library IEEE;
use IEEE.STD\_LOGIC\_1164.ALL;
use IEEE.STD\_LOGIC\_ARITH.ALL;
use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity Agricultural\_Robot is
    Port (
        clk : in std\_logic;
        reset : in std\_logic;
        start : in std\_logic;
        seed\_type : in std\_logic\_vector(1 downto 0);
        temp\_sensor : in integer range 0 to 50;
        light\_sensor : in integer range 0 to 1000;
        soil\_moisture : in integer range 0 to 100;
        battery\_level : in integer range 0 to 100;
        irrigation\_done : in std\_logic;
        energy\_harvesting\_done : in std\_logic;
        season : in std\_logic\_vector(1 downto 0);
        obstacle\_detected : in std\_logic; -- شناسایی مانع
        move\_forward : out std\_logic;
        move\_backward : out std\_logic;
        move\_left : out std\_logic;
        move\_right : out std\_logic;
        alert\_robots : out std\_logic;
        heating : out std\_logic;
        cooling : out std\_logic;
        lighting : out std\_logic;
        pump : out std\_logic;
        done : out std\_logic;
        current\_state : out std\_logic\_vector(3 downto 0);
        depth\_setting : out integer range 0 to 10;
        spacing\_setting : out integer range 0 to 100
    );
end Agricultural\_Robot;

architecture Behavioral of Agricultural\_Robot is

    type state\_type is (
        IDLE, INIT\_MEMORY, PLANTING, IRRIGATION, DRAINAGE, TEMP\_CTRL, LIGHT\_CTRL, HARVESTING, ENERGY\_HARVESTING, MOVING
    );
    signal state, next\_state : state\_type := IDLE;

signal depth\_memory : integer := 0;
    signal spacing\_memory : integer := 0;

    constant seed\_depth\_small : integer := 1;
    constant seed\_depth\_medium : integer := 2;
    constant seed\_depth\_large : integer := 3;

    constant seed\_spacing\_small : integer := 15;
    constant seed\_spacing\_medium : integer := 40;
    constant seed\_spacing\_large : integer := 75;

    constant moisture\_threshold\_low : integer := 30;
    constant moisture\_threshold\_high : integer := 80;

    constant spring\_temp\_low : integer := 15;
    constant spring\_temp\_high : integer := 25;
    constant summer\_temp\_low : integer := 20;
    constant summer\_temp\_high : integer := 35;
    constant autumn\_temp\_low : integer := 10;
    constant autumn\_temp\_high : integer := 20;
    constant winter\_temp\_low : integer := 0;
    constant winter\_temp\_high : integer := 10;

    constant spring\_light\_low : integer := 300;
    constant spring\_light\_high : integer := 700;
    constant summer\_light\_low : integer := 500;
    constant summer\_light\_high : integer := 900;
    constant autumn\_light\_low : integer := 200;
    constant autumn\_light\_high : integer := 600;
    constant winter\_light\_low : integer := 100;
    constant winter\_light\_high : integer := 500;

    signal alternate\_direction : std\_logic := '0'; -- جهت جایگزین برای حرکت (چپ یا راست)

begin

    -- تنظیم حافظه اولیه
    process (clk, reset)
    begin
        if reset = '1' then
            depth\_memory <= 0;
            spacing\_memory <= 0;
        elsif rising\_edge(clk) then
            if state = INIT\_MEMORY then
                case seed\_type is
                    when "00" =>
                        depth\_memory <= seed\_depth\_small;
                        spacing\_memory <= seed\_spacing\_small;
                    when "01" =>
                        depth\_memory <= seed\_depth\_medium;
                        spacing\_memory <= seed\_spacing\_medium;
                    when others =>
                        depth\_memory <= seed\_depth\_large;
                        spacing\_memory <= seed\_spacing\_large;
                end case;
            end if;
        end if;
    end proce

* مدیریت انتقال وضعیت

 Process (clk, reset)

 Begin

 If reset = '1' then

 State <= IDLE;

 Elsif rising\_edge(clk) then

 State <= next\_state;

 End if;

 End process;

* منطق تعیین وضعیت بعدی

 Process (state, start, temp\_sensor, light\_sensor, soil\_moisture, irrigation\_done, energy\_harvesting\_done, obstacle\_detected)

 Begin

 Case state is

 When IDLE =>

 If start = '1' then

 Next\_state <= INIT\_MEMORY;

 Else

 Next\_state <= IDLE;

 End if;

 When INIT\_MEMORY =>

 Next\_state <= PLANTING;

 When PLANTING =>

 Next\_state <= IRRIGATION;

 When IRRIGATION =>

 If irrigation\_done = '1' then

 If soil\_moisture < moisture\_threshold\_low then

 Next\_state <= DRAINAGE;

 Else

 Next\_state <= TEMP\_CTRL;

 End if;

 Else

 Next\_state <= IRRIGATION;

 End if;

 When DRAINAGE =>

 If soil\_moisture > moisture\_threshold\_high then

 Next\_state <= TEMP\_CTRL;

 Else

 Next\_state <= DRAINAGE;

 End if;

 When TEMP\_CTRL =>

 Next\_state <= LIGHT\_CTRL;

 When LIGHT\_CTRL =>

 Next\_state <= HARVESTING;

 When HARVESTING =>

 Next\_state <= ENERGY\_HARVESTING;

 When ENERGY\_HARVESTING =>

 If energy\_harvesting\_done = '1' then

 Next\_state <= MOVING;

 Else

 Next\_state <= ENERGY\_HARVESTING;

 End if;

 When MOVING =>

 If obstacle\_detected = '1' then

 Next\_state <= MOVING; -- ادامه تغییر جهت

 Else

 Next\_state <= IDLE;

 End if;

 When others =>

 Next\_state <= IDLE;

 End case;

 End process;

* خروجی‌ها و منطق حرکت

 Process (state, obstacle\_detected, alternate\_direction)

 Begin

 Heating <= '0';

 Cooling <= '0';

 Lighting <= '0';

 Pump <= '0';

 Done <= '0';

 Move\_forward <= '0';

 Move\_backward <= '0';

 Move\_left <= '0';

 Move\_right <= '0';

 Depth\_setting <= 0;

 Spacing\_setting <= 0;

 Current\_state <= "0000";

 Case state is

 When INIT\_MEMORY =>

 Depth\_setting <= depth\_memory;

 Spacing\_setting <= spacing\_memory;

 Current\_state <= "0001";

 When TEMP\_CTRL =>

 Current\_state <= "0010";

 When LIGHT\_CTRL =>

 Current\_state <= "0011";

 When DRAINAGE =>

 If soil\_moisture > moisture\_threshold\_high then

 Pump <= '1';

 End if;

 Current\_state <= "0100";

 When ENERGY\_HARVESTING =>

 If energy\_harvesting\_done = '1' then

 Done <= '1';

 End if;

 Current\_state <= "0101";

 When MOVING =>

 If obstacle\_detected = '1' then

 Move\_forward <= '0';

 Move\_backward <= '1'; -- حرکت به عقب

 If alternate\_direction = '0' then

 Move\_left <= '1'; -- حرکت به چپ

 Alternate\_direction <= '1'; -- تغییر جهت به راست برای بار بعد

 Else

 Move\_right <= '1'; -- حرکت به راست

 Alternate\_direction <= '0'; -- تغییر جهت به چپ برای بار بعد

 End if;

 Else

 Move\_forward <= '1'; -- حرکت به جلو

 End if;

 Current\_state <= "0110";

 When others =>

 Current\_state <= "1111";

 End case;

 End process;

End Behavioral;