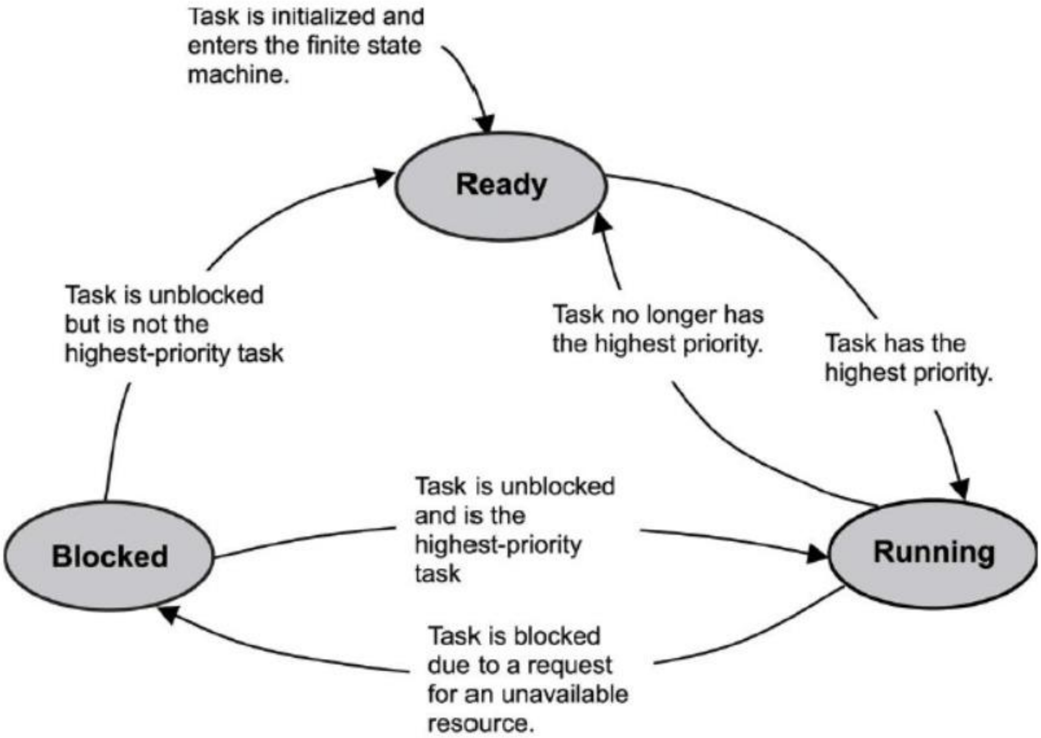
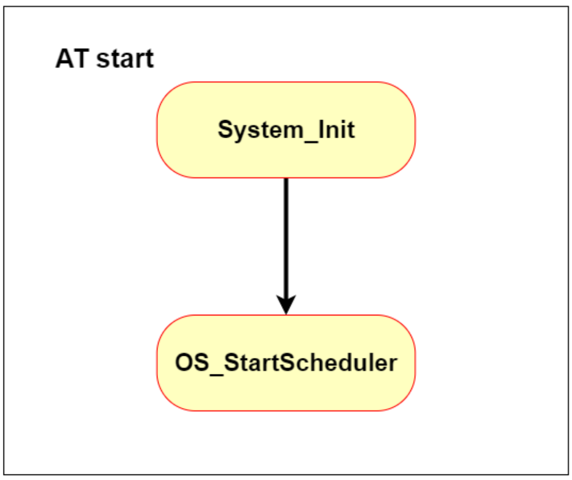
❖ Automotive door control system design

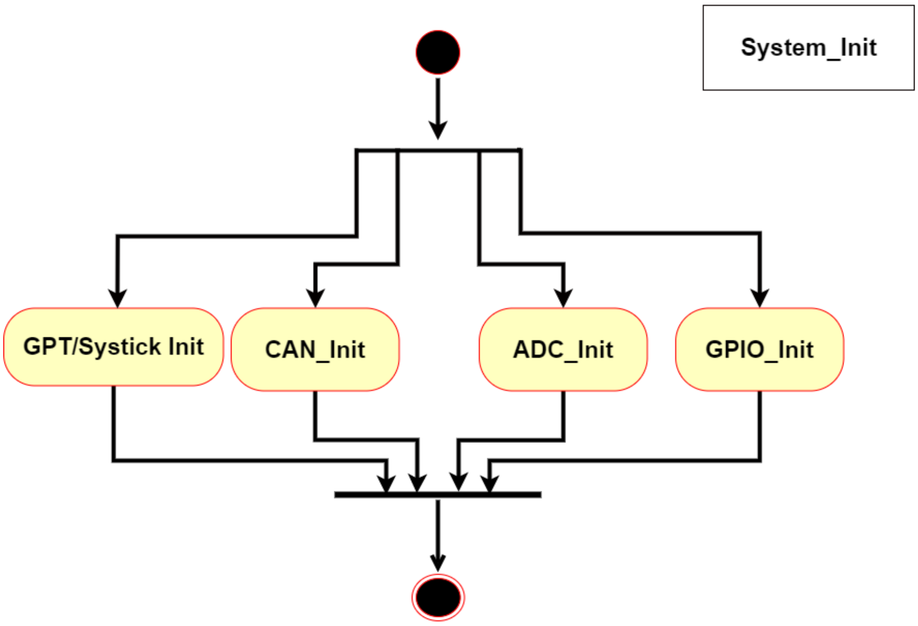
(Dynamic Design)

- **OS(RTOS) StateMachine :**

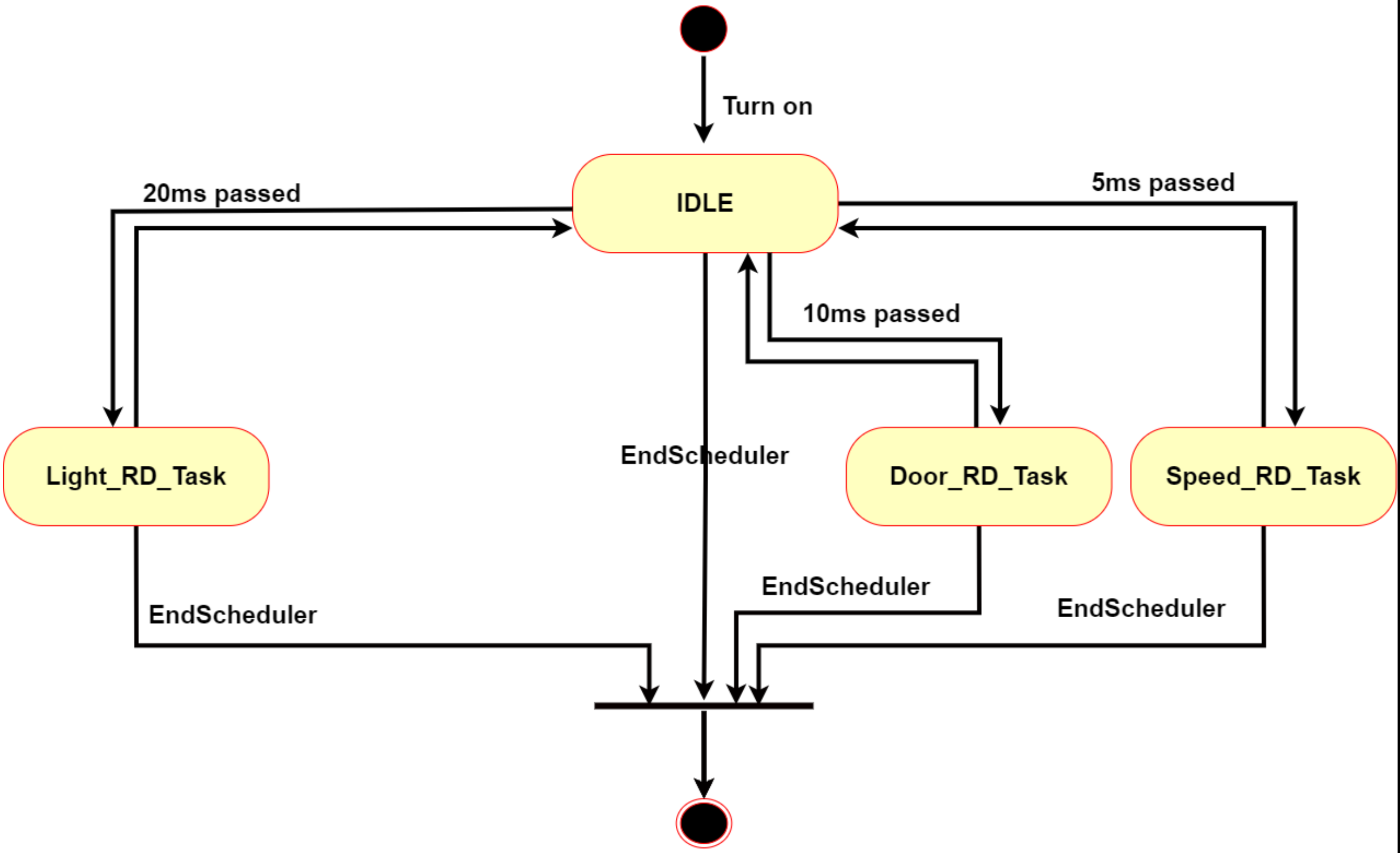




- **FOR ECU1:**

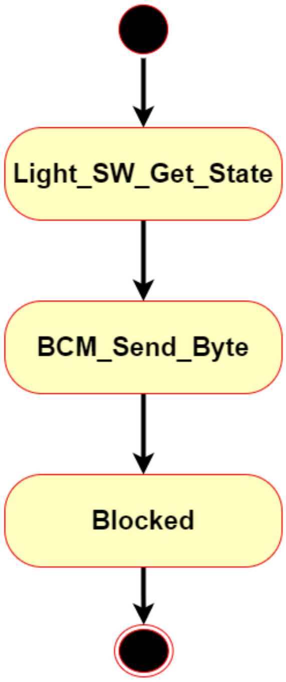


- **ECU1 Operation StateMachine :**

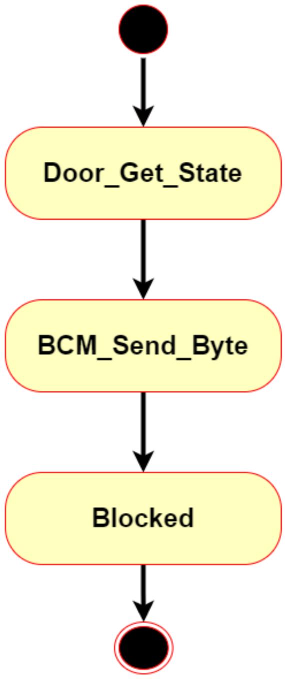


- **ECU1 Components:**

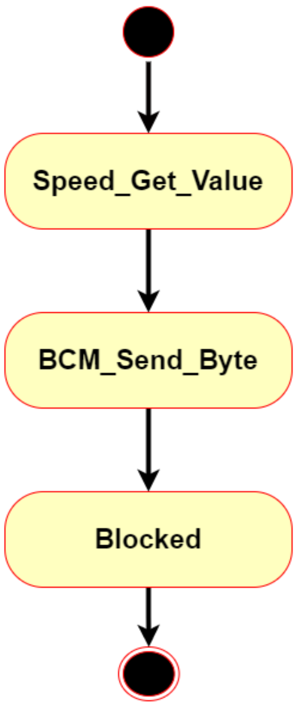
Light_RD_Task



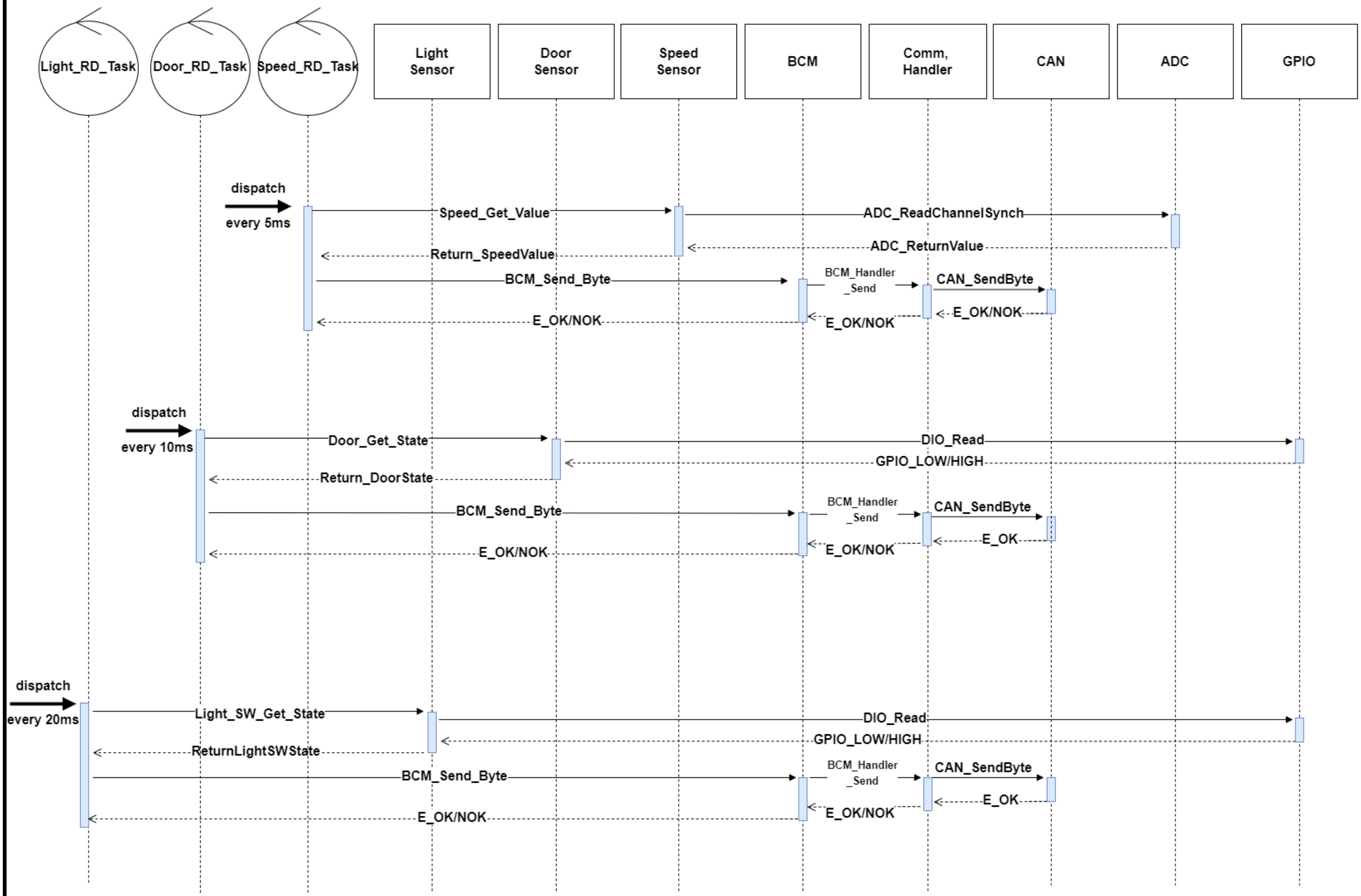
Door_RD_Task



Speed_RD_Task



▪ ECU1 Sequence Diagram:



▪ ECU1 CPU Load:

3 tasks (Light_RD_Task /Door_RD_Task/Speed_RD_Task)

CPU Load = (E1+E2+E3)/hyperperiod

$$=\frac{E1}{P1}+\frac{E2}{P2}+\frac{E3}{P3}$$

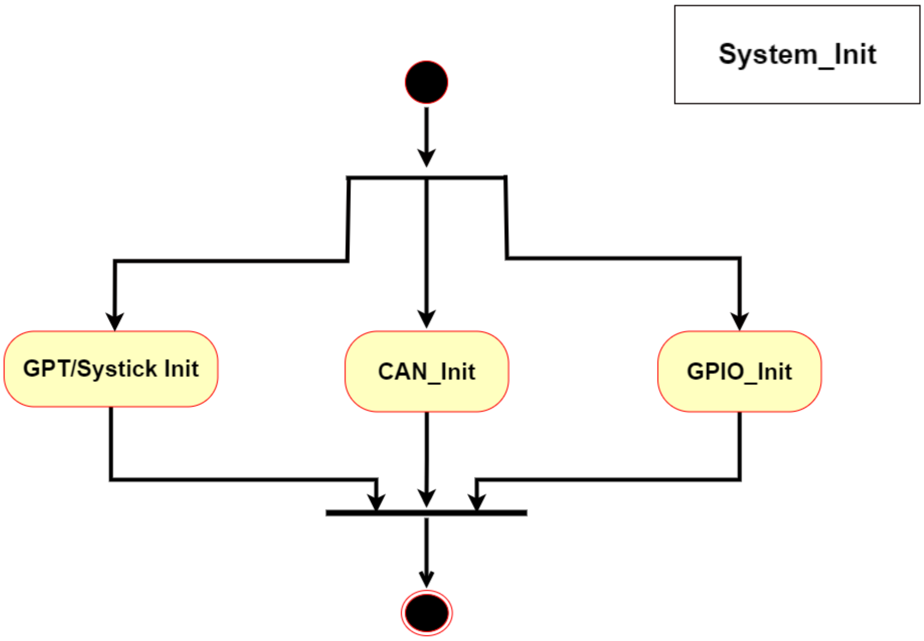
Assume E1 → 150 us

Assume E2 → 150 us

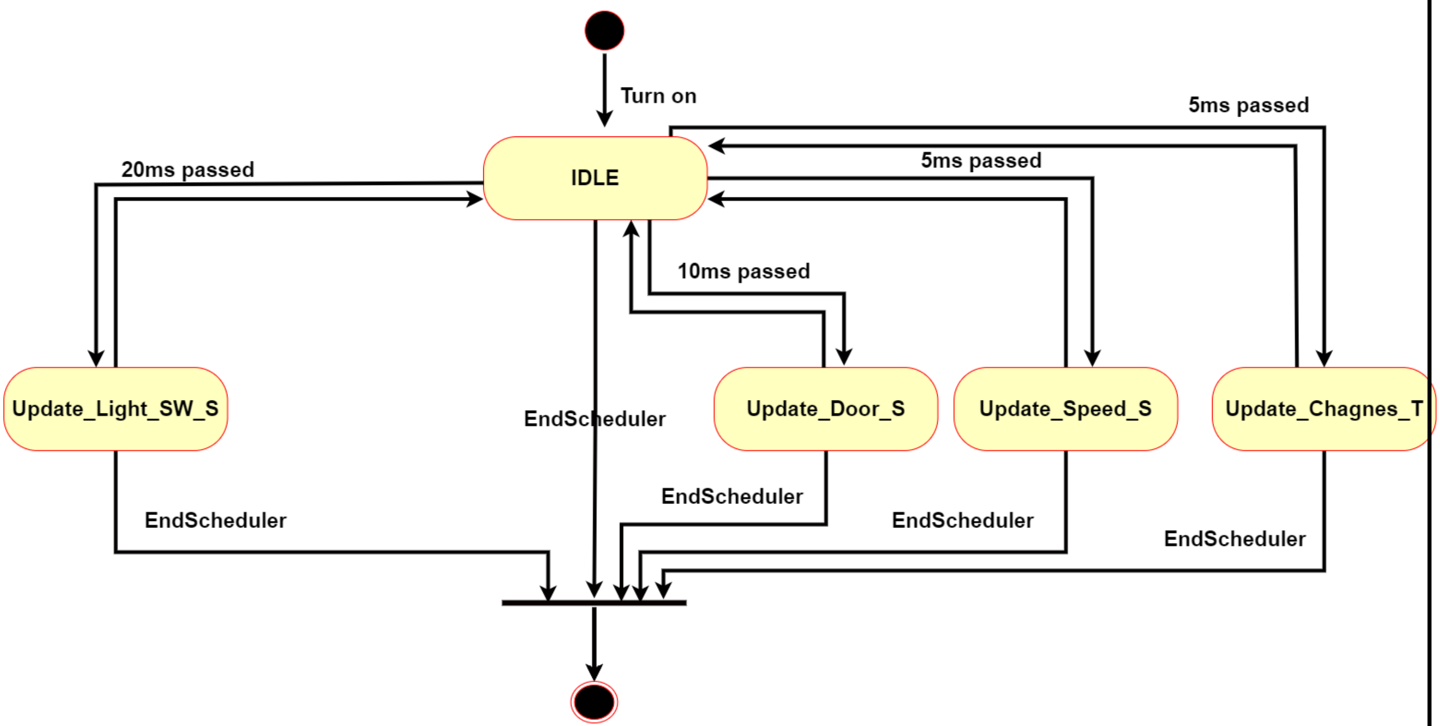
Assume E3 → 200 us

SO.. CPU Load $=\frac{150us}{5ms}+\frac{150us}{10ms}+\frac{200us}{20ms}$ = 5.5 %

- **FOR ECU2:**



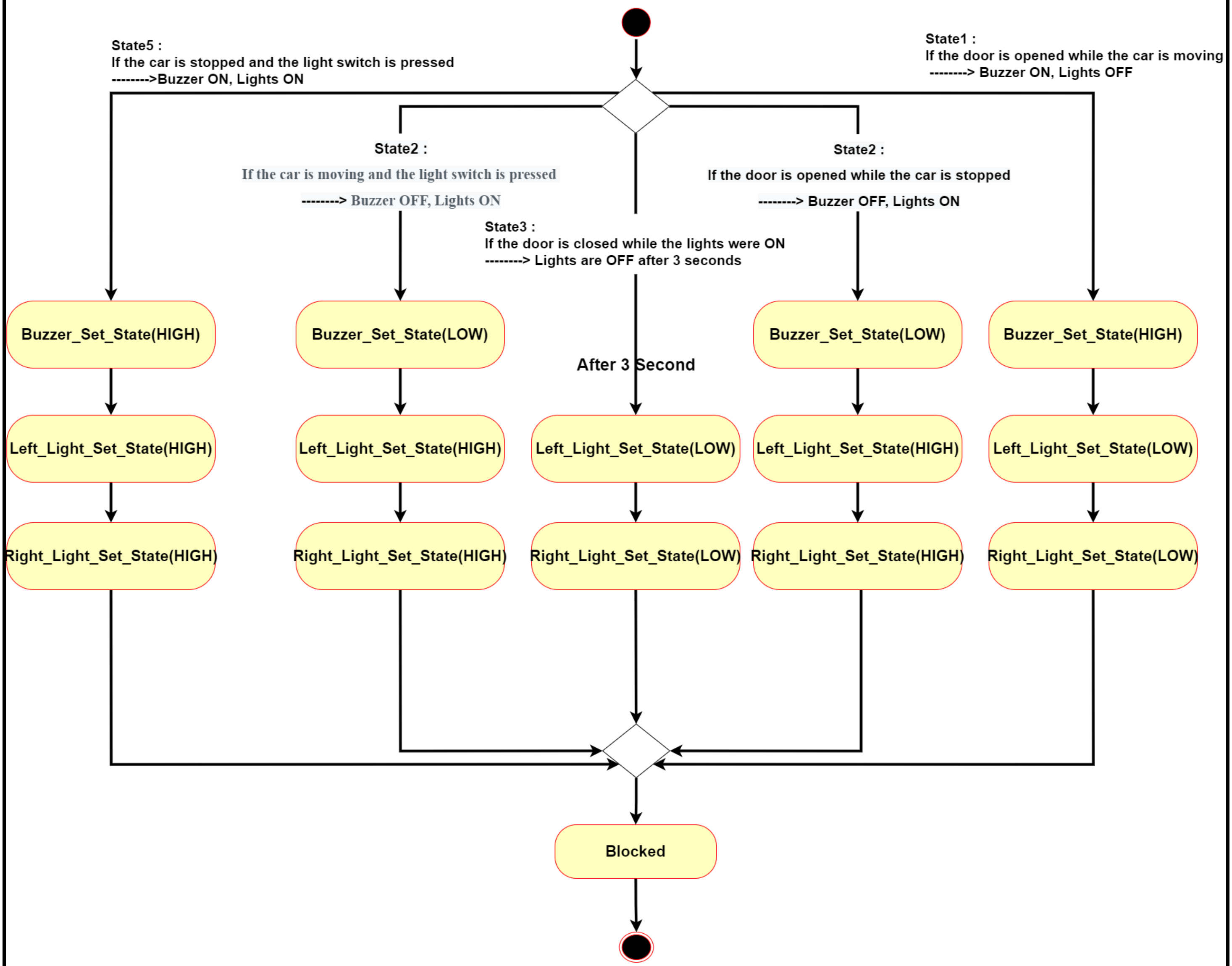
- **ECU2 Operation StateMachine :**

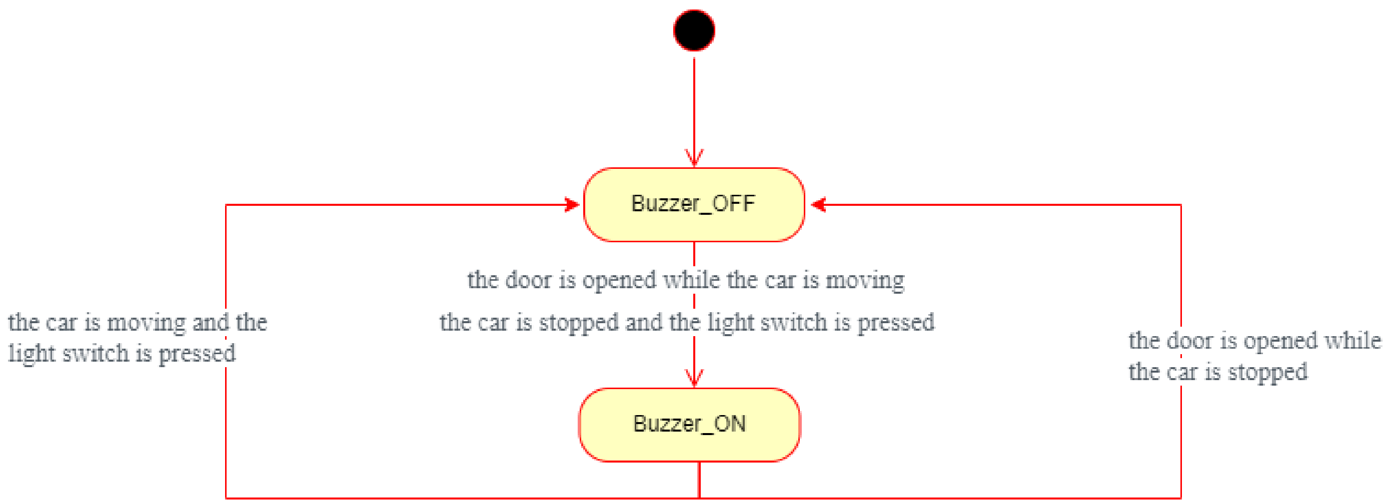


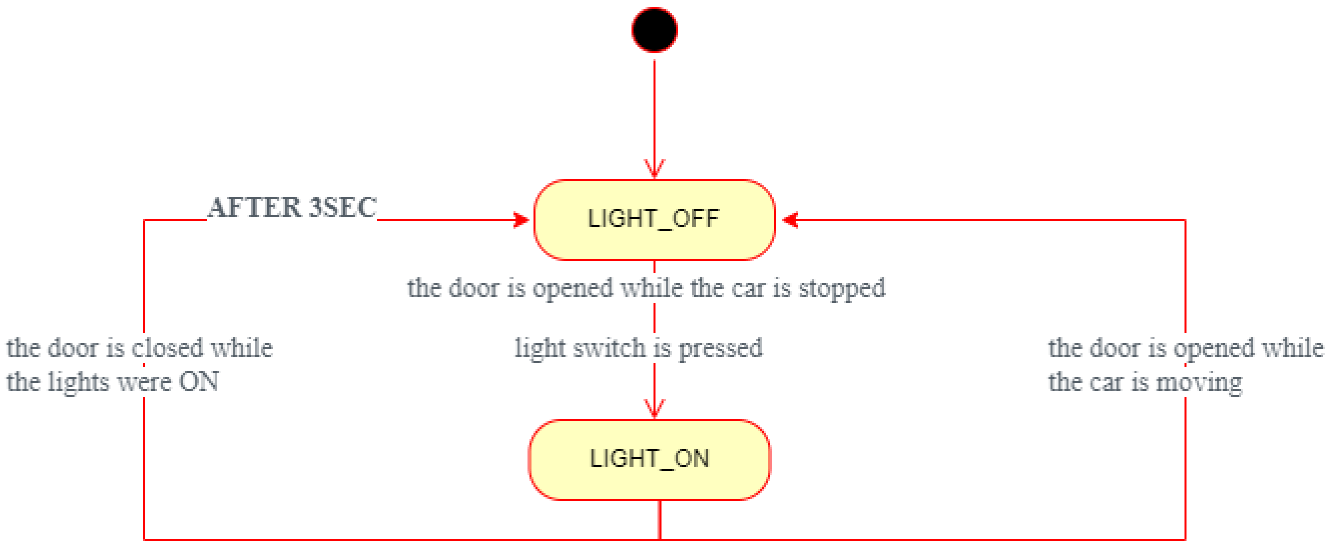
Note : Update_Changes_T has lesser priority than other tasks (still bigger than idle task)

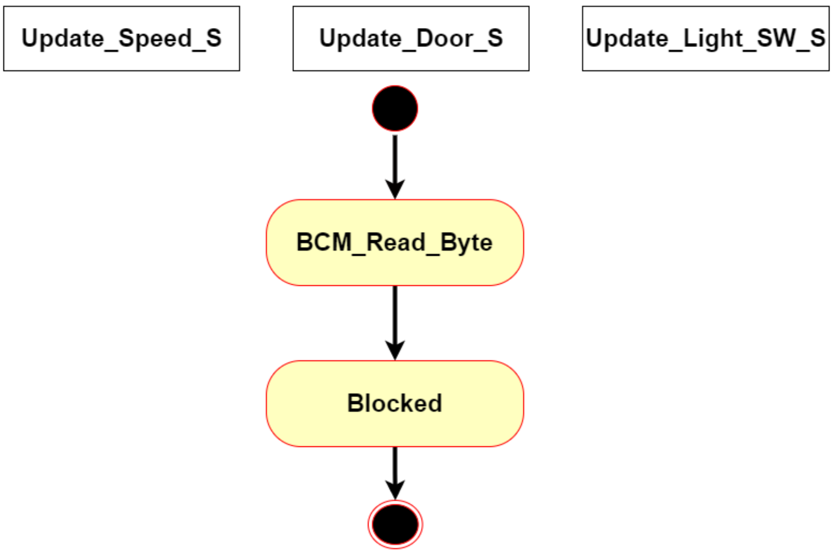
▪ ECU2 Components:

Update_Chagnes_T

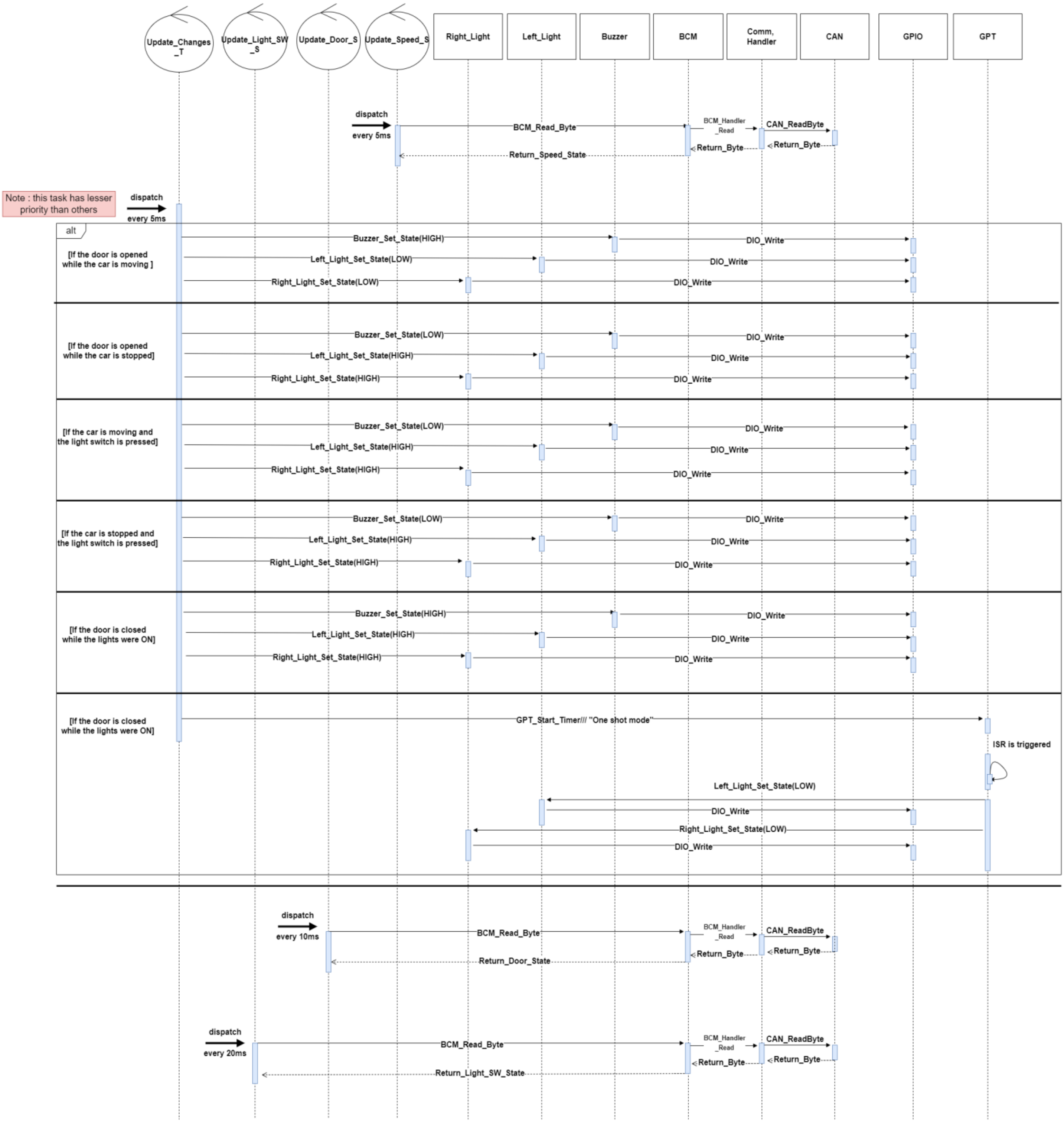





LIGHT_OFF
AFTER 3SEC
the door is opened while the car is stopped
light switch is pressed
the door is closed while
the lights were ON
the door is opened while
the car is moving
LIGHT_ON


Update_Speed_S
Update_Door_S
Update_Light_SW_S
BCM_Read_Byte
Blocked

- # ECU2 Sequence Diagram:



- **ECU2 CPU Load:**

4tasks(Update_Light_SW_S/Update_Door_S /Update_Speed_S/ Update_Changes_Task)

CPU Load = (E1+E2+E3+E4)/hyperperiod

$$=\frac{E1}{P1}+\frac{E2}{P2}+\frac{E3}{P3}+\frac{E4}{P4}$$

Assume E1 → 100 us

Assume E1 → 100 us

Assume E1 → 100 us

Assume E1 → 80 us

SO.. CPU Load $=\frac{100us}{5ms}+\frac{100us}{10ms}+\frac{100us}{20ms}+\frac{80us}{5ms}$ = 5.1 %

- **Bus Load:**

* Assume using 500 kBit/s bit rate

* Assuming standard identifier, a CAN frame consists of:

- 1 bit start bit.
- 11 bit identifier
- 1 bit RTR
- 6 bit control field
- 0 to 64 bit data field
- 15 bit CRC
- Bit stuffing is possible in the above, for every sequence of 5 consecutive bits of same level. Somewhere around 19 bits worst case.
- 3 bit delimiter, ack etc.
- 7 bit end of frame
- 3 bit intermission field after frame.

Ignoring stuffing, you have an overhead of 47 bits/frame.
If we make our data frame always 8 bit (byte send) so the maximum frame size will be = 47 +8 + 8 stuffing bits at worst case = 63 bit/frame

SO :-

bit time = 1/bit rate = 1/(500k) = 2us

Approx, time to transfer one frame = 2us * 63bit = 126 us

Having multiple sending intervals :

1 Frame every 5ms = 200 Frame every 1000 ms

1 Frame every 10ms = 100 Frame every 1000 ms

1 frame every 20ms = 50 Frame every 1000 ms

Total time on bus = 350 * 126 us

Bus load per 1 second = (350*126 us) / (1000ms) = 4.41 %