Laboratory Report # 7

Name: Lauron, John Enrico D. **Date Completed:** December 5, 2025

Laboratory Exercise Title: Finite State Machines (FSMs)

Target Course Outcomes:

CO1: Create descriptions of digital hardware components such as in combinational and sequential circuits using synthesizable Verilog HDL constructs.

CO2: Verify the functionality of HDL-based components through design verification tools.

Exercise 7A: Complex Counter

In Exercise 7A, we designed a 3-bit complex counter using a Moore-type FSM. Our task was to construct a counter that could operate in either binary or Gray-code mode depending on the value of the mode input. We created the state diagram and transition table, defined the next-state logic, and implemented the design in Verilog using a synchronous active-low reset and negative-edge clocking. Overall, we focused on how the FSM transitions produce the correct output sequence for both counting modes.

M	Current (State)	Next (State)
0	000	001
0	001	010
0	010	011
0	011	100
0	100	101
0	101	110
0	110	111
0	111	000
1	000	001
1	001	011
1	011	010
1	010	110
1	110	111

1	111	101
1	101	100
1	100	000

Table 1. State Transition Table (Current -> Next)

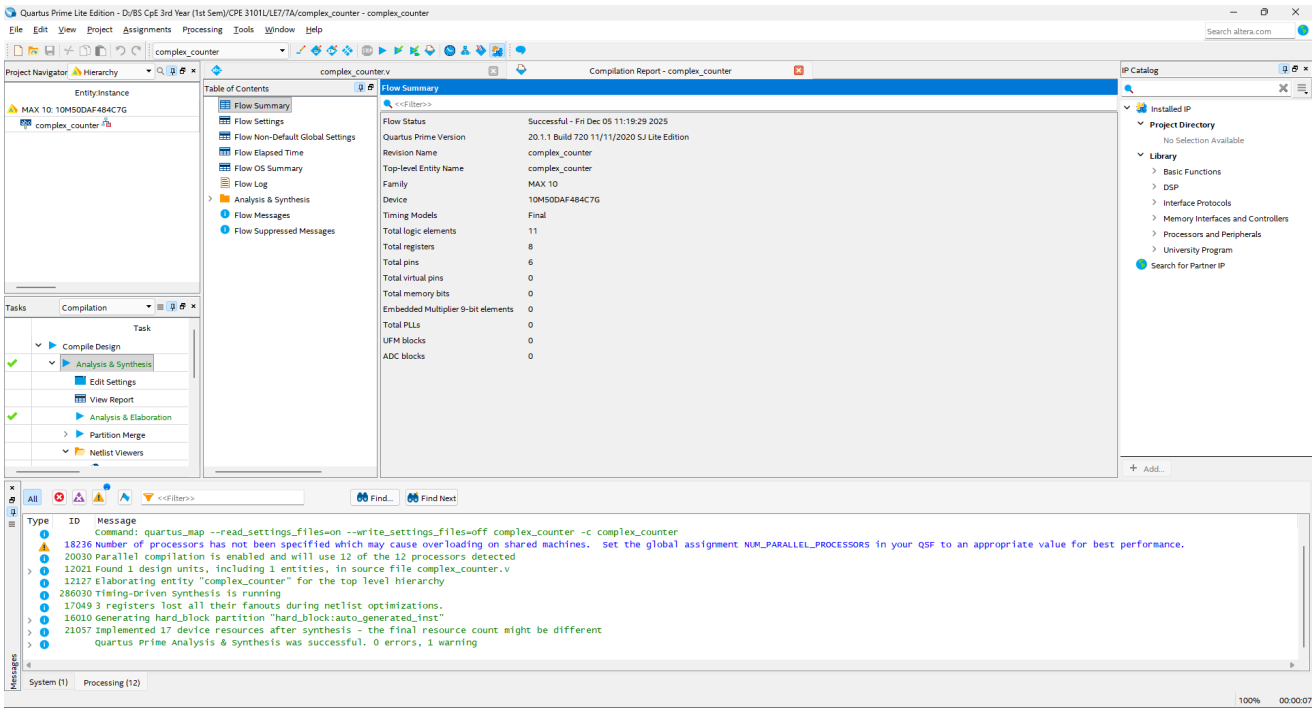

Figure 1. Flow Summary and Design Synthesis Result of Complex Counter

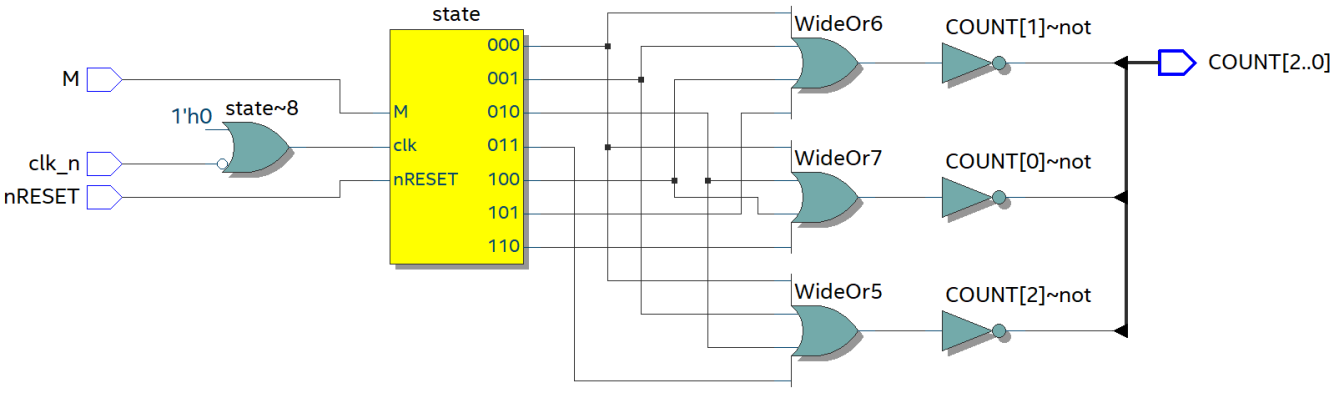

Figure 2. RTL View of Complex Counter

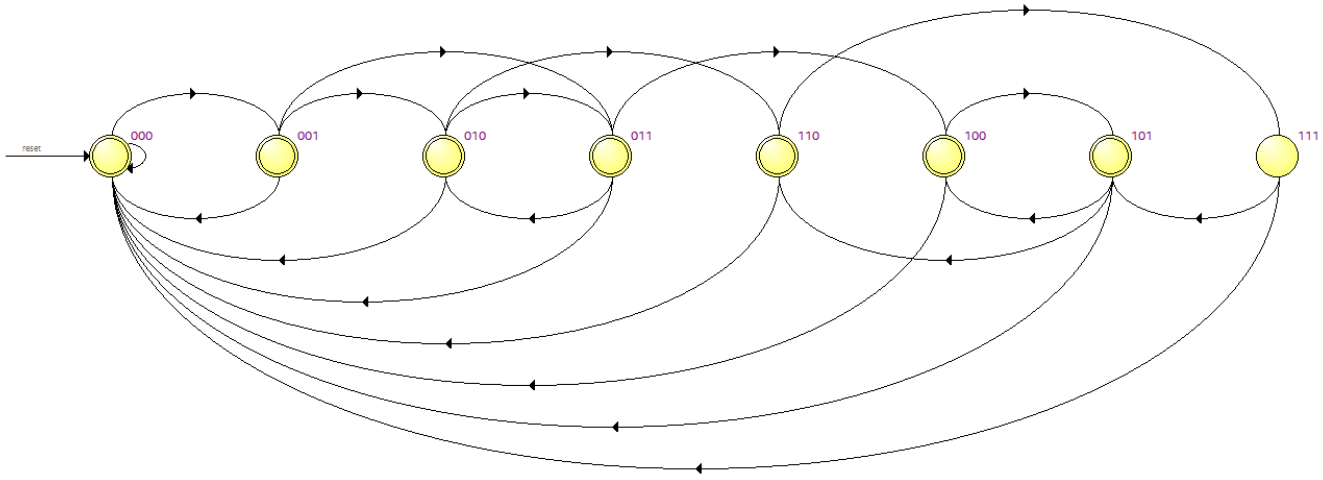

Figure 3. State Machine View of Complex Counter

	Source State	Destination State	Condition
1	000	001	(nRESET)
2	000	000	(!nRESET)
3	001	010	(!M).(nRESET)
4	001	011	(M).(nRESET)
5	001	000	(!nRESET)
6	010	011	(!M).(nRESET)
7	010	110	(M).(nRESET)
8	010	000	(!nRESET)
9	011	010	(M).(nRESET)
10	011	100	(!M).(nRESET)
11	011	000	(!nRESET)
12	100	000	(!M). (!nRESET) + (M)
13	100	101	(!M).(nRESET)
14	101	100	(M).(nRESET)
15	101	110	(!M).(nRESET)
16	101	000	(!nRESET)
17	110	000	(!nRESET)
18	110	111	(nRESET)
19	111	000	(!M) + (M). (!nRESET)
20	111	101	(M).(nRESET)

Table 2. State Transition Table of Complex Counter

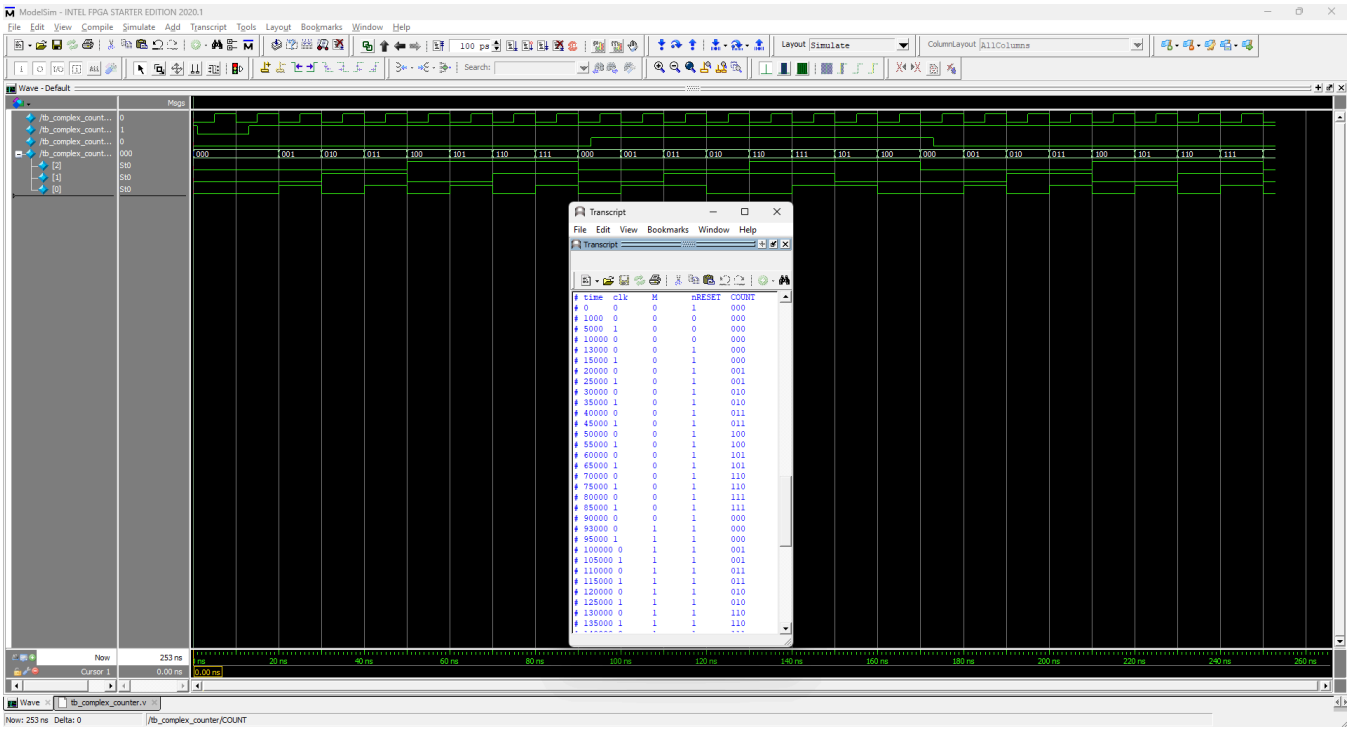

Figure 4. Simulation Results of Complex Counter

Exercise 7B: Race Lights Controller

In Exercise 7B, we developed a race-lights controller using a Mealy-type FSM. Our design responds to a START signal and controls the RED, YELLOW, and GREEN outputs with specific timing requirements before returning to the idle state. We defined the states, timing behavior, and Mealy transitions influenced by the input signal, then wrote the Verilog implementation using an asynchronous active-low reset and negative-edge clocking. This exercise allowed us to apply Mealy FSM concepts to a real-world timing-based control system.

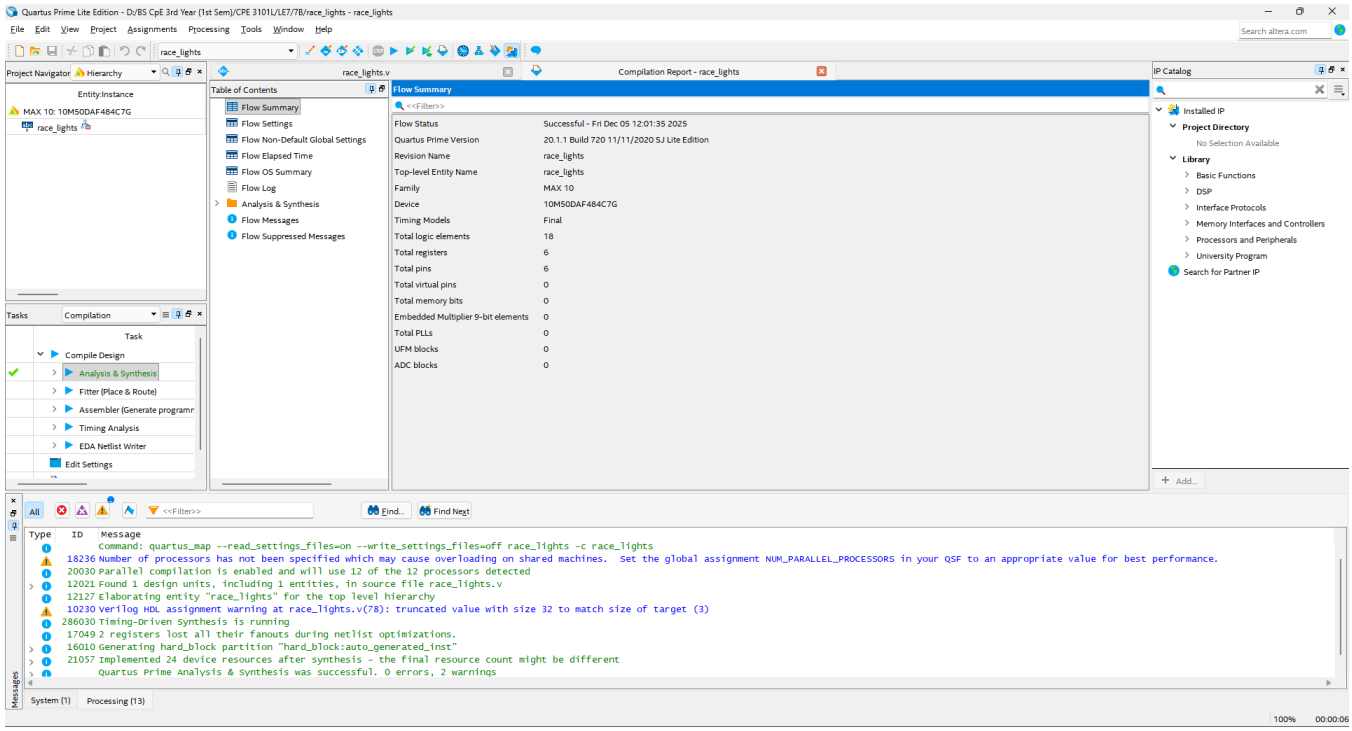

Figure 5. *Flow Summary and Design Synthesis Result of Race Lights Controller*

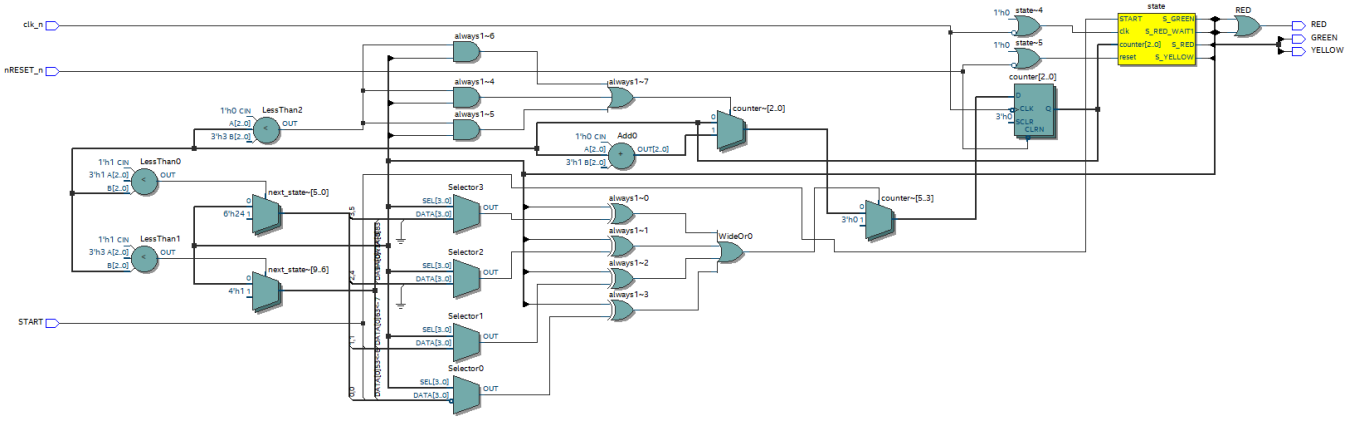

Figure 6. RTL View of Race Lights Controller

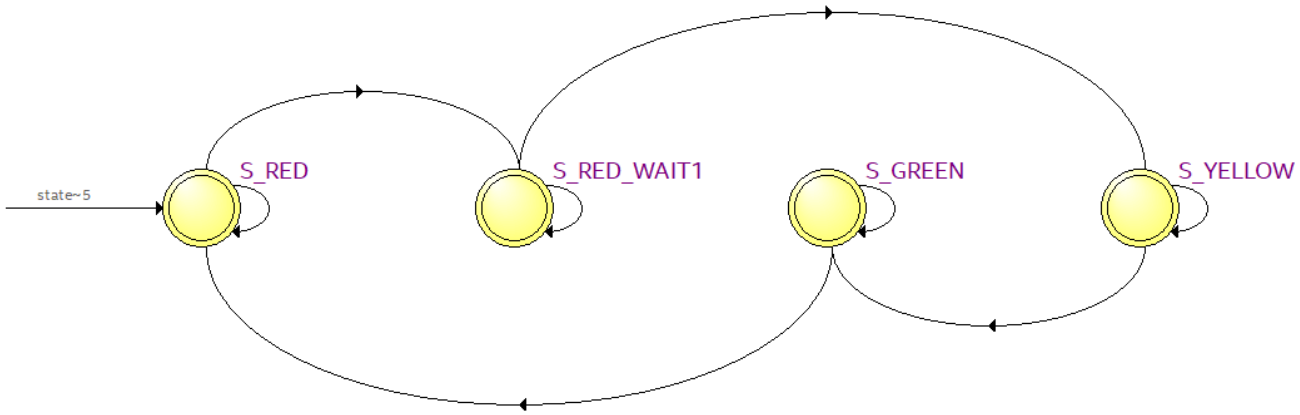

Figure 7. State Machine View of Race Lights Controller

	Source State	Destination State	Condition
1	S_GREEN	S_RED	(!counter[0]).(counter[2]) + (counter[0]).(!counter[1]).(counter[2]) + (counter[0]).(counter[1])
2	S_GREEN	S_GREEN	(!counter[0]).(!counter[2]) + (counter[0]).(!counter[1]).(!counter[2])
3	S_RED	S_RED_WAIT1	(START)
4	S_RED	S_RED	(!START)
5	S_RED_WAIT1	S_YELLOW	(!counter[0]).(!counter[1]).(counter[2]) + (!counter[0]).(counter[1]) + (counter[0])
6	S_RED_WAIT1	S_RED_WAIT1	(!counter[0]).(!counter[1]).(!counter[2])
7	S_YELLOW	S_YELLOW	(!counter[0]).(!counter[1]).(!counter[2])
8	S_YELLOW	S_GREEN	(!counter[0]).(!counter[1]).(counter[2]) + (!counter[0]).(counter[1]) + (counter[0])

Table 3. State Transition Table of a Race Lights Controller

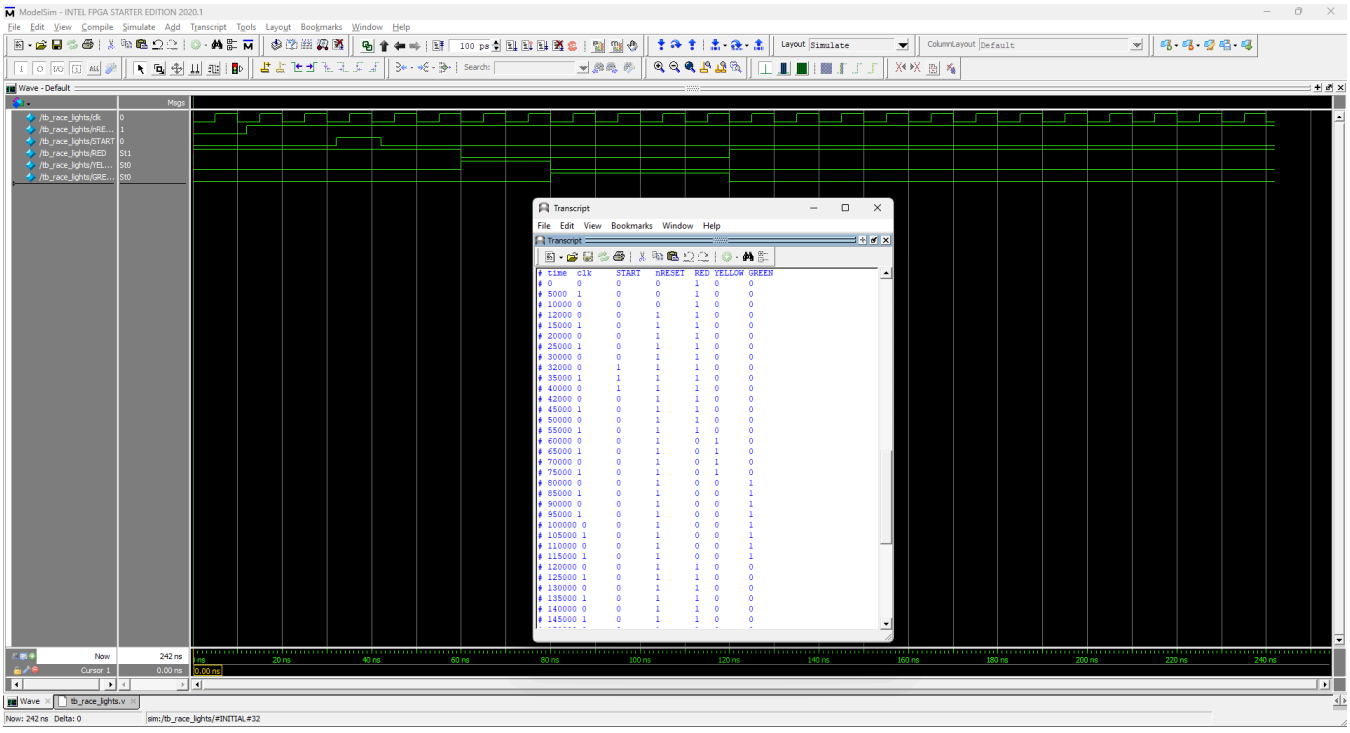

Figure 8. Simulation Results of a Race Lights Controller