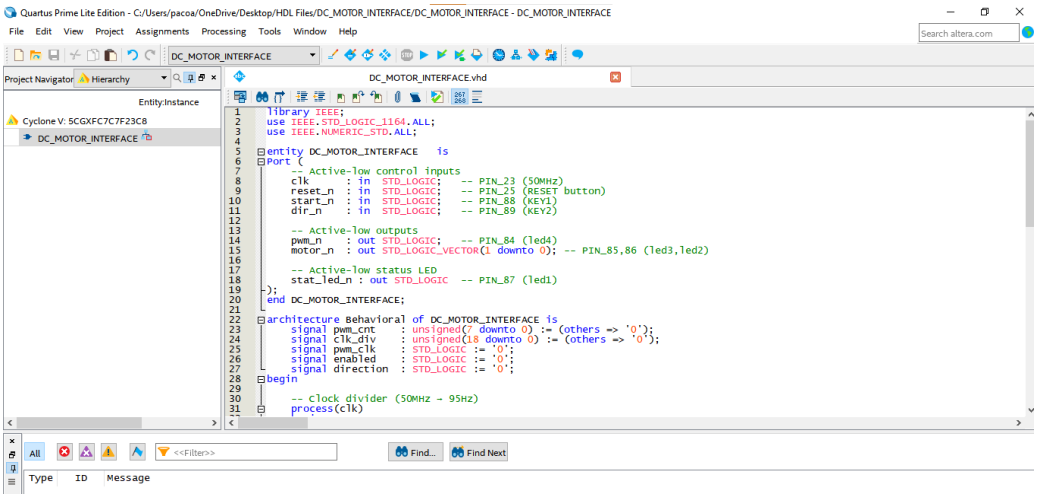
PACO, ARMIN R.
BSCpE – 3A

DC MOTOR INTERFACE



CODE:

```
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;
```