**Collision Avoidance**

**Collision avoidance systems “sometimes referred to as crash avoidance systems “are driver- assistance systems that use sensors and algorithms to detect potential hazards or obstacles and provide the driver with warnings to help avoid a collision.**

**• We’ll work on a simple version to learn state machines and how to implement the project into multi modules.**

**Requirements:**

* **when the distance becomes less than or equal 50; the car will stop & its speed becomes Zero.**

**Sequences:**

**We build three modules:**

**- US Module “Ultra Sonic”: It read the distance by a function make a random value “US\_Get\_distance\_random” & send it to CA Module.**

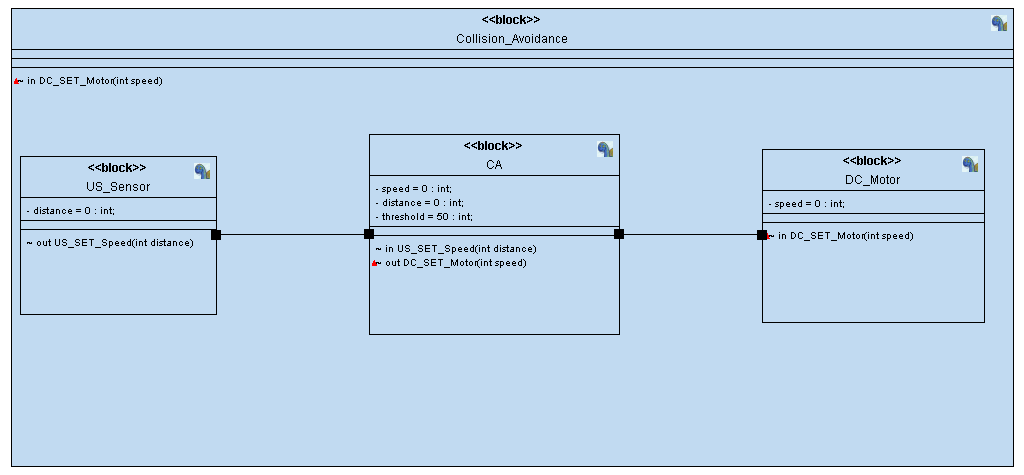
**- CA Module “Collision Avoidance”: it’s implemented with the algorithm of the project, take the distance from US and check the conditions then send the speed to DC**

**- DC Module “DC Motor”: it receives the speed from CA and starts to interface with PWM...etc.**

**Diagrams & implements:**

* **In this part we will discuss the design of diagrams with state machines and flow charts then implement it with C code.**

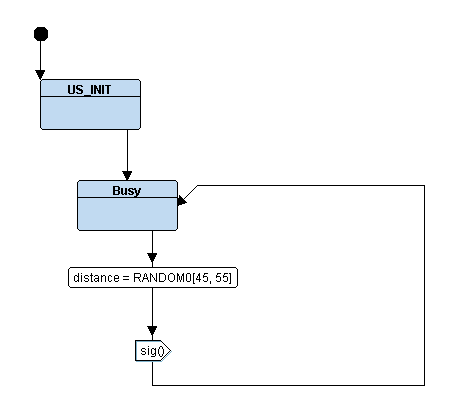
**• Diagram:**

****

**• US Module:**

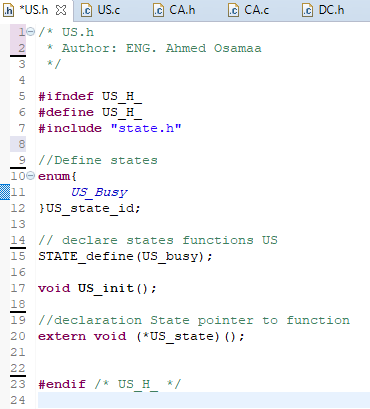
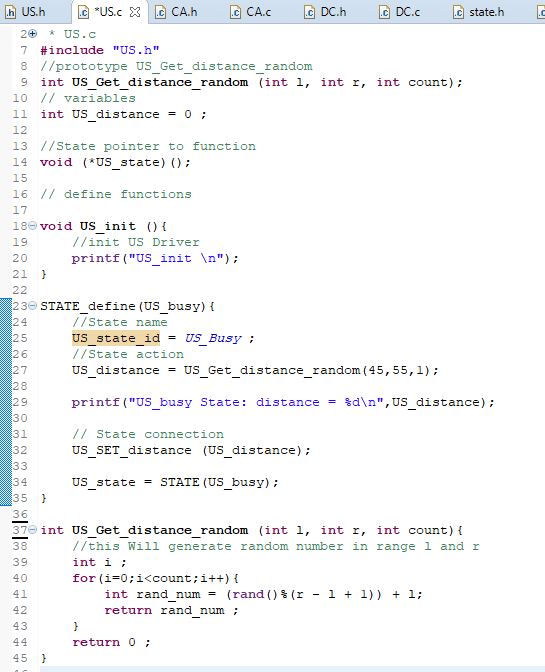
* + **Diagram**

**it contains initialization state and Busy state.**

****

* + **C Code:**

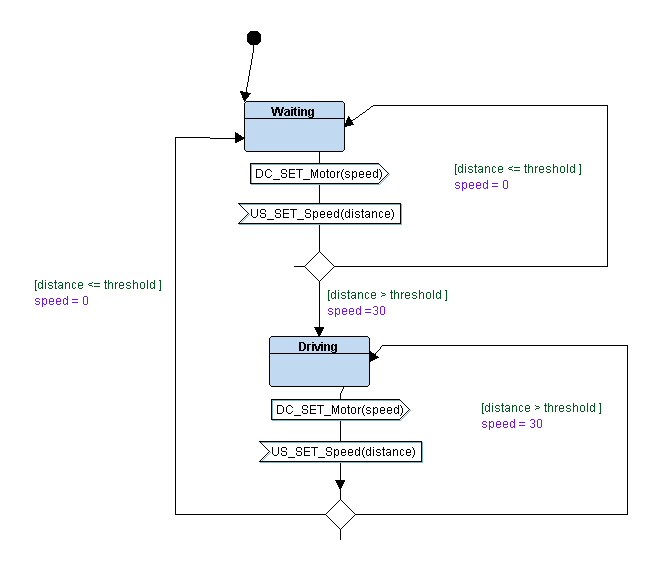
**>US.h >US.c**

****

**• CA Module:**

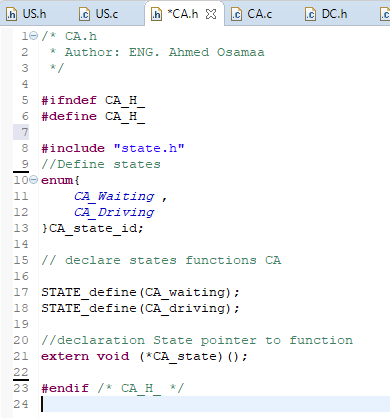
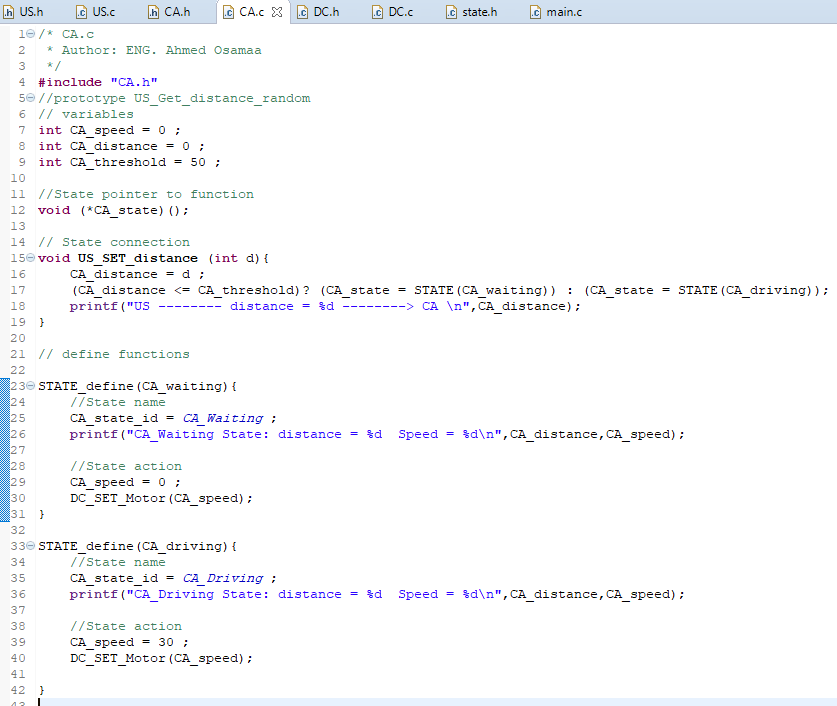
* + **Diagram**

**it contains Waiting state and Driving state.**

****

* + **C Code:**

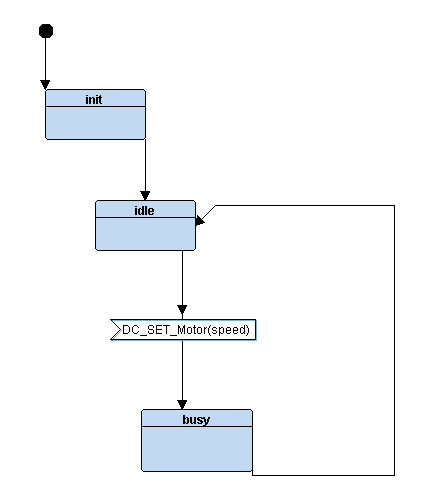
**>CA.h >CA.c**

****

**• DC Module:**

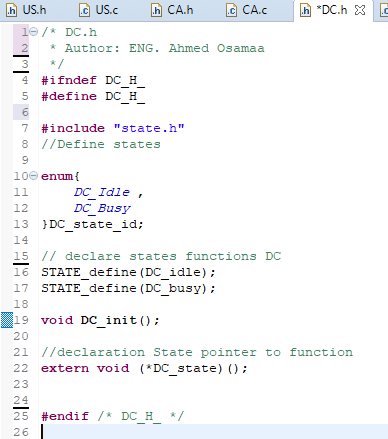
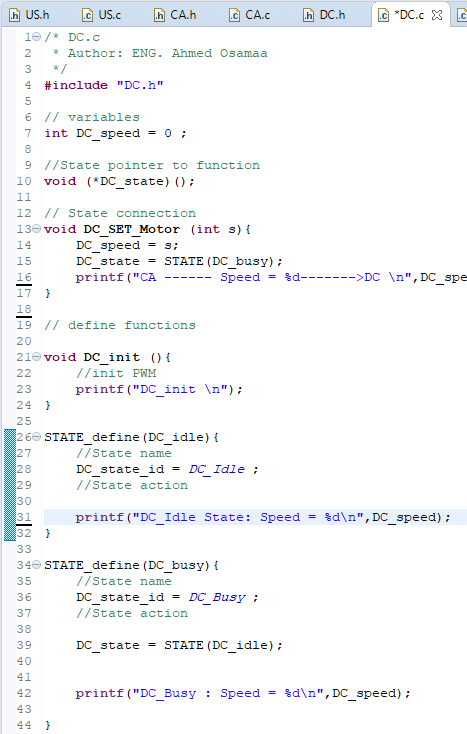
* + **Diagram**

**it contains initialization state, Idle state and Busy state.**

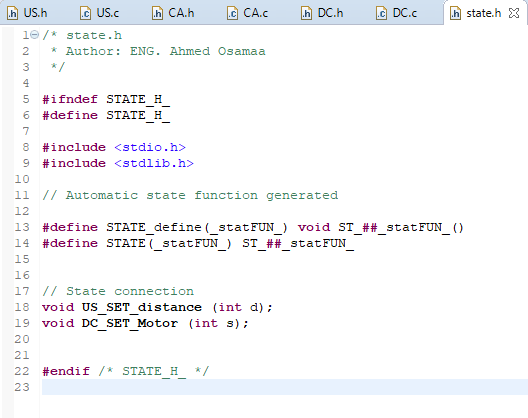
****

* + **C Code:**

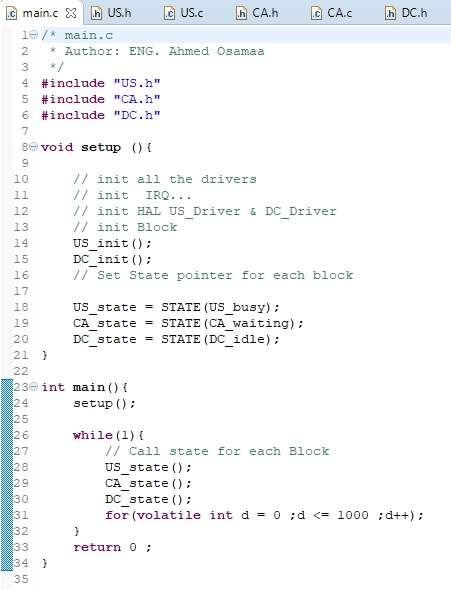
**>DC.h >DC.c**

****

**• state.h:**



**• main.c:**



**• Result:**

**>implemented in C >at Diagram**

