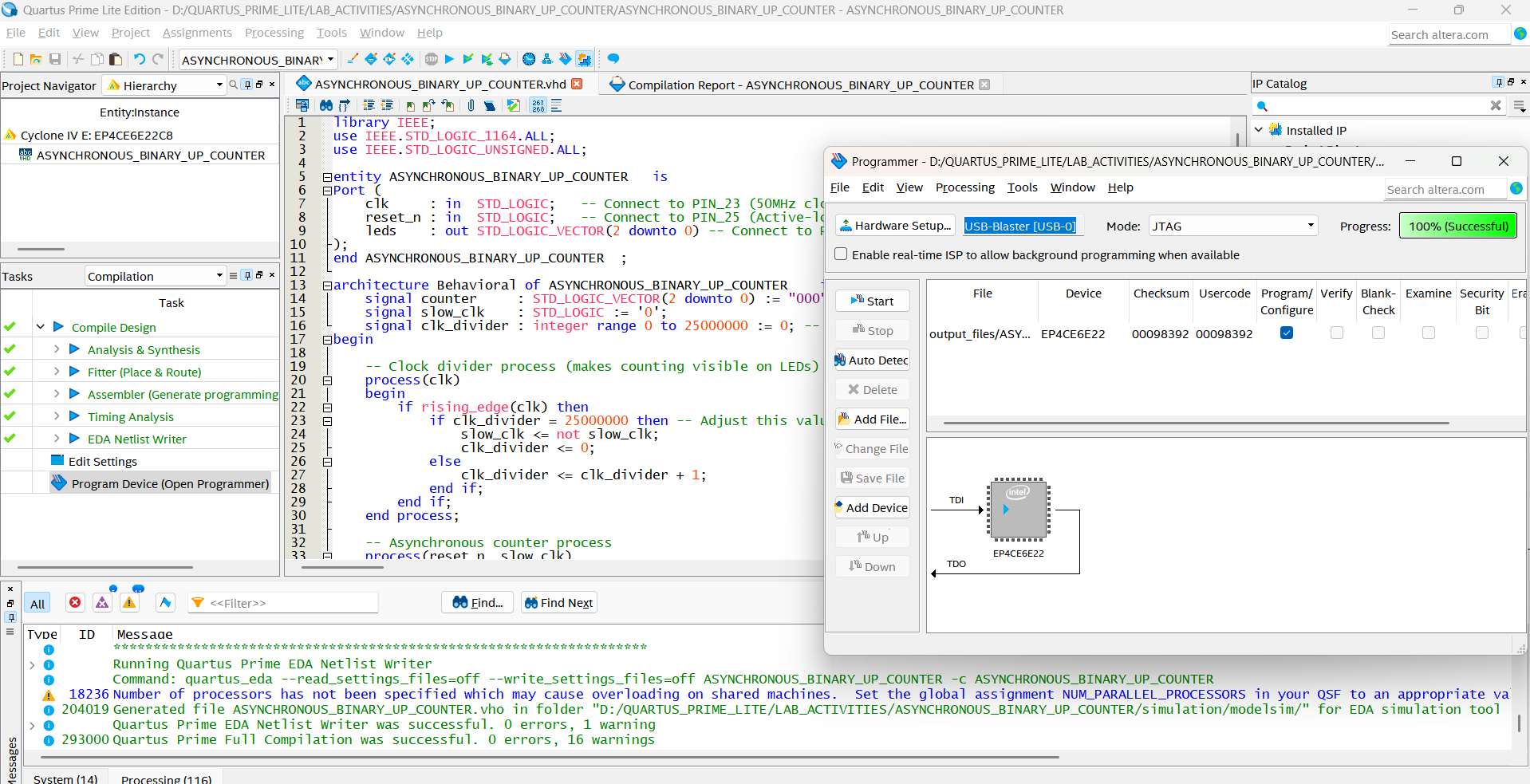
Name: Richard Raymond J. Canda

C.Y.S.: BSCpE - 3A



**ASYNCHRONOUS BINARY UP COUNTER**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity ASYNCHRONOUS\_BINARY\_UP\_COUNTER is

Port (

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

reset\_n : in STD\_LOGIC; -- Connect to PIN\_25 (Active-low reset button)

leds : out STD\_LOGIC\_VECTOR(2 downto 0) -- Connect to Pins 87,86,85

);

end ASYNCHRONOUS\_BINARY\_UP\_COUNTER ;

architecture Behavioral of ASYNCHRONOUS\_BINARY\_UP\_COUNTER is

signal counter : STD\_LOGIC\_VECTOR(2 downto 0) := "000";

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

signal clk\_divider : integer range 0 to 25000000 := 0; -- For 1Hz @ 50MHz

begin

-- Clock divider process (makes counting visible on LEDs)

process(clk)

begin

if rising\_edge(clk) then

if clk\_divider = 25000000 then -- Adjust this value for speed

slow\_clk <= not slow\_clk;

clk\_divider <= 0;

else

clk\_divider <= clk\_divider + 1;

end if;

end if;

end process;

-- Asynchronous counter process

process(reset\_n, slow\_clk)

begin

if reset\_n = '0' then -- Active-low reset

counter <= "000";

else

-- First stage (LSB)

if rising\_edge(slow\_clk) then

counter(0) <= not counter(0);

end if;

-- Second stage

if falling\_edge(counter(0)) then

counter(1) <= not counter(1);

end if;

-- Third stage (MSB)

if falling\_edge(counter(1)) then

counter(2) <= not counter(2);

end if;

end if;

end process;

-- Active-high LED outputs (invert if your board needs active-low)

leds <= counter;

end Behavioral;

