Joshua S. Rebuta

BSCpE 3A

**Stepper Motor Interface**

**VHDL Code:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity stm\_st 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 stm\_st;

architecture Behavioral of stm\_st 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";

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;

end process;

-- Control signal processing (active-low to active-high)

enabled <= not enable\_n;

direction <= not dir\_n;

end Behavioral;