David Elliott

Evan Yeh

EE125 Homework #4

Exercise 1:

1. Code:

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

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

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

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

05 **use** ieee**.**math\_real**.all;**

06 -------------------------------------------------------------------------

07 **entity** log2\_prob **is**

08 **generic** **(**BITS**:** natural **:=** 8**);**

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

10 -- generically:

11 -- y: out std\_logic\_vector(ceil(log2(BITS)) downto 0);

-- not synthesizeable

12 -- for simulation:

13 y**:** **out** std\_logic\_vector**(**3 **downto** 0**));**

14 **end** **entity;**

15 -------------------------------------------------------------------------

16 **architecture** problem\_one **of** log2\_prob **is**

17 **function** ceil\_log2 **(**input**:** std\_logic\_vector**(**BITS **-** 1 **downto** 0**))**

18 **return** integer **is**

19 **variable** i**:** integer **:=** 0**;**

20 **variable** inp\_int**:** integer**:=** 0**;**

21 **begin**

22 inp\_int **:=** **to\_integer(**unsigned**(**x**));**

23 **while** **(**2 **\*\*** i **<** inp\_int and i **<=** BITS**)** **loop**

24 i **:=** i **+** 1**;**

25 **end** **loop;**

26 **return** i**;**

27 **end** **function** ceil\_log2**;**

28

29 **signal** x\_int**:** integer**;**

30 **begin**

31 -- generically:

32 -- y <= std\_logic\_vector(to\_unsigned(ceil\_log2(x), ceil\_log2(BITS));

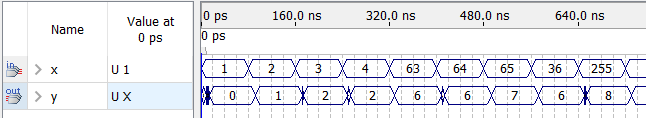
33 -- for simulation:

34 y **<=** std\_logic\_vector**(to\_unsigned(**ceil\_log2**(**x**),** 4**));**

35 **end** **architecture;**

36 -------------------------------------------------------------------------

1. Simulation:



Exercise 2:

1. Code:

natural\_to\_thermometer.vhd

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

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

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

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

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

06 **package** natural\_to\_thermometer **is**

07 **constant** IN\_BITS**:** integer **:=** 7**;**

08 **function** natural\_to\_thermometer**(signal** num**:** natural**)** **return** std\_logic\_vector**;**

09 **end** natural\_to\_thermometer**;**

10 -------------------------------------------------------------------------

11 **package** **body** natural\_to\_thermometer **is**

12 **function** natural\_to\_thermometer**(signal** num**:** natural**)** **return** std\_logic\_vector **is**

13 **constant** OUT\_BITS**:** integer **:=** 2**\*\***IN\_BITS**;**

14 **variable** thermometer**:** std\_logic\_vector**(**OUT\_BITS **-** 1 **downto** 0**);**

15 **begin**

16 **for** i **in** 0 **to** OUT\_BITS **-** 1 **loop**

17 **if** i **<** num **then**

18 thermometer**(**i**)** **:=** '1'**;**

19 **else**

20 thermometer**(**i**)** **:=** '0'**;**

21 **end** **if;**

22 **end** **loop;**

23 **return** thermometer**;**

24 **end** **function;**

25 **end** **package** **body;**

26 -------------------------------------------------------------------------

natural\_to\_thermometer\_test.vhd

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

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

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

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

05 **library** work**;**

06 **use** work**.**natural\_to\_thermometer**.all;**

07 -------------------------------------------------------------------------

08 **entity** natural\_to\_thermometer\_test **is**

09 **generic(**IN\_BITS**:** integer **:=** 7**);**

10 **PORT** **(**num**:** **in** std\_logic\_vector**(**IN\_BITS**-**1 **downto** 0**);**

11 thermometer**:** **out** std\_logic\_vector**(**2**\*\***IN\_BITS **-** 1 **downto** 0**));**

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

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

14 **architecture** test **of** natural\_to\_thermometer\_test **is**

15 **begin**

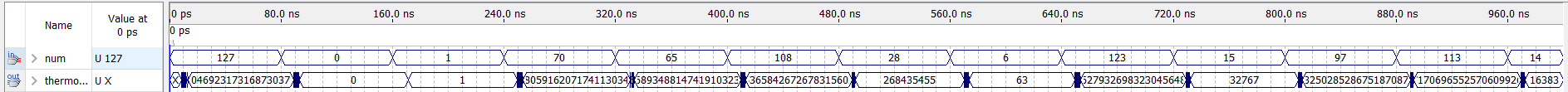
16 thermometer **<=** natural\_to\_thermometer**(to\_integer(**unsigned**(**num**)));**

17 **end** **architecture;**

18 -------------------------------------------------------------------------

1. Simulation:

Binary values are too large to display in the waveform, displaying with unsigned decimals instead:



Exercise 3:

1. Code:

01 ---- tflipflop:----------------------------------------------------------

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

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

04 -------------------------------------------------------------------------

05 **entity** tflipflop **is**

06 **port** **(**a**,** b**,** clk**:** **in** std\_logic**;**

07 x**:** **out** std\_logic**;**

08 q**:** **buffer** std\_logic**);**

09 **end** **entity;**

10 -------------------------------------------------------------------------

11 **architecture** tflipflop **of** tflipflop **is**

12 **begin**

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

14 **begin**

15 x **<=** a and b**;**

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

17 q **<=** **(**a and b**)** xor q**;**

18 **end** **if;**

19 **end** **process;**

20 **end** **architecture;**

21 ----Main code:-----------------------------------------------------------

22 **library** ieee**;**

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

24 **entity** sync\_counter **is**

25 **generic** **(**BITS**:** natural **:=** 4**);**

26 **port** **(**a**,** b**,** clk**:** **in** std\_logic**;**

27 q\_vec**:** **buffer** std\_logic\_vector**(**BITS **-** 1 **downto** 0**));**

28 **end** **entity;**

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

30 **architecture** sync\_counter **of** sync\_counter **is**

31 **signal** x\_vec**:** std\_logic\_vector**(**BITS **-** 1 **downto** 0**);**

32

33 **component** tflipflop **is**

34 **port** **(**a**,** b**,** clk**:** **in** std\_logic**;**

35 x**:** **out** std\_logic**;**

36 q**:** **buffer** std\_logic**);**

37 **end** **component;**

38

39 **begin**

40 cell**:** tflipflop **PORT** **MAP** **(**a**,** b**,** clk**,** x\_vec**(**0**),** q\_vec**(**0**));**

41 gen**:** **for** i **in** 1 **to** BITS **-** 1 **generate**

42 cells**:** tflipflop **PORT** **MAP** **(**x\_vec**(**i**-**1**),** q\_vec**(**i**-**1**),** clk**,** x\_vec**(**i**),**

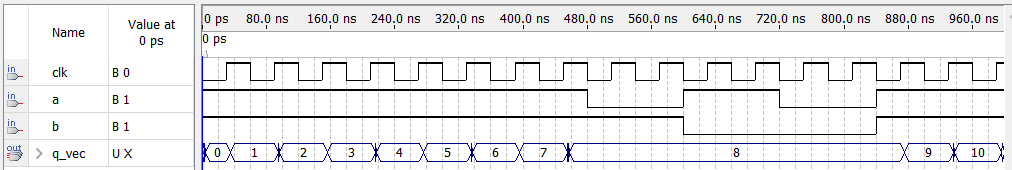
q\_vec**(**i**));**

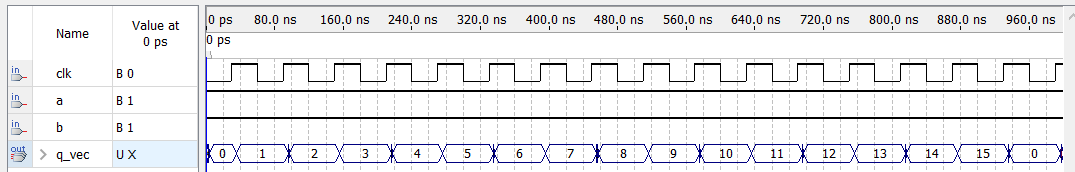
43 **end** **generate** gen**;**

44

45 **end** **architecture;**

1. Simulation:





Exercise 4

1. Code:

pwm\_demo.vhd

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

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

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

04 -------------------------------------------------------------------------

05 **entity** pwm\_demo **is**

06 **generic** **(**F\_CLK**:** integer**:=**50**;**

07 T\_CLK\_PWM**:** integer**:=**120**;**

08 BITS\_DUTY**:** integer**:=**3**);**

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

10 duty**:** **in** std\_logic\_vector**(**BITS\_DUTY**-**1 **downto** 0**);**

11 pwm\_out**:** **out** std\_logic**;**

12 count**:** **out** std\_logic\_vector**(**BITS\_DUTY**-**1 **downto** 0**));**

13 **END** **ENTITY;**

14 -------------------------------------------------------------------------

15 **architecture** pwm\_demo **of** pwm\_demo **is**

16 **signal** pwm\_clk**:** std\_logic**;**

17 **begin**

18 pwm**:** **entity** work**.**pwm**(**pwm**)** **generic** **map(**BITS\_DUTY**)**

19 **port** **map(**duty**,** pwm\_clk**,** pwm\_out**,** count**);**

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

21

22 **constant** CYCLES\_MAX**:** integer **:=** **(**T\_CLK\_PWM **\*** F\_CLK**)** **/** 2000 **-** 1**;**

23 **variable** cycles**:** integer **range** 0 **to** CYCLES\_MAX **:=** CYCLES\_MAX**;**

24 **begin**

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

26 **if** cycles **=** CYCLES\_MAX **then**

27 cycles **:=** 0**;**

28 **else**

29 cycles **:=** cycles **+** 1**;**

30 **end** **if;**

31 **if** cycles **=** 0 **then**

32 pwm\_clk **<=** NOT**(**pwm\_clk**);**

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

34 **end** **if;**

35 **end** **process;**

36 **end** **architecture;**

37 -------------------------------------------------------------------------

pwm.vhd

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

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

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

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

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

06 **entity** pwm **is**

07 **generic** **(**BITS\_DUTY**:** integer **:=** 3**);**

08 **PORT** **(**duty**:** **in** std\_logic\_vector**(**BITS\_DUTY**-**1 **downto** 0**);**

09 pwm\_clk**:** **in** std\_logic**;**

10 pwm**:** **out** std\_logic**;**

11 count\_vec**:** **out** std\_logic\_vector**(**BITS\_DUTY**-**1 **downto** 0**));**

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

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

14 **architecture** pwm **of** pwm **is**

15 **signal** duty\_int**:** integer **range** 0 **to** 2**\*\***BITS\_DUTY **-** 1**;**

16 **begin**

17 duty\_int **<=** **to\_integer(**unsigned**(**duty**));**

18 **process(**pwm\_clk**)**

19 **constant** COUNT\_MAX**:** integer **:=** 2**\*\***BITS\_DUTY **-** 2**;**

20 **variable** count**:** integer **range** 0 **to** COUNT\_MAX **:=** COUNT\_MAX**;**

21 **begin**

22 **if** **rising\_edge(**pwm\_clk**)** **then**

23 **if** count **=** COUNT\_MAX **then**

24 count **:=** 0**;**

25 **else**

26 count **:=** count **+** 1**;**

27 **end** **if;**

28 **if** count **=** 0 **then**

29 pwm **<=** '1'**;**

30 **end** **if;**

31 **if** count **=** duty\_int **then**

32 pwm **<=** '0'**;**

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

34 **end** **if;**

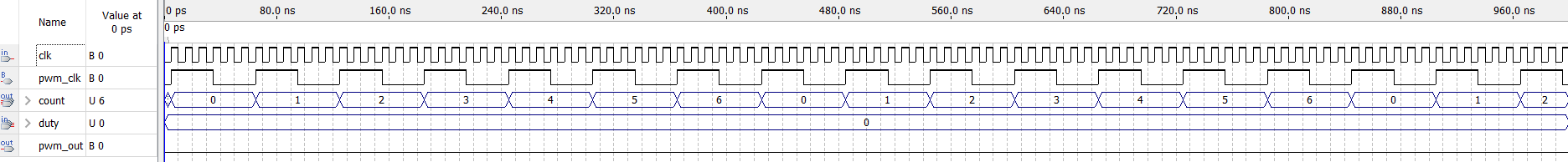
35 count\_vec **<=** std\_logic\_vector**(to\_unsigned(**count**,** BITS\_DUTY**));**

36 **end** **process;**

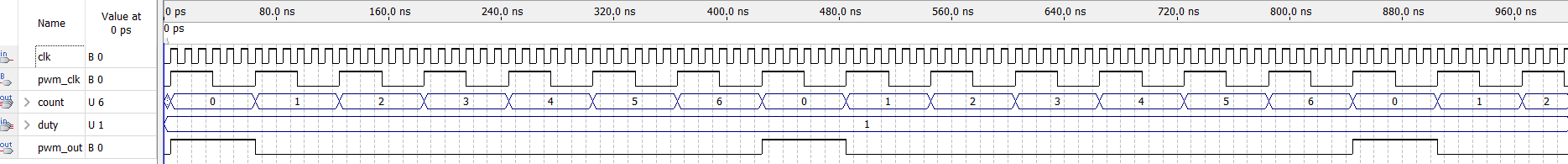
37 **end** **architecture;**

38 -------------------------------------------------------------------------

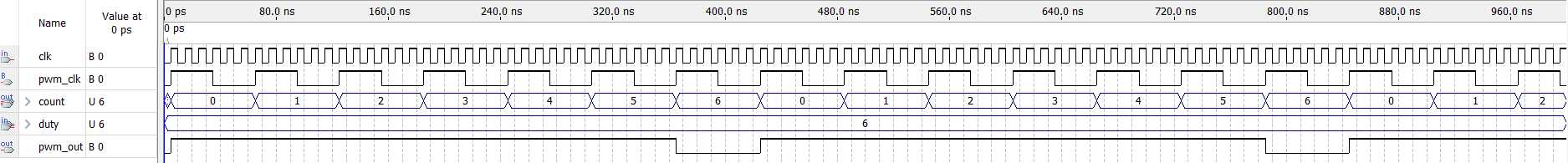
Duty = 0



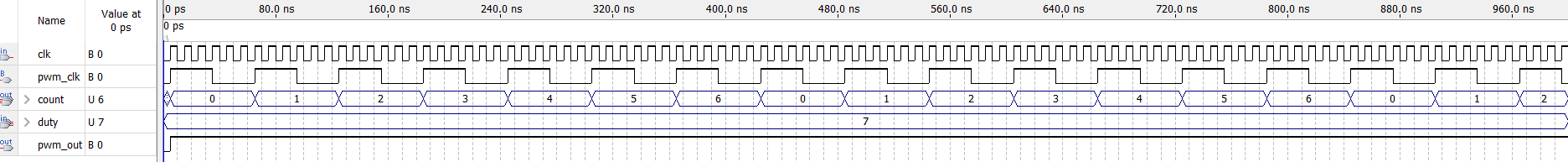
Duty = 1



Duty = 6



Duty = 7



1. Simulations are similar to the figure and the pwm pulse does start at zero.
2. To demonstrate the PWM functionality, I used pwm\_demo to divide the 50hz CLK frequency down to CLK\_PWM. I specified BITS\_DUTY=3, and used three switches on the Terasic as the three duty input bits. I also set the pwm output to RLED0, and observed that the LED was dimmed more when the duty cycle was less, turned off completely when the duty cycle was zero, and turned on completely when the duty cycle was 7.