Mohammadreza Taraz and Tien Mai

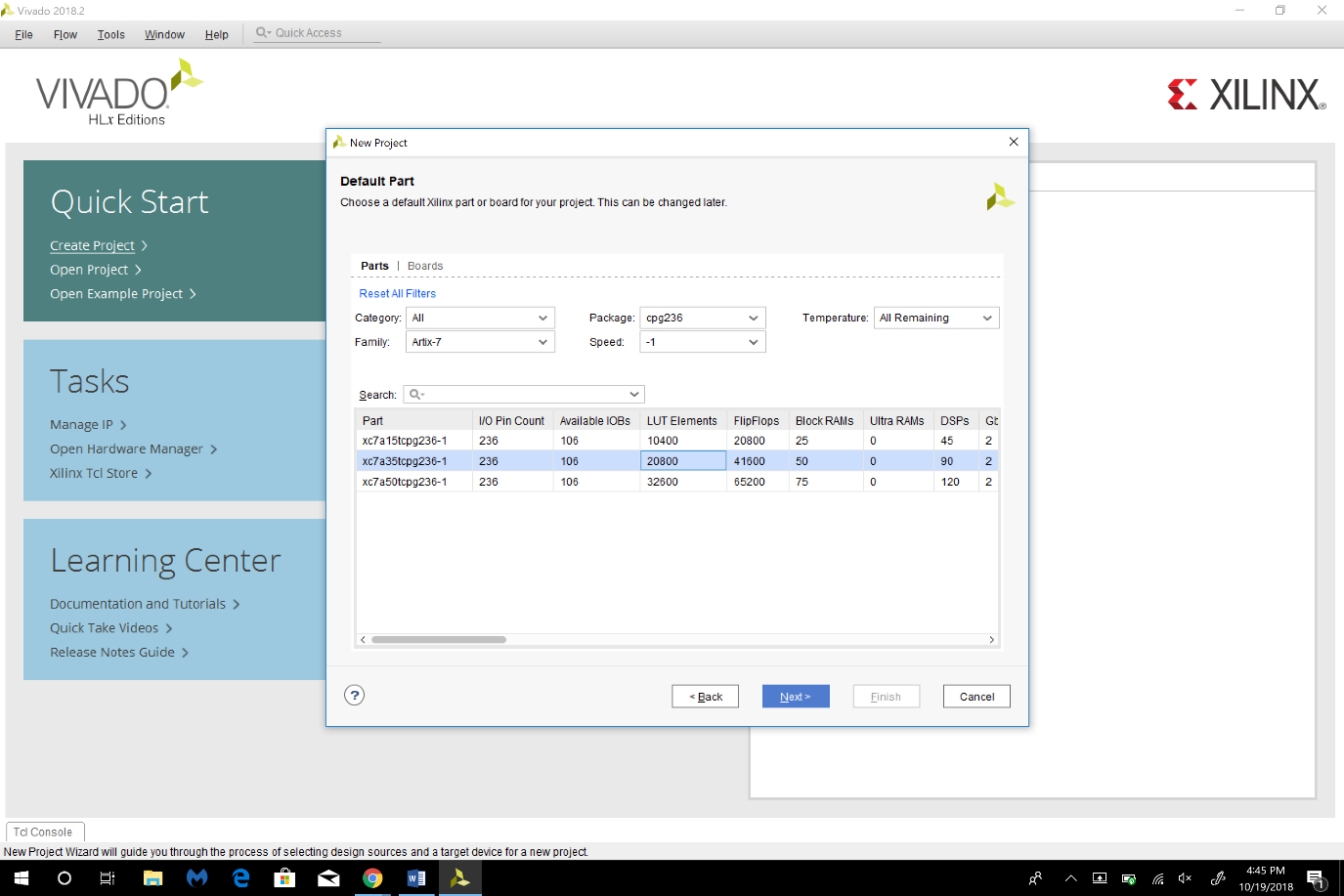
Dr. Mihail Cutitaru

ECGR 2181-002

October 19, 2018

VHDL Report

We start by creating a directory, and me made it as Register Transfer Level (RTL) project. After we had to change the simulation language to VHDL.

here we changed Family: Artix-7, Package: CPG236, Speed: -1, and we choose the one with 41600 FlipFlops.

Create A, B, Cin as input for out project and outputs are Sum and Cout.

TASK 1:

We Followed all the steps that was listed and that is our result for the first task.

Our code:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity task1 is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

Cin : in STD\_LOGIC;

Sum : out STD\_LOGIC;

Cout : out STD\_LOGIC);

end task1;

architecture Behavioral of task1 is

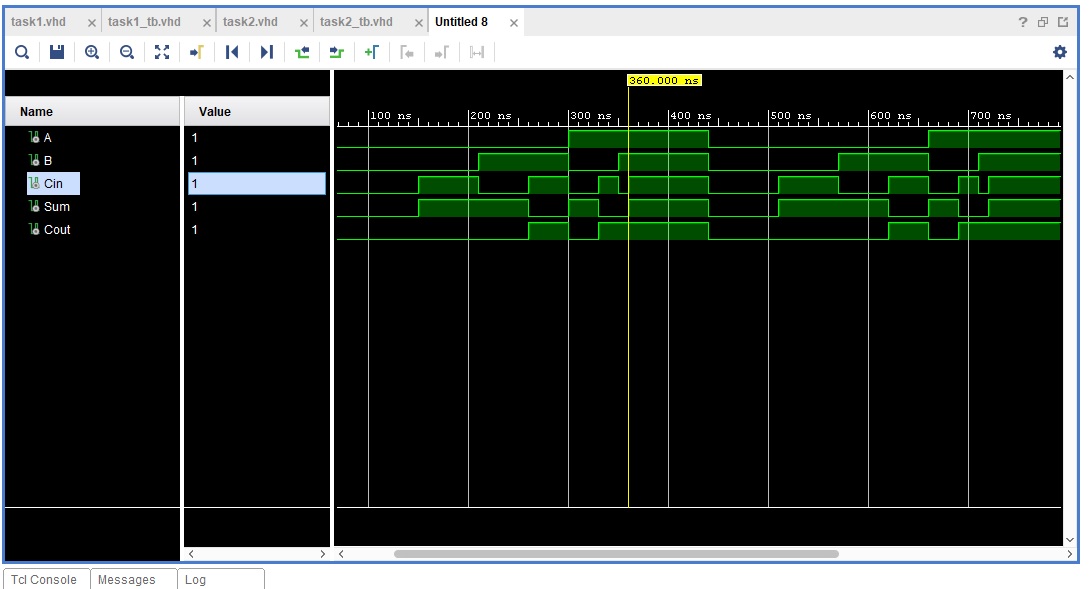
begin

Sum <= A XOR B XOR Cin;

Cout <= (A AND B) OR (A AND Cin) OR (B AND Cin);

end Behavioral;

Result:



One problem we had was that we forgot to add code into table-bench and we had a hard time writing a table-bench.

PART 1 Task2: The problem we face was simplifying the implement function to the canonical SOP for the second part of the project.

Our code:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity task2 is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

C : in STD\_LOGIC;

D : in STD\_LOGIC;

F : out STD\_LOGIC);

end task2;

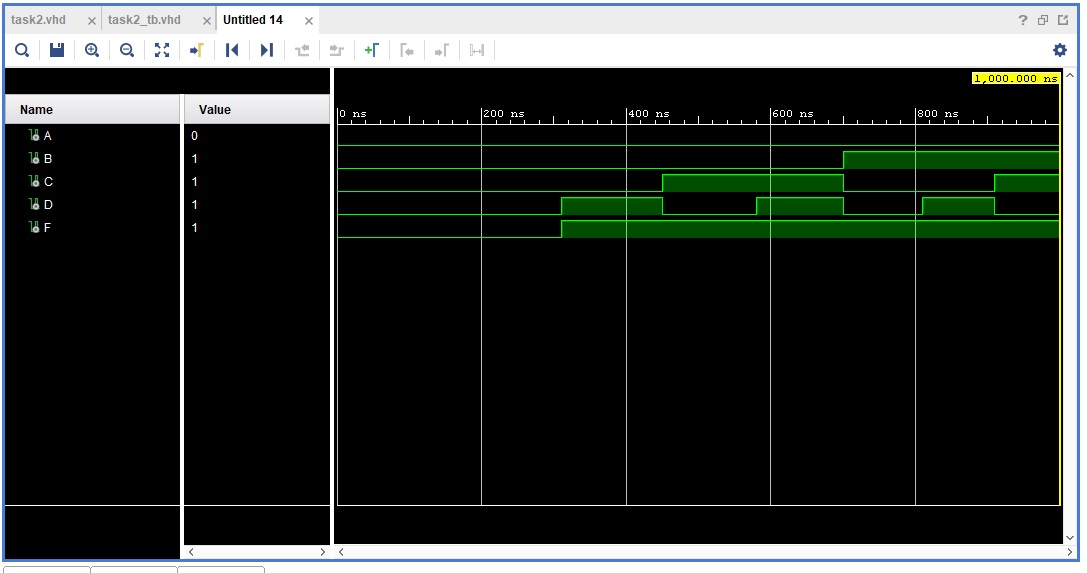
architecture Behavioral of task2 is

begin

F <= A OR B OR C OR D OR (A AND B AND C AND D) OR (A AND (NOT B) AND (NOT C) AND (NOT D));

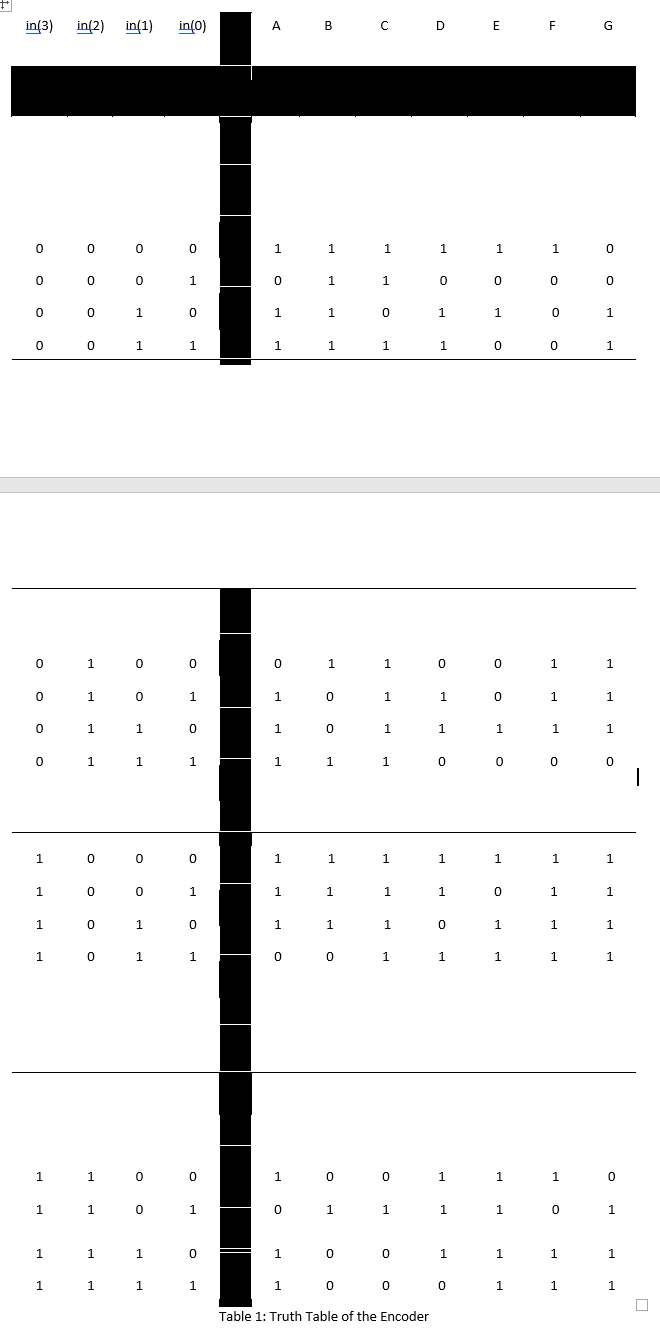
end Behavioral;

Result:



PART 2:

In the second part of the project we write and simulate a hex to seven segment display converter. We filled up the truth table for it on the figure bellow wherever we have a segment of the value will be 1 and when it’s on it will be 0.



Our code for PART 2:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity task3 is

Port (

hex\_in : in STD\_LOGIC\_VECTOR (3 downto 0);

letters : out STD\_LOGIC\_vector (6 downto 0)

);

end task3;

architecture Behavioral of task3 is

signal temphex\_in: integer;

begin

process (hex\_in)

begin

temphex\_in <= to\_integer(unsigned(hex\_in));

case hex\_in is

when "0000" => letters <= "0000001";

when "0001" => letters <= "1001111";

when "0010" => letters <= "0010010";

when "0011" => letters <= "0000110";

when "0100" => letters <= "1001100";

when "0101" => letters <= "0100100";

when "0110" => letters <= "0100000";

when "0111" => letters <= "0001111";

when "1000" => letters <= "0000000";

when "1001" => letters <= "0000100";

when "1010" => letters <= "0001000";

when "1011" => letters <= "1100000";

when "1100" => letters <= "0110001";

when "1101" => letters <= "1000010";

when "1110" => letters <= "0110000";

when others => letters <= "0111000";

end case;

end process;

end Behavioral;

Result:

