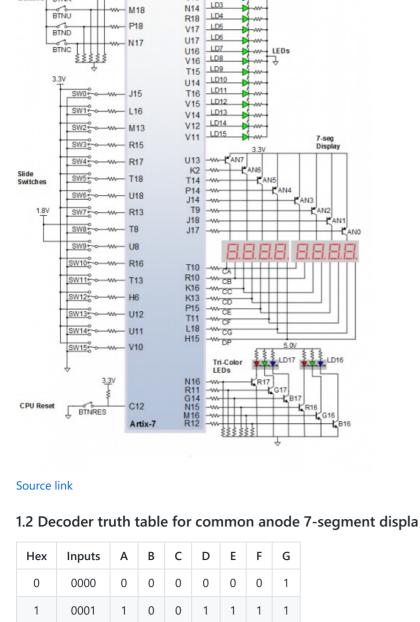
Assignment 4 Link to this Assignment Link to top of repository 1. Preparation tasks 1.1 Table with connection of 7-segment displays on Nexys A7 board LD0 H17

BTNL LD1 K15 M17 BTNR LD2 J13



3 4

5

7

8

9

2

0010

0011

0100

0101

0111

1000

1001

0

0

1

0

0

0

0

0

1 0

0

0

0

when "1000" =>

when "1001" =>

when "1010" =>

when "1011" =>

when "1100" =>

when "1101" =>

when "1110" =>

when "1111" =>

when others =>

2.2 Listing of VHDL stimulus process

tb_stimuls: process begin

wait for 100 ns; s_hex <= "0100"; wait for 100 ns; s_hex <= "0101"; wait for 100 ns; s_hex <= "0110";

wait for 100 ns; s_hex <= "0111"; wait for 100 ns; s_hex <= "1000"; wait for 100 ns; s_hex <= "1001";

wait for 100 ns; s_hex <= "1010"; wait for 100 ns; s_hex <= "1011";

wait for 100 ns; s_hex <= "1110"; wait for 100 ns; s_hex <= "1111"; wait for 100 ns;

wait;

Name

⊌ s_hex[3:0] 1 [3] l [2] la [1] [0] **v** s_seg[6:0]

> **6** [6] **[5] [**4] T [3] J [2] l [1] **[**0]

entity top is

end process tb_stimuls;

Value

38

s_hex <= "0000"; -- 0 wait for 100 ns; s_hex <= "0001"; wait for 100 ns; s_hex <= "0010"; wait for 100 ns; s_hex <= "0011";

end process p_7seg_decoder;

end Behavioral;

seg_o <= "0000000";

seg_o <= "0000100";

seg_o <= "0001000";

seg_o <= "1100000";

seg_o <= "0110001";

seg_o <= "1000010";

seg_o <= "0110000";

seg_o <= "0111000";

seg_o <= "0000000";

report "Stimulus process started" severity note;

-- 3

-- 6

-- 9

report "Stimulus process ended" severity note;

2.3 Screenshot with simulated time waveforms

200.000 ns

12

06

400.000 ns

4⊂

2.4 Listing of VHDL code from source file top.vhd with 7-segment module instantiation

24

20

600.000 ns 800.000 ns

0f

00

|0.000 ns

01

4f

Port (SW : in STD_LOGIC_VECTOR (3 downto 0);

CA : out STD_LOGIC; CB : out STD_LOGIC;

seg_o(1) => CF, $seg_o(0) \Rightarrow CG$

-- Display input value on LEDs

-- Connect one common anode to 3.3V

-- Turn LED(4) on if input value is equal to 0, ie "0000" $\,$

-- Turn LED(6) on if input value is odd, ie 1, 3, 5, \dots

-- Turn LED(5) on if input value is greater than "1001", ie 9

-- Turn LED(7) on if input value is a power of two, ie 1, 2, 4, or 8

LED(4) <= '1' when (SW = "0000") else '0';

LED(5) <= '1' when (SW >"1001") else '0';

LED(7) <= '1' when SW = "0001" else

'1' when SW = "0010" else '1' when SW = "0100" else '1' when SW = "1000" else

3.1 Truth table and listing of VHDL code for LEDs(7:4)

LED6

0

1

0

1

0

1

0

1

0

1

0

1

0

1

-- Turn LED(4) on if input value is equal to 0, ie "0000"

-- Turn LED(5) on if input value is greater than "1001", ie 9

LED7

1

1

0

1

0

0

0

1

0

0

0

0

LED5

0

0

0

0

0

0

0

0

0

1

1

1

1

LED(4) <= '1' when (SW = "0000") else '0';

);

AN <= b"1111_0111";

LED(3 downto 0) <= SW;

-- LED(7:4) indicators

 $LED(6) \leftarrow SW(0);$

'0';

LED4

1

0

0

0

0

0

0

0

0

0

0

0

0

0

-- Display input value on LEDs

LED(3 downto 0) <= SW;

-- LED(7:4) indicators

end architecture Behavioral;

3. LED(7:4) indicators

Inputs

0000

0001

0010

0011

0100

0101

0110

0111

1000

1001

1010

1011

1100

1111

Hex

0

1

2

3

4

5

6

7

8

9

Α

b C

F

7 [7] 🌡 [6] 🌡 [5] J [4] 🌡 [3] 🌡 [2] 7 [1] 🌡 [0] lo s_CA s_CB å s_CC d s_CD d s_CE s_CF l₀ s_CG **⊌** s_AN[7:0]

f7

-- A

-- B

-- D

-- E

-- undefined

1

0

0

0

0

0

6 0110 0 0 0 0 0 0 1

0

0

1

0

1

0

0

0 1

1 1

1 0

1 0

1

0 0

1

0

0

0

0

0

1

0

0

| b | | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
|-----|---|--|--|---|---|---|-----------------------|---------------------------------------|
| | 1011 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| С | 1100 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| C | 1100 | U | ' | ' | U | U | U | 1 |
| d | 1101 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| Е | 1110 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| F | 1111 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| • | | | | | | Ū | | |
| gin | | decode | er: | | x_7se | | | |
| | - A combi | | | | | | _ | |
| - | ·- A combin ·- Any tim ·- Output | e "hex | (_i" | is ch | anged | , the | pro | cess |
| - | Any timo Output p | e "hex oin se | <_i" | is ch 6) co | anged rresp | , the | e pro | cess |
| - | Any time Output p | e "hex pin se oder : | c_i" : eg_o(pro | is ch 6) co | anged rresp | , the | e pro | cess egmen |
| - | Any time Output | e "hex pin se coder : ex_i i | <pre>c_i" : eg_o(pro is 000" :</pre> | is ch 6) co cess(=> | anged rresp hex_i | , the onds)begi | e pro | cess egmen |
| - | Any timo Output p | e "hex pin se oder : ex_i i en "00 seg_ | (_i" : eg_o(: pro is 000" : | is ch 6) co cess(=> "000 | anged rresp | , the onds)begi | to so | cess egmen |
| - | Any timo Output p | e "hexoin seconder : ex_i i en "00 seg_en "00 | <pre>c_i" : eg_o(pro is 000" : 000" : 001" :</pre> | is ch 6) co cess(=> "000 | anged rresp hex_i 0001" | , the onds)begi | to so | cess egmen |
| - | Any timo | e "hexoin seconder : ex_i i en "00 seg_en "00 | c_i": eg_o(profis 000": 0 <= 001": | is ch 6) co cess(=> "000 => | anged rresp hex_i | , the onds)begi | to so | cess egmen |
| - | Any tim Output Outp | e "hexoin seconder : ex_i i en "00 seg_ en "00 seg_ en "00 seg_ | <pre>c_i": eg_o(pro is 000": 0 <= 001": 0 <= 010": 0 <=</pre> | is ch 6) co cess("000 => "100 => "001 | anged rresp hex_i 0001" | , the onds | e prod to so in | cess egmen |
| - | Any tim Output Outp | e "hexoin seconder : ex_i i en "00 seg_en "00 seg_en "00 seg_en "00 | c_i": eg_o(procis 000": 0 <= 001": 0 <= 010": 0 <= 011": | is ch 6) co cess(=> "000 => "100 => | anged rresp hex_i 0001" 1111" | , the onds | e prod to so in | cess egmen 0 1 |
| - | Any tim Output | e "hexoin seconder : coder : c | c_i": eg_o(: pro is 000" o <= 001" o <= 010" o <= 011" o <= | is ch 6) co cess(=> "000 => "001 => | anged rresp hex_i 0001" | , the onds | (| cess egmen 2 |
| - | Any tim Output | e "hey poin se poder : ex_i i en "00 seg_ en "00 seg_ en "00 seg_ en "00 seg_ en "00 | (_i": eg_o((: pro is 000": _o <= 001": _o <= 011": _o <= | is ch 6) co cess("000 => "100 => "001 => "000 => | anged rresp hex_i 0001" 1111" 0010" | , the onds) begi | e proce to se | cess egmen 2 |
| - | Any time Output | e "hey poin se poder : ex_i i en "00 seg_ en "00 seg_ en "00 seg_ en "00 seg_ en "00 | c_i": pro is pool": o <= 001": o <= 011": o <= 011": o <= 011": o <= 010": o <= 011": o <= 010": o | is ch 6) co cess("000 => "100 => "001 => "000 => "1000 => | anged rresp hex_i 0001" 1111" | , the onds) begi | (| cess egmen 2 1 1 2 |
| - | Any time Output Output | e "hexpoin seconder : coder : ex_i i en "00 seg_en "00 seg_en "00 seg_en "01 seg_en "01 seg_en "01 seg_en "01 | <pre>c(_i" : reg_o() r</pre> | is ch6) co cess("000 => "100 => "001 => "100 => "100 => "100 => "100 => "100 => "100 => "100 | anged rresp hex_i 0001" 1111" 0010" | , the conds conds conds conds conds conds conds conds conds cond cond cond cond cond cond cond cond | :: | cess egmen a a 1 1 2 4 4 |
| - | Any time Output Output | e "hexpoin seconder : coder : ex_i i en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "01 seg_en "01 | <pre> (<i" ''="" ''<="" td=""><td>is ch6) co cess("000 => "100 => "001 => "100 => "100 => "100 =></td><td>anged rresp hex_i 0001" 1111" 0010" 1100"</td><td>, theedonds</td><td> :</td><td>cess egmen a a 1 1 2 4 4</td></i"></pre> | is ch6) co cess("000 => "100 => "001 => "100 => "100 => "100 => | anged rresp hex_i 0001" 1111" 0010" 1100" | , theedonds | : | cess egmen a a 1 1 2 4 4 |
| - | Any time Output Output Tseg_deco case he who who who who who who | e "hexpoin seconder : coder : ex_i i en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "01 seg_en "01 | <pre>(<i" ''="" ''<="" td=""><td>is ch6) co cess("000 => "100 => "001 => "010 => "010</td><td>anged rresp hex_i 0001" 1111" 0010" 0110"</td><td>, theodores</td><td> ::</td><td>cess egmen</td></i"></pre> | is ch6) co cess("000 => "100 => "001 => "010 => "010 | anged rresp hex_i 0001" 1111" 0010" 0110" | , theodores | :: | cess egmen |
| - | Any time Output Output Tseg_deco case he who who who who who who who | e "hexpoin seconder: coder: ex_i i ex_i i en "00 seg_en "00 seg_en "00 seg_en "00 seg_en "01 seg_en "01 seg_en "01 seg_en "01 | <pre>(<i" :="pro" is="" pro="" pro<="" td=""><td>is ch6) co cess("000 => "100 => "000 => "010 => "010 => "010 => "010</td><td>anged rresp hex_i 0001" 1111" 0110" 1100" 0100" 1111"</td><td>, thee</td><td> :: :: ::</td><td>cess egmen 3 1 2 4 5 7</td></i"></pre> | is ch6) co cess("000 => "100 => "000 => "010 => "010 => "010 => "010 | anged rresp hex_i 0001" 1111" 0110" 1100" 0100" 1111" | , thee | :: :: :: | cess egmen 3 1 2 4 5 7 |

wait for 100 ns; s_hex <= "1100"; -- C wait for 100 ns; s_hex <= "1101"; -- D

| | CC : out STD_LOGIC; |
|--------------|---|
| | CD : out STD_LOGIC; |
| | CE : out STD_LOGIC; |
| | CF : out STD_LOGIC; |
| | CG : out STD_LOGIC; |
| | AN : out STD_LOGIC_VECTOR (7 downto 0); |
| | LED : out STD_LOGIC_VECTOR (7 downto 0)); |
| end top; | |
| | |
| | |
| - Architec | ture body for top level |
| 711 0112 000 | court body for cop level |
| | |
| | re Behavioral of top is |
| oegin | |
| | |
| | |
| Inst | cance (copy) of hex_7seg entity |
| hex2seg | g : entity work.hex_7seg |
| por | t map(|
| | hex_i => SW, |
| | $seg_o(6) \Rightarrow CA$, |
| | $seg_o(5) \Rightarrow CB$, |
| | $seg_o(4) \Rightarrow CC$, |
| | $seg_o(3) \Rightarrow CD$, |
| | |

1,600.000 ns

38

1,400.000 ns

30

1,000.000 ns |1,200.000 ns

60

31

42

08

04

1101 d 0 1 1 0 Ε 1110 0 1 0 0

| | \- I \ | when (SW >"1 | 001) C1. | se 0; | | | | | | | | | | | | |
|------------------------------------|-------------------------|---|-------------------------------------|-------------|---------|---------|---------|---------|---------|---|--------------|-------|---------|--------|--------------|--------|
| | en LED(6) | on if input | value i | s odd, : | ie 1, 3 | 3, 5, . | • • | | | | | | | | | |
| | <= '1' \\ '1' \\ '1' \\ | on if input when SW = "0 when SW = "0 when SW = "0 when SW = "1 | 001" elso 010" elso 100" elso | e e e | er of t | two, ie | 1, 2, | 4, or 8 | 3 | | | | | | | |
| | | | | | | | | | | | | | | | | |
| 3.2 Screen | shot wi | th simulat | ed time | e wave | eform | ıs | | | | | | | | | | |
| 3.2 Screen | | th simulat | ed time | | eform | | 600.000 | ns | 800.000 |) ns | 1,000.0 | 00 ns | 1,200.0 | 100 ns | 1,400.0 |)00 ns |
| Name | Value | | | | | | 600.000 | ns | 800.000 |) ns 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | 1,000.0 | 00 ns | 1,200.0 | 100 ns | 1,400.0 | 000 ns |
| | Value | | 200.000 | ns L | |) ns | | ns 7 | 800.000 |) ns | 1,000.0 | | 1,200.0 | 00 ns | 1,400.0 e | 000 ns |
| Name ∨ ⊌ s_hex[3:0] | Value f | | 200.000 | ns L | |) ns | | ns 7 | 800.000 |) ns | 1,000.0 a | | 1,200.0 | 00 ns | 1,400.0 e | 000 ns |
| Name ∨ ▼ s_hex[3:0] 1 [3] | Value f 1 | | 200.000 | ns L | |) ns | | ns 7 | 800.000 |) ns | 1,000.0 a | | 1,200.0 | 000 ns | 1,400.0 | 000 ns |
| Name > * s_hex[3:0] 1 [3] 1 [2] | Value f 1 | | 200.000 | ns L | |) ns | | ns 7 | 800.000 | 0 ns | 1,000.0 | | 1,200.0 | 00 ns | 1,400.0 e | 000 ns |