**VERILOG CODE FOR TRAFFIC LIGHT CONTROLLER**

**Aim:** Verilog code for a traffic light controller that helps to clear traffic at highway and farmway junction. There is a sensor on the farm way which detects the vehicles and gives a signal which gives required signals for the farm vehicles to go and when there are no vehicles it changes back the traffic lights to green for highway and red for farmway.

Source code:

**module traffic\_light\_controller (**

**input wire clk, // clock signal (50 MHz)**

**input wire rst\_n, // active-low reset**

**input wire sensor, // sensor input**

**output reg [2:0] highway\_light,**

**output reg [2:0] farm\_light**

**);**

**typedef enum logic [1:0] {**

**HGRE\_FRED = 2'b00, // Highway green, farm red**

**HYEL\_FRED = 2'b01, // Highway yellow, farm red**

**HRED\_FGRE = 2'b10, // Highway red, farm green**

**HRED\_FYEL = 2'b11 // Highway red, farm yellow**

**} state\_t;**

**state\_t state, next\_state;**

**localparam integer YELLOW\_TIME = 150\_000\_000; // 3 seconds**

**reg [27:0] timer;**

**always @(posedge clk or negedge rst\_n) begin**

**if (!rst\_n) begin**

**state <= HGRE\_FRED;**

**timer <= 0;**

**end**

**else**

**begin**

**if (timer > 0) begin**

**timer <= timer - 1;**

**end**

**else**

**begin**

**state <= next\_state;**

**if (next\_state == HYEL\_FRED || next\_state == HRED\_FYEL) begin**

**timer <= YELLOW\_TIME;**

**end**

**else**

**begin**

**timer <= 0;**

**end**

**end**

**end**

**end**

**always @(\*)**

**begin**

**next\_state = state;**

**case (state)**

**HGRE\_FRED:**

**begin**

**highway\_light = 3'b001; // Green**

**farm\_light = 3'b100; // Red**

**if (sensor)**

**begin**

**next\_state = HYEL\_FRED;**

**end**

**end**

**HYEL\_FRED:**

**begin**

**highway\_light = 3'b010; // Yellow**

**farm\_light = 3'b100; // Red**

**if (timer == 0)**

**begin**

**next\_state = HRED\_FGRE;**

**end**

**end**

**HRED\_FGRE:**

**begin**

**highway\_light = 3'b100; // Red**

**farm\_light = 3'b001; // Green**

**if (!sensor)**

**begin**

**next\_state = HRED\_FYEL;**

**end**

**end**

**HRED\_FYEL:**

**begin**

**highway\_light = 3'b100; // Red**

**farm\_light = 3'b010; // Yellow**

**if (timer == 0)**

**begin**

**next\_state = HGRE\_FRED;**

**end**

**end**

**endcase**

**end**

**endmodule**

**Test bench:**

**module traffic\_light\_controller\_tb;**

**localparam integer CLOCK\_PERIOD = 20; // Clock period in ns (50 MHz)**

**localparam integer SIMULATION\_TIME = 500\_000\_000;**

**reg clk, rst\_n, sensor;**

**wire [2:0] highway\_light, farm\_light;**

**traffic\_light\_controller uut (.clk(clk), .rst\_n(rst\_n), .sensor(sensor), .highway\_light(highway\_light), .farm\_light(farm\_light));**

**always # (CLOCK\_PERIOD / 2) clk = ~clk;**

**initial begin**

**clk = 0; rst\_n = 0; sensor = 0;**

**# (CLOCK\_PERIOD \* 5) rst\_n = 1; // Reset**

**# (CLOCK\_PERIOD \* 10) sensor = 1; // vehicle detection**

**# (CLOCK\_PERIOD \* 300\_000\_000) sensor = 0; // No vehicle detected after some time**

**# (SIMULATION\_TIME - (CLOCK\_PERIOD \* 10 + CLOCK\_PERIOD \* 300\_000\_000)) $stop; // Stop simulation**

**end**

**initial begin**

**$monitor("Time: %d, Highway Light: %b, Farm Light: %b, Sensor: %b", $time, highway\_light, farm\_light, sensor); // Monitor outputs**

**end**

**endmodule**