Ministry of Communications and Information Technology

egyptfwd

مستقبلنا رقمي

Sprints

JDACITY

Embedded System Advanced Track

EgFWD-Aug Cohort,2022

Embedded Software Design Graduation Project

Automotive Door Control System Design

Part 2

Fully Dynamic Design

Mohamed Hassaneen Ahmed

September, 2022

Content:

Introduction:

In the automotive industry, ECUs are communicating together to provide safety and driver's comfort. ECUs are microcontrollers connected with input and output devices, input devices to sense the surrounding environments, and output devices to perform actions according to readings that came from input devices.

Project Requirements

1. Provide Fully Static Design
2. Provide a Fully Dynamic design

For two ECUs communicating together to control car lights according to door state, light switch state, and car speed state.

Project Specification:

Dynamic design analysis	For ECU 1: 1. Draw a state machine diagram for each ECU component 2. Draw a state machine diagram for the ECU operation 3. Draw the sequence diagram for the ECU 4. Calculate CPU load for the ECU For ECU 2: 1. Draw a state machine diagram for each ECU component 2. Draw a state machine diagram for the ECU operation 3. Draw the sequence diagram for the ECU 4. Calculate CPU load for the ECU Calculate bus load in your system: With what percentage of system bus was busy per 1 second You should deliver a pdf file containing all your work and a video recording where you will discuss your work (maximum 5min long)

Flow Chart:

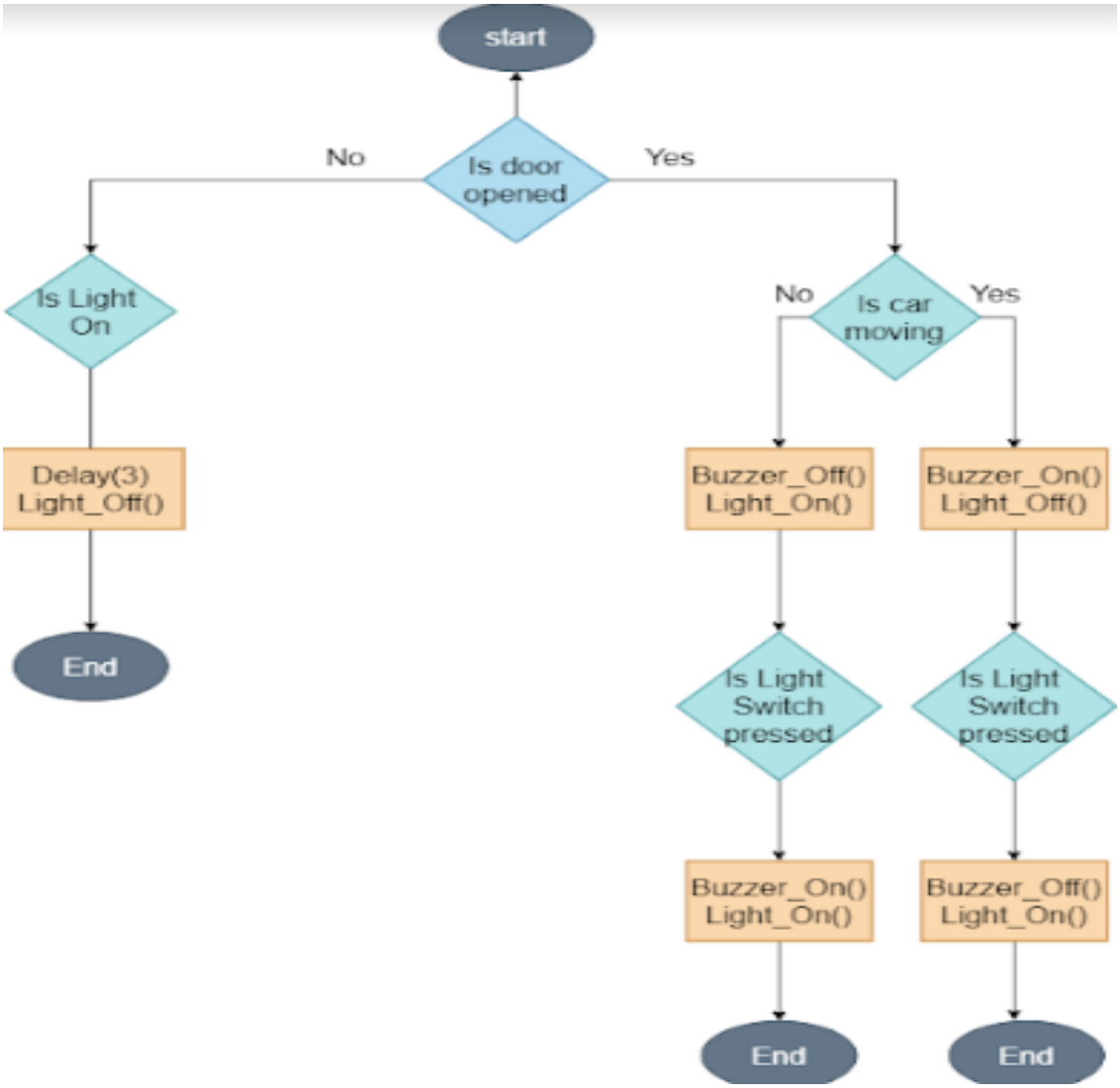
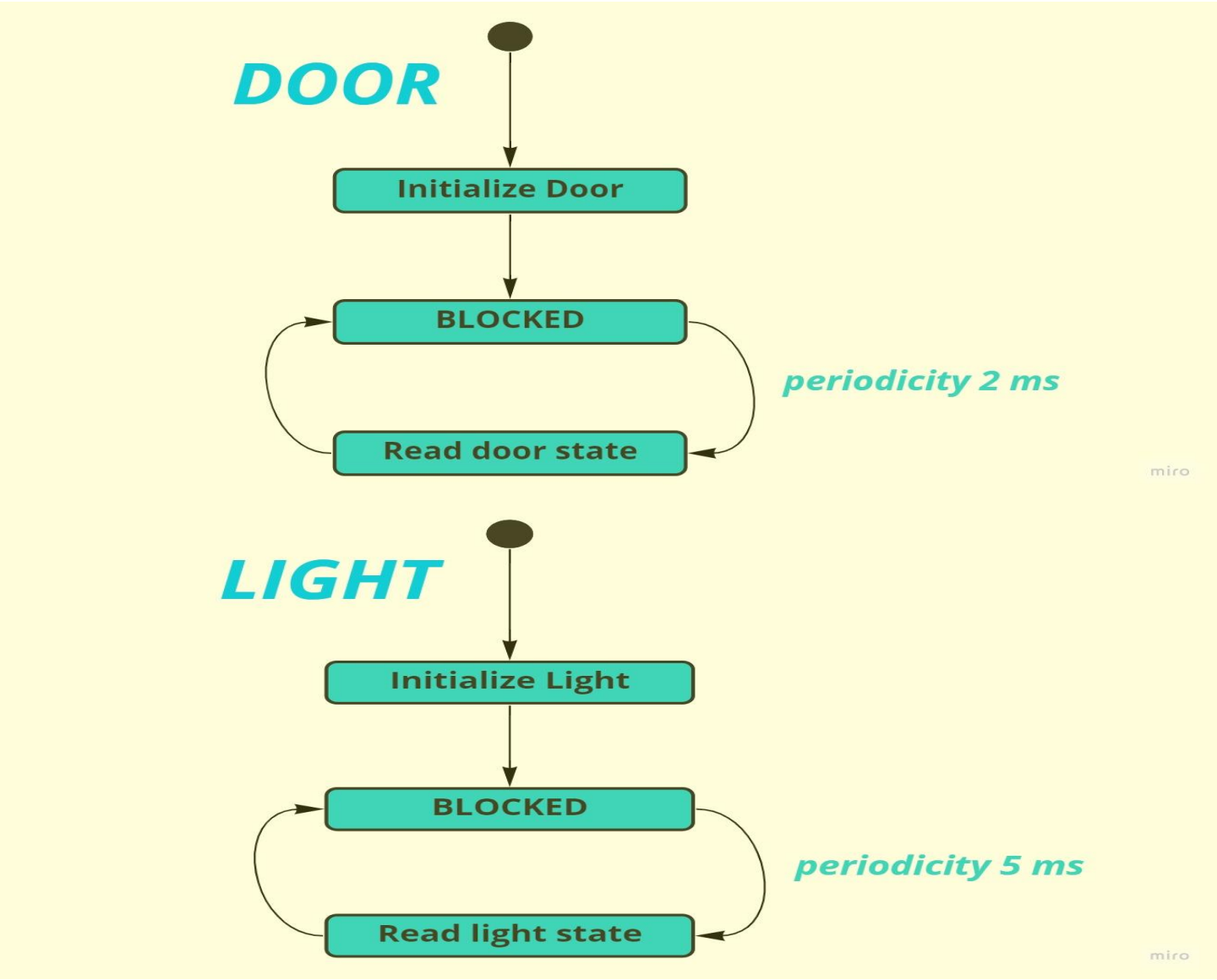
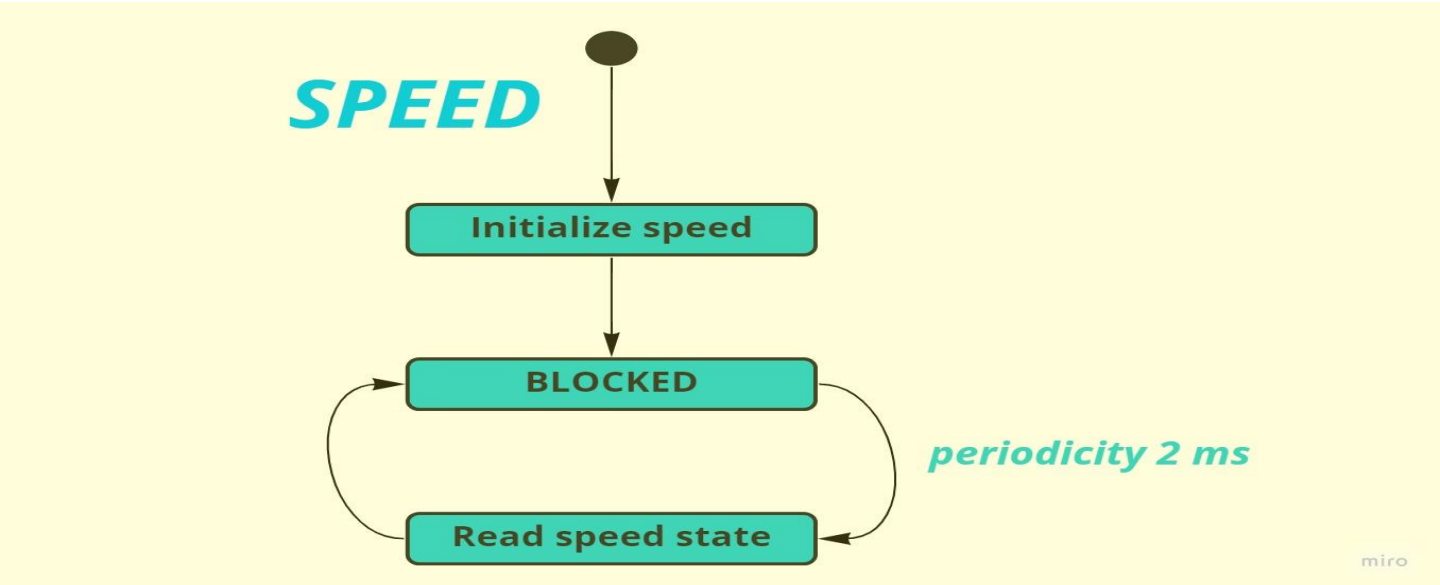
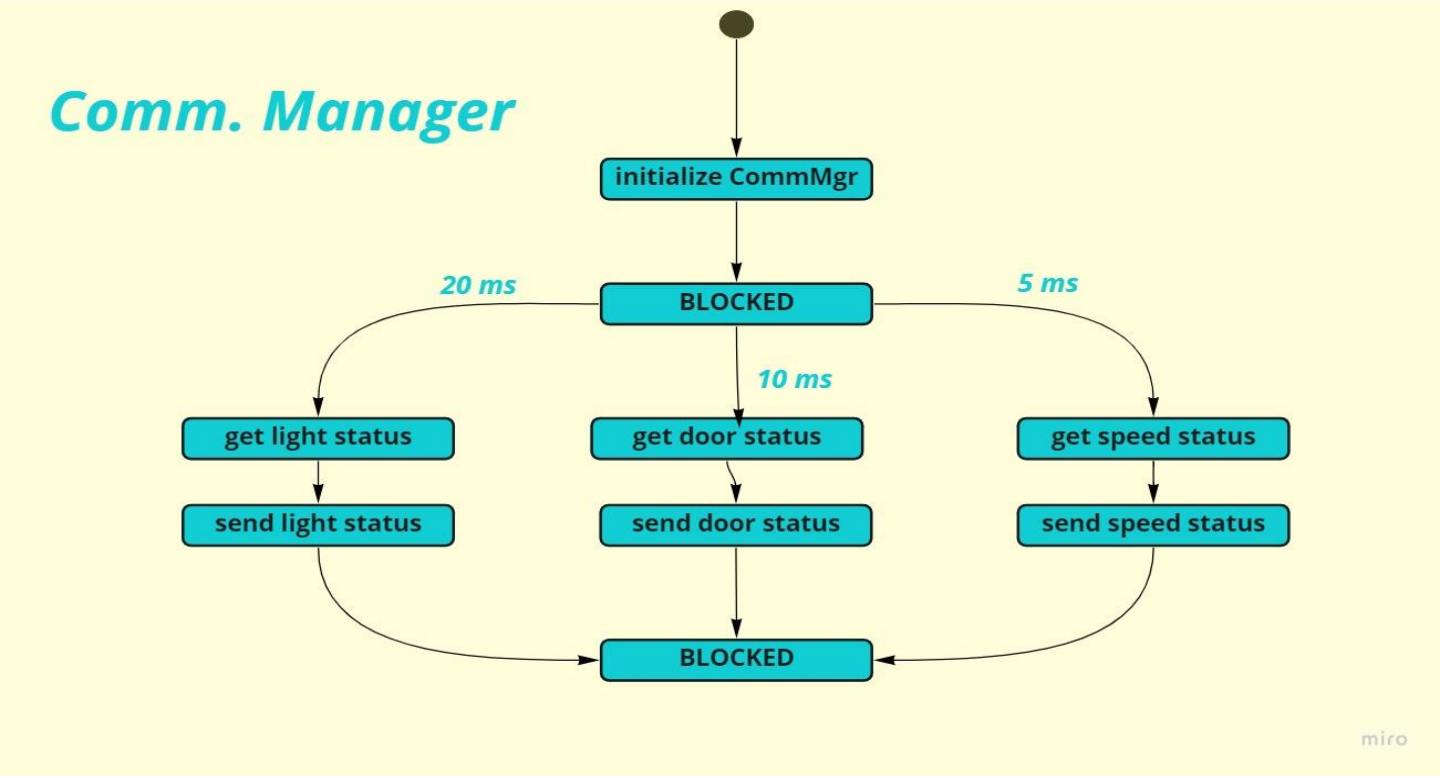


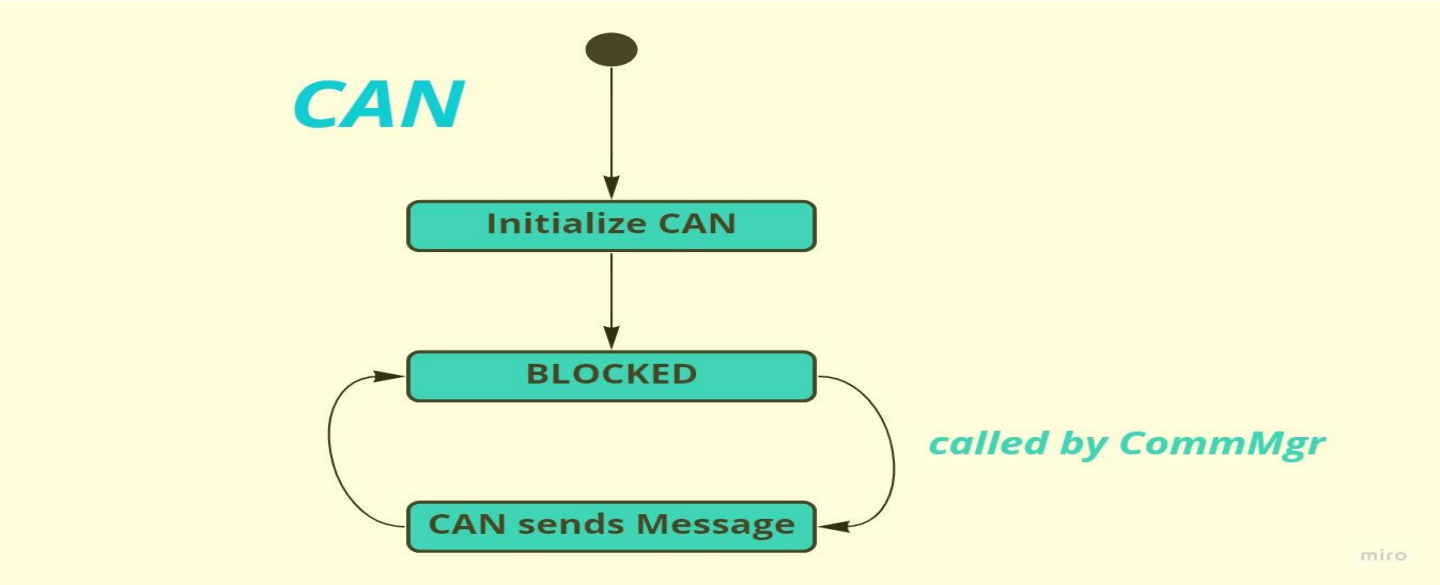
Figure 1.Flow Chart Diagram

State machine diagram for each ECU1 component:

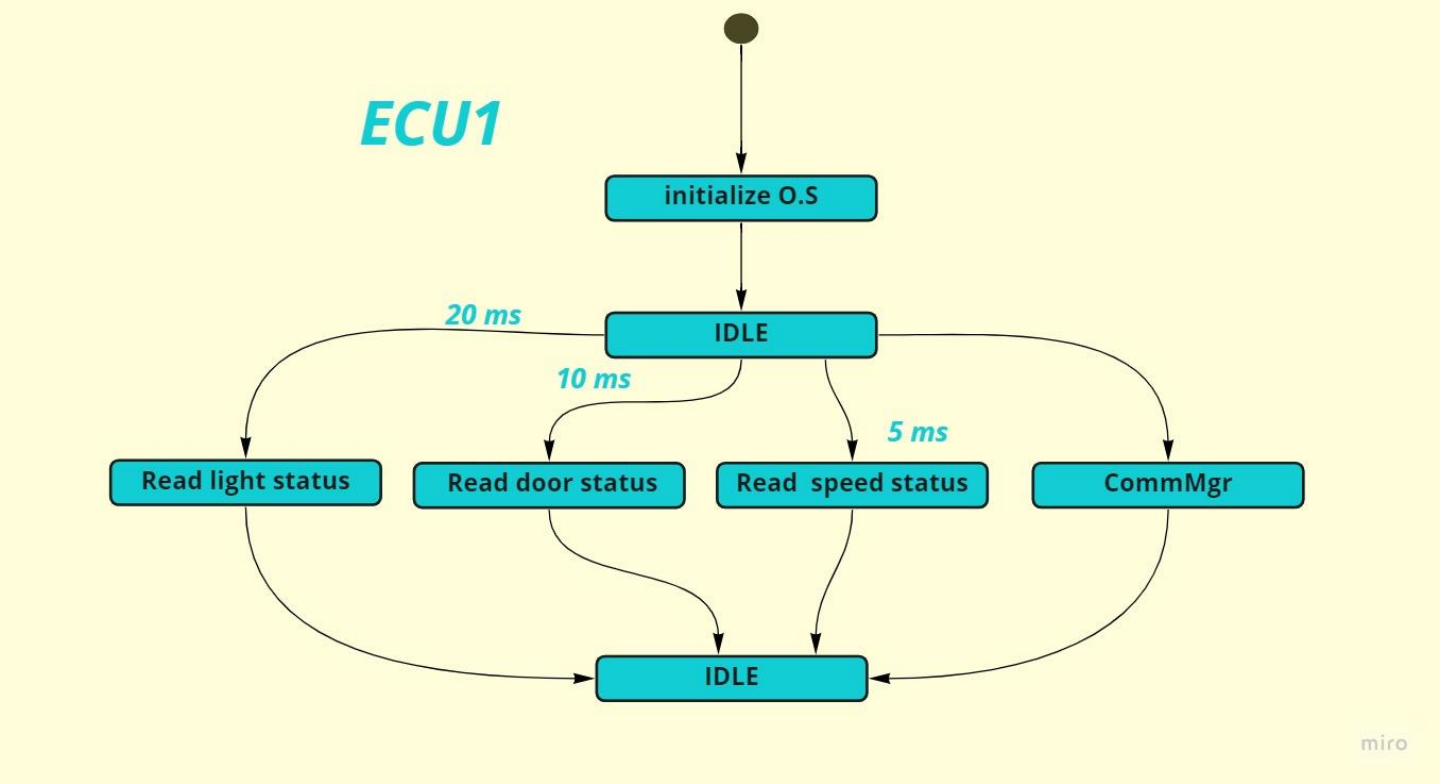





Comm. Manager
initialize CommMgr
20 ms
BLOCKED
5 ms
10 ms
get light status
get door status
get speed status
send light status
send door status
send speed status
BLOCKED
miro


CAN
Initialize CAN
BLOCKED
called by CommMgr
CAN sends Message
miro

State machine diagram for the ECU1 operation:



Sequence diagram for the ECU1:

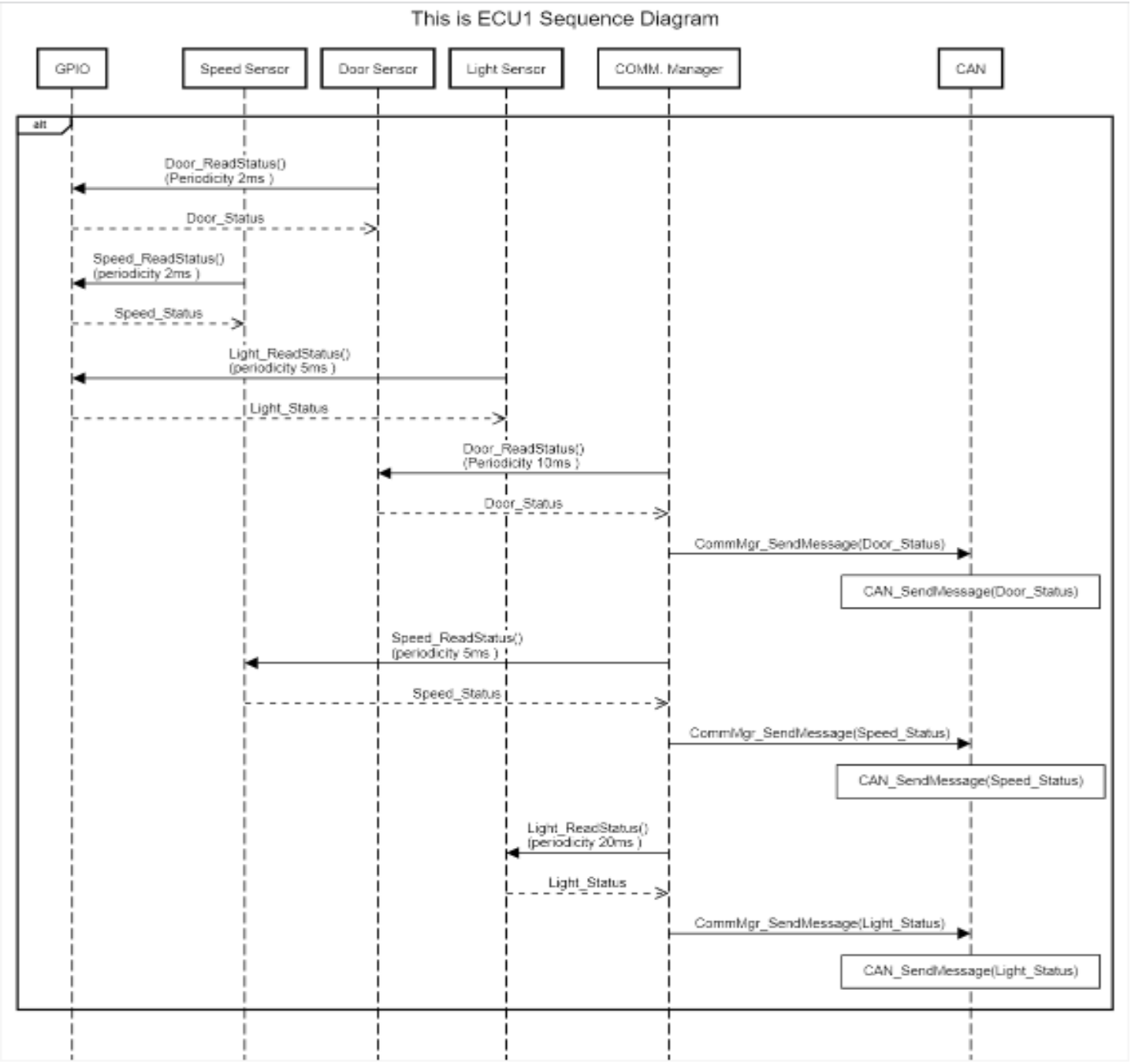


Figure 2.Sequence Diagram ECU1

```
title This is ECU1 Sequence Diagram
alt

participant GPIO

participant Speed Sensor

participant Door Sensor

participant Light Sensor

participant COMM. Manager

participant CAN



//------------------------------------------------------------------
Door Sensor->GPIO:Door_ReadStatus()\n(Periodicity 2ms )
GPIO -->> Door Sensor : Door_Status

Speed Sensor->GPIO:Speed_ReadStatus()\n(periodicity 2ms )
GPIO -->> Speed Sensor : Speed_Status


Light Sensor->GPIO :Light_ReadStatus()\n(periodicity 5ms )
GPIO -->> Light Sensor : Light_Status
//------------------------------------------------------------------

COMM. Manager->Door Sensor :Door_ReadStatus()\n(Periodicity 10ms )
Door Sensor -->> COMM. Manager : Door_Status
COMM. Manager->CAN:CommMgr_SendMessage(Door_Status)
box over CAN  : CAN_SendMessage(Door_Status)

COMM. Manager->Speed Sensor :Speed_ReadStatus()\n(periodicity 5ms )
Speed Sensor -->> COMM. Manager : Speed_Status
COMM. Manager->CAN:CommMgr_SendMessage(Speed_Status)
box over CAN  : CAN_SendMessage(Speed_Status)


COMM. Manager->Light Sensor :Light_ReadStatus()\n(periodicity 20ms )
Light Sensor -->> COMM. Manager : Light_Status
COMM. Manager->CAN:CommMgr_SendMessage(Light_Status)
box over CAN  : CAN_SendMessage(Light_Status)
//------------------------------------------------------------------


end
```

Figure 3.Code of Sequence Diagram ECU1

CPU load for the ECU1:

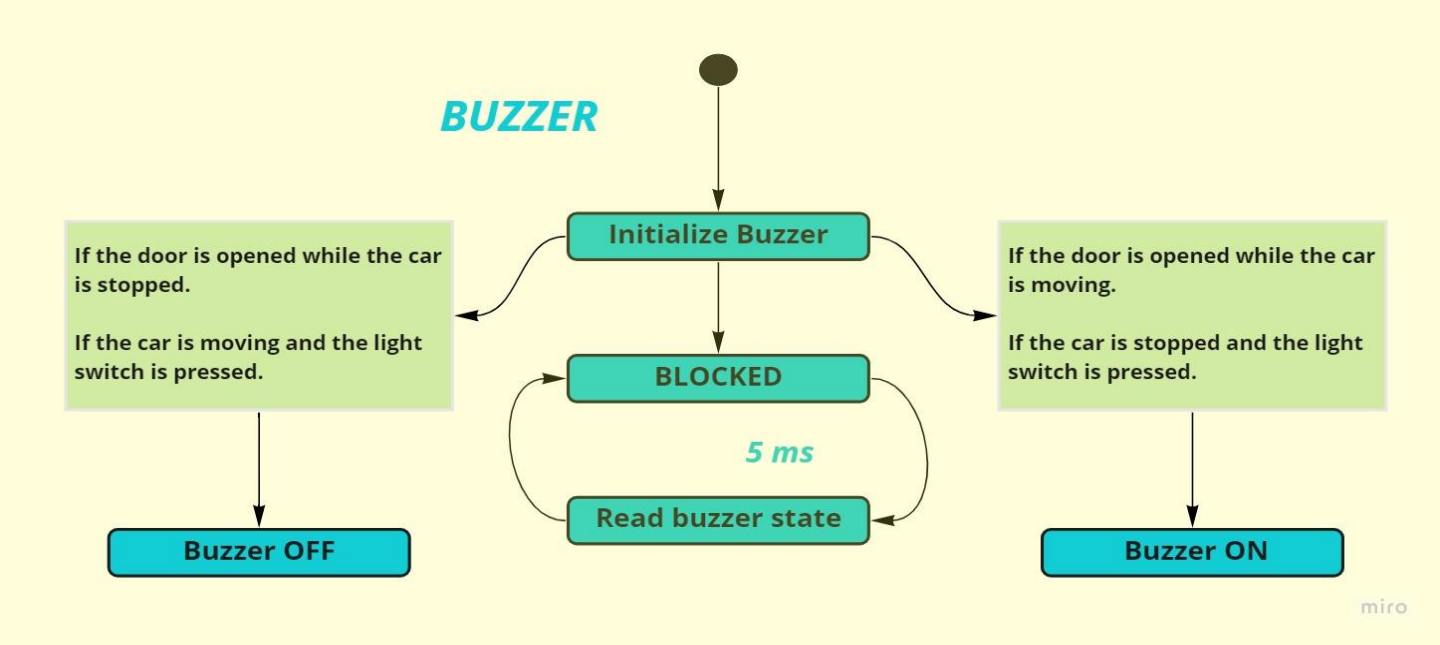
Task Name	Periodicity(ms)	Execution Time (Assumption)(ms)
Light_ReadStatus	5	0.05
Door_ReadStatus	2	0.05
Speed_ReadStatus	2	0.05
CommMgr_SendMessage(light)	20	2
CommMgr_SendMessage(door)	10	2
CommMgr_SendMessage(speed)	5	2

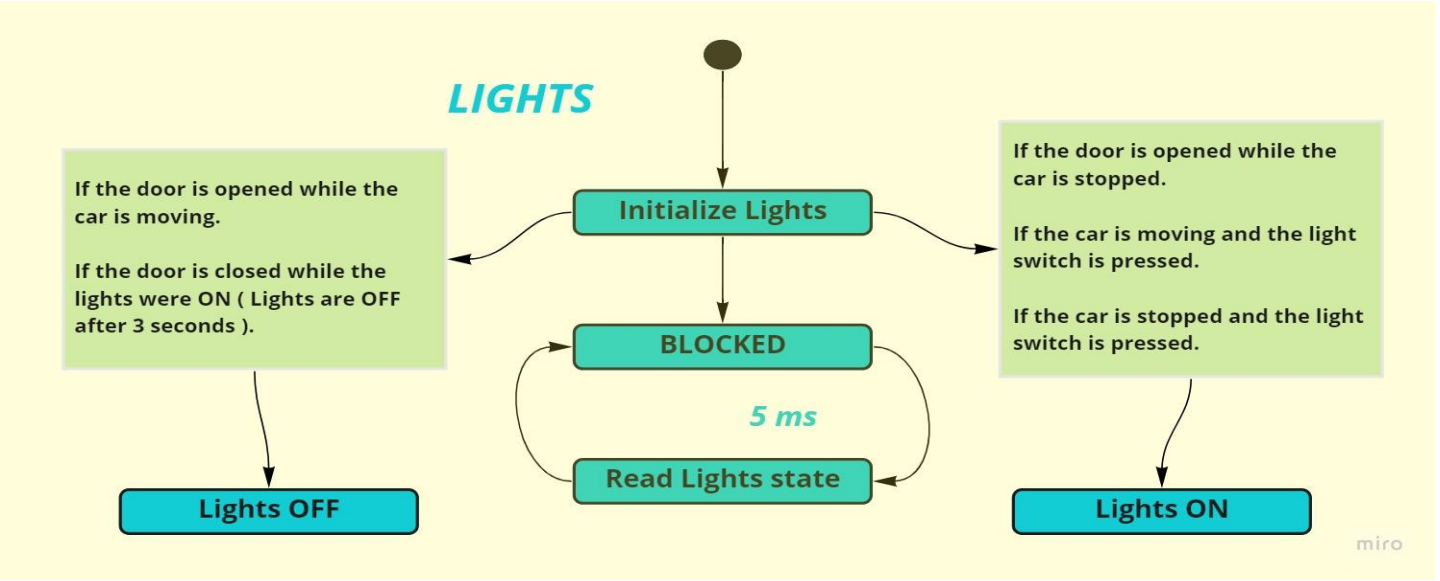
Hyper period = 20 ms.

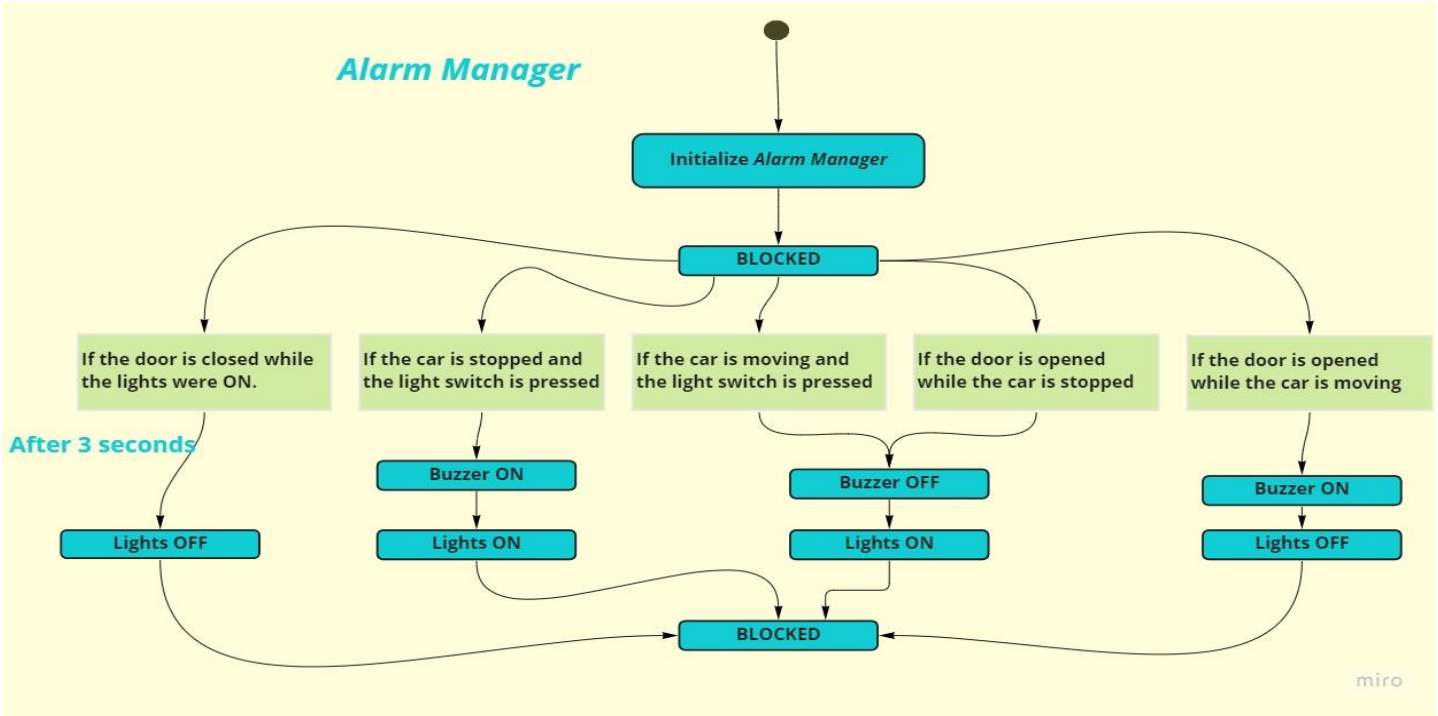
Execution Time = (0.05*4) + (0.05*10) + (0.05*10) + (2* 4) + (2* 2) + (2* 1) = 15.2 ms.

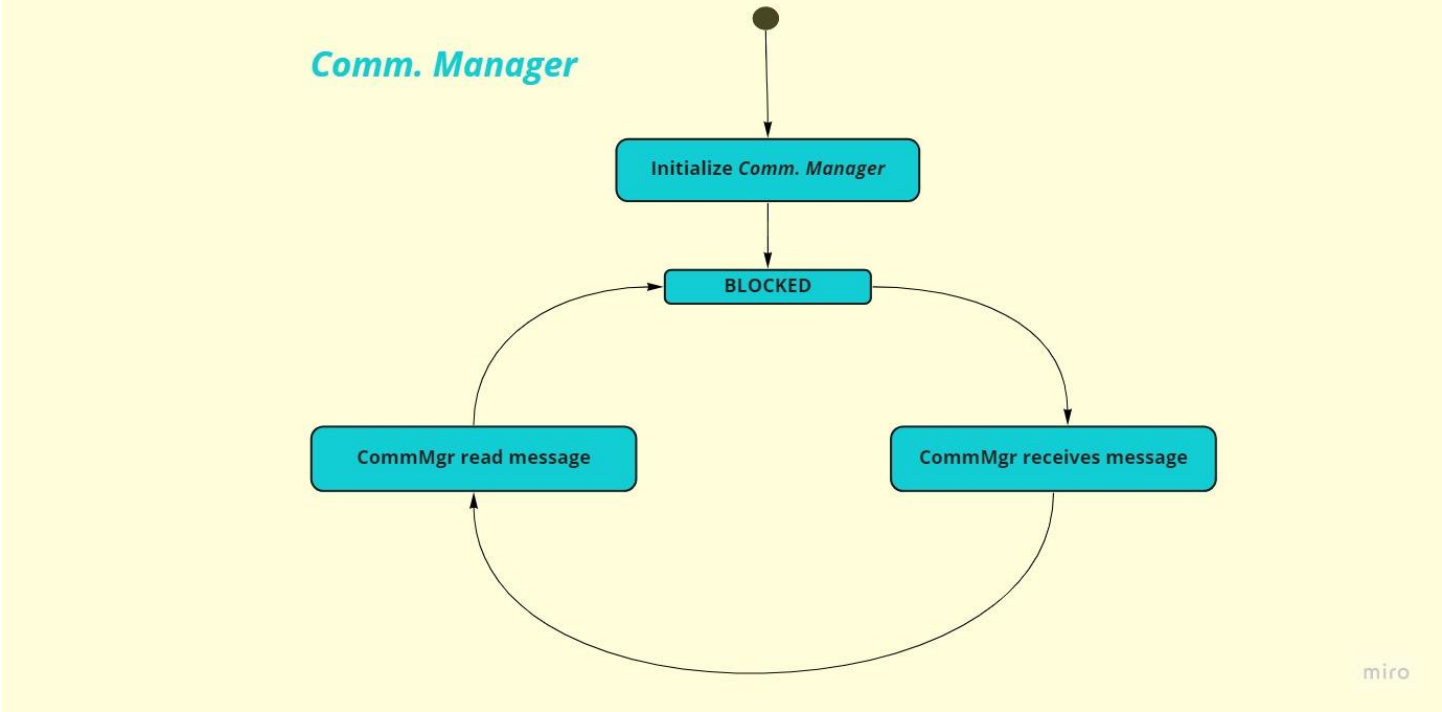
CPU Load (ECU1) = 15.2 / 20 = 76%

State machine diagram for each ECU2 component:

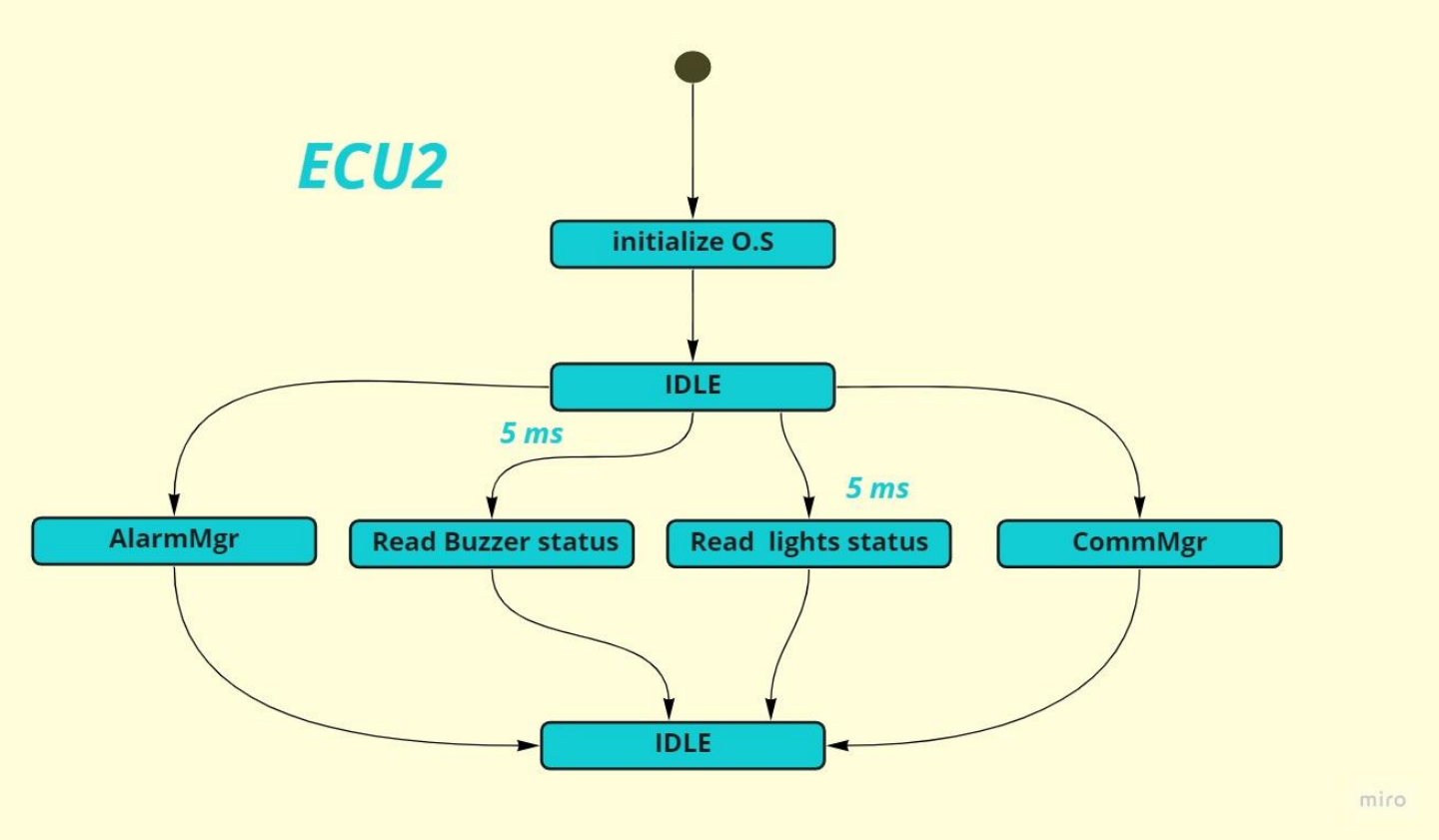





Alarm Manager
Initialize Alarm Manager
BLOCKED
If the door is closed while the lights were ON.
If the car is stopped and the light switch is pressed
If the car is moving and the light switch is pressed
If the door is opened while the car is stopped
If the door is opened while the car is moving
After 3 seconds
Buzzer ON
Buzzer OFF
Buzzer ON
Lights OFF
Lights ON
Lights ON
Lights OFF
BLOCKED
miro


Comm. Manager
Initialize Comm. Manager
BLOCKED
CommMgr read message
CommMgr receives message
miro

State machine diagram for the ECU2 operation:



Sequence diagram for the ECU2:

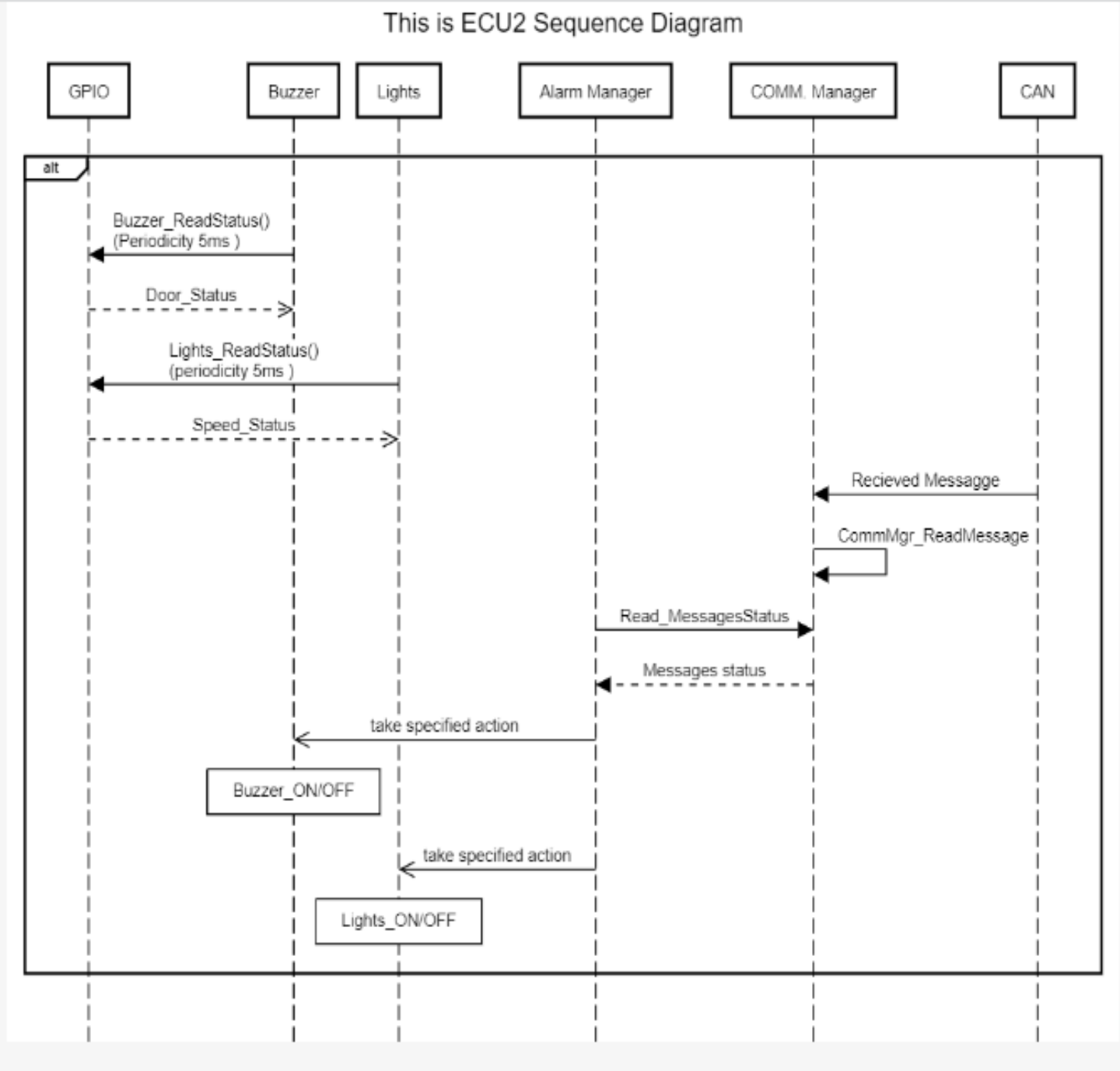


Figure 4.Sequence diagram for the ECU2

```
title This is ECU2 Sequence Diagram
alt

// ADD elements

participant GPIO
participant Buzzer
participant Lights
participant Alarm Manager
participant COMM. Manager
participant CAN

//-----------------------------------------------------------------
Buzzer->GPIO:Buzzer_ReadStatus()\n(Periodicity 5ms )
GPIO -->> Buzzer : Door_Status

Lights->GPIO:Lights_ReadStatus()\n(periodicity 5ms )
GPIO -->> Lights : Speed_Status

//-----------------------------------------------------------------
CAN  ->COMM. Manager: Recieved Messagge
COMM. Manager ->COMM. Manager: CommMgr_ReadMessage

Alarm Manager ->COMM. Manager: Read_MessagesStatus
COMM. Manager --> Alarm Manager :Messages status
Alarm Manager  ->>Buzzer: take specified action
box over Buzzer : Buzzer_ON/OFF

Alarm Manager  ->>Lights: take specified action
box over Lights: Lights_ON/OFF

end
```

Figure 5.Code of Sequence diagram for the ECU2

CPU load for the ECU2:

Task Name	Periodicity(ms)	Execution Time (Assumption)(ms)
Light_ReadStatus	5	0.05
Buzzer_ReadStatus()	5	0.05
AlarmMgr_Task	5	1
CommMgr_RecieveMessage(light)	20	1
CommMgr_RecieveMessage(door)	10	1
CommMgr_RecieveMessage(speed)	5	1
CommMgr_ReadMessage() (No. of received messages=7)	No. of received messages=7	0.05

Hyper period = 20 ms.

Execution Time = (0.05*4) +(0.05*4) + (1*4) +(1*1) + (1* 2) + (1* 4) + (0.05* 7) = 11.75 ms.

CPU Load (ECU2) = 11.75 / 20 = 58.75%.

Bus load:

Assume a complete message takes one mille second to send.

Total time of messages = 4+2+1 = 7ms / hyper period (20 ms).

Bus Load = 7/20 = 35 % Second