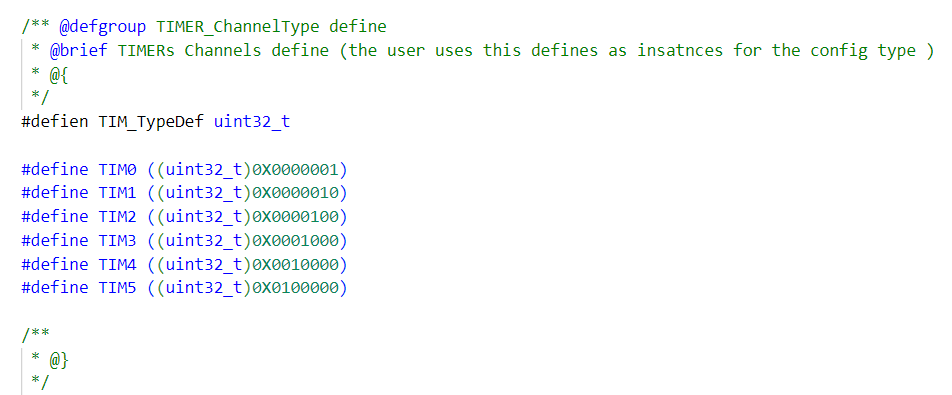
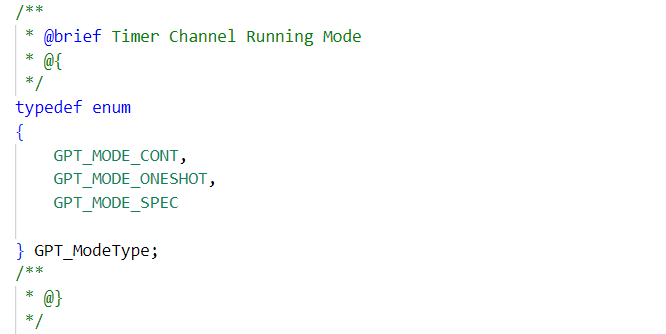
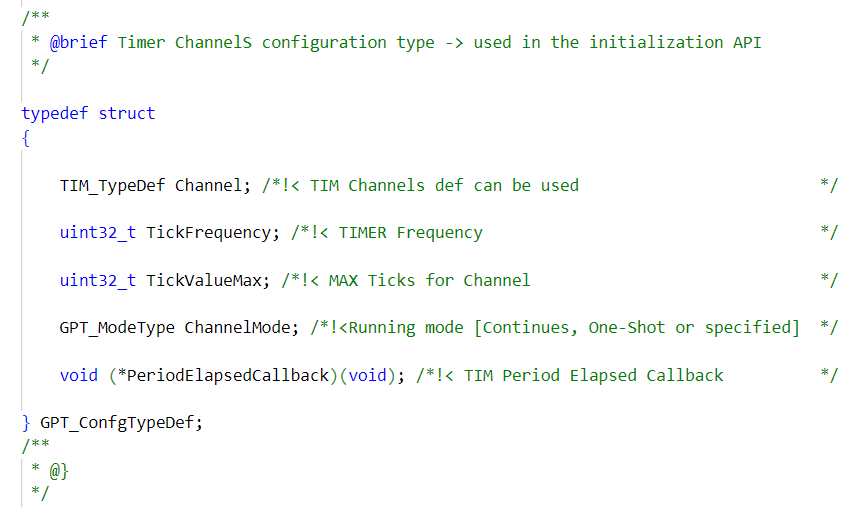
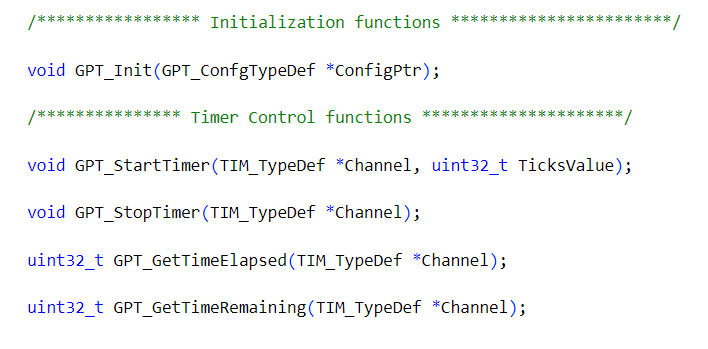
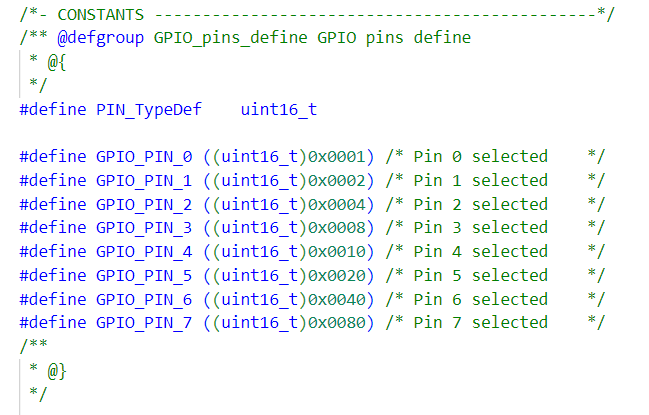
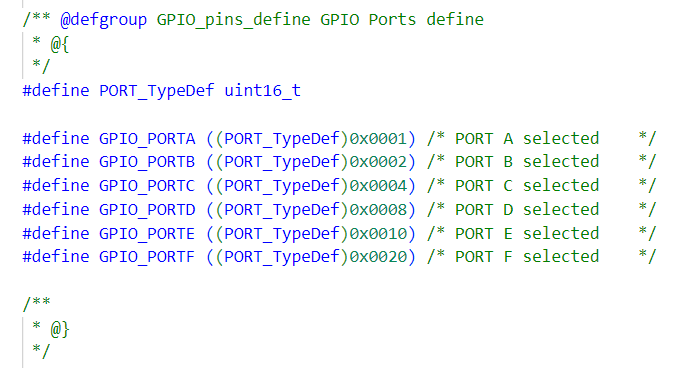
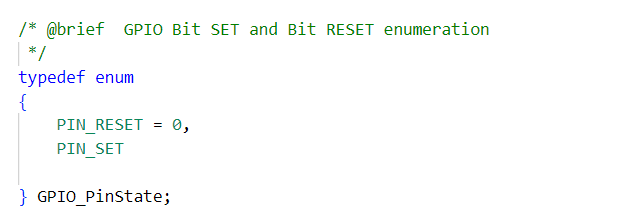
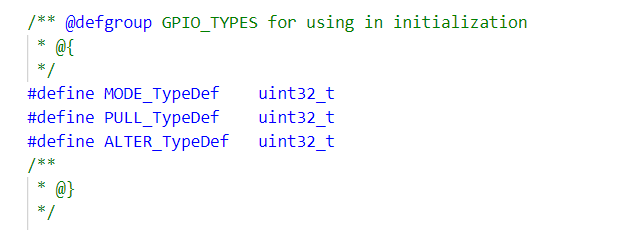
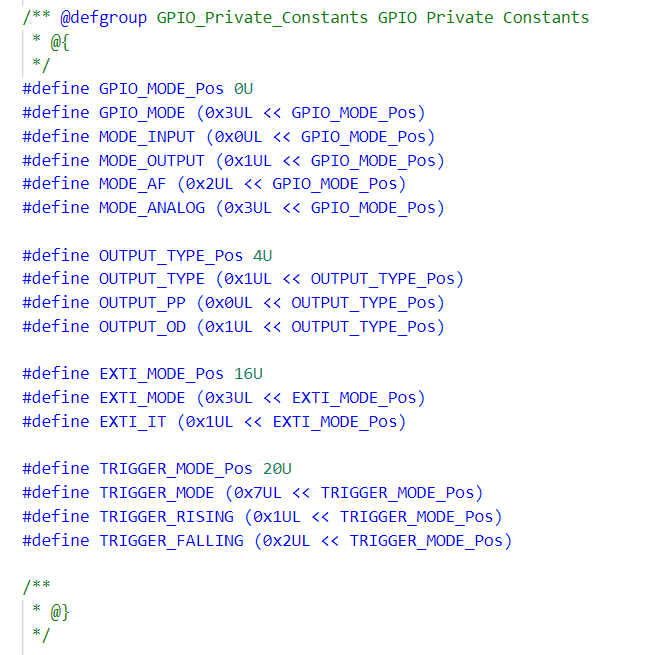
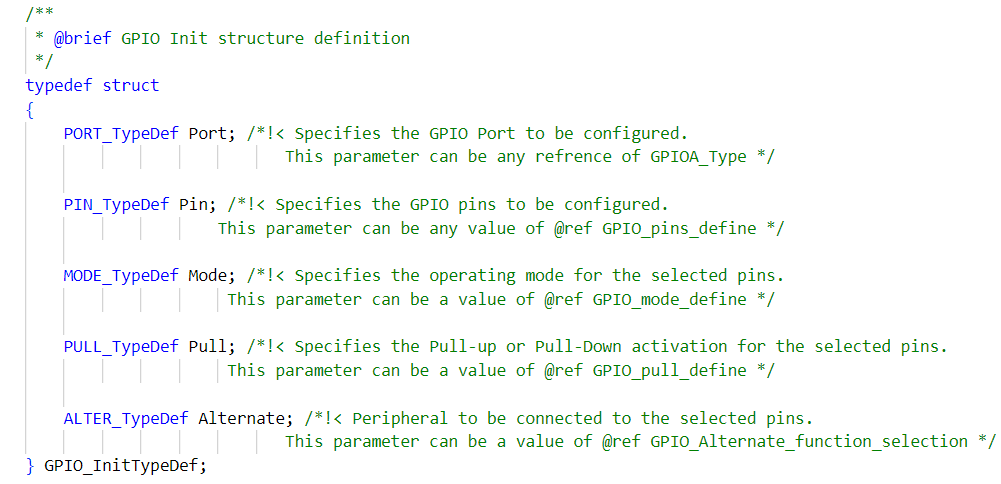
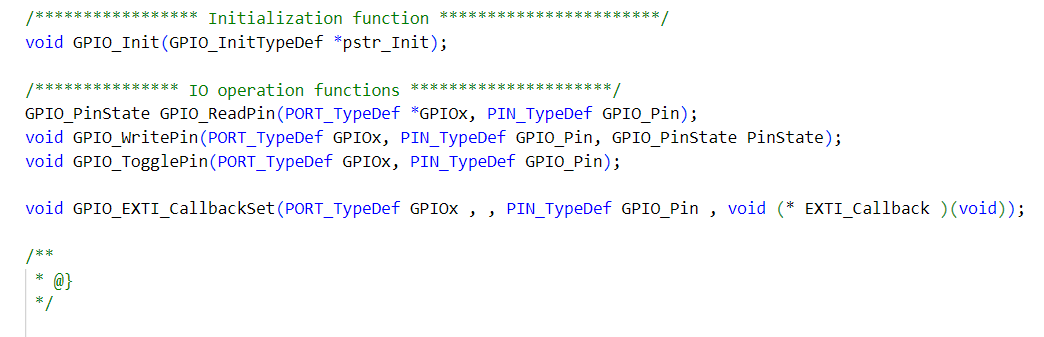
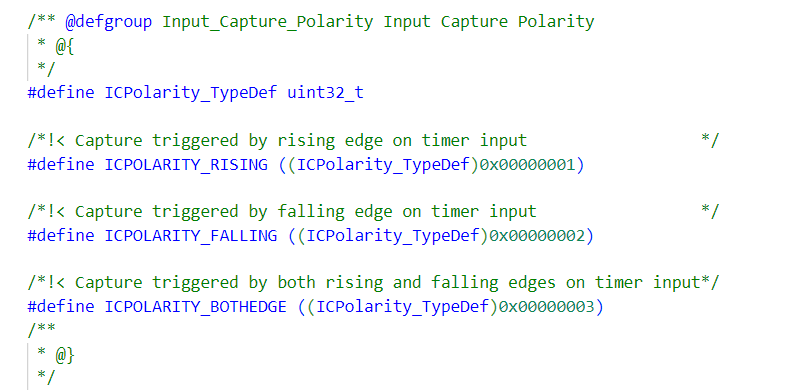
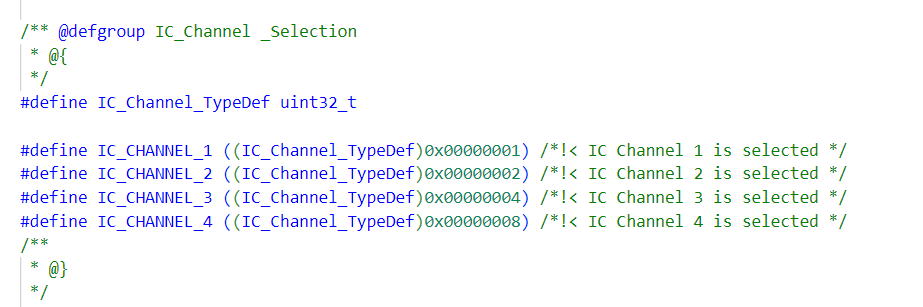
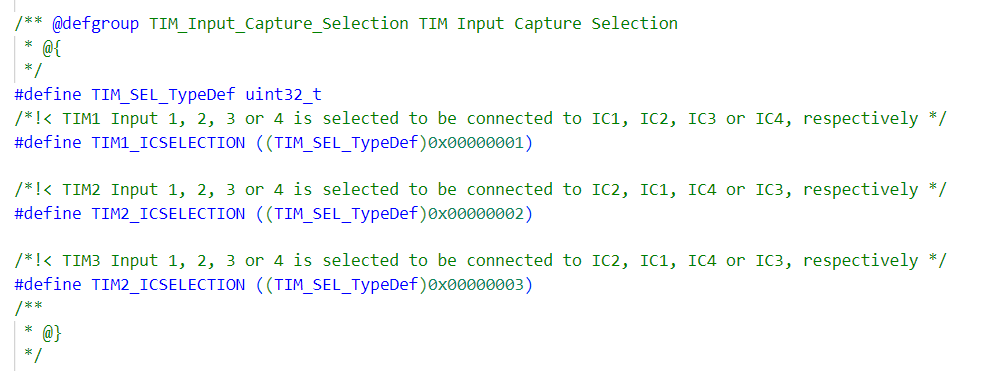
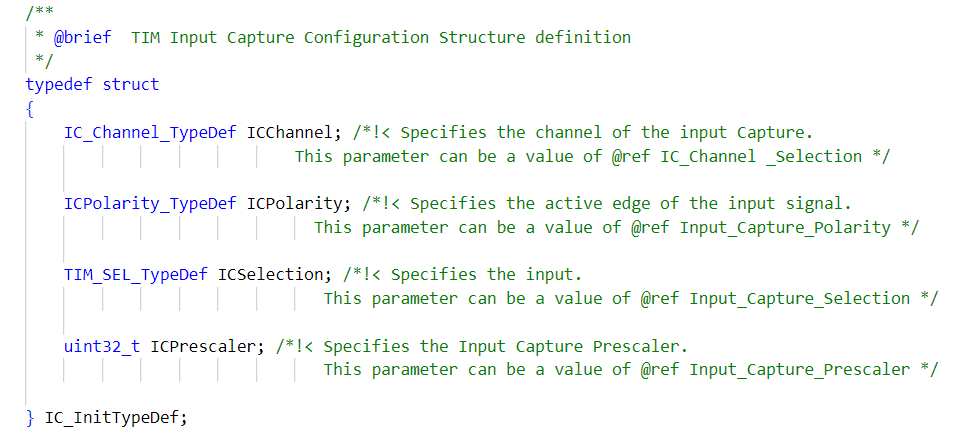
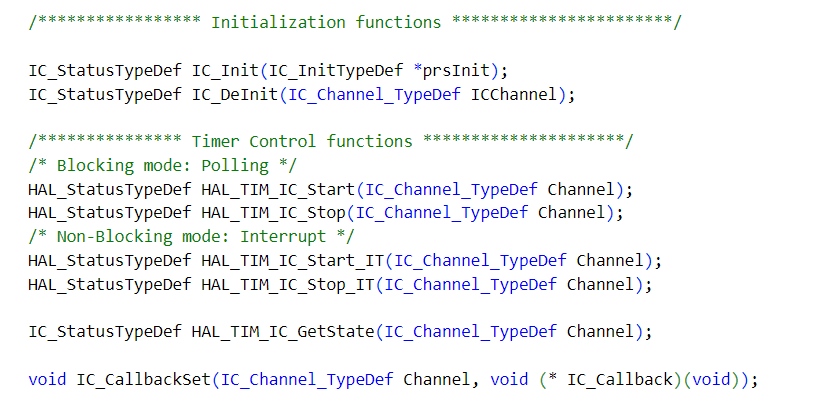
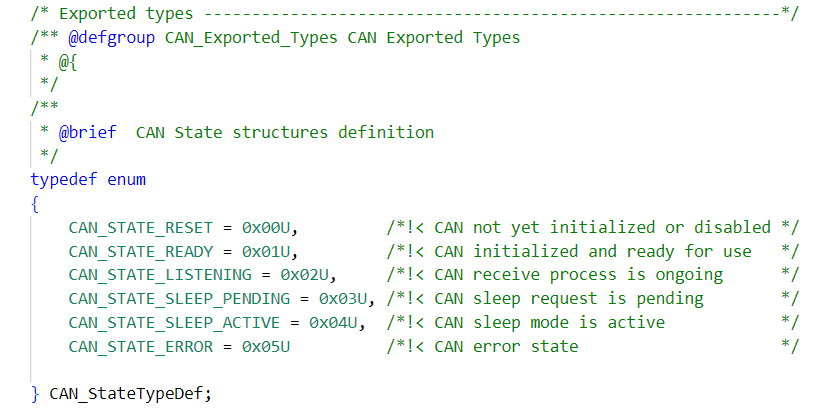
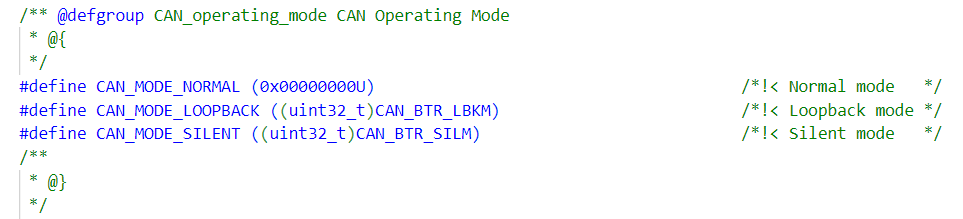
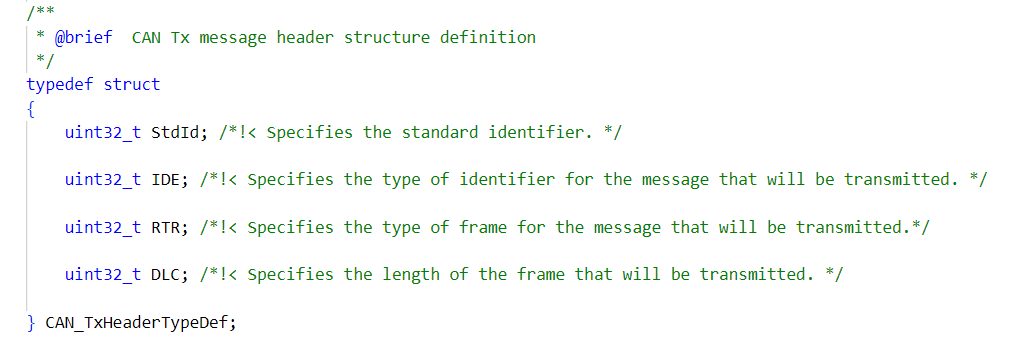
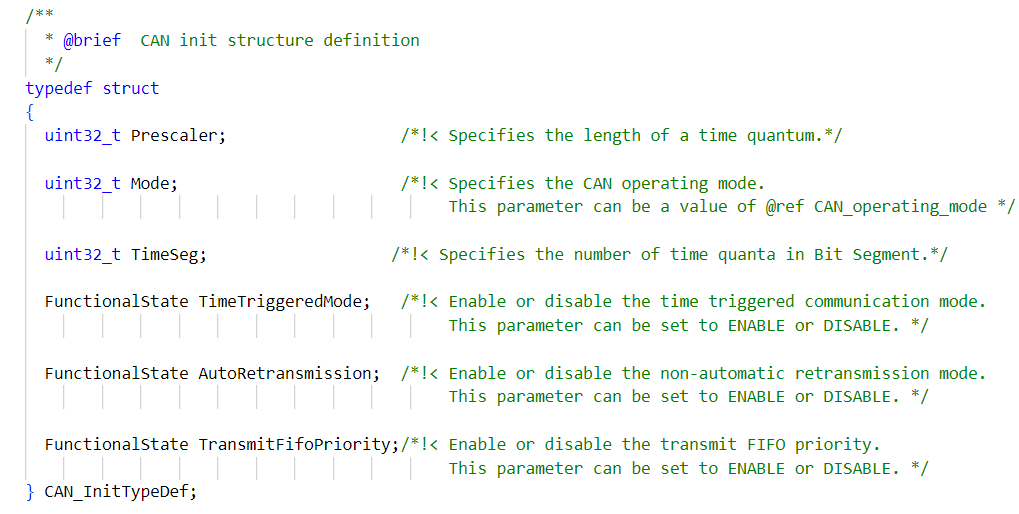
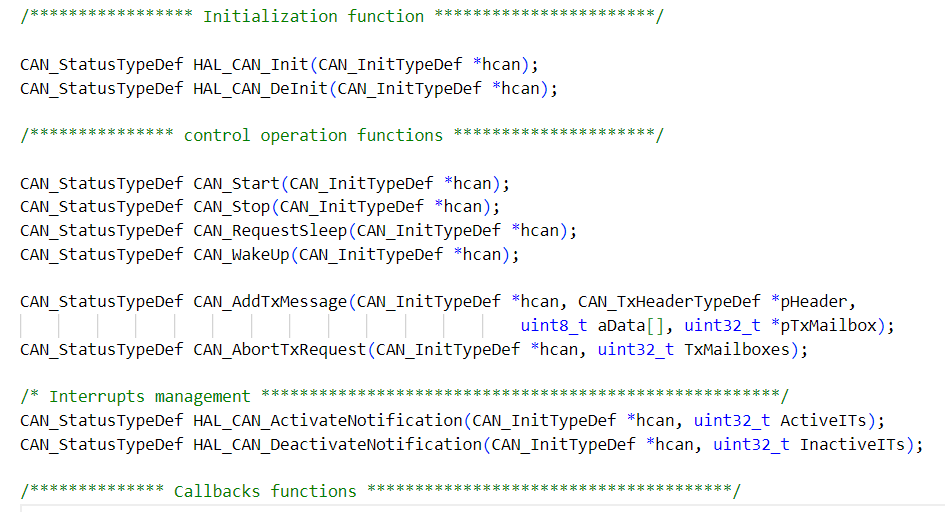
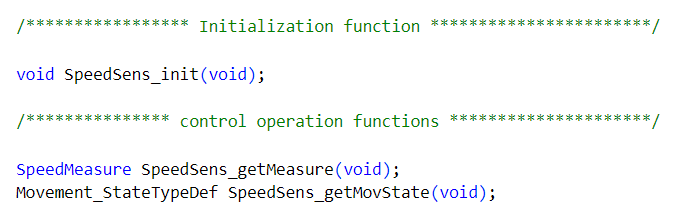
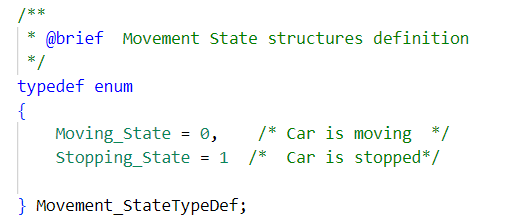
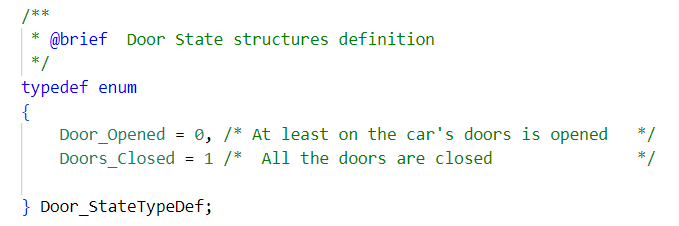
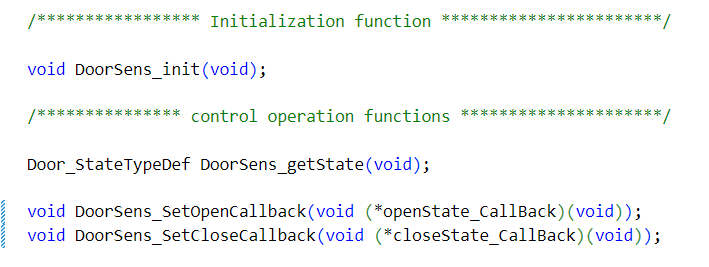
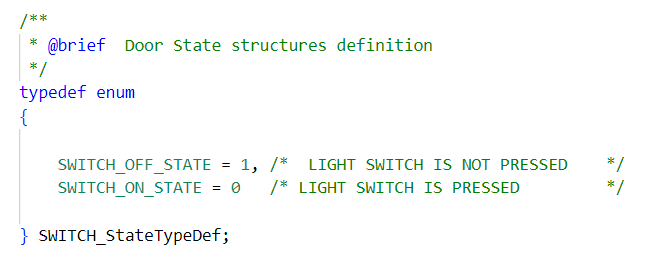
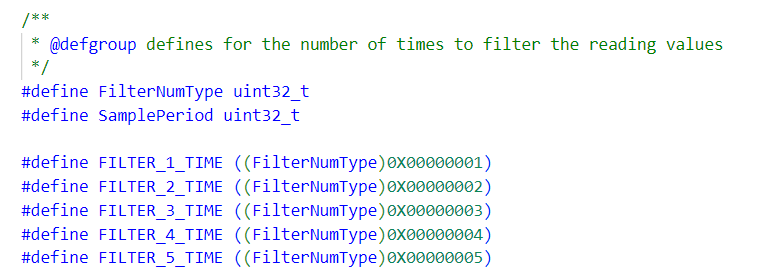
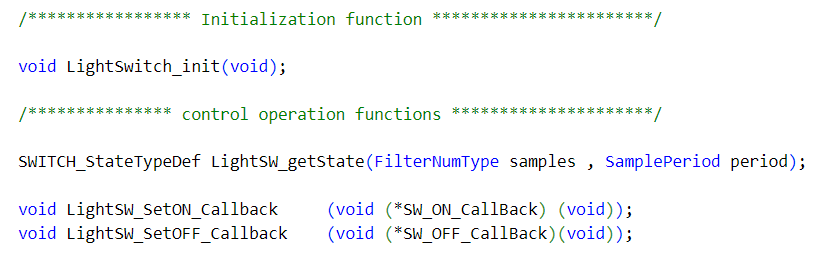
Here we specify each component and module in the ECU abstraction layer as well as the Low Layers Drivers. We start with the low layers to build such infrastructure for the higher layers as ECUAL and Application.

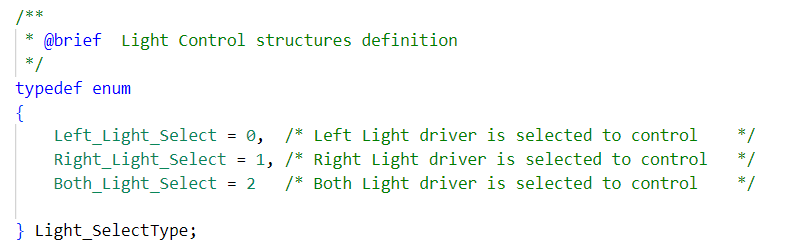
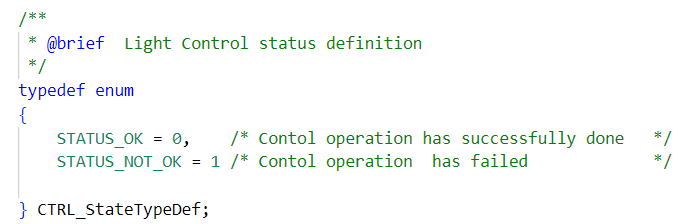
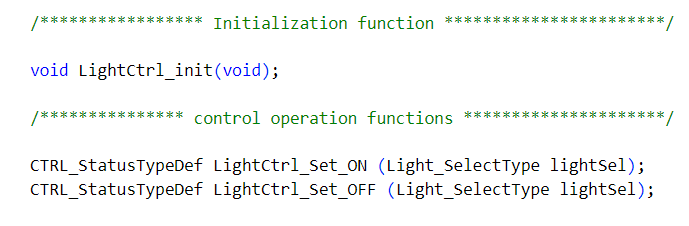
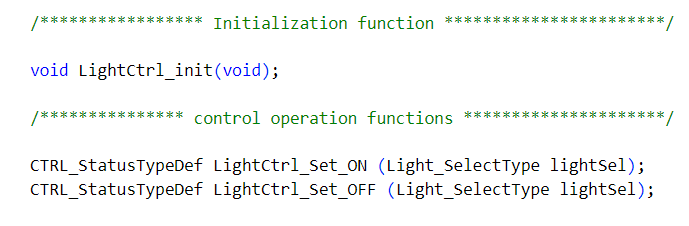
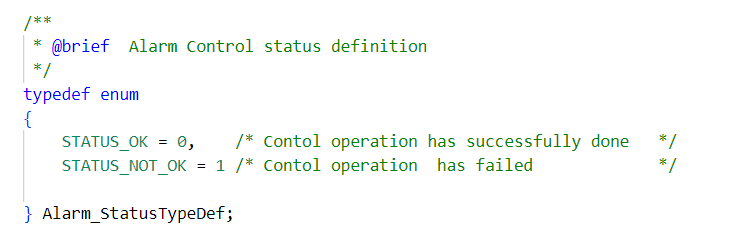
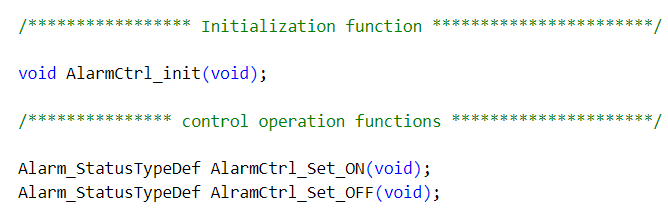
* **Low Layers**
* Microcontroller Abstraction Layer (MCAL):

1. Timer Module Driver  
     
   As specified in HRS this target micro controller is up to attach with number of sensors that will utilize the timer module for ti- ming management and synchronization of the communicati-on bus as it periodically transmit the tracing data on a CAN bus.  
     
   This Driver must provide APIs that utilize any of the hardware   
   timers inside the MCU and generate accurate time based event   
   triggering for specified number of times, API for providing the current counter of ticks as well as initialization functions.  
     
   API Type used for initialization the channels : \_  
     
   API Type used for configure the modes: \_  
   API Type used for struct the configuration parameters: \_  
     
   API functions used for initialization the driver and control operations : \_
2. DIO Module Driver   
     
   Digital input/output will be used by the ECUAL layer to com-  
   munication with sensor and switches attached to the MCU.  
     
   It’s required to Provide full functional APIs to control the op-  
   eration of the GPIO Module from reading and writing data and also controlling external interrupts on the pins.  
   API Type used for initialization the Pins :\_  
   API Type used for specify the required PORT to control:\_  
     
   API Type enum used to read and control the pins state:\_  
   API Types used for the configuration parameters:\_  
     
     
     
   API Types used to configure the operation modes of the pins:\_  
     
   Note : it’s up for the developer to define the required macros   
    for the bits positions according to the target used.   
     
     
     
     
   API Type used to struct the configuration parameters and passing to the initializing API:\_  
   API functions to initialize the DIO module and control operations:\_
3. Input Capture Module   
     
   Dealing with sensors requires signals measurements, that’s why the IC Driver must provide such APIs functions to measure the   
   timing between rising and falling edges of the signal coming from the sensors, IC Driver utilize a timer unit on the selected target.  
     
   ECUAL layer depend on this driver to implement its components APIs. So that, it must be implemented accurately in order to evaluate such correct data from the sensors.  
     
   Here we illuminate the Module API types and functions ..  
   API type to specify the channels to be configured:\_  
     
     
   API type to select the utilized Timer to configure in IC mode :\_  
     
      
   API type used to identify the Input Capture Polarity :\_  
     
   API type to struct the configuration parameters of the module and pass it to the initialization API function.:\_  
     
   API functions to initialize the IC Module and control its operation.  
     
     
     
     
     
    \*\*\*
4. CAN Module Driver   
     
   It’s specified that the first ECU target will pridoically transmit the collected data from the sensors on a CAN bus. In the higher levels   
   we declared the communication module that will need the CAN driver APIs to achieve its purposes. So we define drivers APIs   
   that will facilitate the required communication specifications.  
     
   API type for the CAN Module status :\_  
     
   API type to determine the operation mode of the module on the bus:  
   API type for definition of the Transmit header structure :\_  
   API type to struct the initialization parameters to configure the module before starting operation :\_  
     
     
      
   API functions to allow initialization and control operations and manage interrupts for the Module:\_  
     
     
     
   \*\*\*

* ECU Abstraction LayerHere we define the APIs of the ECUAL layer which on the application layer will depend directly. Where the components of this layer are different from the the two ECUs in the system we define each components group alone. Starting with the component of the ECU 1.
* ECU 1 Components :  
    
    
  1. Speed sensor   
    
     
  That sensor is supposed to provide API functions to get measurements from the hardware sensor attached to the target.  
  Also to specify the car movement state for those situations we just care about the general state not a specific measure for the   
  speed.   
    
  API type for retrieving the car movement state:\_  
    
    
    
  API function to control operation and retrieve measurements ans state:\_  
    
  2. Door Sensor   
    
  That sensor is supposed to provide APIs functions to get doors   
  state as well as Callbacks on changing in either state.  
    
  API type to identify the state of the car doors:\_  
    
  APIs functions to initialization and control operation and set Callbacks handlers:\_

3. Light switch   
  
This component will be responsible for reading the light switch input and get state of the switch on that moment it’s API called.  
  
To make sure that the value returned from reading the switch is accurate and not effect by any kind on noise on the pins we assume the implementation will take care of that by reading the pins state by specified number of times on specified periods which   
could be controlled by the component APIs.  
  
API type to identify the current state of the switch:\_  
  
API type used to define the filter numbers of samples used to decide the state of the switch and the period between each sample  
  
 API function to initialize the component and get switches state as well as set the callback functions that called on changing the state.

* ECU 2 Components :  
    
  Target Microcontroller in ECU 2 is supposed to attach with two light drivers for left and right lights and mosfet gate to control a   
  buzzer or alarm.   
    
  we depend on the structure pattern to combine both the two light drivers controllers in one Module components and to provide API types to facilitate controlling both of them.

1.  Light Control  
     
   Here we define types and api functions to control on both of the right and left lights.  
     
   API types that used in selecting between the two drivers to control:\_  
     
     
   API type used to provide status of the performed operation:\_  
     
     
     
   API functions that used to initialization and control operations:\_
2. Alarm Control   
     
     
   Here we define the APIs used to Control the Alarm operations.  
     
   API type used to identify the performed operation condition:\_  
     
   API function to initialize the module and control operations:\_