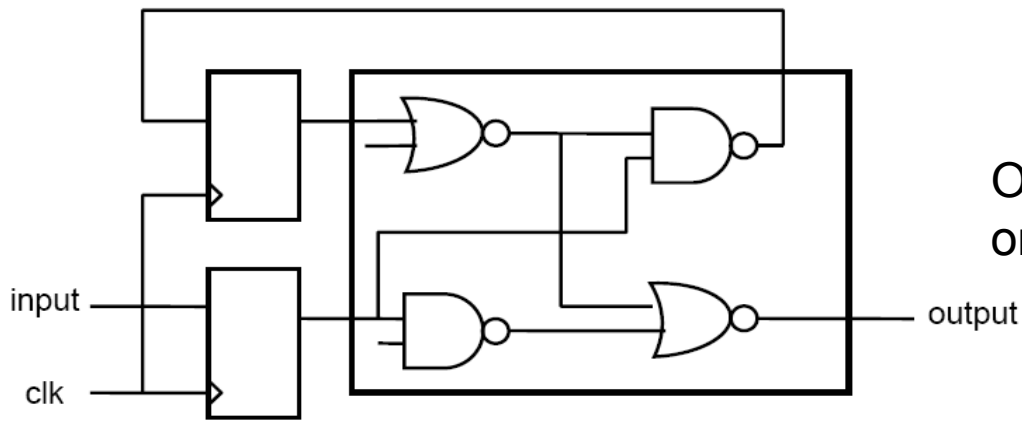
Finite State Machines (FSM)

- *Finite State Machines:*
  Outputs are Function of State (and Inputs)
  Next States are Functions of State and Inputs
  Used to implement circuits that control other circuits
  "Decision Making" logic

- *Application of Sequential Logic Design Techniques*
  Word Problems
  Mapping into formal representations of FSM behavior
  Case Studies

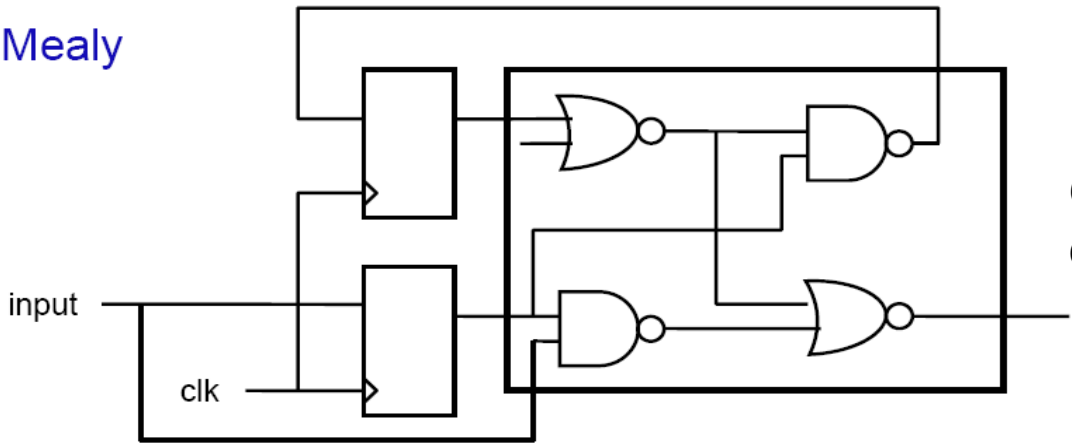
Finite State Machines (FSM)

Moore



Output is function of only the current state

Mealy



Output is function of current state and inputs

NJIT

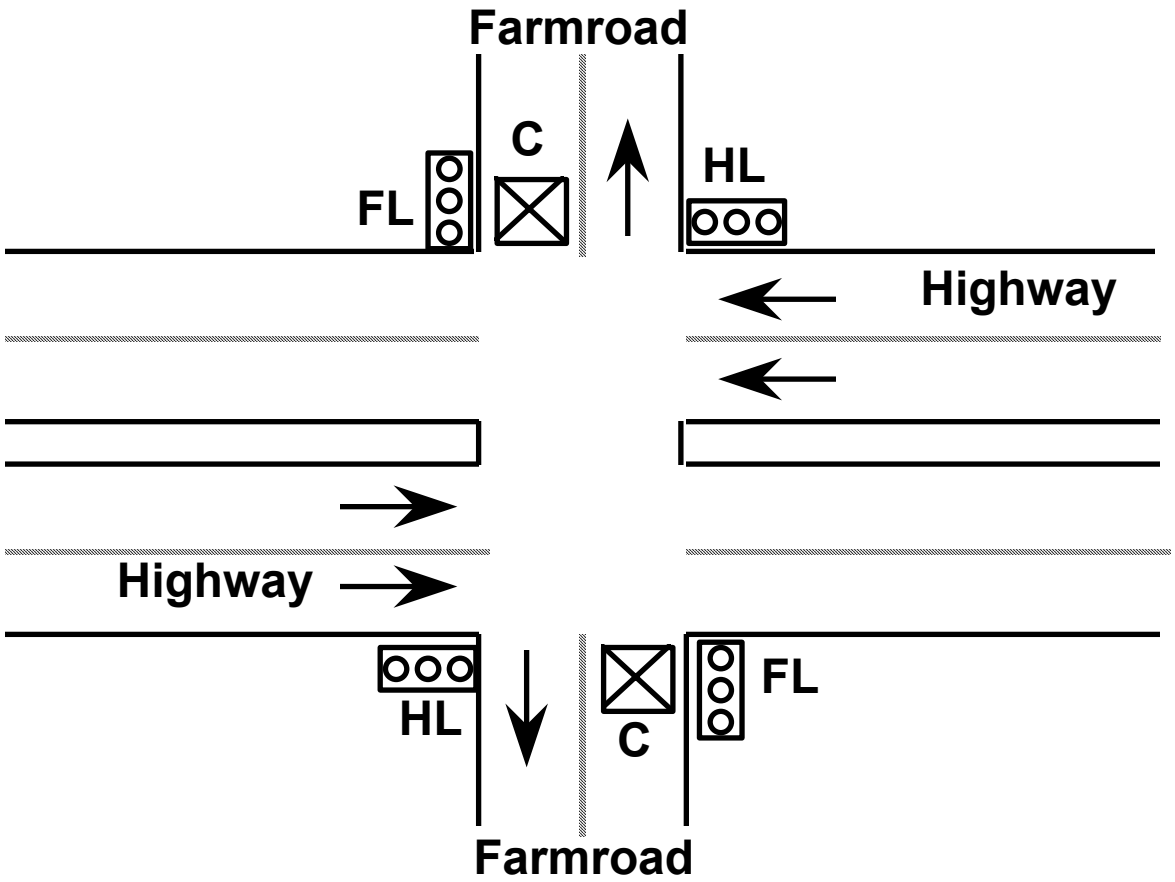
Traffic Light Controller

A busy highway is intersected by a little used farmroad. Detectors C sense the presence of cars waiting on the farmroad. With no car on farmroad, light remain green in highway direction. If vehicle on farmroad, highway lights go from Green to Yellow to Red, allowing the farmroad lights to become green. These stay green only as long as a farmroad car is detected but never longer than a set interval. When these are met, farm lights transition from Green to Yellow to Red, allowing highway to return to green. Even if farmroad vehicles are waiting, highway gets at least a set interval as green.

Assume you have an interval timer that generates a short time pulse (TS) and a long time pulse (TL) in response to a set (ST) signal. TS is to be used for timing yellow lights and TL for green lights.

Traffic Light Controller

Picture of Highway/Farmroad Intersection:



NJIT

Traffic Light Controller

- **Tabulation of Inputs and Outputs:**

Input Signal	*Description*
reset	place FSM in initial state
C	detect vehicle on farmroad
TS	short time interval expired
TL	long time interval expired

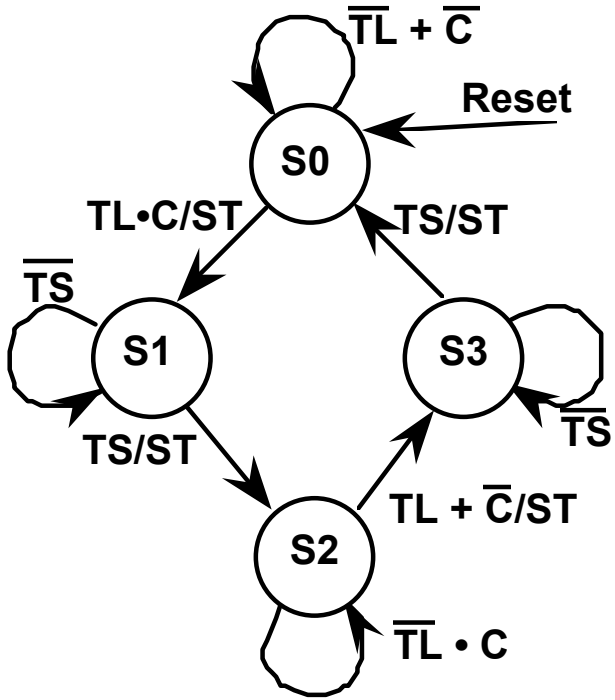
Output Signal	*Description*
HG, HY, HR	assert green/yellow/red highway lights
FG, FY, FR	assert green/yellow/red farmroad lights
ST	start timing a short or long interval

- **Tabulation of Unique States: Some light configuration imply others**

State	*Description*
S0	Highway green (farmroad red)
S1	Highway yellow (farmroad red)
S2	Farmroad green (highway red)
S3	Farmroad yellow (highway red)

Traffic Light Controller

Compare with state diagram:



S0: HG

S1: HY

S2: FG

S3: FY

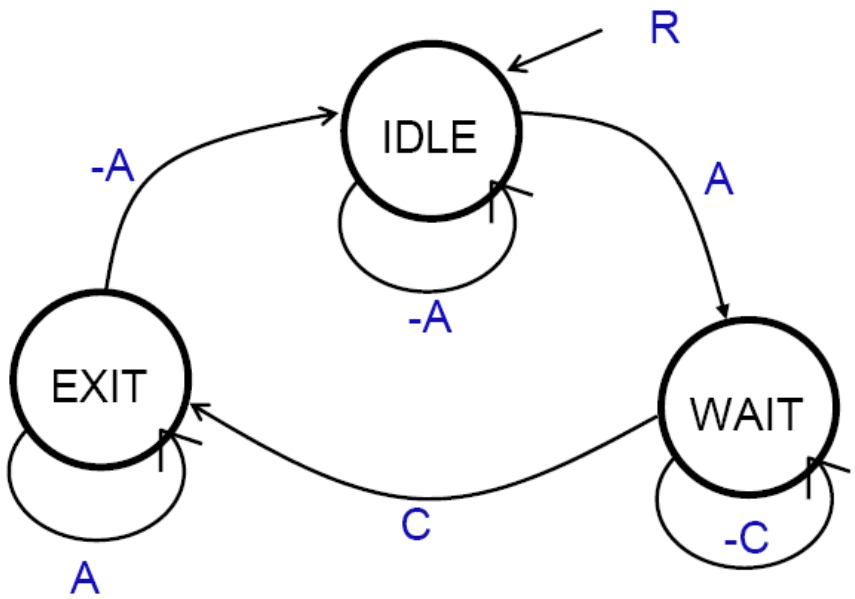
Advantages of State Charts:

- **Concentrates on paths and conditions for exiting a state**
- **Exit conditions built up incrementally, later combined into single Boolean condition for exit**
- **Easier to understand the design as an algorithm**

NJIT

Tollbooth Controller

- FSM design procedure
  - Draw the state-transition diagram
  - Check the state diagram
  - Write state equations (Write HDL)
- An example of state-transition diagram



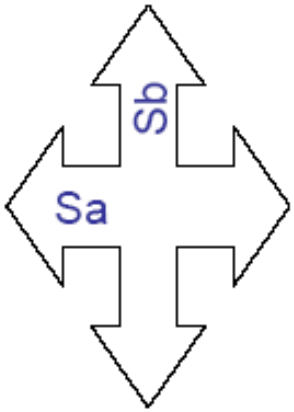
IDLE: (S1,S0)=(00)
WAIT: (S1,S0)=(01)
EXIT: (S1,S0)=(10)
A: car-in
C: change-ok
R: rst

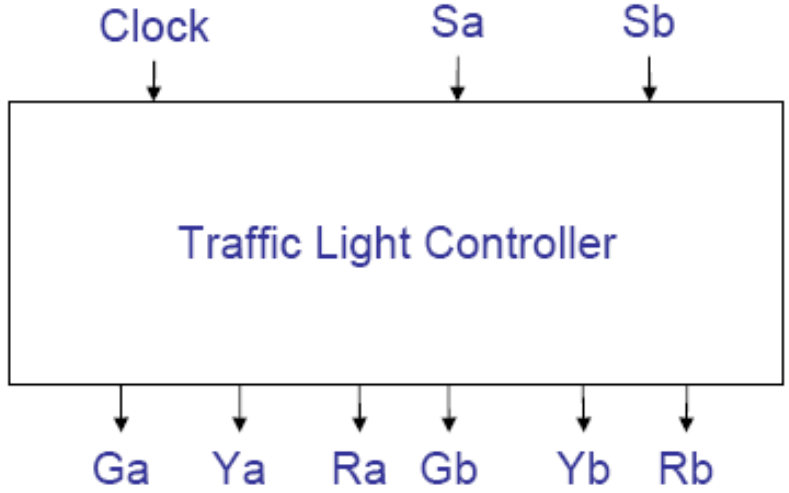


VHDL Modeling of FSMs

- Controller design should
  - Avoid asynchronous feedback, race conditions
  - Split large counters
  - Initialize design to a known state
- Minimize the number of latches
  - A flow-through latch is implied when a signal or variable in a combinational process is not fully specified
- Prefer case over IF-THEN-ELSE
  - IF statement operates on priority encoded basis

Traffic Light Controller

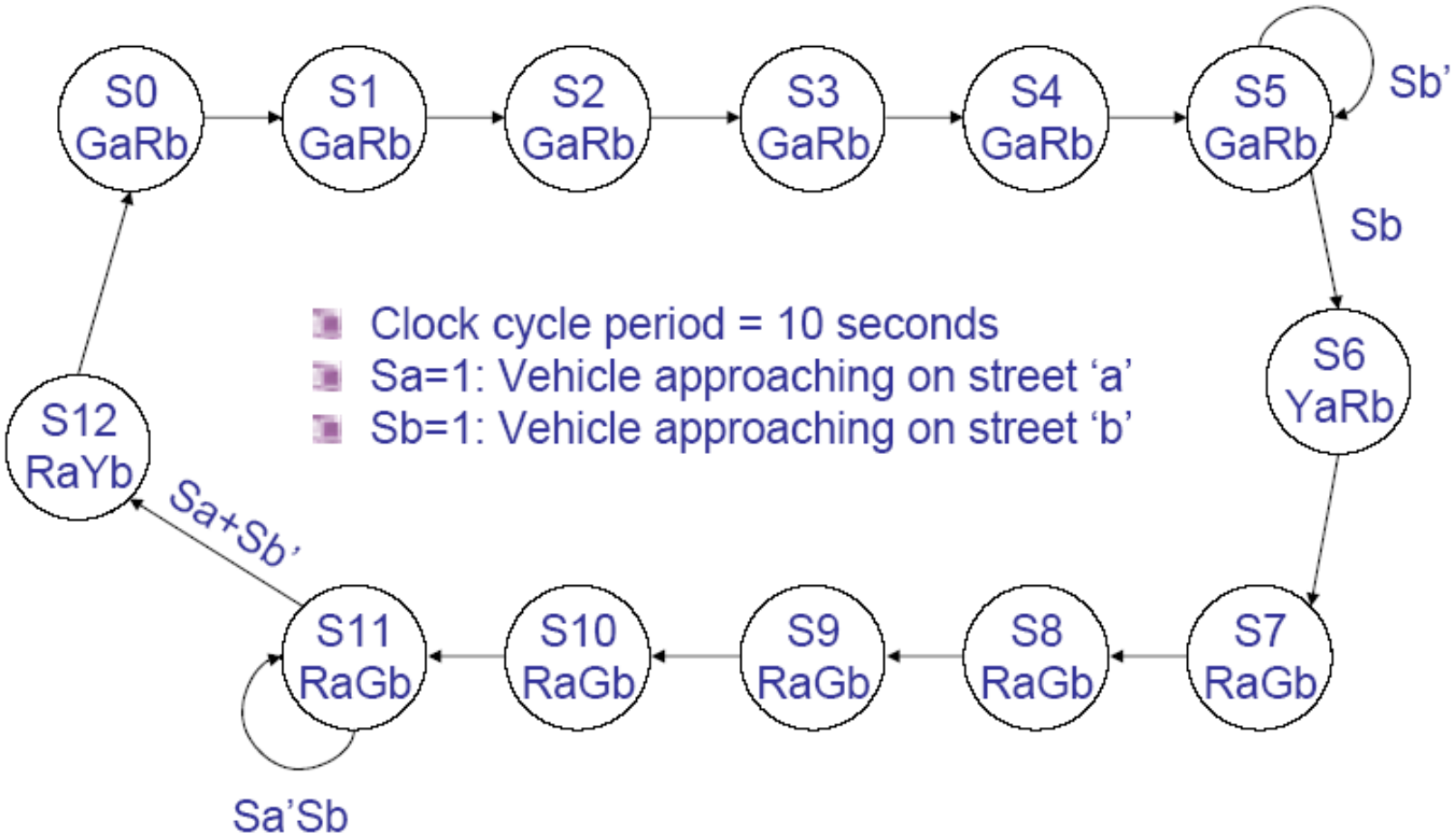




- Street a is the main street and street b is the side street
- Sa=1: Vehicle approaching on street ‘a’
- Sb=1: Vehicle approaching on street ‘b’
- Light on ‘a’ stays green until vehicle on ‘b’
- Then light on ‘b’ changes green and changes back after 50 seconds
- If another car on ‘b’ and no car on ‘a’ then light on ‘b’ remains green for 10 more seconds
- When ‘a’ is green remains green for at least 60 seconds

NJIT

Traffic Light Controller Mealy FSM





Traffic Light Controller VHDL Model

```
entity traffic_light is port ( clk, Sa , Sb: in bit;
                               Ra, Rb, Ga, Gb, Ya, Yb: inout bit );
end traffic_light;

architecture behave of traffic_light is
    signal state, nextstate: integer range 0 to 12;
    type light is ( R,Y, G );
    signal lightA, lightB: light;
begin
    process( clk )
    begin
        if ( clk'event and clk = '1' ) then
            state <= nextstate;
        end if;
    end process;
    lightA <= R when Ra ='1' else Y when Ya = '1' else G when Ga ='1';
    lightB <= R when Rb ='1' else Y when Yb = '1' else G when Gb ='1';
```

State transition on clock edge

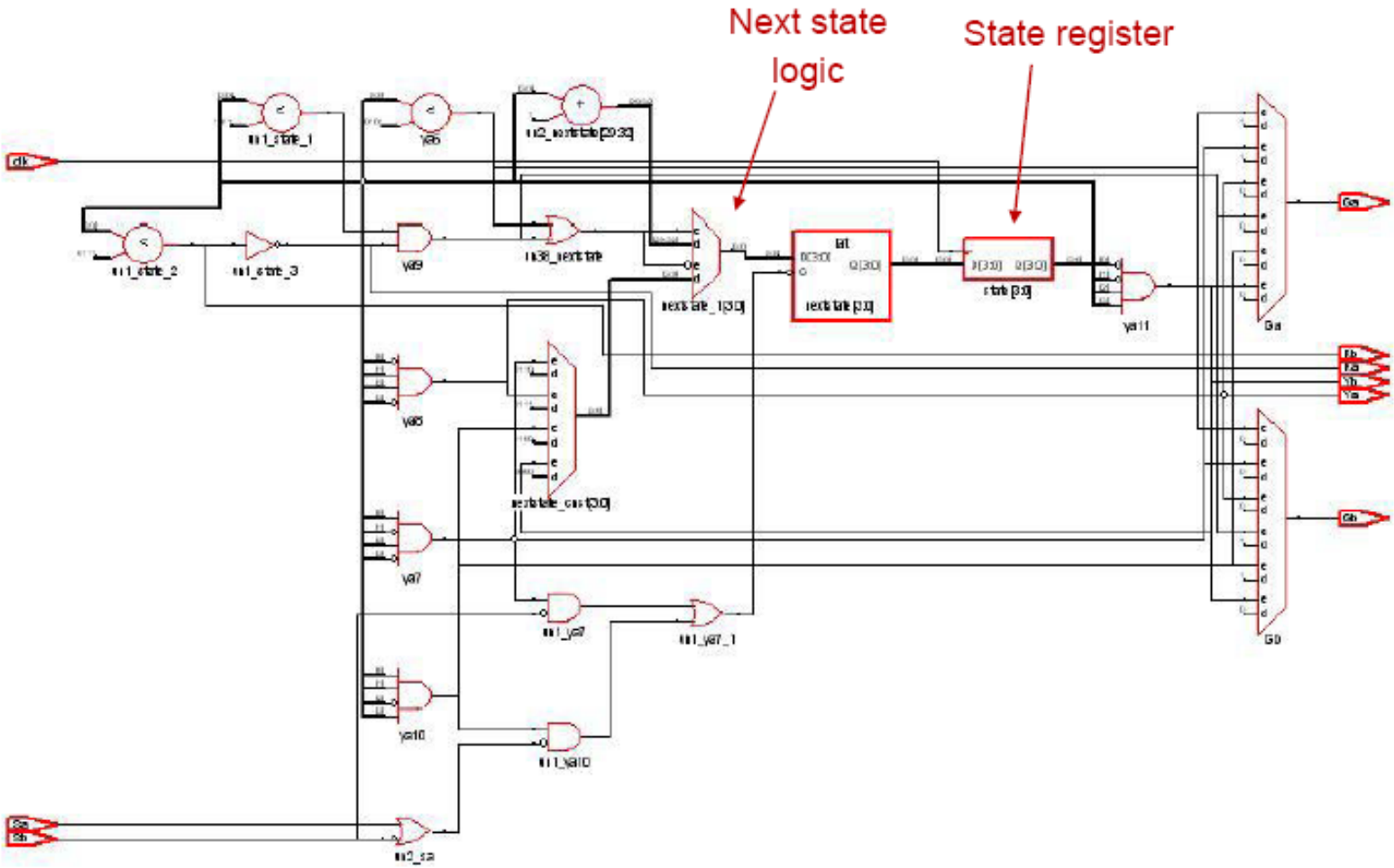
NJIT

Traffic Light Controller VHDL Model (Cont.)

```
process ( state, Sa, Sb )
begin
        Ra <='0'; Rb<='0'; Ga<='0';
        Gb<='0'; Ya<='0'; Yb<= '0';
        case state is
            when 0 to 4 => Ga <= '1'; Rb <= '1'; nextstate <= state+1;
            when 5 => Ga <= '1'; Rb<='1';
                if Sb = '1'then nextstate <=6; end if;
            when 6 => Ya <= '1'; Rb <= '1'; nextstate <= 7;
            when 7 to 10 =>Ra <='1'; Gb <='1'; nextstate <= state +1;
            when 11=> Ra <= '1'; Gb <= '1';
                if (Sa ='1' or Sb ='0') then nextstate <= 12; end if;
            when 12 => Ra <= '1'; Yb <='1'; nextstate <= 0;
            when others => NULL;
        end case;
    end process;
end behave;
```

Output and next state logic

Synthesized Traffic Light Controller



NJIT

Simulated Traffic Light Controller

State changes on rising clock edge as a result of change in Sb

Output follows current state

Timeout from state 11 after 10 Sec

