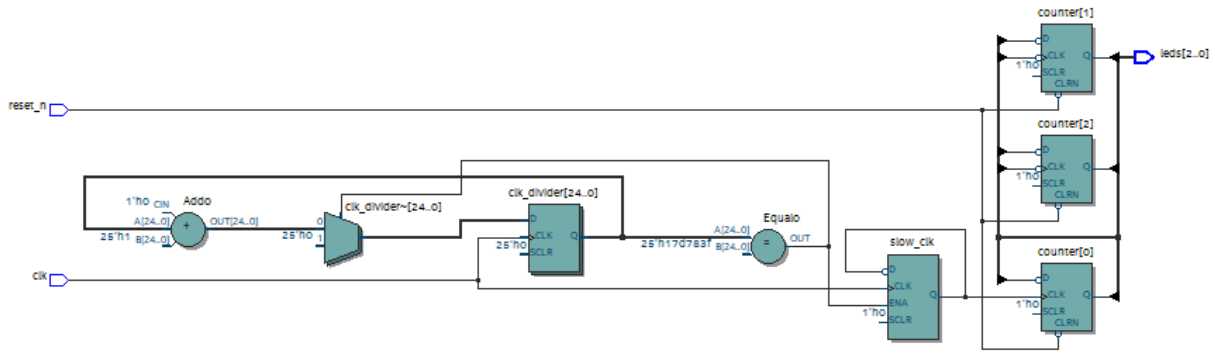
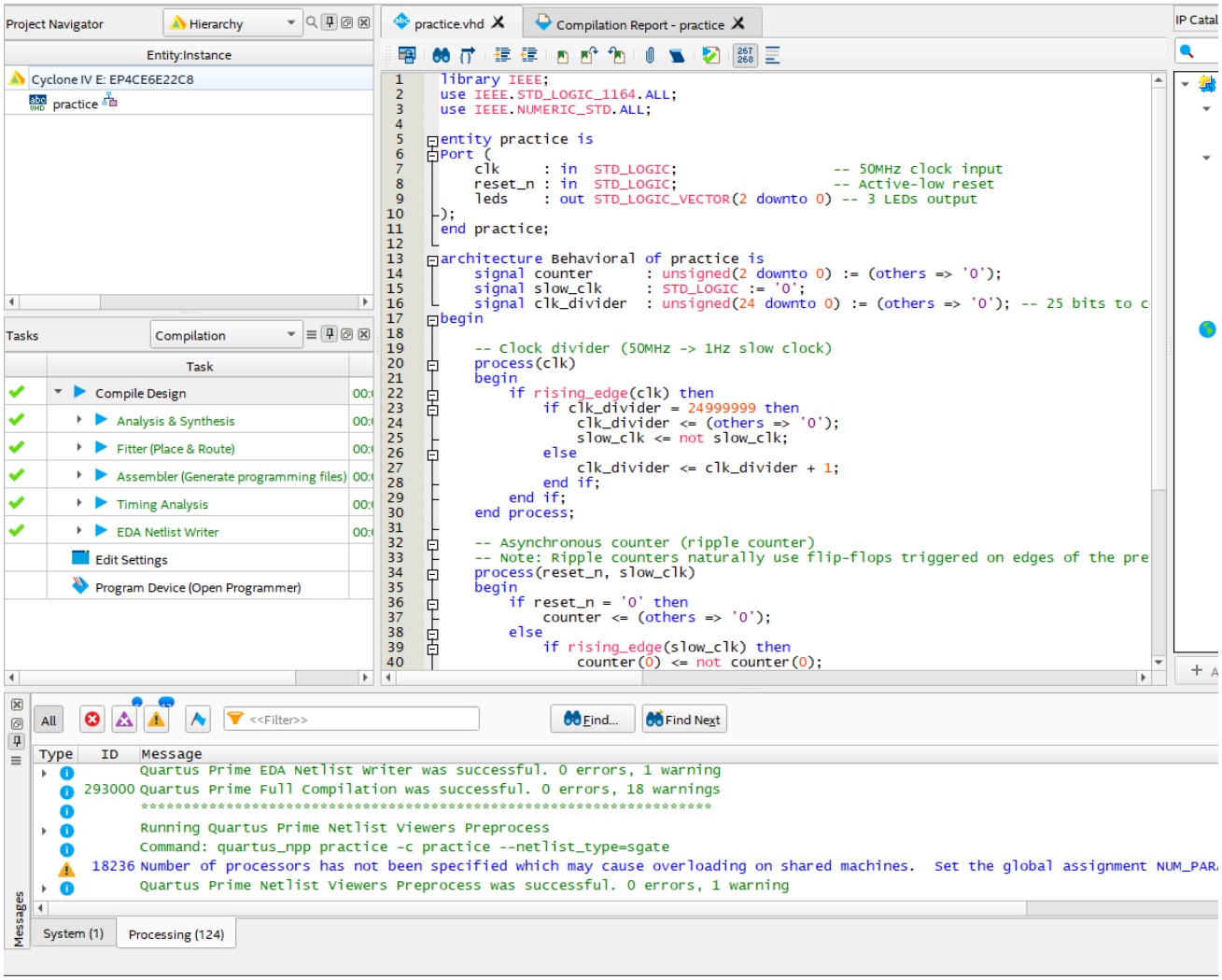
EMMAN ACE G MENION
BSCPE 3A

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;
```





14. ASYNCHRONOUS BINARY UP COUNTER:

3-bit Asynchronous up counter			
Clock	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0
9	0	0	1