```vhdl
-- ROBOT.VHD --

library ieee;
use ieee.std_logic_1164.all;

entity Robot is
    port(reset, clk, athome, findfood, lostfood, closetofood,
    success, aboverestth, abovesearchth, scantimeup: in std_logic;
    rest, search, food: out std_logic);
end Robot;

architecture automate_robot of Robot is

    type States is (IDLE, RESTING, RANDOMWALK, SCANAREA, HOMING, MOVETOFOOD,
    MOVETOHOME, DEPOSIT, GRABFOOD) ;
    Signal state, nextstate : States := IDLE;

begin
    -- Calcul de l'état suivant
    -- Comme on est en std_logic,"elsif ='0'" et non "else", car le signal peux
    avoir d'autre valeur
    process (state, athome, findfood, lostfood, closetofood, success, aboverestth,
    abovesearchth, scantimeup)
    begin
        case state is
            when IDLE => nextstate <= RESTING;
            when RESTING =>
                if aboverestth = '1' then nextstate <= RANDOMWALK;
                elsif aboverestth = '0' then nextstate <= RESTING;
                end if;

            when RANDOMWALK =>
                if abovesearchth = '1' then nextstate <= HOMING;
                elsif  abovesearchth = '0' then
                    if findfood = '1' then nextstate <= MOVETOFOOD;
                    elsif findfood = '0' then
                        nextstate <= RANDOMWALK;
                    end if;
                end if;

            when SCANAREA =>
                if abovesearchth = '1' then nextstate <= HOMING;
                elsif  abovesearchth = '0' then
                    if findfood = '1' then nextstate <= MOVETOFOOD;
                    elsif findfood = '0' then
                        if scantimeup = '1' then nextstate <= RANDOMWALK;
                        elsif scantimeup = '0' then nextstate <= SCANAREA;
                        end if;
                    end if;
                end if;
            when HOMING => nextstate <= RESTING;
            when MOVETOFOOD =>
                if abovesearchth = '1' then nextstate <= HOMING;
                elsif  abovesearchth = '0' then
                    if lostfood = '1' then nextstate <= SCANAREA;
                    elsif lostfood = '0' then
                        if closetofood = '1' then nextstate <= GRABFOOD;
                        elsif closetofood = '0' then
                            nextstate <= MOVETOFOOD;
                        end if;
                    end if;
                end if;
            when GRABFOOD =>
                if success = '1' then nextstate <= MOVETOHOME;
                elsif success = '0' then nextstate <= GRABFOOD;
                end if;
            when MOVETOHOME =>
                if athome = '1' then nextstate <= DEPOSIT;
                elsif athome = '0' then nextstate <= MOVETOFOOD;
                end if;
            when DEPOSIT =>
                if success = '1' then nextstate <= RESTING;
                elsif success = '0' then nextstate <= DEPOSIT;
```

```
71                  end if;
72          end case;
73      end process;
74
75      -- MISE A JOUR DU REGISTRE D'ETAT
76
77      process(reset, clk)
78      begin
79          -- RESET : asynchrone haut
80          if reset = '1' then state <= IDLE;
81          -- HORLOGE : front montant
82          elsif (clk'event and clk = '1') then
83              state <= nextstate;
84          end if;
85      end process;
86
87
88      -- MISE A JOUR DES OUTPUTS
89      rest <= '1' when (( state = DEPOSIT and success = '1' ) OR (state = IDLE) OR
        (state = HOMING and athome = '1') ) else '0';
90      search <= '1' when (state = RESTING and aboverestth = '1' ) else '0';
91      food <= '1' when  (state = MOVETOFOOD and abovesearchth = '0' and lostfood = '0'
        and closetofood ='1') else '0';
92
93
94
95  end automate_robot;
96
97
```