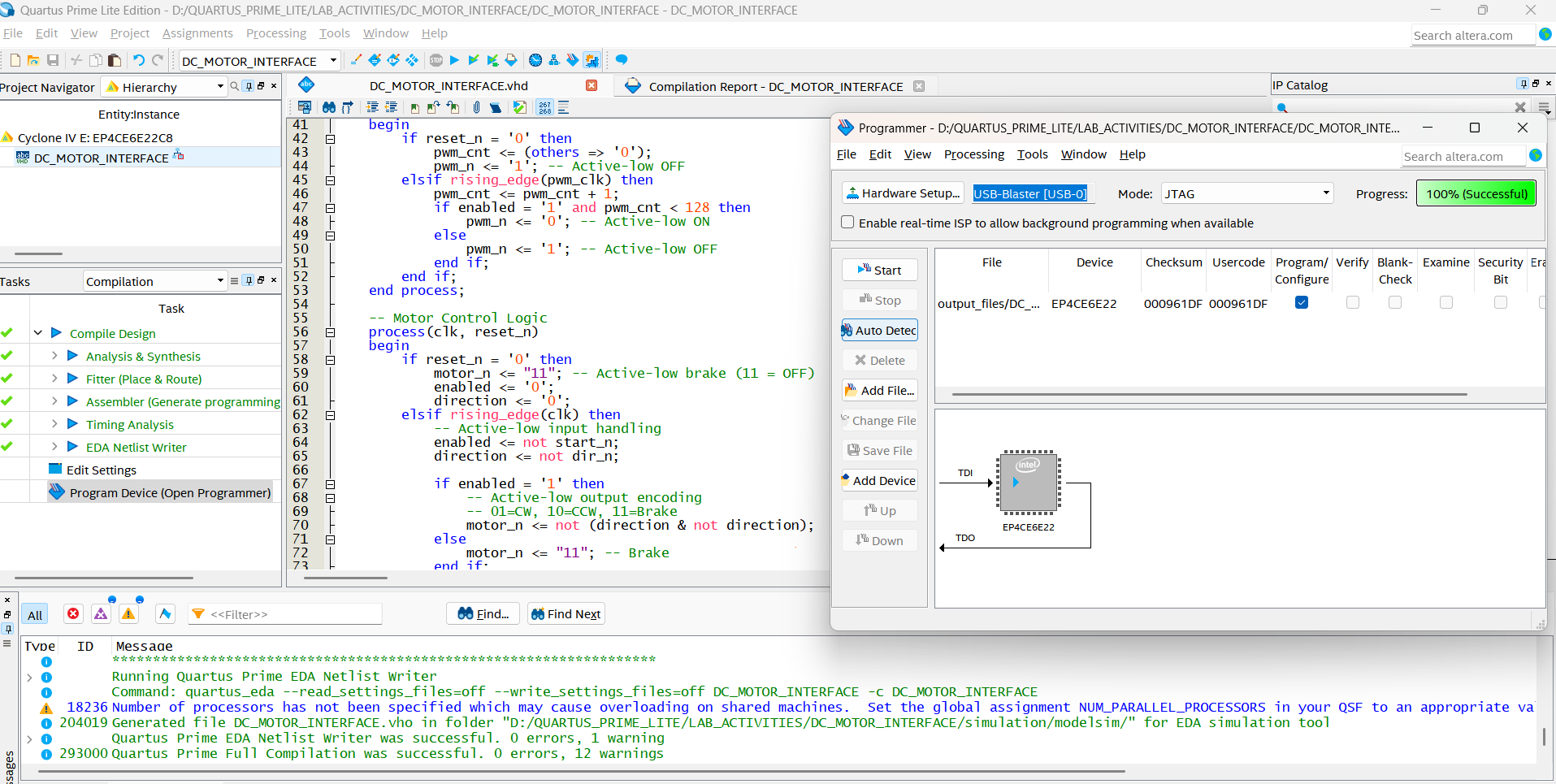
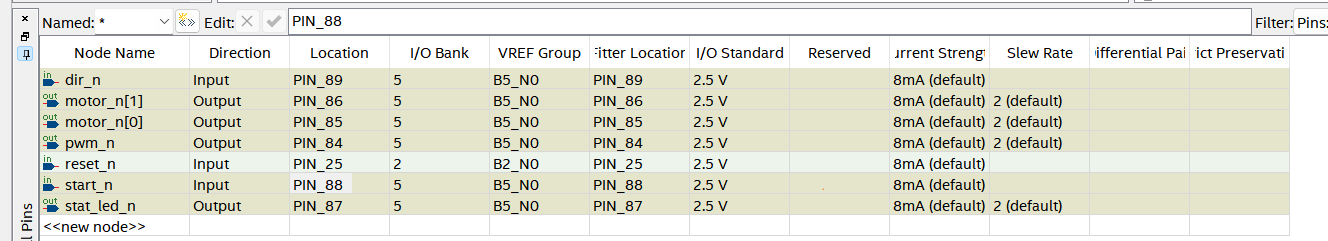
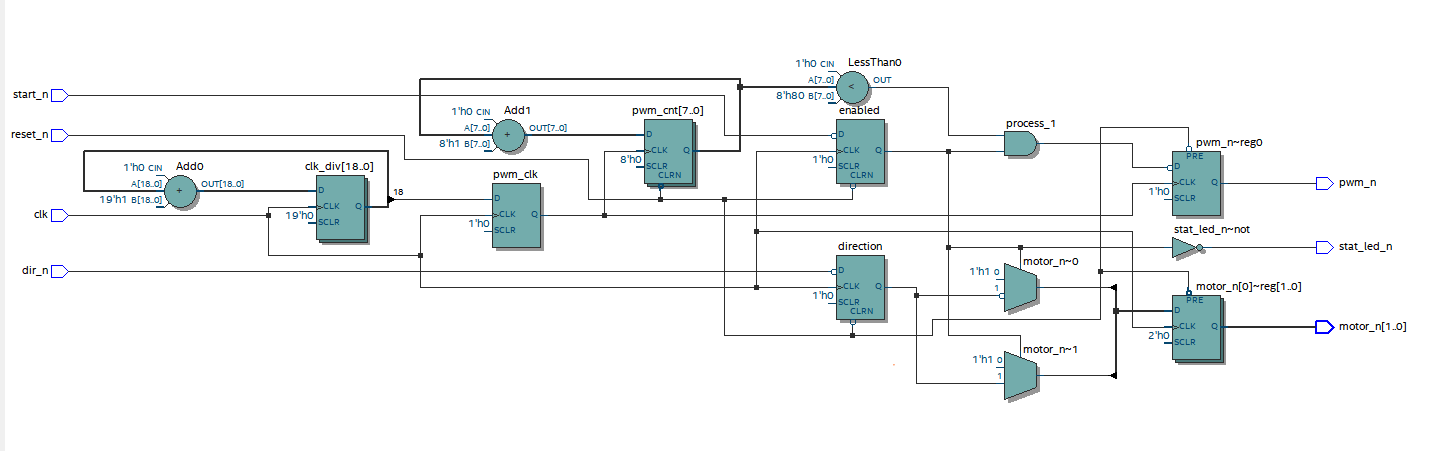
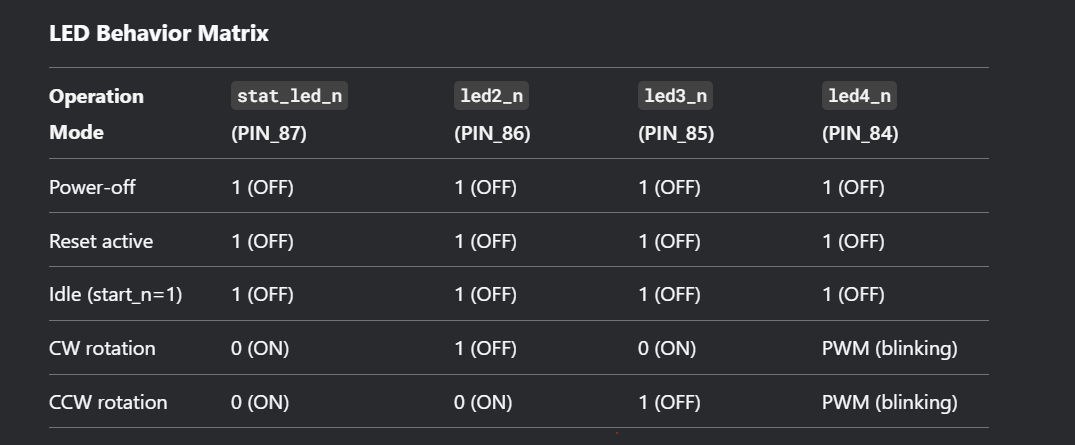
Name: Richard Raymond J. Canda

C.Y.S.: BSCpE - 3A









**DC MOTOR INTERFACE**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity DC\_MOTOR\_INTERFACE is

Port (

-- Active-low control inputs

clk : in STD\_LOGIC; -- PIN\_23 (50MHz)

reset\_n : in STD\_LOGIC; -- PIN\_25 (RESET button)

start\_n : in STD\_LOGIC; -- PIN\_88 (KEY1)

dir\_n : in STD\_LOGIC; -- PIN\_89 (KEY2)

-- Active-low outputs

pwm\_n : out STD\_LOGIC; -- PIN\_84 (led4)

motor\_n : out STD\_LOGIC\_VECTOR(1 downto 0); -- PIN\_85,86 (led3,led2)

-- Active-low status LED

stat\_led\_n : out STD\_LOGIC -- PIN\_87 (led1)

);

end DC\_MOTOR\_INTERFACE;

architecture Behavioral of DC\_MOTOR\_INTERFACE is

signal pwm\_cnt : unsigned(7 downto 0) := (others => '0');

signal clk\_div : unsigned(18 downto 0) := (others => '0');

signal pwm\_clk : STD\_LOGIC := '0';

signal enabled : STD\_LOGIC := '0';

signal direction : STD\_LOGIC := '0';

begin

-- Clock divider (50MHz → 95Hz)

process(clk)

begin

if rising\_edge(clk) then

clk\_div <= clk\_div + 1;

pwm\_clk <= clk\_div(18);

end if;

end process;

-- PWM Generator (active-low output)

process(pwm\_clk, reset\_n)

begin

if reset\_n = '0' then

pwm\_cnt <= (others => '0');

pwm\_n <= '1'; -- Active-low OFF

elsif rising\_edge(pwm\_clk) then

pwm\_cnt <= pwm\_cnt + 1;

if enabled = '1' and pwm\_cnt < 128 then

pwm\_n <= '0'; -- Active-low ON

else

pwm\_n <= '1'; -- Active-low OFF

end if;

end if;

end process;

-- Motor Control Logic

process(clk, reset\_n)

begin

if reset\_n = '0' then

motor\_n <= "11"; -- Active-low brake (11 = OFF)

enabled <= '0';

direction <= '0';

elsif rising\_edge(clk) then

-- Active-low input handling

enabled <= not start\_n;

direction <= not dir\_n;

if enabled = '1' then

-- Active-low output encoding

-- 01=CW, 10=CCW, 11=Brake

motor\_n <= not (direction & not direction);

else

motor\_n <= "11"; -- Brake

end if;

end if;

end process;

-- Active-low status LED (ON when enabled)

stat\_led\_n <= not enabled;

end Behavioral;