From SystemVerilog Assertions Handbook 4th Edition, Ben Cohen

SystemVerilog Assertions for traffic light controller

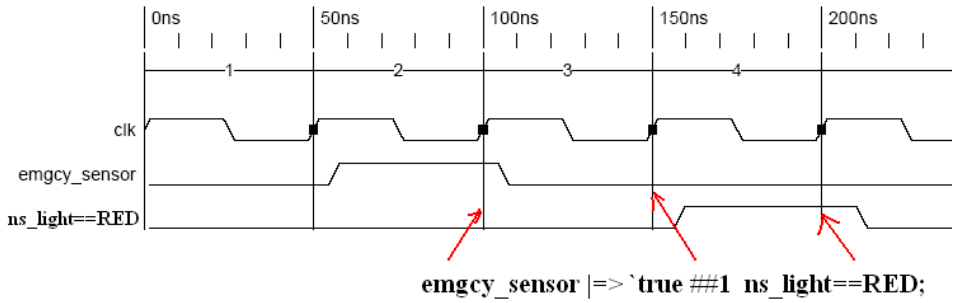
Based upon the requirements, several assertions can be expressed prior to writing the RTL. Below (in file *tlight/tlight_props1.sv*[1]) are some example assertions written for this design. Note that in order to later bind this checker module to the actual design, all its ports are declared as inputs. Its ports are all inputs and outputs of the actual design, plus some internal states needed to model the requirements. The internal states are defined in compilation unit scope as enumerations so they can be easily shared in the RTL. In this case we choose to include the internal timer *ns_green_timer* in the port list.

```
`define true 1
`ifndef MULTIPLE_FILE_COMPILE
 typedef enum {OFF, RED, YELLOW, GREEN, PRE_GREEN} lights_t;  `endif
module tlight_props  ( // or checker
  input  lights_t ns_light,  // North/South light status, Main road
  input  lights_t ew_light,  // East/West light status
  input    ew_sensor, // East/West sensor for new car
  input            emgcy_sensor, // emergency sensor
  input            reset_n,   // synchronous reset
  input            clk,       // master clock
  input [1:0]  ns_green_timer
  );
 parameter FAIL   = 1'b0;

 // ***************************************************
 // Safety property
 property Never_NS_EW_ALL_GREEN;
    disable iff (!reset_n)
   not (ns_light==GREEN && ew_light==GREEN);
 endproperty : Never_NS_EW_ALL_GREEN
 Never_NS_EW_ALL_GREEN_1 : assert property(@ (posedge clk) Never_NS_EW_ALL_GREEN);
 // ***************************************************
 // State of lights at reset
 property nsLightAtReset;
   // disable iff (!reset_n)  // <-- this causes the assertion to always be vacuous
   reset_n==1'b0 |=> ns_light==OFF;
 endproperty : nsLightAtReset
 nsLightAtReset_1 : assert property(@ (posedge clk) nsLightAtReset);
 //
 property ewLightAtReset;
   // disable iff (!reset_n) // <-- this causes the assertion to always be vacuous
  reset_n==1'b0 |=> ew_light==OFF;  // RED???
  endproperty : ewLightAtReset
 ewLightAtReset_1 : assert property(@ (posedge clk) ewLightAtReset);
 // ***************************************************
 // State of lights during emergency
 // Lights switch from GREEN to YELLOW to RED
 property NsLightsWhenEmergency;
   disable iff (!reset_n)
  emgcy_sensor |=> `true[*2] ##1  ns_light==RED;
  endproperty : NsLightsWhenEmergency
 NsLightsWhenEmergency_1 : assert property(@ (posedge clk) NsLightsWhenEmergency);
```

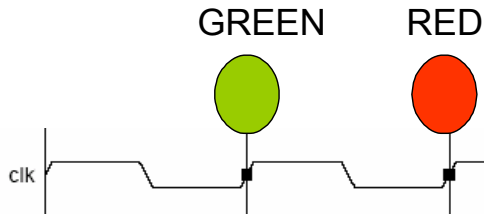
The following is preferred (see 8.3.3)
(ns_light==GREEN |-> !ew_light==GREEN)

1 Please refer to file model ***tlight/trafficlight.sv*** for the complete RTL code.




```
 property EwLightsWhenEmergency;
   disable iff (!reset_n)
   emgcy_sensor |=> `true[*2] ##1 ew_light==RED;
 endproperty : EwLightsWhenEmergency
 EwLightsWhenEmergency_1 : assert property(@ (posedge clk) EwLightsWhenEmergency);
// Safety, GREEN to RED is illegal.  Need YELLOW
 property NsNeverFromGreenToRed;
   disable iff (!reset_n)
    not(ns_light==GREEN ##1 ns_light==RED);
 endproperty : NsNeverFromGreenToRed
 NsNeverFromGreenToRed_1 : assert property(@ (posedge clk) NsNeverFromGreenToRed);
```

See 8.3.3 for guidelines. This code is best rewritten as:
ns_light==GREEN |=> ! ns_light==RED



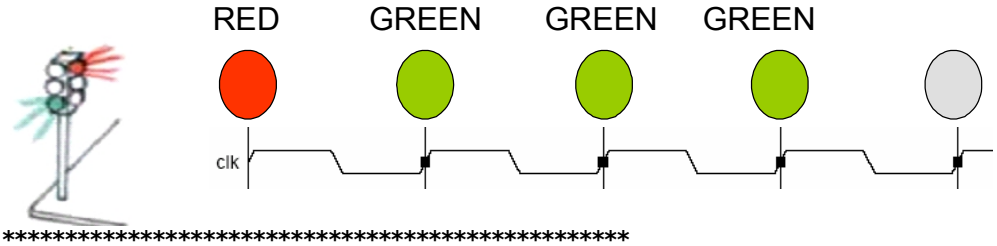
See 8.3.3 for guidelines. Rewrite as
ew_light==GREEN |=> ! ew_light==RED

```
 property EwNeverFromGreenToRed;
   disable iff (!reset_n)
    not(ew_light==GREEN ##1 ew_light==RED);
endproperty : EwNeverFromGreenToRed
 EwNeverFromGreenToRed_1 : assert property(@ (posedge clk) EwNeverFromGreenToRed);
// ************************************************
 // The NorthSouth light is the main street light.
 // If ns is green and no emergency or ew sensor, then next cycle is also GREEN
 property NsGreenNext;
   (ns_light==GREEN) && ($past(emgcy_sensor)==1'b0 && reset_n==1'b1)
            |=> ns_light==GREEN;
endproperty : NsGreenNext
NsGreenNext_1:  assert property (@ (posedge clk) NsGreenNext);
```

See 8.3.3 for guidelines. Rewrite as
(ew_light==GREEN |-> !ns_light==YELLOW) and
(ns_light==GREEN |-> !ew_light==YELLOW)

```
 // GREEN-YELLOW at the same time
 property NeverGreenYellow;
   not ((ew_light==GREEN && ns_light==YELLOW) ||
       (ns_light==GREEN && ew_light==YELLOW));
 endproperty : NeverGreenYellow
 NeverGreenYellow_1: assert property (@ (posedge clk) NeverGreenYellow);
 // ****************************************************
 // The NorthSouth light is the main street light.
```

```
// It must remain GREEN for ns_green_timer == 3 before it can switch.
// Timer ns_green_timer will count to 3, and remain at 3 until light changes.
 property NsGreenForMin3Ccyles;
  @ (posedge clk) disable iff (!reset_n || emgcy_sensor)
    $rose(ns_light==GREEN) && !$past(emgcy_sensor) |=>
          ns_light==GREEN[*2]; // abort emgcy_sensor);
 endproperty : NsGreenForMin3Ccyles
NsGreenForMin3Ccyles_1 : assert property (NsGreenForMin3Ccyles);
```




```
// ***************************************************
// East-West North-South Lights with East-West sensor
// If ew_sensor is activated (new car), then light will switch for the ew_light
// when minimum time for ns_light is satisfied.  ew_green_timer will count to 3,
// at which time, the ns_green_timer will regain control of GREEN.
 property EwNewSensorActivation;
  @ (posedge clk) disable iff (!reset_n || emgcy_sensor)
   ( (ew_sensor==1'b1) && $rose(ns_green_timer==2'b11 )) &&
                !$past(emgcy_sensor) && ns_light!= RED
                    |=>   ns_light==YELLOW ##1 ew_light==GREEN;
 endproperty : EwNewSensorActivation
 EwNewSensorActivation_1 : assert property (EwNewSensorActivation);
// End of new properties 09/10/09
 endmodule : tlight_props

bind trafficlight tlight_props tlight_props1 (.*);
```