SIMULATION & IMPLEMENTATION TOOLS

This Project is Firstly Designed and verified using “TTool” then implemented using simple “Editor”

Build & Compiled → Native MinGW Tool-Chain

Debugger → Native GDB Debugger

BY

Omar Ahmed Hassan

CONTACT US

Phone| (+20) 1141563470 |
omarsaudiy@gmail.com
github.com/OmarSaudiy
linkedin.com/in/omarsaudiy

PROJECT NAME

OBJECT DETECTION

An embedded system object detection project involves designing and implementing a system that can detect objects in-front and stop vehicle accordingly.

SYSTEMARCHITECTURE

- CASE STUDY

System detects the distance between the vehicle and objects in front of it. Then decides whether to stop or to continue according to compare with the threshold value.

Assuming power, sensors and actuators never fail.

- METHODOLOGY

Water-fall Model for 2 reasons

1. I am both Customer and Developer, that's grantee there is not any misunderstanding between the end-user and the developer.
2. The Water-fall Model suffers from teams' dependency and blocking; However, I am working alone and that means whatever Model I choose there will be blocking anyway.

- REQUIREMENTS

The Requirements Diagram is identified next page.

- SPACE-EXPLORATION/ PARTITIONING

We will use Native Host Target x86 Processor for this project.

- SYSTEM ANALYSIS

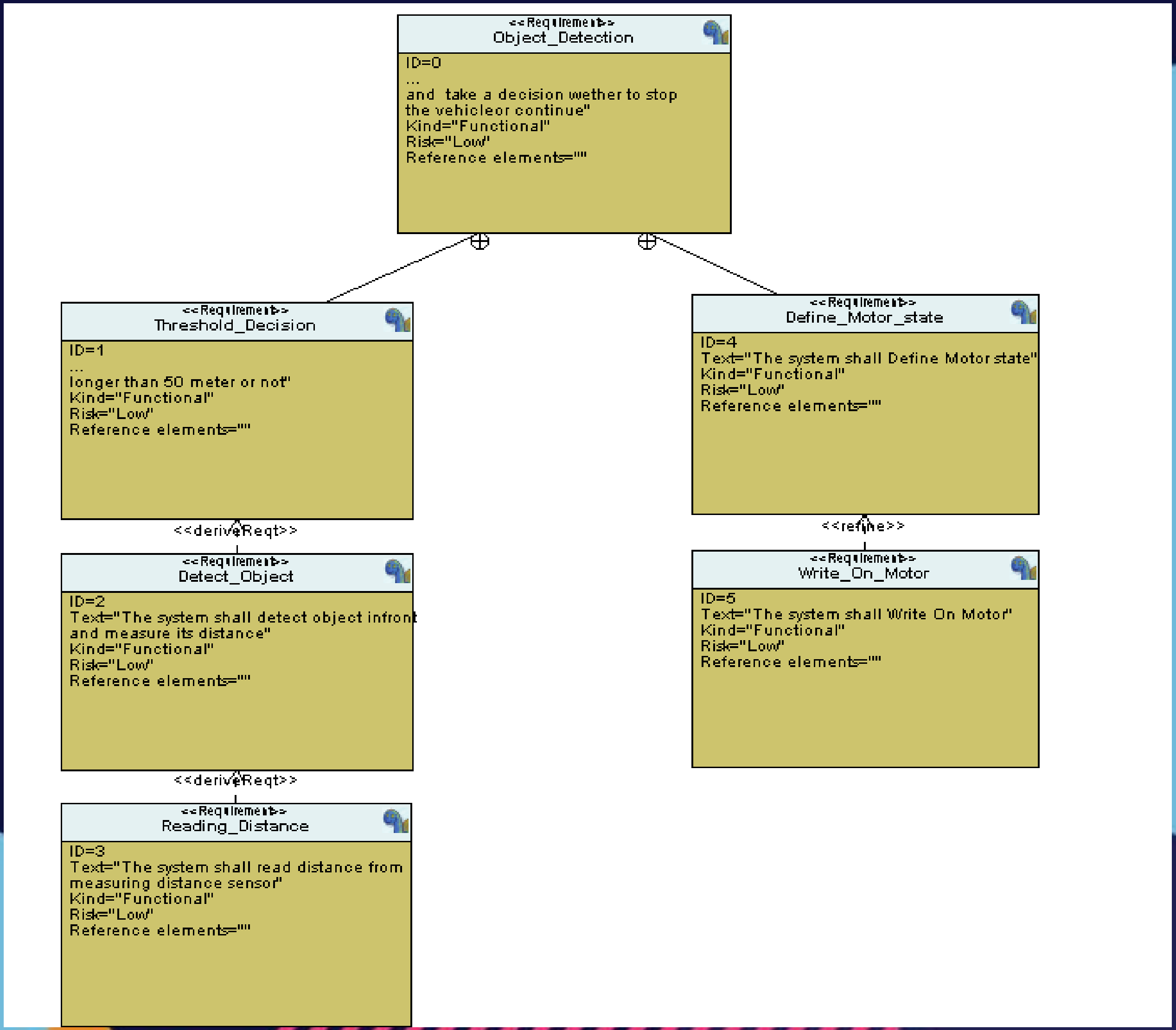
The system analysis Diagrams are identified in the following pages.

- SYSTEM DESIGN

The system Design Diagrams are identified in the following pages.

> **"Learn in depth and you will be deeply happy."**
>
> \- Omar Ahmed -

REQUIREMENTS DIAGRAM



<<Requirement>>
Object_Detection
ID=0
...
and take a decision wether to stop
the vehicleor continue"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Threshold_Decision
ID=1
...
longer than 50 meter or not"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Detect_Object
ID=2
Text="The system shall detect object infront
and measure its distance"
Kind="Functional"
Risk="Low"
Reference elements=""

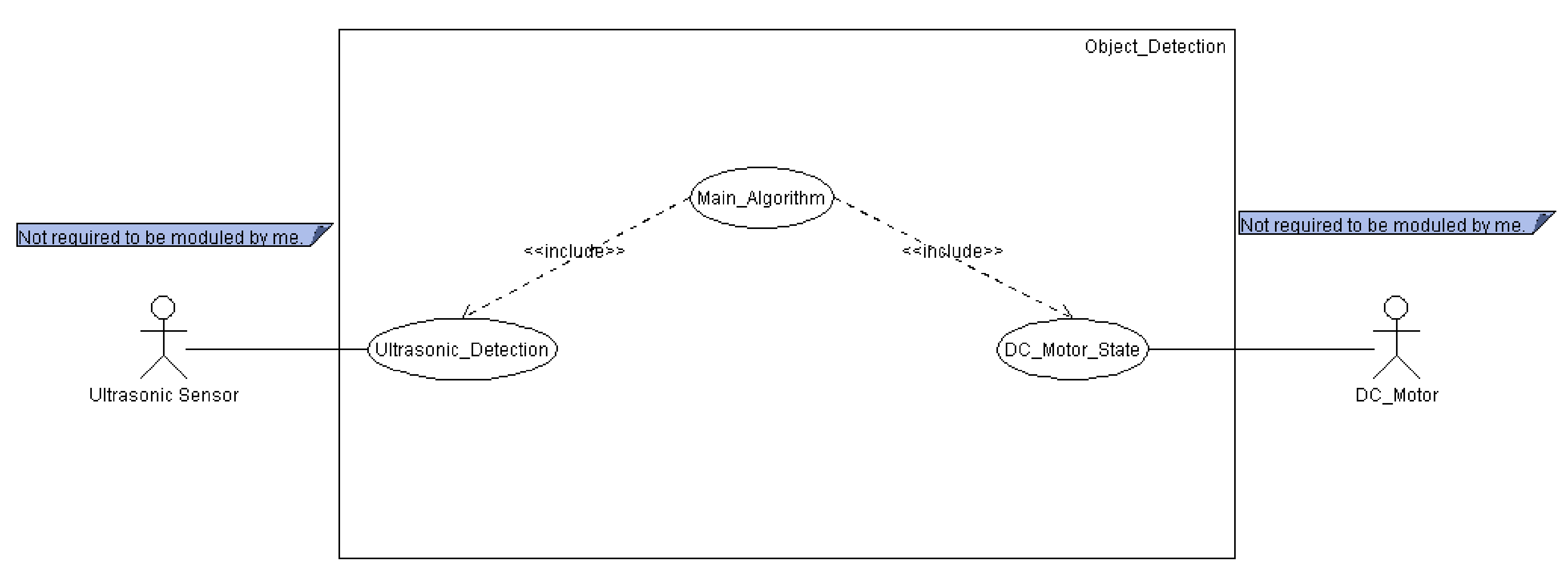
<<Requirement>>
Reading_Distance
ID=3
Text="The system shall read distance from
measuring distance sensor"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Define_Motor_state
ID=4
Text="The system shall Define Motor state"
Kind="Functional"
Risk="Low"
Reference elements=""

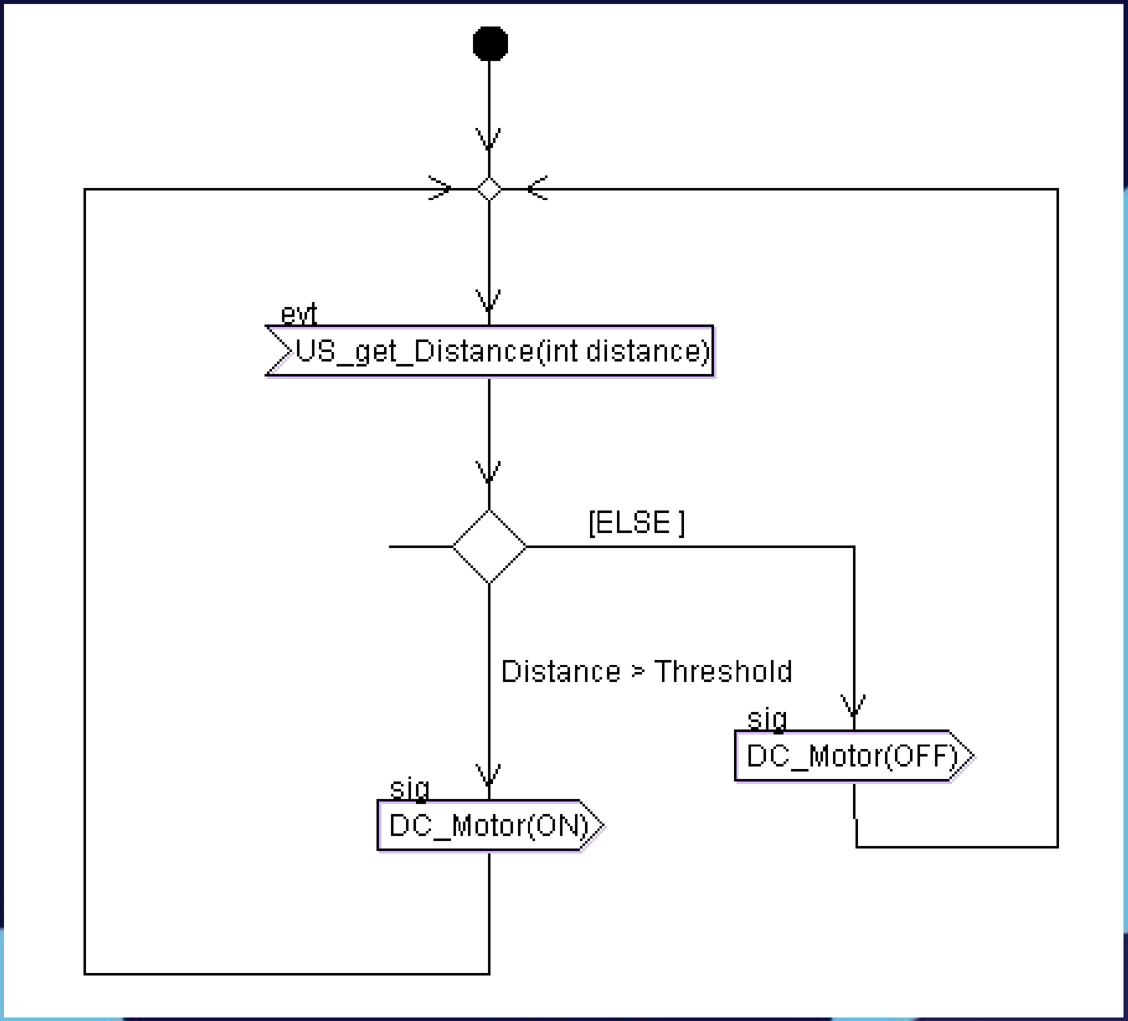
<<Requirement>>
Write_On_Motor
ID=5
Text="The system shall Write On Motor"
Kind="Functional"
Risk="Low"
Reference elements=""

SYSTEM ANALYSIS

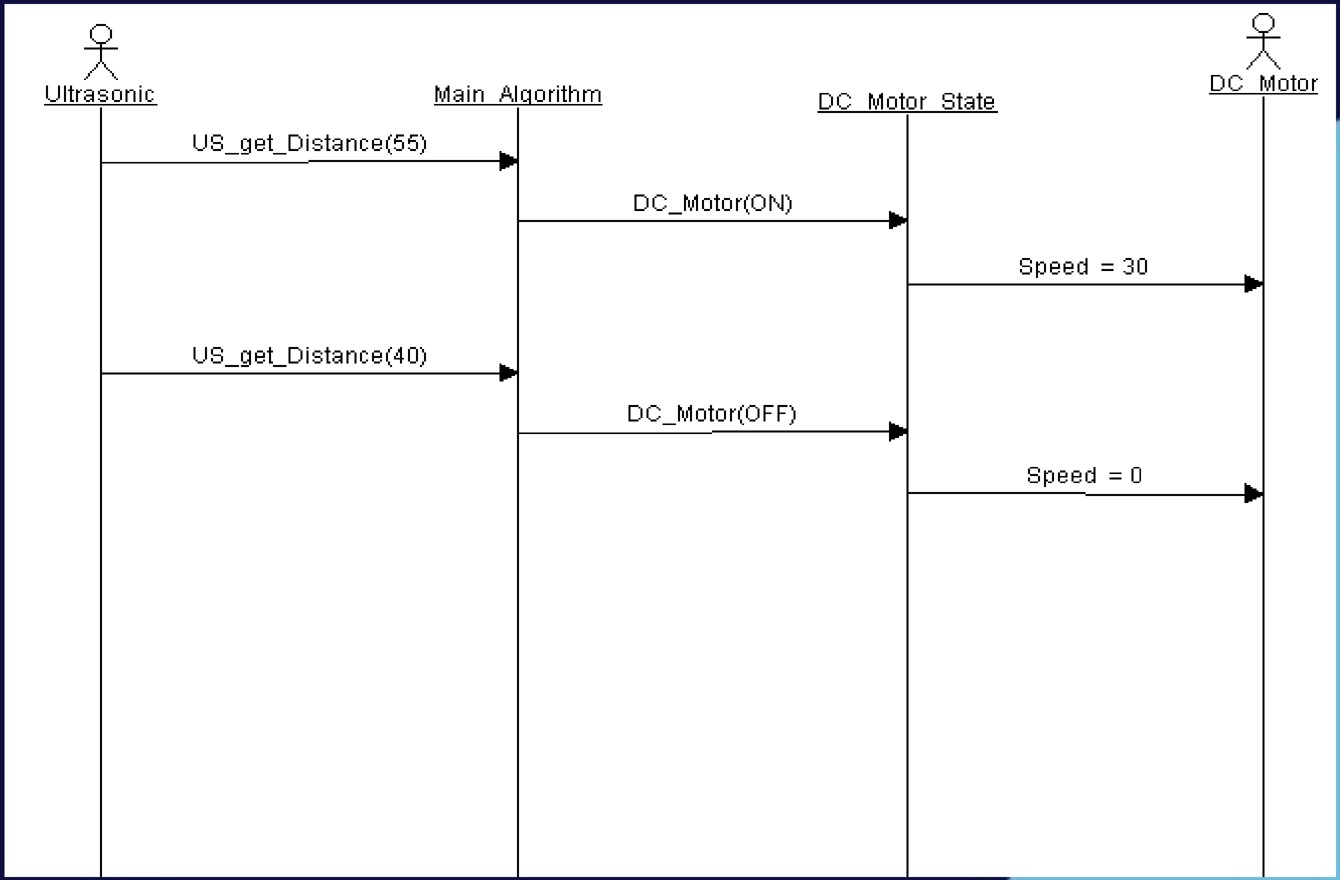
- USE CASE DIAGRAM



- ACTIVITY DIAGRAM

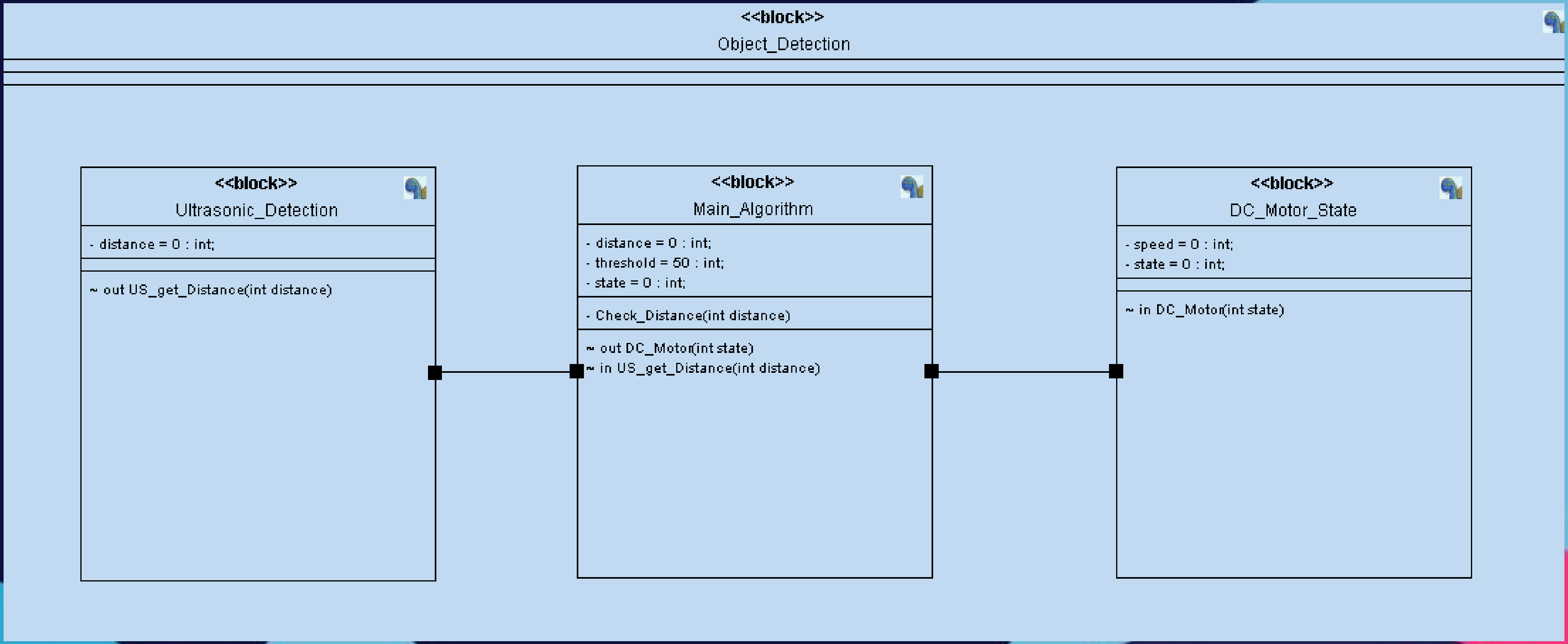


- SEQUENCE DIAGRAM



SYSTEM DESIGN

- BLOCK DIAGRAM



<<block>>
Object_Detection

<<block>>
Ultrasonic_Detection

- distance = 0 : int;

~ out US_get_Distance(int distance)

<<block>>
Main_Algorithm

- distance = 0 : int;
- threshold = 50 : int;
- state = 0 : int;

- Check_Distance(int distance)

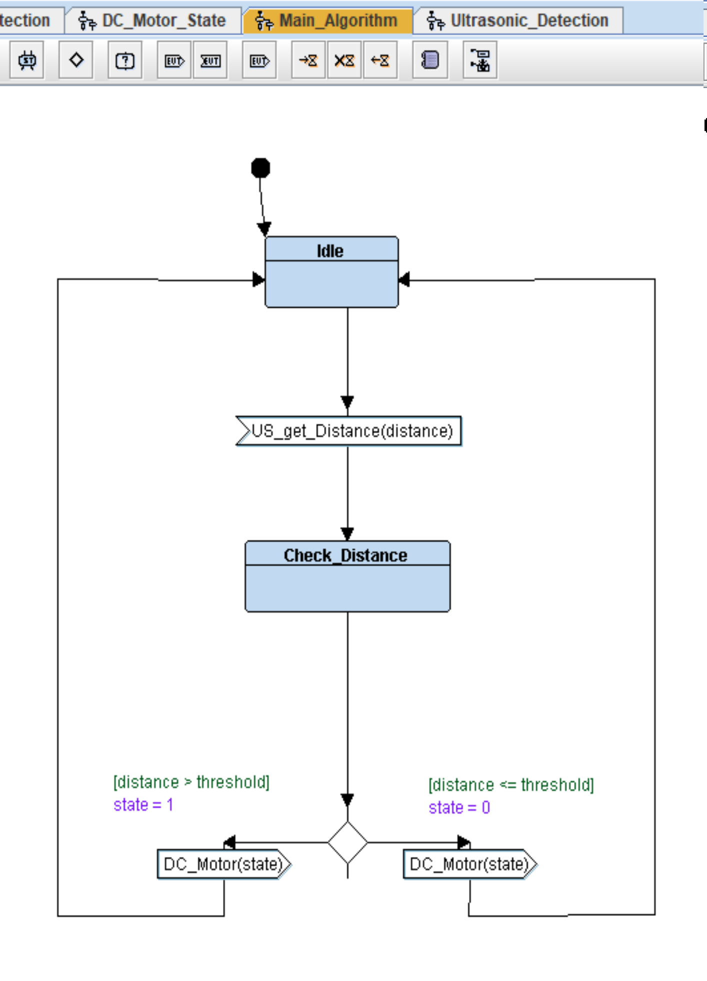
~ out DC_Motor(int state)
~ in US_get_Distance(int distance)

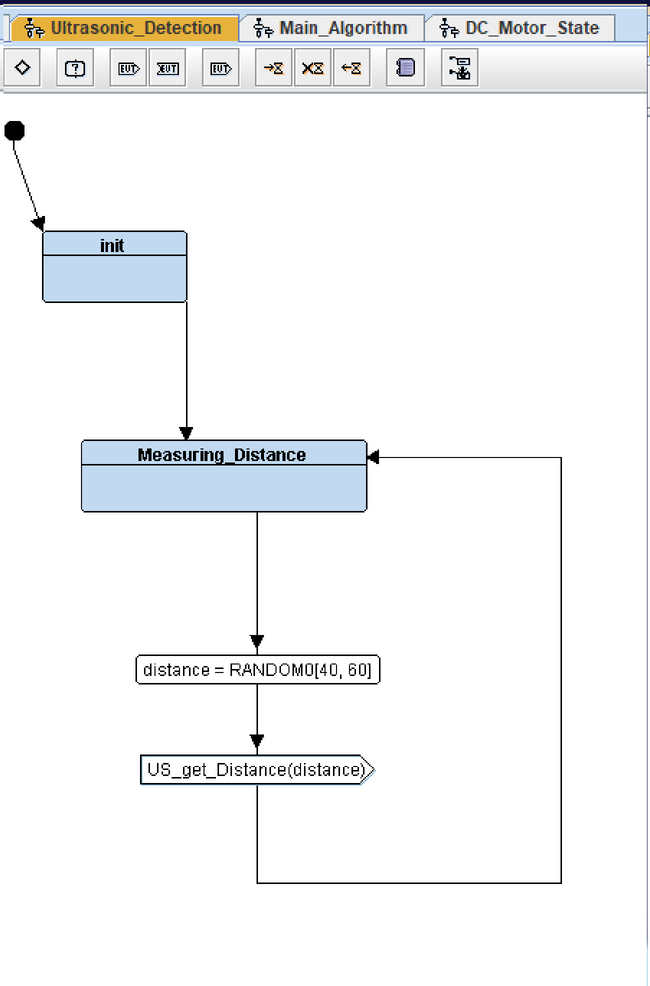
<<block>>
DC_Motor_State

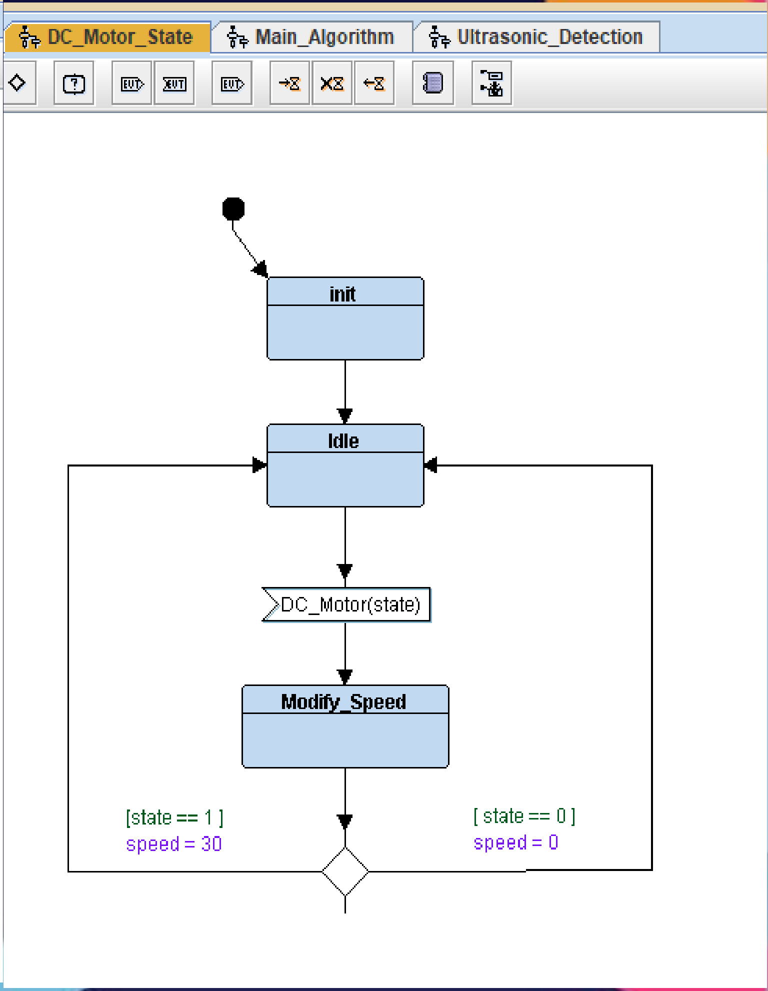
- speed = 0 : int;
- state = 0 : int;

~ in DC_Motor(int state)

- STATE DIAGRAM OF EACH BLOCK


tection
DC_Motor_State
Main_Algorithm
Ultrasonic_Detection
Idle
US_get_Distance(distance)
Check_Distance
[distance > threshold]
state = 1
[distance <= threshold]
state = 0
DC_Motor(state)
DC_Motor(state)


Ultrasonic_Detection
Main_Algorithm
DC_Motor_State
init
Measuring_Distance
distance = RANDOM0[40, 60]
US_get_Distance(distance)


DC_Motor_State
Main_Algorithm
Ultrasonic_Detection
init
Idle
DC_Motor(state)
Modify_Speed
[state == 1]
speed = 30
[state == 0]
speed = 0