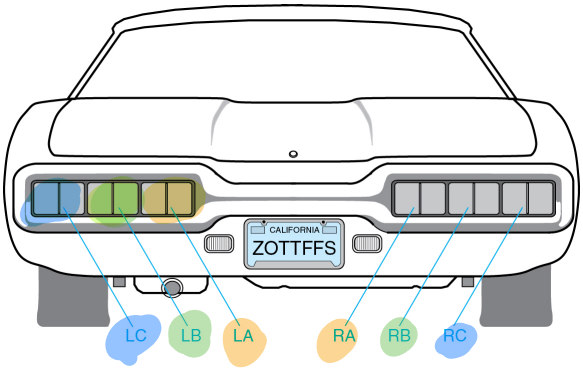
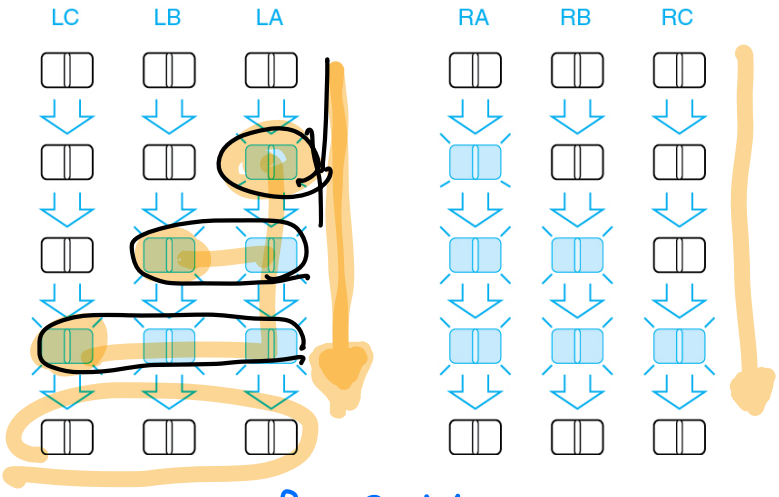
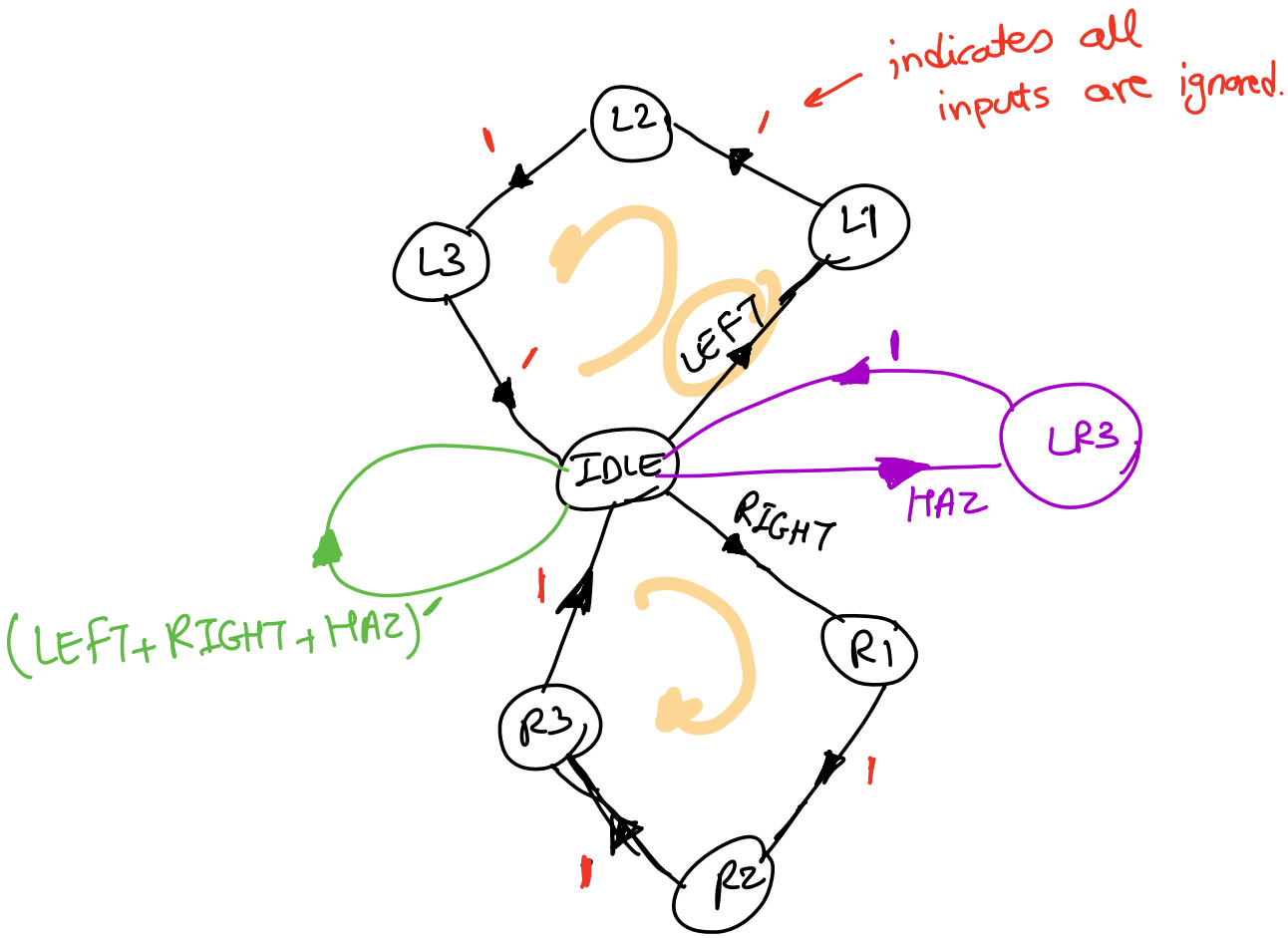
Example: Ford Thunderbird Bird Tail Lights- There are 3 lights on each side (left and right), and for turns they operate in sequence to animate the turning direction. The state machine has 3 input signals: LEFT and RIGHT (that carry the driver's request for a left turn or a right turn), and HAZ (that requests the tail lights to be operated in hazard mode—all six lights flashing on and off in unison).



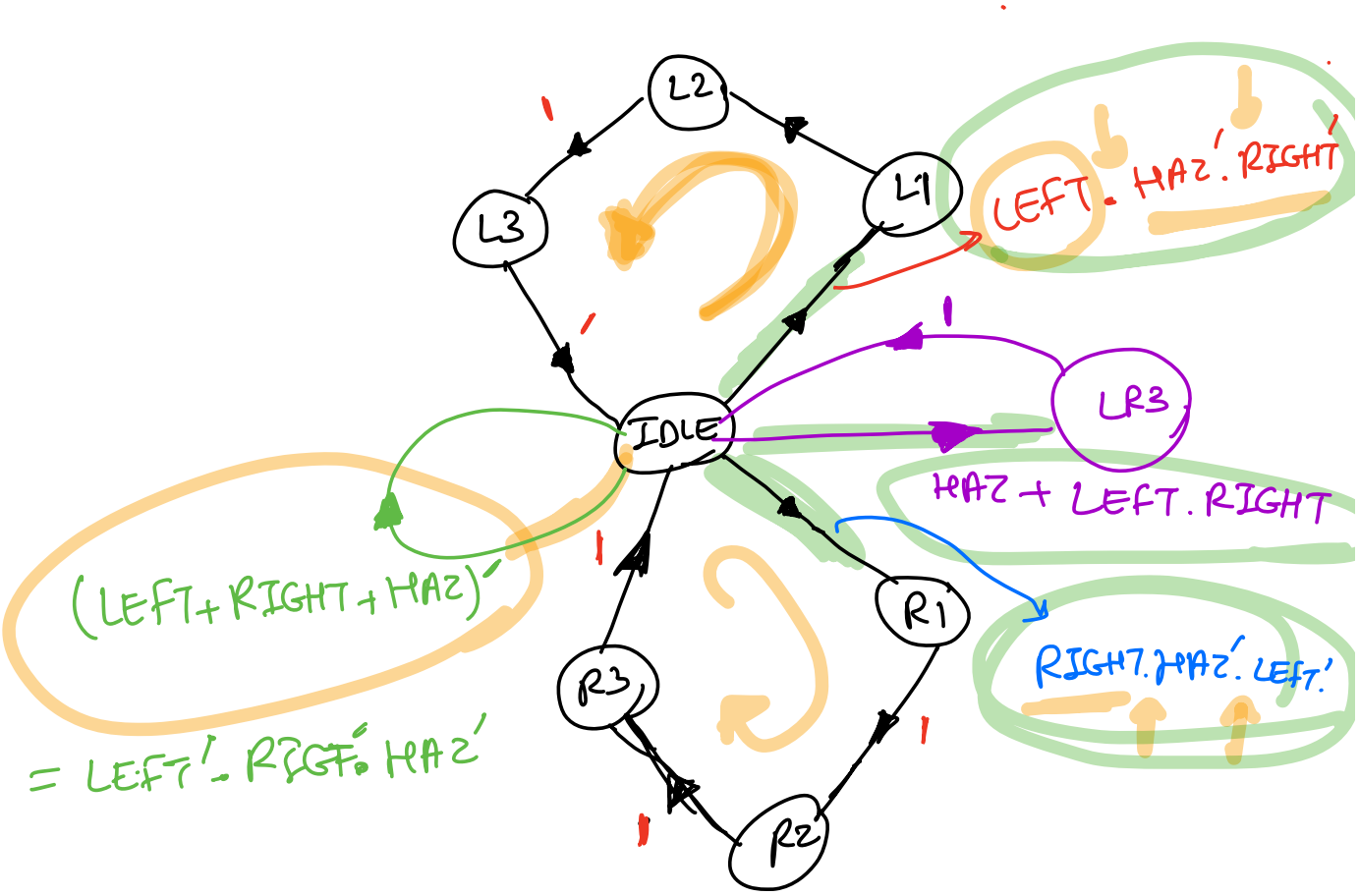


Identify Inputs/Outputs: { 3 inputs: Left, Right, HAZ
6 outputs: 3 on Left (LA, LB, LC)
3 on Right (RA, RB, RC)

State	LC	LB	LA	RA	RB	RC
IDLE	0	0	0	0	0	0
L1	0	0	1	0	0	0
L2	0	1	1	0	0	0
L3	1	1	1	0	0	0
R1	0	0	0	1	0	0
R2	0	0	0	1	1	0
R3	0	0	0	1	1	1
LR3	1	1	1	1	1	1



Problem: The above state diagram does not properly handle multiple inputs asserted at the same time. (e.g., what if in IDLE state, both "RIGHT" and "HAZ" Commands are asserted) → This is an "ambiguous" state diagram.



Ambiguity in State diagrams:

- → Mutual Exclusion: logical product of each possible pair of transition expressions on arcs leaving the state 0
- → All Inclusion: logical sum of the transition expressions on all arcs leaving the state is 1.

In the example State diagram, the "IDLE" state has more than "1" arc leaving the state. So we need to check ambiguity for this state.