

EMMAN ACE G. MENION
BSCPE 3A

19. VHDL CODE FOR STEPPER MOTOR INTERFACE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity practice is
Port (
    -- Active-low control inputs
    clk      : in  STD_LOGIC; -- PIN_23 (50MHz)
    reset_n  : in  STD_LOGIC; -- PIN_25 (RESET button)
    enable_n : in  STD_LOGIC; -- PIN_88 (KEY1)
    dir_n    : in  STD_LOGIC; -- PIN_89 (KEY2)

    -- Active-low outputs (ULN2003 driver compatible)
    coil_n   : out STD_LOGIC_VECTOR(3 downto 0) -- PIN_84-87 (led4-led1)
);
end practice ;

architecture Behavioral of practice is
    signal step_counter : unsigned(1 downto 0) := "00";
    signal clk_div      : unsigned(20 downto 0) := (others => '0');
    signal step_clk      : STD_LOGIC := '0';
    signal enabled       : STD_LOGIC := '0';
    signal direction     : STD_LOGIC := '0';

    -- Full-step sequence (active-low)
    type step_sequence is array (0 to 3) of std_logic_vector(3 downto 0);
    constant full_step : step_sequence := (
        "1100", -- Phase A (0x8)
        "0110", -- Phase B (0x4)
        "0011", -- Phase C (0x2)
        "1001"  -- Phase D (0x1)
    );
begin

    -- Clock divider (50MHz → ~12Hz stepping)
    process(clk)
    begin
        if rising_edge(clk) then
            clk_div <= clk_div + 1;
            step_clk <= clk_div(20); -- 50MHz/2^21 ≈ 12Hz
        end if;
    end process;

    -- Stepping control
    process(step_clk, reset_n)
    begin
        if reset_n = '0' then
            step_counter <= "00";
        end if;
    end process;
end Behavioral;
```

```

coil_n <= "1111"; -- All coils OFF (active-low)
elsif rising_edge(step_clk) then
  if enabled = '1' then
    if direction = '1' then
      step_counter <= step_counter + 1; -- CW
    else
      step_counter <= step_counter - 1; -- CCW
    end if;

    -- Output current step phase (active-low)
    coil_n <= full_step(to_integer(step_counter));
  else
    coil_n <= "1111"; -- Disable all coils
  end if;
end if;

-- Control signal processing (active-low to active-high)
enabled <= not enable_n;
direction <= not dir_n;

```

```
end Behavioral;
```

