FH JOANNEUM
System Test Engineering

Digital Circuit Design – Laboratory 2

a traffic light logic coded as a state machine

Patrick Lampl

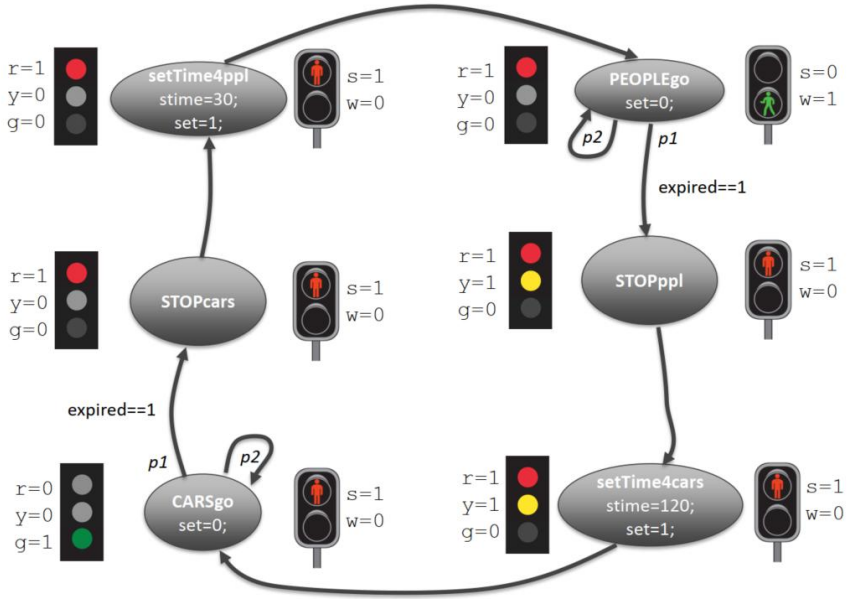
09.10.2025

FH JOANNEUM Gesellschaft mbH
Alte Poststraße 149, A-8020 Graz, AUSTRIA, T.: +43 (0)316 5453-8800, E.: info@fh-joanneum.at, www.fh-joanneum.at

DVR 0813559, UID-Nr ATU 42361001, FN 125888 f, Landesgericht für ZRS Graz
Bankverbindung: Landes-Hypothekenbank Steiermark AG, Kto.Nr. 20141232330, BLZ 56000, IBAN: AT885600020141232330, BIC: HYSTAT2G

1 STATEMACHINE

Create the traffic light state machine demonstrated in Lecture DCD-LV6



States and state transition conditions of the traffic light

Modify the state diagram as such, that the CARSgo state stays there indefinitely. Add a user input (push button) for pedestrians to request passage. Only upon pedestrian request the state machine propagates away from the CARSgo state, stops the cars, lets the pedestrians cross and afterwards arrives back at the CARSgo state, waiting for another pedestrian request.

direction, bit-width and name of signal	Functional description
input i_button	pushbutton input pedestrians (from any side of the road)
input i_expired	external timing input
input i_fsmstep	external strobe input to define step time of FSM (e.g. a strobe each second)
output o_stop	output for pedestrian red light
output o_walk	output for pedestrian green light
output o_red	output for car red light
output o_yellow	output for car yellow light
output o_green	output for car green light
output [7:0] o_stime	8bit output to define a time for the external timer
output o_set	output to start external timer
input i_clk	100MHz clock
input i_rst,	Active high synchronous reset: Resets all internal memories to zero

Create a simple manual test bench where you test three cases:

- start the FSM in the STOPcars stat and let it propagate until it gets stuck in the CARSgo state.
- press the button to trigger a pedestrian cycle, make sure the FSM stops again at CARSgo.
- simulate a user keeping the button pressed forever, FSM should behave as before.

1.1 Documentation

Document your state machine logic, the test bench for your three test cases and the results of the simulation. (interpret you results! does the state machine behave as expected?)

1.2 Tipps

Remember to code the state machine in a way that the state change is coded separately from the actions performed within said states (as shown within the lecture slides).

Every state that has an immediate follower can directly switch to the next state whenever the i_fsmstep input is asserted.
Actual waiting, where you set “next_state = current_state” is only required within the CARSgo and PEOPLEgo states.

How to add memory to your state machine? There are two options:

1. If you want to stick with a strict MOORE or MEALY style FSM, you must code your memories as states in your state diagram. This will cause you to have many more states than the states shown on page 1.
2. But as shown in LV6, you can have the state machine communicate with external memories or timers, operating concurrently with the FSM.