PACO, ARMIN R. BSCpE – 3A

```
DC MOTOR INTERFACE
Quartus Prime Lite Edition - C:/Users/pacoa/OneDrive/Desktop/HDL Files/DC_MOTOR_INTERFACE/DC_MOTOR_INTERFACE - DC_MOTOR_INTERFACE
 File Edit View Project Assignments Processing Tools Window Help
 oject Navigator A Hierarchy 🔻 🔾 📮 🗗 🗴 💠
                                                    DC_MOTOR_INTERFACE.vhd

DC_MOTOR_INTERFACE.vhd

DC_MOTOR_INTERFACE.vhd

DC_MOTOR_INTERFACE.vhd

DC_MOTOR_INTERFACE.vhd

DC_MOTOR_INTERFACE.vhd
                             Entity:Instance
     Cyclone V: 5CGXFC7C7F23C8
                                                           library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
   DC_MOTOR_INTERFACE <sup>™</sup>
                                                           □entity DC_MOTOR_INTERFACE is □Port (
                                                                  t (
- Active-low control inputs
clk : in STD_LOGIC; -- PIN_23 (50MHz)
reset_n : in STD_LOGIC; -- PIN_25 (RESET
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;
                                                               rchitecture Behavioral of DC_MOTOR_INTERFACE is signal pom.cnt unsigned(7 downto 0) := (others => '0'); signal pom.clk signal pom.clk signal enabled signal enabled stop.locic := '0'; signal enabled sig
                                                                  -- Clock divider (50MHz - 95Hz)
process(clk)
66 Find... 66 Find Next
CODE:
library IEEE;
use IEEE.STD LOGIC 1164.ALL;
use IEEE.NUMERIC STD.ALL;
entity DC MOTOR INTERFACE
                                                                                                                  is
Port (
       -- Active-low control inputs
                        : 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 \Rightarrow '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 \rightarrow 95Hz)
      process(clk)
      begin
             if rising edge(clk) then
```

```
clk div \le clk div + 1;
       pwm clk \le 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;</pre>
       if enabled = '1' and pwm cnt < 128 then
         pwm n <= '0'; -- Active-low ON
         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 \le "11"; -- Active-low brake (11 = OFF)
       enabled \leq 10';
       direction <= '0';
    elsif rising edge(clk) then
       -- Active-low input handling
       enabled <= not start n;</pre>
       direction <= not dir n;
       if enabled = '1' then
         -- Active-low output encoding
         -- 01=CW, 10=CCW, 11=Brake
         motor n \le n (direction & not direction);
       else
         motor n <= "11"; -- Brake
    end if;
  end process;
  -- Active-low status LED (ON when enabled)
  stat led n <= not enabled;
end Behavioral;
```