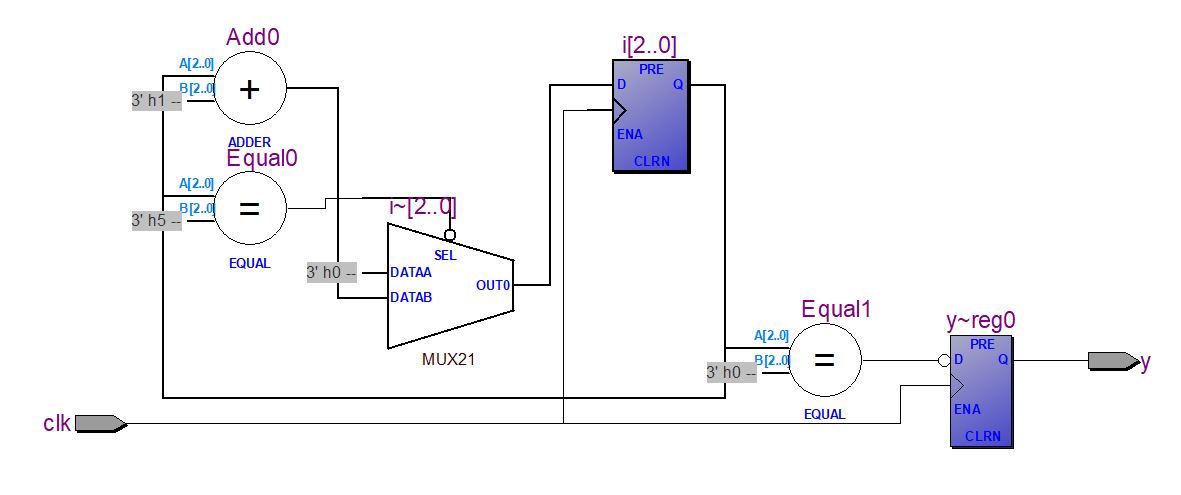
d.) RTL View:

Code1:



There are a total of 4 flip-flops in the generated view, same as our prediction.

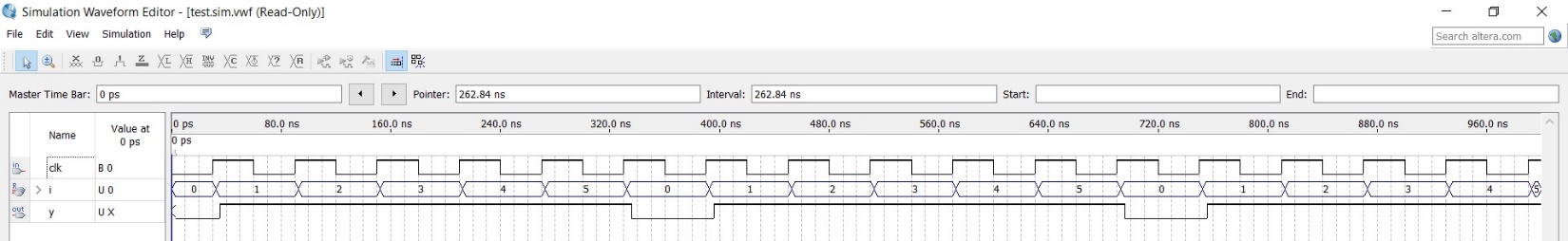
Code2:



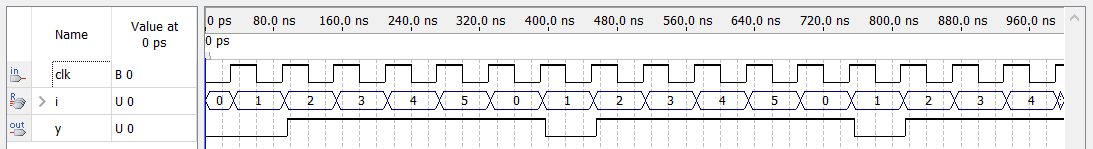
There are still 4 flip-flops in the generated view, same as our prediction.

e.) Simulation Results

Code1:



Code2:



Both waveforms align with our predictions.

Exercise 3:

01 -------------------------------------------------------------------------

02 **library** ieee**;**

03 **use** ieee**.**std\_logic\_1164**.all;**

04 **use** ieee**.**numeric\_std**.all;**

05 -------------------------------------------------------------------------

06 **entity** leading\_ones\_sequential **is**

07 **generic** **(**N**:** integer **:=** 8 **);** --number of input bits

08 **port** **(**x**:** **in** std\_logic\_vector**(**N**-**1 **downto** 0**);**

09 clk**:** **in** std\_logic**;**

10 ssd**:** **out** std\_logic\_vector**(**6 **downto** 0**);**

11 cnt**:** **buffer** integer**);**

12 **end** **entity;**

13 -------------------------------------------------------------------------

14 **architecture** leading\_ones\_sequential **of** leading\_ones\_sequential **is**

15 **begin**

16 **process(**x**,**clk**)**

17 **variable** count**:** integer **:=** 0**;**

18 **begin**

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

20 count **:=** 0**;**

21 **for** i **in** N**-**1 **downto** 0 **loop**

22 **if** **(**x**(**i**)** **=** '1'**)** **then**

23 count **:=** count **+** 1**;**

24 **else**

25 **exit;**

26 **end** **if;**

27 **end** **loop;**

28

29 cnt **<=** count**;**

30

31 --encode into ssd bits

32

33 **case** count **is**

34 **when** 0 **=>** ssd **<=** "0000001"**;**

35 **when** 1 **=>** ssd **<=** "1001111"**;**

36 **when** 2 **=>** ssd **<=** "0010010"**;**

37 **when** 3 **=>** ssd **<=** "0000110"**;**

38 **when** 4 **=>** ssd **<=** "1001100"**;**

39 **when** 5 **=>** ssd **<=** "0100100"**;**

40 **when** 6 **=>** ssd **<=** "0100000"**;**

41 **when** 7 **=>** ssd **<=** "0000001"**;**

42 **when** 8 **=>** ssd **<=** "0000000"**;**

43 **when** 9 **=>** ssd **<=** "0000100"**;**

44 **when** **others** **=>** ssd **<=** "0110000"**;**

45 **end** **case;**

46 **end** **if;**

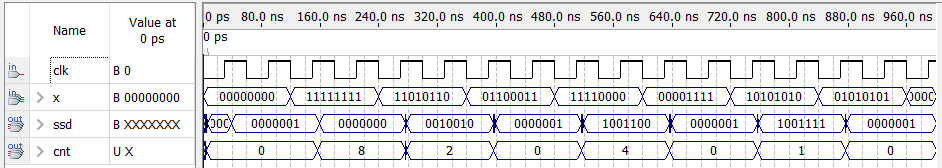
47 **end** **process;**

48 **end** **architecture;**

49 -------------------------------------------------------------------------

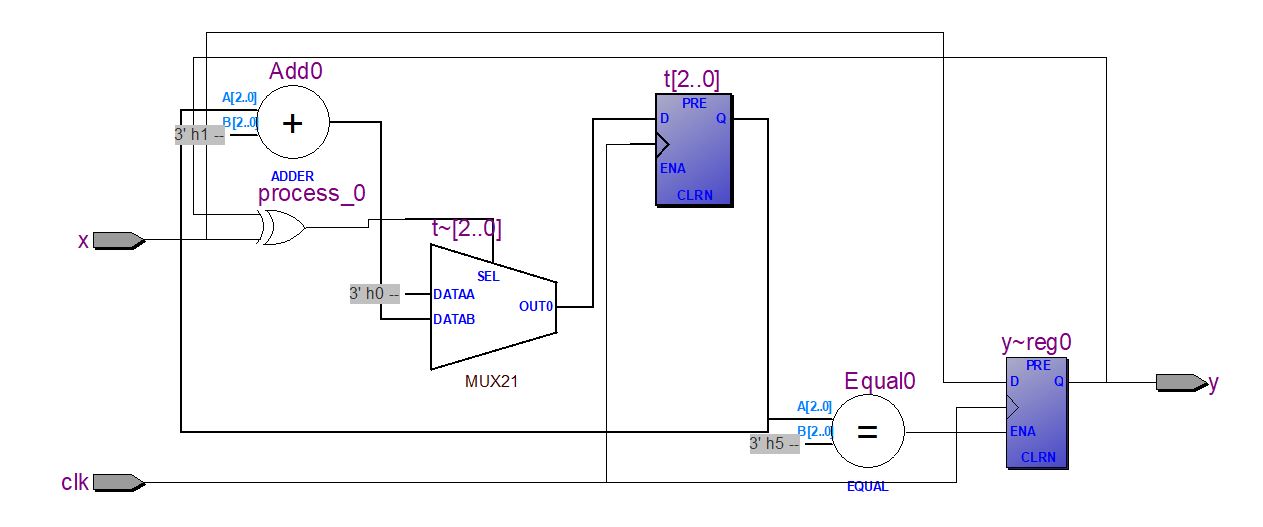
b.) Sequential code is easier to write. It is more intuitive and requires fewer signals.

c.) Using the same test cases as last week:



Exercise 4

1. For simulation, we counted 5 clock cycles before debouncing, so our debounce counter was 3 bits -> 3 flip flops. Then there is one more flip flop for if the button is debounced. (4 total flip flops) For synthesis, we want to count 1000000 clocks, so 20 bits would be needed for the counter and then one more for if the button is debounced (21 total flip flops).



The above RTL view agrees with our prediction of 4 flip flops (for simulation).

Code:

debouncer\_demo.vhd

1 -------------------------------------------------------------------------

2 **library** ieee**;**

3 **use** ieee**.**std\_logic\_1164**.all;**

4 -------------------------------------------------------------------------

5 **entity** debouncer\_demo **is**

6 **generic** **(**t\_debounce**:** integer **:=** 1000000**);**

7 **PORT** **(**x**:** **in** std\_logic**;**

8 clk**:** **in** std\_logic**;**

9 rst**:** **in** std\_logic**;**

10 ssd\_debounced\_pins**:** **OUT** STD\_LOGIC\_VECTOR**(**6 **DOWNTO** 0**);**

11 ssd\_bounced\_pins**:** **OUT** STD\_LOGIC\_VECTOR**(**6 **DOWNTO** 0**));**

12 **END** **ENTITY;**

13 -------------------------------------------------------------------------

14 **architecture** debouncer\_demo **of** debouncer\_demo **is**

15 **signal** y**:** std\_logic**;**

16 **signal** num\_debounced**:** integer **range** 0 **to** integer'**high;**

17 **signal** num\_bounced**:** integer **range** 0 **to** integer'**high;**

18 **begin**

19 debouncer**:** **entity** work**.**switch\_debouncer**(**switch\_debouncer**)** **port** **map(**x**,** y**,** clk**);**

20 ssd\_debounced **:** **entity** work**.**ssd**(**ssd**)** **port** **map(**num\_debounced**,** ssd\_debounced\_pins**);**

21 ssd\_bounced **:** **entity** work**.**ssd**(**ssd**)** **port** **map(**num\_bounced**,** ssd\_bounced\_pins**);**

22 **process(**x**,** y**,** rst**)**

23 **begin**

24 **if** rst **=** '0' **then**

25 num\_debounced **<=** 0**;**

26 num\_bounced **<=** 0**;**

27 **else**

28 **if** **falling\_edge(**x**)** **then**

29 num\_bounced **<=** num\_bounced **+** 1**;**

30 **if** num\_bounced **=** 9 **then**

31 num\_bounced **<=** 0**;**

32 **end** **if;**

33 **end** **if;**

34 **if** **falling\_edge(**y**)** **then**

35 num\_debounced **<=** num\_debounced **+** 1**;**

36 **if** num\_debounced **=** 9 **then**

37 num\_debounced **<=** 0**;**

38 **end** **if;**

39 **end** **if;**

40 **end** **if;**

41 **end** **process;**

42 **end** **architecture;**

43 -------------------------------------------------------------------------

switch\_debouncer.vhd

1 -------------------------------------------------------------------------

2 **library** ieee**;**

3 **use** ieee**.**std\_logic\_1164**.all;**

4 -------------------------------------------------------------------------

5 **entity** switch\_debouncer **is**

6 **generic** **(**t\_debounce**:** integer **:=** 1000000**);**

7 **PORT** **(**x**:** **in** std\_logic**;**

8 y**:** **buffer** std\_logic**;**

9 clk**:** **in** std\_logic**);**

10 **END** **ENTITY;**

11 -------------------------------------------------------------------------

12 **architecture** switch\_debouncer **of** switch\_debouncer **is**

13 **begin**

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

15 **variable** t**:** integer **range** 0 **to** t\_debounce**;**

16 **begin**

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

18 **if** t **=** t\_debounce **then**

19 y **<=** x**;**

20 **end** **if;**

21 **if** x **/=** y **then**

22 t **:=** t **+** 1**;**

23 **else**

24 t **:=** 0**;**

25 **end** **if;**

26 **end** **if;**

27 **end** **process;**

28 **end** **architecture;**

29 -------------------------------------------------------------------------

ssd.vhd

1 -------------------------------------------------------------------------

2 **LIBRARY** ieee**;**

3 **USE** ieee**.**std\_logic\_1164**.all;**

4 -------------------------------------------------------------------------

5 **ENTITY** ssd **IS**

6 **PORT** **(**num**:** **in** integer**;**

7 ssd**:** **OUT** STD\_LOGIC\_VECTOR**(**6 **DOWNTO** 0**));**

8 **END** **ENTITY;**

9 -------------------------------------------------------------------------

10 **ARCHITECTURE** ssd **OF** ssd **IS**

11 **BEGIN**

12 **with** num **select**

13 ssd **<=** "0000001" **when** 0**,**

14 "1001111" **when** 1**,**

15 "0010010" **when** 2**,**

16 "0000110" **when** 3**,**

17 "1001100" **when** 4**,**

18 "0100100" **when** 5**,**

19 "0100000" **when** 6**,**

20 "0001111" **when** 7**,**

21 "0000000" **when** 8**,**

22 "0000100" **when** 9**,**

23 "0110000" **when** **others;**

24 **END** **ARCHITECTURE;**

25 -------------------------------------------------------------------------