VHDL for Sequential Logic

ELEC 311

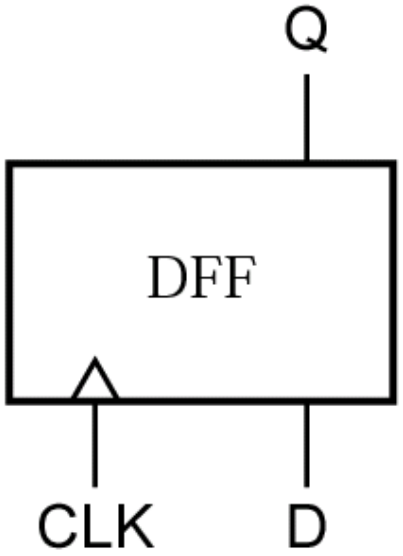
Digital Logic and Circuits

Dr. Ron Hayne

Images Courtesy of Cengage Learning

THE CITADEL
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

D Flip-Flop




```
process (CLK)
begin
     if CLK'event and CLK = '1' -- rising edge of CLK
          then Q <= D;
     end  if;
end  process;
```

Asynchronous Clear

```
process (CLK, ClrN)
begin
     if ClrN = '0' then Q <= '0';
          else  if CLK'event and CLK = '1'
                then Q <= D;
                end  if;
     end  if;
end  process;
```

Shift Register

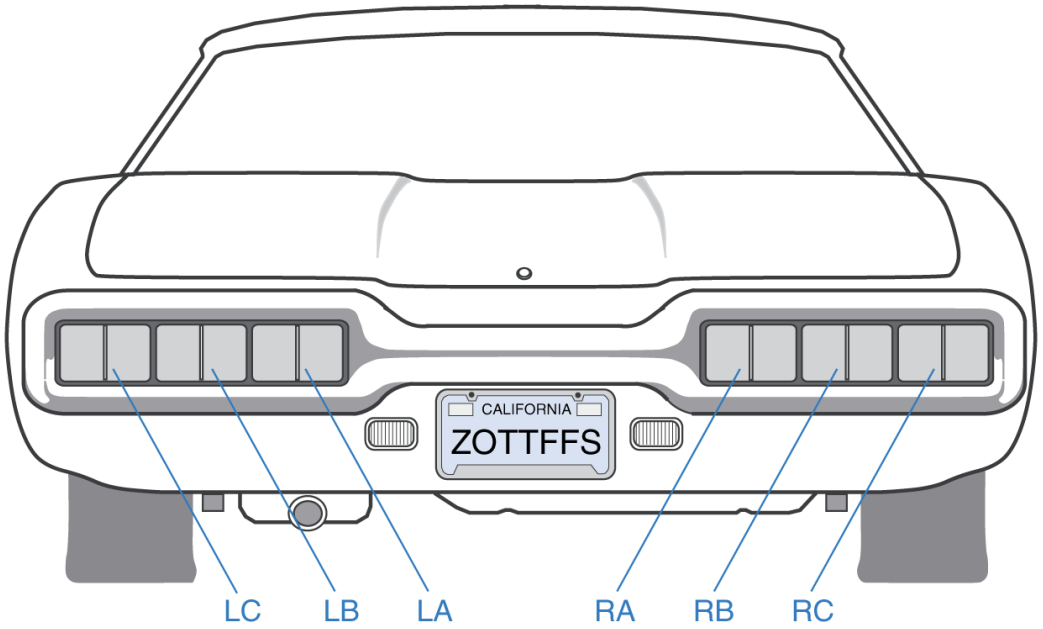
```
process (CLK)
begin
    if CLK'event and CLK = '1' then
       if CLR = '1' then Q <= "0000";
          elsif Ld = '1' then Q <= D;
          elsif LS = '1' then Q <= Q(2 downto 0)& Rin;
       end  if;
    end  if;
end  process;
```

Counter

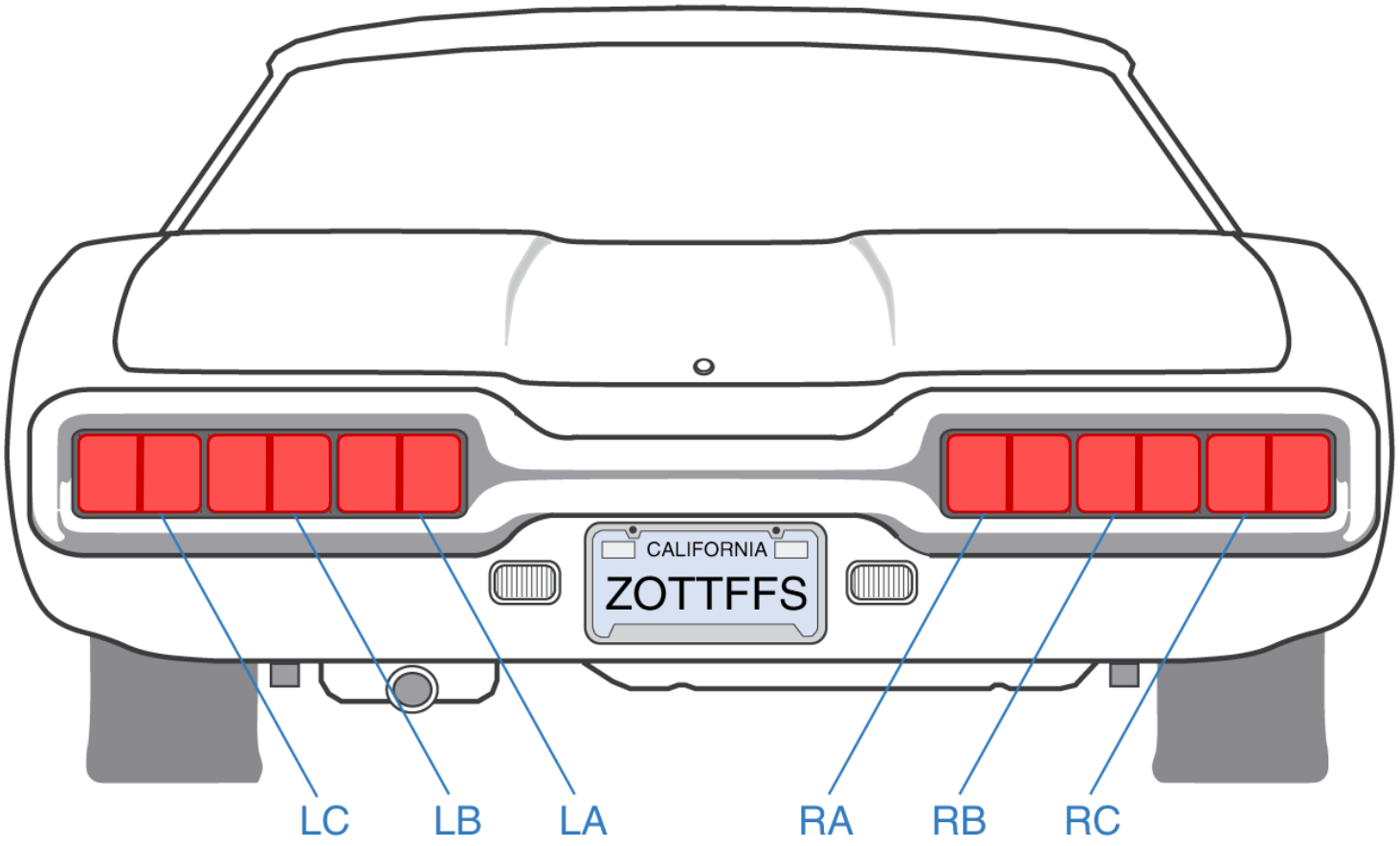
```
signal Q: std_logic_vector(3 downto 0);
------------
process (CLK)
begin
     if CLK'event and CLK = '1' then
          if ClrN = '0' then Q <= "0000";
               elsif En = '1' then Q <= Q + 1;
          end  if;
     end  if;
end  process;
```

Design Example

- T-bird Tail Lights
  - Left Turn
  - Right Turn
  - Hazard

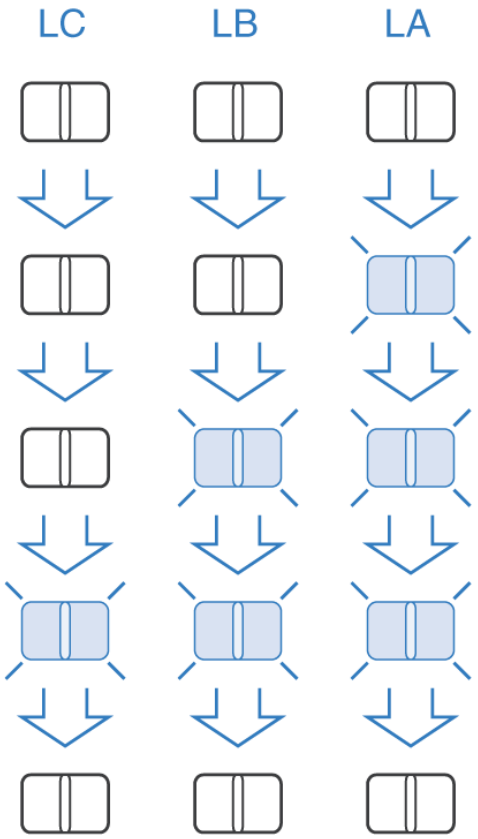


Flashing Sequence

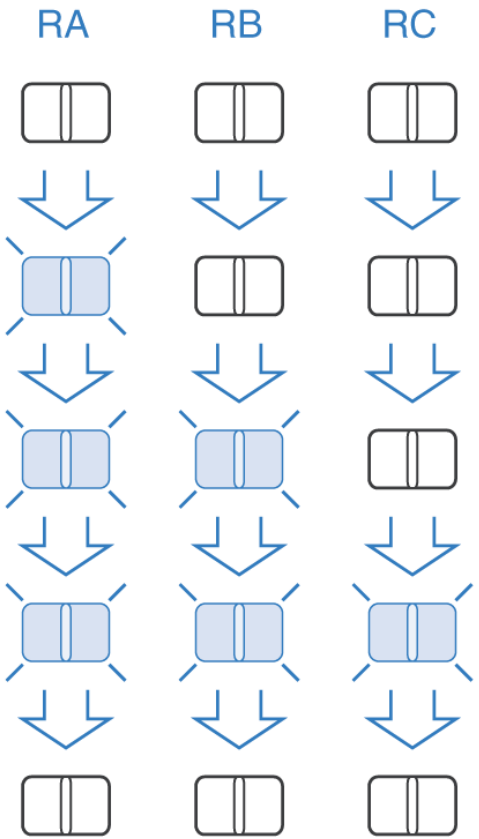


Flashing Sequence

- Left Turn

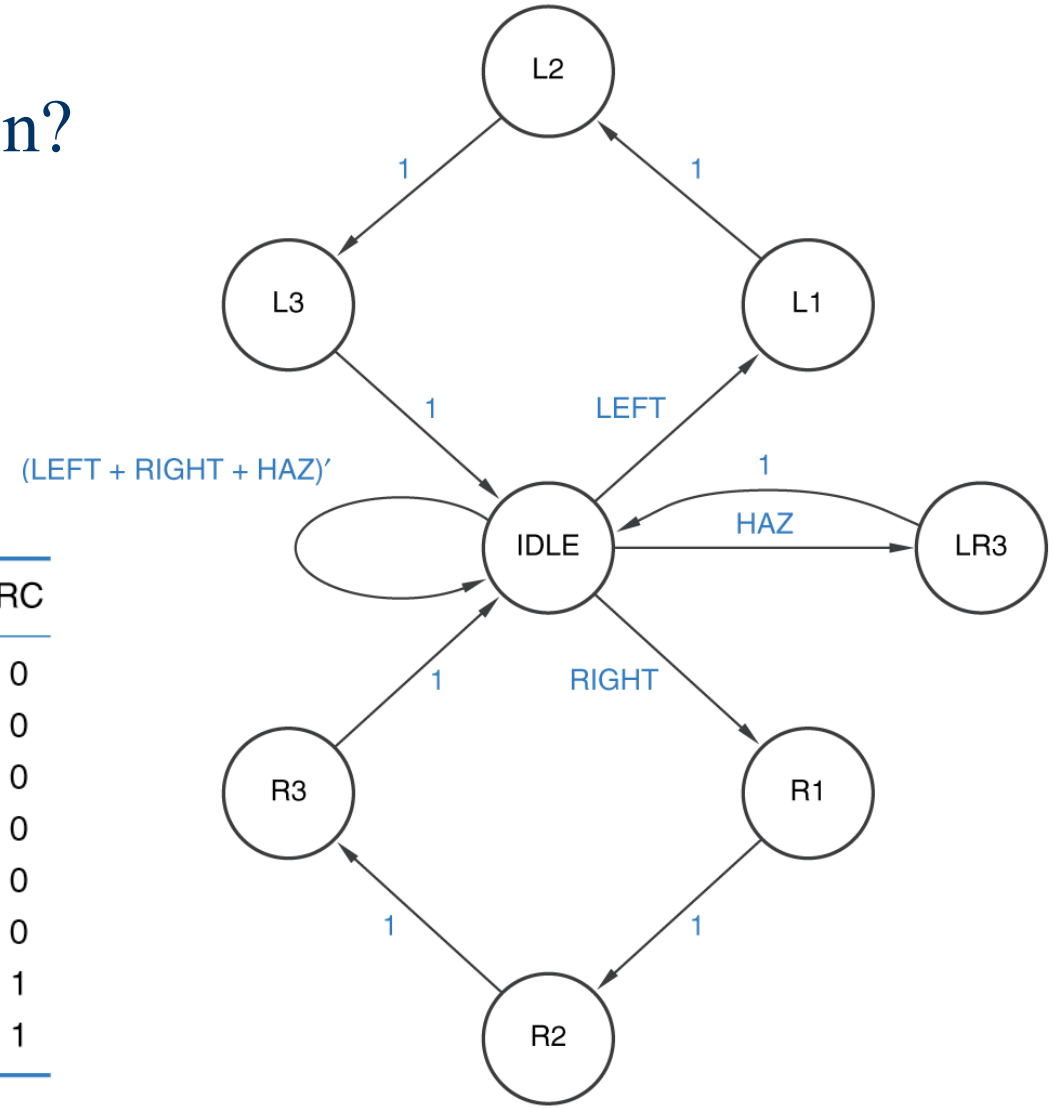


- Right Turn



Initial State Graph

- Mutual Exclusion?
- All Inclusion?

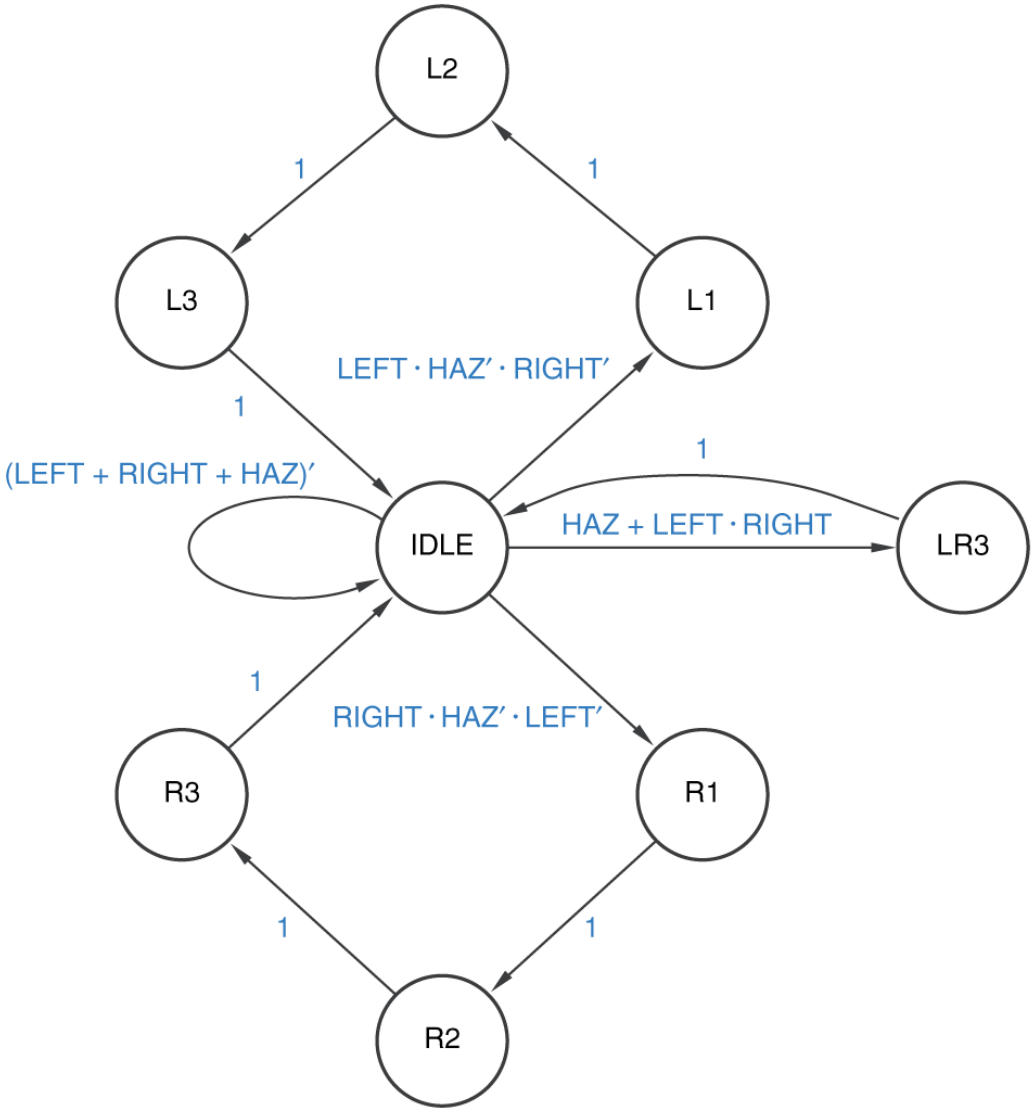


Output Table

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

Corrected State Graph

- Handles multiple inputs asserted simultaneously

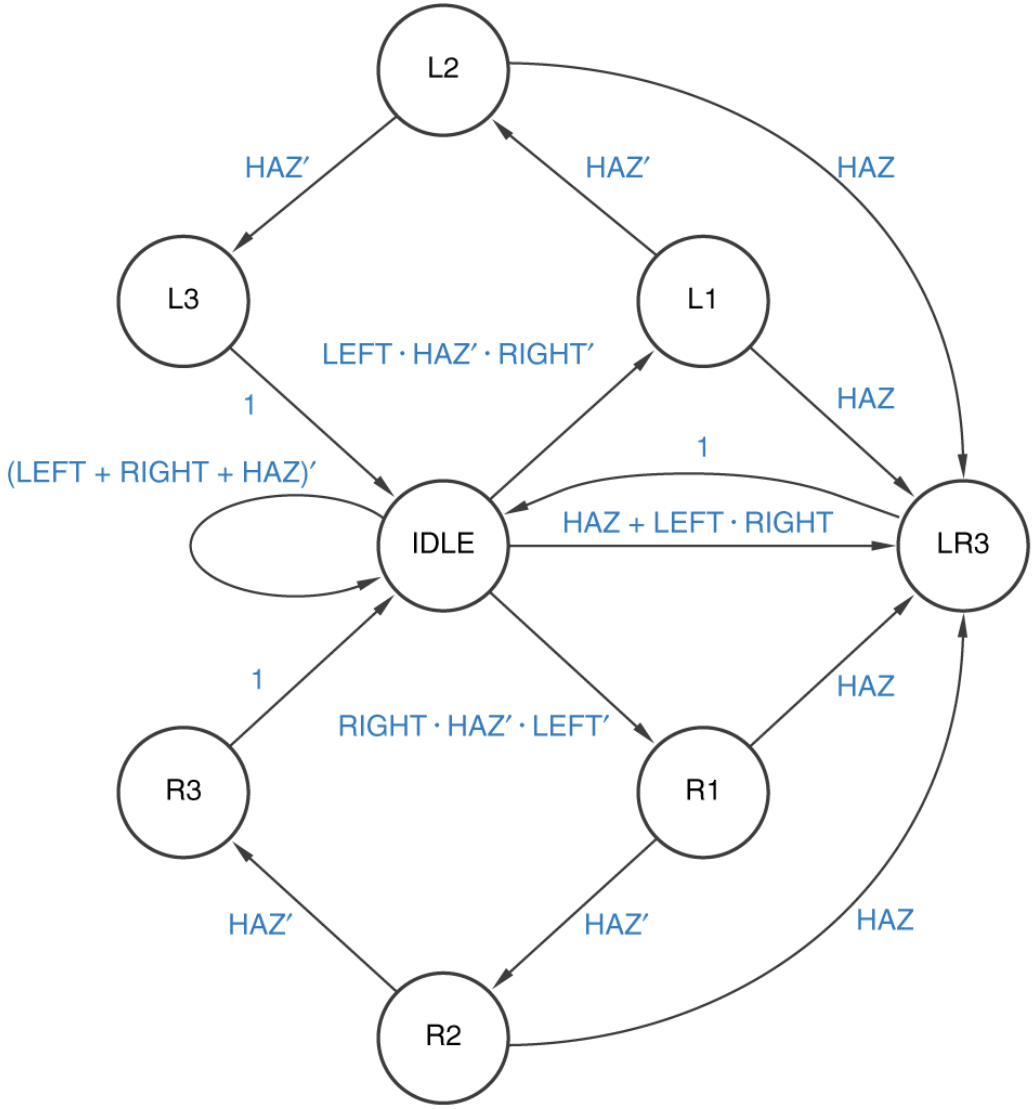


Output Table

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

Enhanced State Graph

- Goes into hazard mode as soon as possible



Output Table

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

VHDL Model

```
entity TBIRD is
  port (clock, left, right, haz : in  std_logic;
        tail : out  std_logic_vector (1 to 6));
end TBIRD;

architecture BEHAVE of TBIRD is
  type State_type is (IDLE,L1,L2,L3,R1,R2,R3,LR3);
  signal State, Next_State: State_type;
  signal input: std_logic_vector(1 to 3);

begin
  input <= left & right & haz;
```

VHDL Outputs

```
with State select
  tail <= "000000" when IDLE,
          "001000" when L1,
          "011000" when L2,
          "111000" when L3,
          "000100" when R1,
          "000110" when R2,
          "000111" when R3,
          "111111" when LR3;
```

VHDL Sequential Machine

```
process (input, State)
  begin
    case State is
      when IDLE =>
        case input is
          when "010" => Next_State <= R1;
          when "100" => Next_State <= L1;
          when "--1" => Next_State <= LR3;
          when others => Next_State <= IDLE;
        end case;
      when L1 =>
        case input is
          when "--1" => Next_State <= LR3;
          when others => Next_State <= L2;
```

VHDL Sequential Machine

```
process (SLOW_CLK)
begin
  if SLOW_CLK'event and SLOW_CLK = '1' then
    State <= Next_State;
  end if;
end process;
```

Summary

- Sequential VHDL
  - process
  - if-then-else
  - case
- Design Example