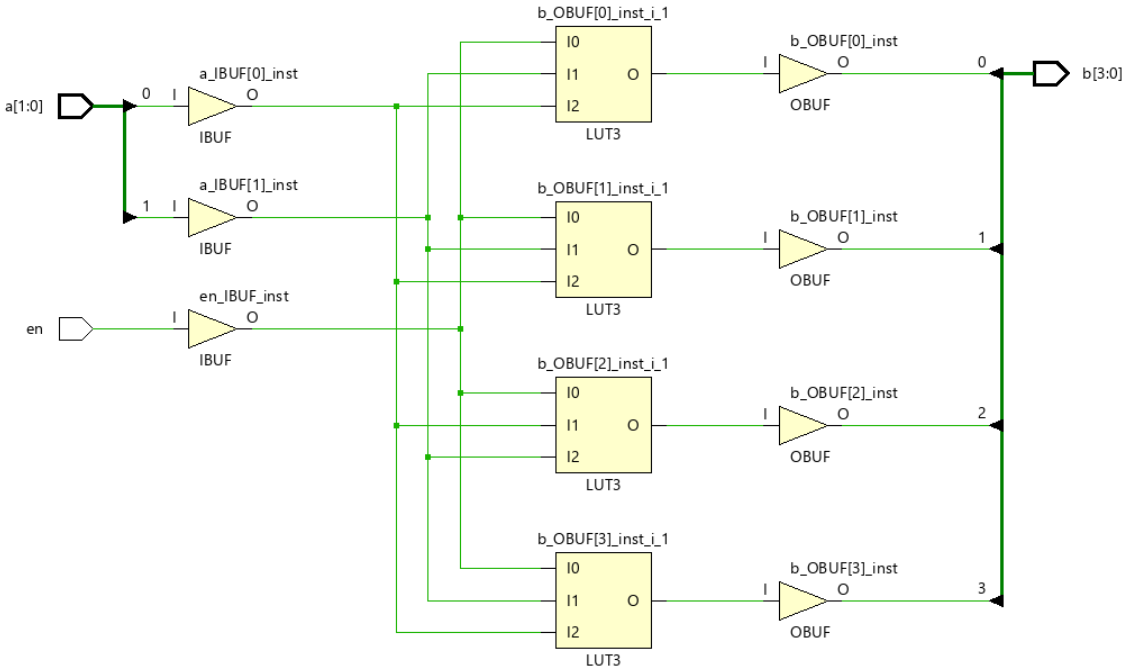
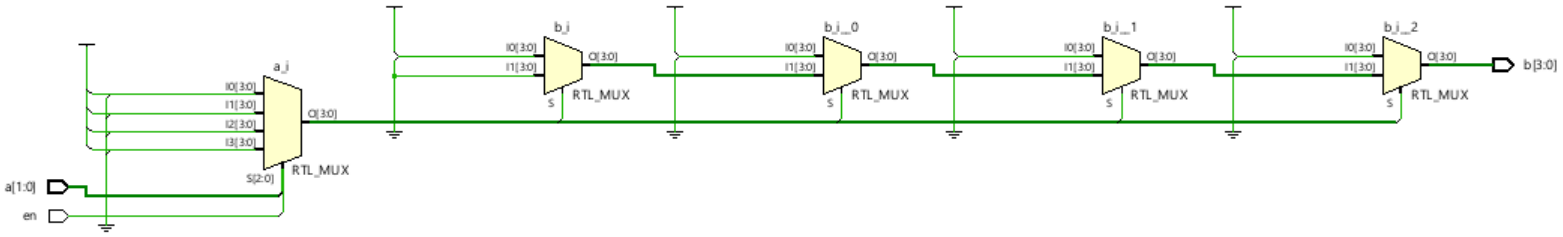
11a)

```
1  ----------------------------------------------------------------------------------
2  -- Author: <Cheuk Ki Liu><V00937822>
3  -- Contact info: <cheukkiliu@uvic.ca>
4  --
5  --Acknowledgements: this code is based in part on website
6  --https://www.ece.uvic.ca/~capson/ece441/
7  --
8  -- Module Name: 2-to-4 decoder with an enable signal (en) - Behavioral
9  --
10 -- Description: Assignment#1 question 11 a
11 --
12 -- Course: ECE 441/ ECE543
13 -- Department of Electrical and Computer Engineering
14 -- University of Victoria
15 --
16 -- Date: May 20, 2023
17 --
18 -- Notes: Inputs are 2 bit of a and enable
19 --        Outputs are 4 bit signals: b
20 --        When en is '1', the decoder operates as usual.
21 --        When en is '0', the decoder is disabled and the output is "0000".
22 ----------------------------------------------------------------------------------
23
24 library ieee;
25 use ieee.std_logic_1164.all;
26
27 ----------------------------------------------------
28
29 entity decoder24 is
30
31 port(
32     a:  in std_logic_vector(1 downto 0);
33     en: in std_logic ;
34     b:  out std_logic_vector (3 downto 0)
35     );
36 end decoder24;
37
38
39 architecture behavioural of decoder24 is
40
41 begin
42     process(a,en)
43     begin
44
45     if ( a="00" and en ='1') then
46          b <= "0001";
47     elsif ( a="01" and en ='1') then
48          b <= "0010";
49     elsif ( a="10" and en ='1') then
50          b <= "0100";
51     elsif ( a="11" and en ='1') then
52          b <= "1000";
53     else
54          b <= "0000";
55          end if;
56          end process;
57
58
59 end behavioural;
```

Synthesis schematics from a Vivado®:



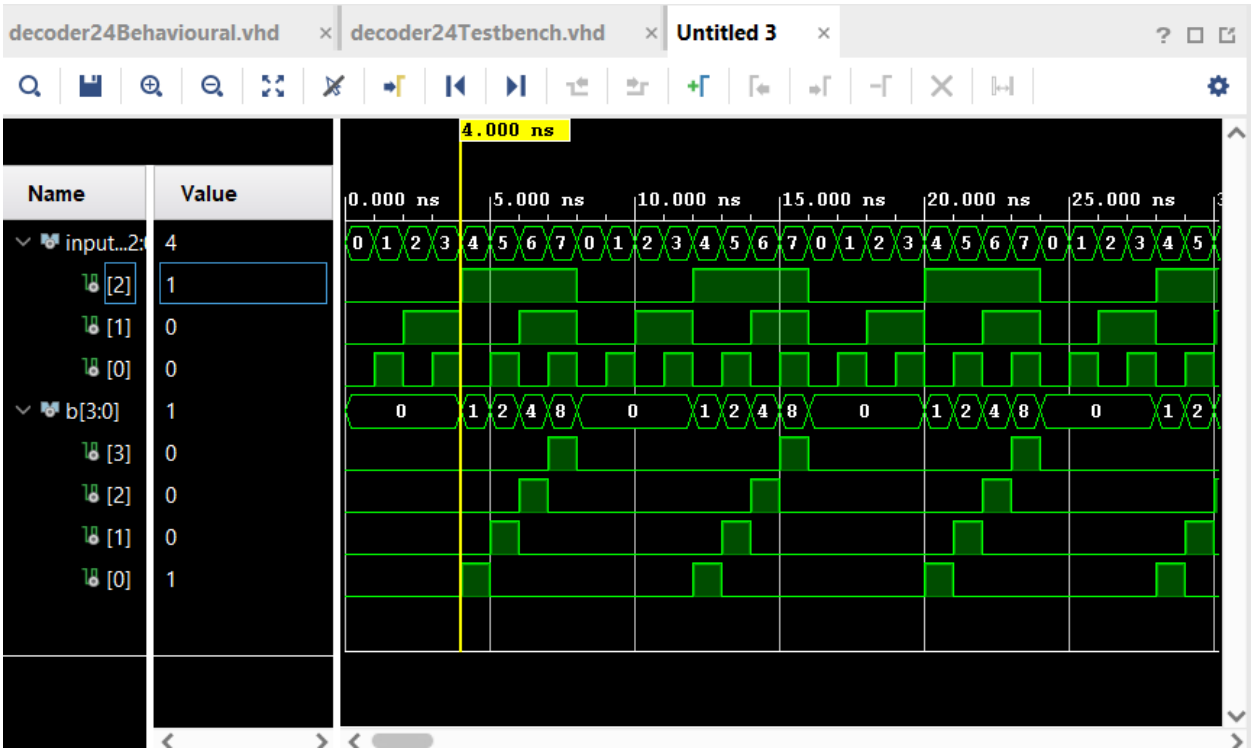
RTL schematics from a Vivado®:



11b)

```
1  ----------------------------------------------------------------------------------
2  -- Author: <Cheuk Ki Liu><V00937822>
3  -- Contact info: <cheukkiliu@uvic.ca>
4  --
5  --Acknowledgements: this code is based in part on website
6  --https://www.ece.uvic.ca/~capson/ece441/
7  --
8  -- Module Name: 2-to-4 decoder with an enable signal (en) - testbench
9  --
10 -- Description: Assignment#1 question 11 b
11 --
12 -- Course: ECE 441/ ECE543
13 -- Department of Electrical and Computer Engineering
14 -- University of Victoria
15 --
16 -- Date: May 20, 2023
17 --
18 -- Notes: Inputs are 2 bit of a and enable
19 --        Outputs are 4 bit signals: b
20 --        When en is '1', the decoder operates as usual.
21 --        When en is '0', the decoder is disabled and the output is "0000".
22 ----------------------------------------------------------------------------------
23
24 library IEEE;
25 use IEEE.STD_LOGIC_1164.ALL;
26 USE IEEE.numeric_std.ALL;
27
28
29 entity decoder24_testbench is
30 end decoder24_testbench;
31
32 architecture simulate of decoder24_testbench is
33
34     signal input_test_pattern: std_logic_vector (2 downto 0);
35
36     signal b: std_logic_vector (3 downto 0);
37
38 begin
39
40 uut : entity work.decoder24 port map(a=>input_test_pattern(1 downto 0), en=>input_test_pattern(2),b => b);
41
42
```

```
43
44 process
45  begin
46
47     for i in 0 to 7 loop  -- by default, the range 0 to 7 makes "i" an integer
48
49        input_test_pattern <= std_logic_vector(to_unsigned(i, 3));
50        wait for 1 ns;
51
52     end loop;
53
54  --  wait;
55
56 end process;
57
58  end simulate;
```



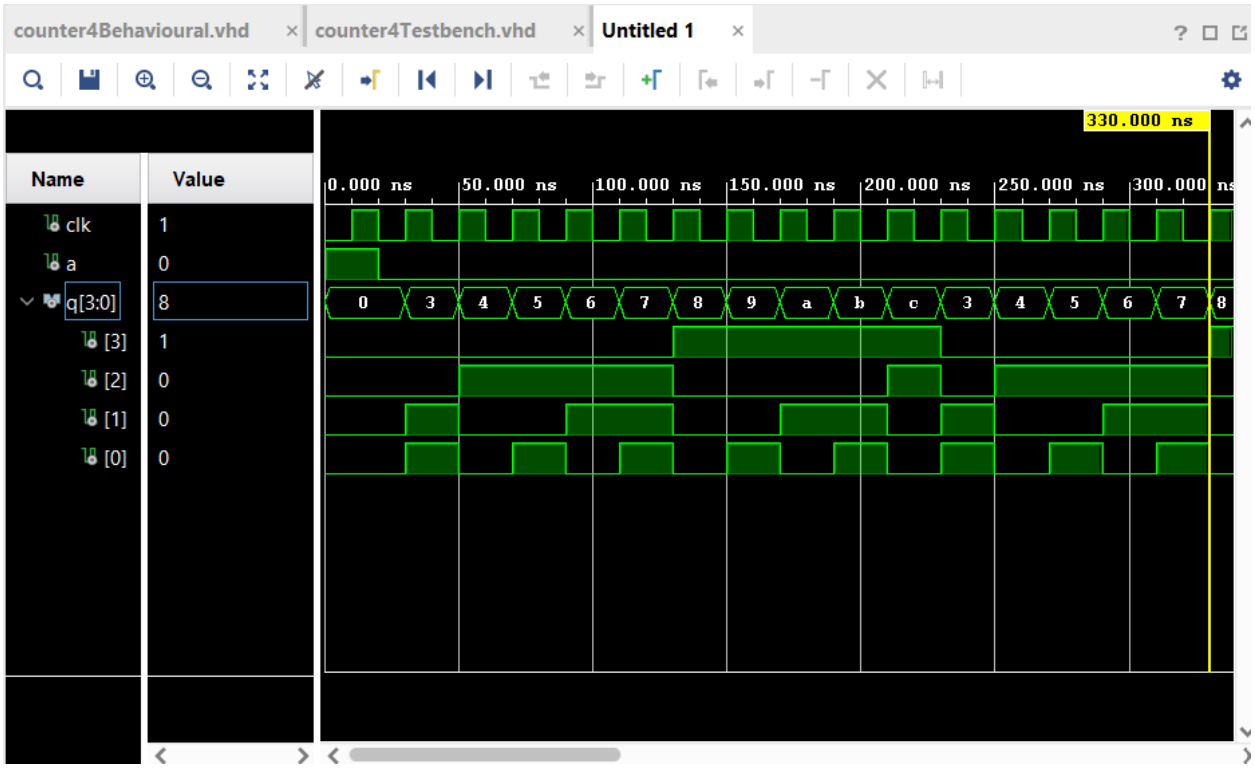
From the timing simulation, when the bit 2 (en) of the input is equal to '0', the output remain 0. When the bit 2 (en) is '1', the decoder operates as usual.

12.

```
1  ----------------------------------------------------------------------------------
2  -- Author: <Cheuk Ki Liu><V00937822>
3  -- Contact info: <cheukkiliu@uvic.ca>
4  --
5  --Acknowledgements: this code is based in part on website
6  --https://www.fpga4student.com/2017/06/vhdl-code-for-counters-with-testbench.html
7  --
8  -- Module Name: 4 bit counter - Behavioral
9  --
10 -- Description: Assignment#1 question 12
11 --
12 -- Course: ECE 441/ ECE543
13 -- Department of Electrical and Computer Engineering
14 -- University of Victoria
15 --
16 -- Date: May 20, 2023
17 --
18 -- Notes: Inputs are 2 bit of clock and set/reset
19 --        Outputs are 4 bit signals: q
20 --        4 bit counter counts from 3 ("0011") to 12 ("1100")
21 ----------------------------------------------------------------------------------
22
23 library ieee;
24 use ieee.std_logic_1164.all;
25 use IEEE.STD_LOGIC_UNSIGNED.ALL;
26 ------------------------------------------------
27
28 entity counter4 is
29
30 port(
31   clk:  in std_logic;  --clock
32     a:  in std_logic ; -- reset
33     q:  out std_logic_vector (3 downto 0) --output
34     );
35 end counter4;
36
37
38 architecture behavioural of counter4 is
39 -- make a counter to another variable to save the output value
40     signal count: std_logic_vector (3 downto 0);
41
42 begin
43     process(a,clk)
44     begin
45
46     if ( a = '1') then
47         count <= "0000";                    -- at the very beginnin
48     elsif (clk'event and clk = '1') then    -- rising edge
49         count <= count + "1";
50               if (count < "0011") then      --if count is less then 3
51                 count <= "0011";            --it have to be 3 after next rising edge
52             elsif (count >= "1100") then     --if count is greater then 12
53         count <= "0011";                    --it have to be 3 after next rising edge
54       end if;
55      end if;
56     end process;
57     q <= count;
58 end behavioural;
```

```vhdl
--------------------------------------------------------------------------------
-- Author: <Cheuk Ki Liu><V00937822>
-- Contact info: <cheukkiliu@uvic.ca>
--
--Acknowledgements: this code is based in part on website
--https://www.fpga4student.com/2017/06/vhdl-code-for-counters-with-testbench.html
--
-- Module Name: 4 bit counter - testbench
--
-- Description: Assignment#1 question 12
--
-- Course: ECE 441/ ECE543
-- Department of Electrical and Computer Engineering
-- University of Victoria
--
-- Date: May 20, 2023
--
-- Notes: Inputs are 2 bit of clock and set/reset
--        Outputs are 4 bit signals: q
--        4 bit counter counts from 3 ("0011") to 12 ("1100")
--------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE IEEE.numeric_std.ALL;


entity counter4_testbench is
end counter4_testbench;

architecture simulate of counter4_testbench is

    signal clk: std_logic;
    signal   a: std_logic;
    signal q: std_logic_vector (3 downto 0);

begin

uut : entity work.counter4 port map(a => a, clk=>clk, q=>q);

-- start the clock from 0 to 1
```

```
43  clock_start :process
44  begin
45       clk <= '0';
46       wait for 10 ns;
47       clk <= '1';
48       wait for 10 ns;
49  end process;
50
51
52  -- start for simulations
53  stimulation_start: process
54  begin
55
56     -- hold reset state for 20 ns. and start for 420 ns and loop for a cycle
57      a <= '1';
58     wait for 20 ns;
59      a <= '0';
60     wait for 420 ns;
61      a<= '1';
62      wait for 20 ns;
63      a <= '0';
64      wait;
65  end process;
66
67  end simulate;
```



From the time simulation, the 4-bit counter start at 0 and which is the unused state, in the next rising edge of the clock (clk), the output q restart from "0011", then counts from 3 ("0011") to 12 ("1100") and then wraps around.