First Assignment

By

Calvin Boostman

Gabriel Cobos Tello

Rick Kloosterman

VHDL

G. Nanninga

# Introdution

The idea behind this first assignment is for us to get a good look at the basics of VHDL, managing to become comfortable with its structure and methodology so that we can move on to more complex tasks and finally achieve a moderate mastery for this programing language.

# Part 2

9.

Afbeelding met schermafbeelding

Beschrijving is gegenereerd met zeer hoge betrouwbaarheid

Afbeelding met schermafbeelding

Beschrijving is gegenereerd met zeer hoge betrouwbaarheid

13.

1. The pin assignments. When using the signal names in the file they are connected to the right pin numbers of the FPGA.
2. A sum of Products is the standard expression that defines the addition of binary numbers, were 1 is always dominant over 0***. F = A’B + AB’ + AB***, this would express the addition of 2 binary digits but can be extrapolated to any number of them, a truth table can be used for this.
3. Instantiation is the term that defines the introduction of entities as components in a VHDL design.

instance\_label: component\_name

**generic map** (generic\_association\_list)

**port map** (port\_association\_list);

u1: component port map (X,Y,S,C);

1. G0 : for i in 0 to 7 generate

M0 : Component port map (X,Y(i),S(i),C(i));

End generate;

1. A top-level entity is the linking file that unites the functions of all components and makes them relate to the world.
2. In the work library.

# Part 3

## Truth table

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C3** | **C2** | **C1** | **C0** | **H0** | **H1** | **H2** | **H3** | **H4** | **H5** | **H6** |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

## Questions

7.

1. 30,216
2. 4 are used as output
3. 4 are used as input
4. Check for whenever the output is supposed to be high, the input. Check from that input which gates are ‘1’. Those gates that are ‘1’are the input for an AND gate. The output of that AND gate is the input of an OR gate.
5. 7, because there are always 7 for the 7 segment display

9.

Number of LEs: 7 for both, because you only need 7. It will always be optimized to the least LEs necessary.

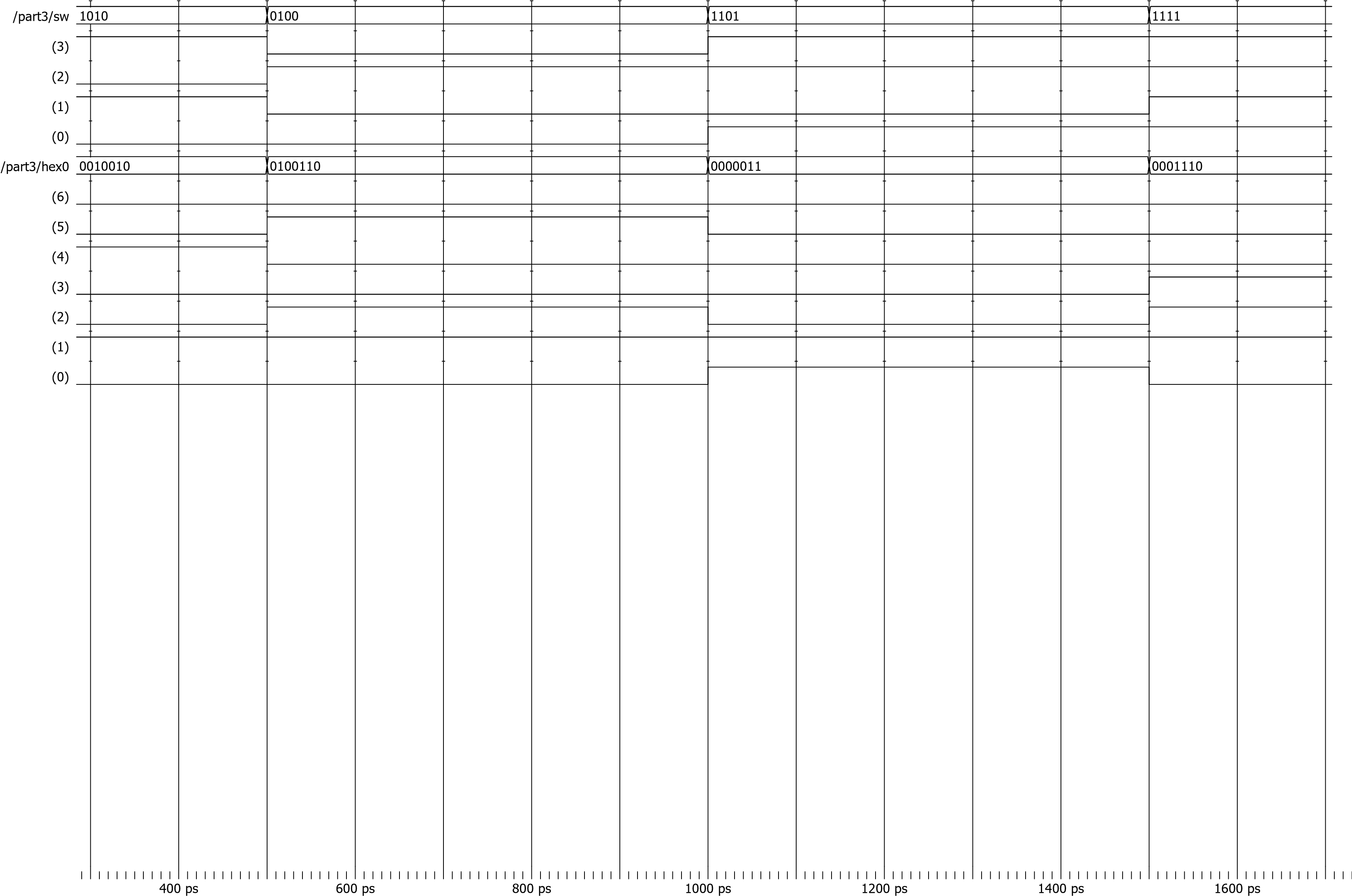
TCL:

force -freeze sim:/part3/sw 1010 0 run

force -freeze sim:/part3/sw 0100 0 run

force -freeze sim:/part3/sw 1101 0 run

force -freeze sim:/part3/sw 1111 0 run



## Code

Afbeelding met schermafbeelding

Beschrijving is gegenereerd met zeer hoge betrouwbaarheid

Afbeelding met schermafbeelding

Beschrijving is gegenereerd met hoge betrouwbaarheid

1. library ieee;

# PART 4

## Truth table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A2** | **A1** | **A0** | **B2** | **B1** | **B0** | **Output** |
| 0 | 0 | 0 | 0 | 0 | 0 | **0** |
| 0 | 0 | 1 | 0 | 0 | 0 | **1** |
| 0 | 1 | 0 | 0 | 0 | 0 | **1** |
| 0 | 1 | 1 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 1 | 0 | 0 | 0 | **1** |
| 1 | 1 | 0 | 0 | 0 | 0 | **1** |
| 1 | 1 | 1 | 0 | 0 | 0 | **1** |
| 0 | 0 | 0 | 0 | 0 | 1 | **0** |
| 0 | 0 | 1 | 0 | 0 | 1 | **0** |
| 0 | 1 | 0 | 0 | 0 | 1 | **1** |
| 0 | 1 | 1 | 0 | 0 | 1 | **1** |
| 1 | 0 | 0 | 0 | 0 | 1 | **1** |
| 1 | 0 | 1 | 0 | 0 | 1 | **1** |
| 1 | 1 | 0 | 0 | 0 | 1 | **1** |
| 1 | 1 | 1 | 0 | 0 | 1 | **1** |
| 0 | 0 | 0 | 0 | 1 | 0 | **0** |
| 0 | 0 | 1 | 0 | 1 | 0 | **0** |
| 0 | 1 | 0 | 0 | 1 | 0 | **0** |
| 0 | 1 | 1 | 0 | 1 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | 0 | **1** |
| 1 | 0 | 1 | 0 | 1 | 0 | **1** |
| 1 | 1 | 0 | 0 | 1 | 0 | **1** |
| 1 | 1 | 1 | 0 | 1 | 0 | **1** |
| 0 | 0 | 0 | 0 | 1 | 1 | **0** |
| 0 | 0 | 1 | 0 | 1 | 1 | **0** |
| 0 | 1 | 0 | 0 | 1 | 1 | **0** |
| 0 | 1 | 1 | 0 | 1 | 1 | **0** |
| 1 | 0 | 0 | 0 | 1 | 1 | **1** |
| 1 | 0 | 1 | 0 | 1 | 1 | **1** |
| 1 | 1 | 0 | 0 | 1 | 1 | **1** |
| 1 | 1 | 1 | 0 | 1 | 1 | **1** |
| 0 | 0 | 0 | 1 | 0 | 0 | **0** |
| 0 | 0 | 1 | 1 | 0 | 0 | **0** |
| 0 | 1 | 0 | 1 | 0 | 0 | **0** |
| 0 | 1 | 1 | 1 | 0 | 0 | **0** |
| 1 | 0 | 0 | 1 | 0 | 0 | **0** |
| 1 | 0 | 1 | 1 | 0 | 0 | **1** |
| 1 | 1 | 0 | 1 | 0 | 0 | **1** |
| 1 | 1 | 1 | 1 | 0 | 0 | **1** |
| 0 | 0 | 0 | 1 | 0 | 1 | **0** |
| 0 | 0 | 1 | 1 | 0 | 1 | **0** |
| 0 | 1 | 0 | 1 | 0 | 1 | **0** |
| 0 | 1 | 1 | 1 | 0 | 1 | **0** |
| 1 | 0 | 0 | 1 | 0 | 1 | **0** |
| 1 | 0 | 1 | 1 | 0 | 1 | **0** |
| 1 | 1 | 0 | 1 | 0 | 1 | **1** |
| 1 | 1 | 1 | 1 | 0 | 1 | **1** |
| 0 | 0 | 0 | 1 | 0 | 1 | **0** |
| 0 | 0 | 1 | 1 | 0 | 1 | **0** |
| 0 | 1 | 0 | 1 | 0 | 1 | **0** |
| 0 | 1 | 1 | 1 | 0 | 1 | **0** |
| 1 | 0 | 0 | 1 | 0 | 1 | **0** |
| 1 | 0 | 1 | 1 | 0 | 1 | **0** |
| 1 | 1 | 0 | 1 | 0 | 1 | **0** |
| 1 | 1 | 1 | 1 | 0 | 1 | **1** |
| 0 | 0 | 0 | 1 | 1 | 1 | **0** |
| 0 | 0 | 1 | 1 | 1 | 1 | **0** |
| 0 | 1 | 0 | 1 | 1 | 1 | **0** |
| 0 | 1 | 1 | 1 | 1 | 1 | **0** |
| 1 | 0 | 0 | 1 | 1 | 1 | **0** |
| 1 | 0 | 1 | 1 | 1 | 1 | **0** |
| 1 | 1 | 0 | 1 | 1 | 1 | **0** |
| 1 | 1 | 1 | 1 | 1 | 1 | **0** |

## Code using boolean expressions

1. library IEEE;
2. use IEEE.STD\_LOGIC\_1164.ALL;
3. use IEEE.STD\_LOGIC\_ARITH.ALL;
4. use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

5

6

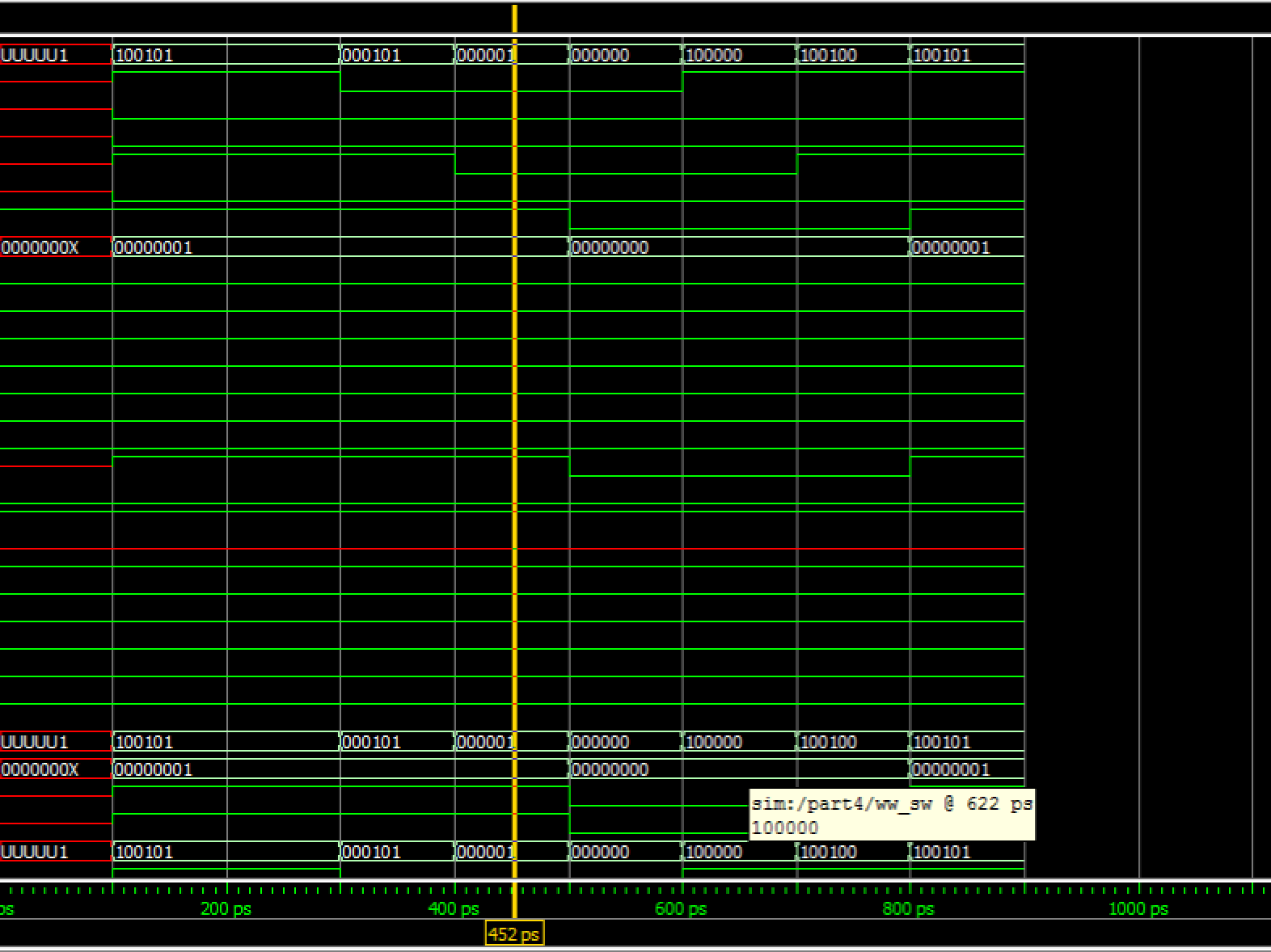
1. entity part4 is
2. Port (
3. SW : in std\_logic\_vector(5 downto 0);
4. LEDG: out std\_logic\_vector(7 downto 0));
5. end part4;

12

1. architecture behavioral of part4 is
2. begin
3. LEDG(0)<=( (SW(2) and not(SW(3)) and not(SW(4)) and not(SW(5))) or
4. (SW(0) and SW(2) and not(SW(4)) and not(SW(5))) or
5. (SW(1) and SW(2) and not(SW(3)) and not(SW(5))) or
6. (SW(0) and SW(1) and SW(2) and not(SW(5))) or
7. (SW(1) and not(SW(3)) and not(SW(4))) or
8. (SW(0) and SW(1) and not(SW(4))) or
9. (SW(0) and not(SW(3))));
10. end behavioral;

23

## Simulation



## tCL

force -freeze sim:/part4/sw(0) 1 0

run

force -freeze sim:/part4/sw(1) 0 0

force -freeze sim:/part4/sw(2) 1 0

noforce sim:/part4/sw(3)

force -freeze sim:/part4/sw(3) 0 0

force -freeze sim:/part4/sw(4) 0 0

force -freeze sim:/part4/sw(5) 1 0

run

add wave \

{sim:/part4/sw(5) }

add wave \

{sim:/part4/sw }

add wave \

{sim:/part4/ledg(0) }

run

force -freeze sim:/part4/sw(5) 0 0

run

force -freeze sim:/part4/sw(2) 0 0

force -freeze sim:/part4/sw(2) 0 0

force -freeze sim:/part4/sw(2) 0 0

run

force -freeze sim:/part4/sw(0) 0 0

run

force -freeze sim:/part4/sw(5) 1 0

run

force -freeze sim:/part4/sw(2) 1 0

run

force -freeze sim:/part4/sw(0) 1 0

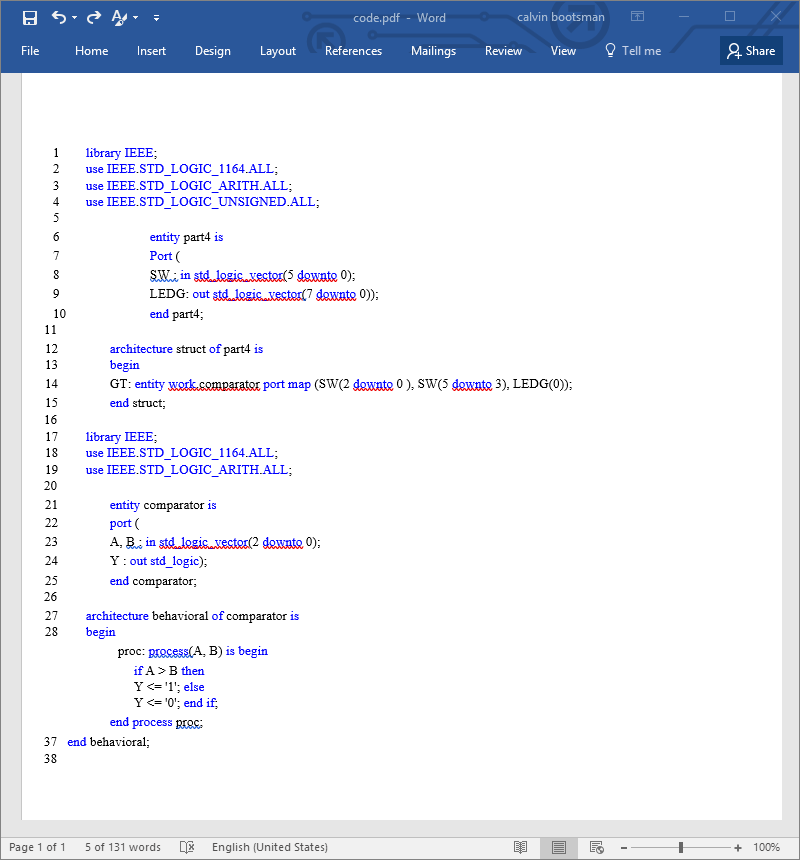
run

## LEs used

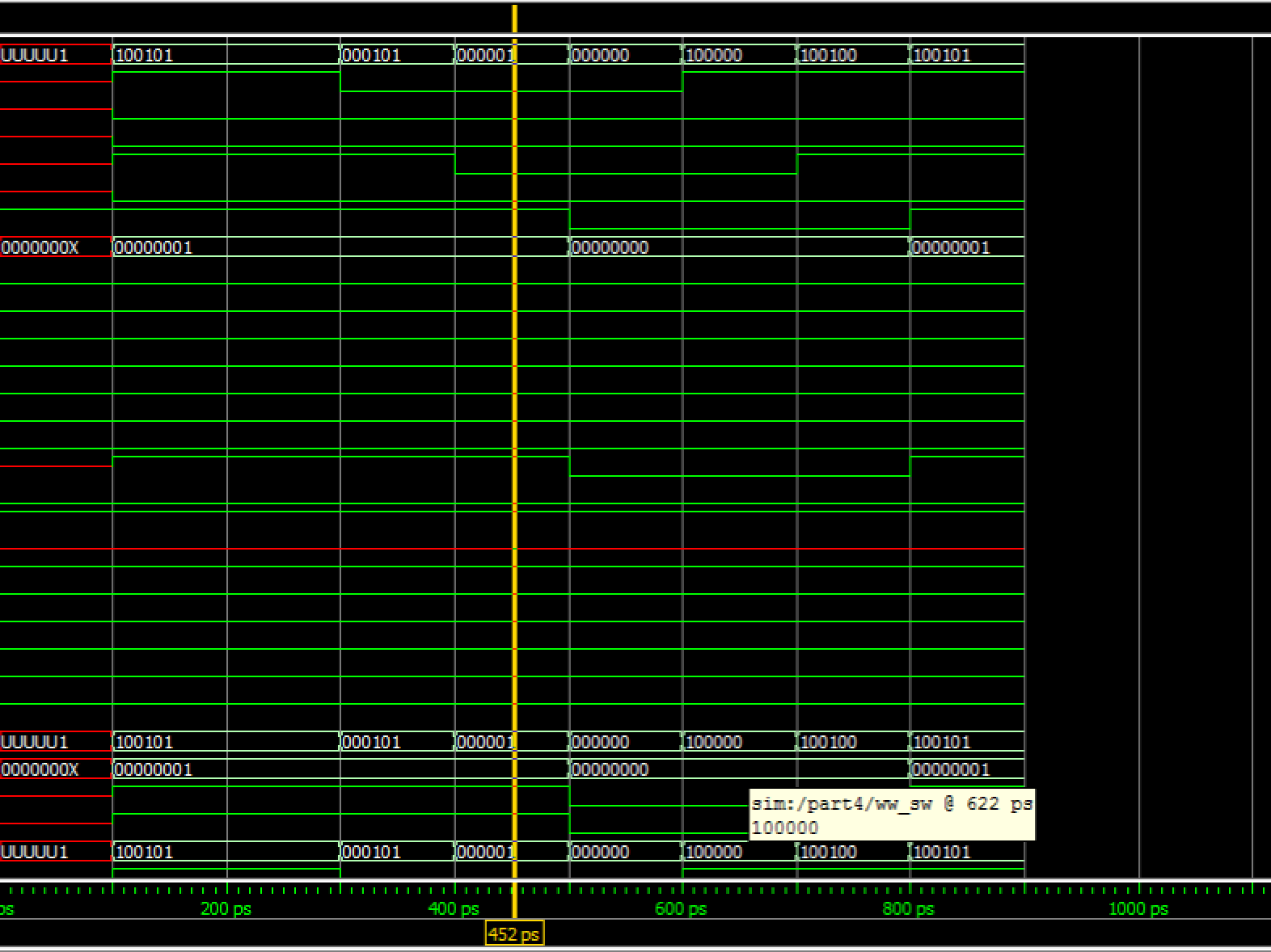
Logic elements used 2

We use a Greater Than operator that consumes 2, 3-bit inputs. The output is one bit.

## Code using behavioural



## Simulation



## Les used

Logic elements used 2

## questions

1. We would get negative numbers. The code have a different range but will still work.
2. Yes, working it out on the board
3. Only if we can program it in behavioral programming.