MOVING ROBOT
Software Design Document
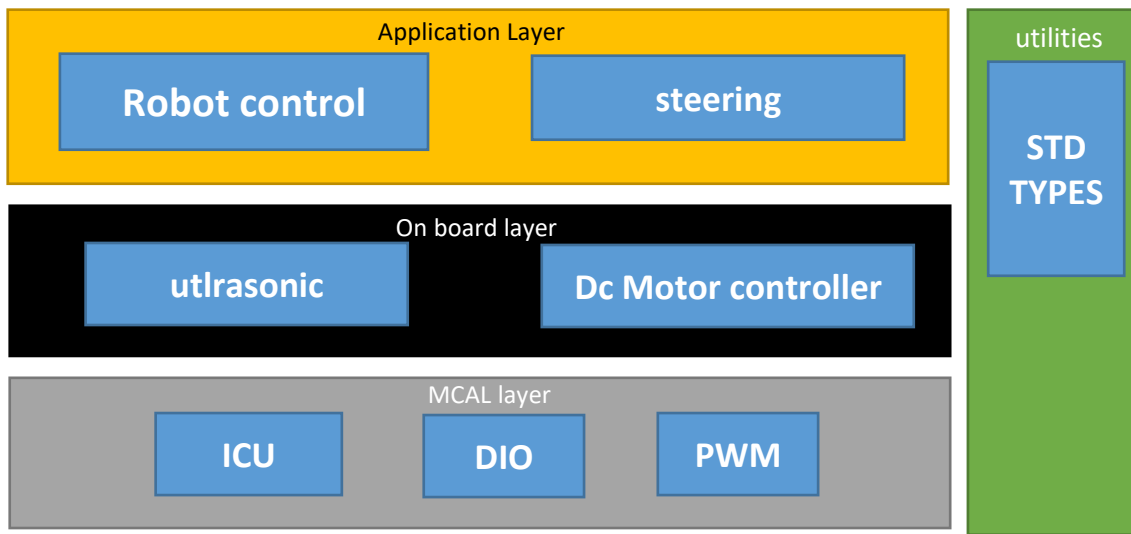
System Description

Requirements :
 - The robot should be moving without stop
 - The robot should be able to avoid obstacles using ultrasonic sensor
 - If there were no object detected the Robot moves forward with 80% of its max speed
   - If there is an object detected 50 cm distant from the Robot then the Robot should slow down to 30% of its maximum speed.
   - If there is an object detected 30 cm distant from the Robot then the Robot will stop then turn right and continue moving under the same distance and speed criteria
   - If there is an object detected less than 30 cm distant from the Robot then the Robot will stop then moves backward until the distance is 30 cm then stop and turn right and continue moving under the same distance and speed criteria

System Architectural Pattern :
   - Monolithic System

Layered Architecture

Layers Diagram :

Application Layer		utilities
Robot control	**steering**	**STD TYPES**

On board layer

utlrasonic	Dc Motor controller

MCAL layer

ICU	DIO	PWM

- MCAL Layer :

Layer type : closed

Description :

    this layer is responsible for providing MCU interface for the required
MCU internal peripheral to the upper layers

Modules :

    - ICU

    - DIO

    - PWM

- On Board Layer :

Layer type : closed

Description :

this layer is responsible for abstracting the MCAL layer from the application layer as it provide the interfaces for the sensors and actuators on the board which is connected to the MCU

Modules :
- ultrasonic module
- Dc motor controller (H bridge)

- Application Layer :

Layer type : closed

Description :

this layer where the application should be written in , and should use the interfaces from the Service layer and on board layer to achieve the system requirement

Modules :
- Robot control
- steering

- utilities

Layer type : open

Description :

this layer used to standard types used by all the software components

Modules :
- standard types

SW Data Types

Name	Uint8
Type	Unsigned char
Range	0 to 255
Description	

Name	Sint8
Type	Signed char
Range	-128 to 127
Description	

Name	Uint16
Type	unsigned short int
Range	0 to 65,535
Description	

Name	Sint16
Type	Signed short int
Range	-32,768 to 32,767
Description	

Name	Uint32
Type	Unsigned long int
Range	0 to 4,294,967,295
Description	

Name	Sint32
Type	Signed long int
Range	-2,147,483,648 to 2,147,483,647
Description	

SW Layers

- MCAL Layer

- PWM

Description : this module should control the MCU hardware timer should be initialized as PWM generator , and should control 4 channel by disable or enable the channel or edit the duty cycle

Module Constrains :

- duty cycle can be between 0 to 65535
- there are only 4 channels can be controlled

- APIs Data Types :

name	Pwm_channel_t
type	enum
range	0-3
description	Pwm channel ID 0 : pwm_ch_1 1 : pwm_ch_2 2 : pwm_ch_3 3 : pwm_ch_4

name	Pwm_channel_duty_t
type	Uint16
range	0-65535
description	Represent the channel duty cycle

name	Pwm_state_t
type	enum
range	0-1
description	Pwm channel state 0 : ch_disabled 1 : ch_enabled

- Configuration data types

name	PWM_Timer_config_t
type	structure
Elements	Precaler :uint8 preload_value :uint16 channel_1_en :Pwm_channel_t channel_2_en :Pwm_channel_t channel_3_en : Pwm_channel_t channel_4_en : Pwm_channel_t init_1_dutycycle : Pwm_channel_duty_t init_2_dutycycle : Pwm_channel_duty_t init_3_dutycycle : Pwm_channel_duty_t init_4_dutycycle : Pwm_channel_duty_t
configuration	Linking configuration
discription	This structure holds the configuration for the pwm timer

- APIs table

Function name	PWM_init
Argument	Inputs : void output : void
Return	void
synchronous	yes
Reentrant	no
description	Used to initialize the pwm module

Function name	PWM_set_state
Argument	Inputs : ID : Pwm_channel_t State : Pwm_state_t output : void
Return	void
synchronous	yes
Reentrant	yes
description	Used to enable or disable the pwm channel

Function name	PWM_set_dutycycle
Argument	Inputs : ID : Pwm_channel_t Duty: Pwm_channel_duty_t output : void
Return	void
synchronous	yes
Reentrant	yes
description	Used to set channel duty cycle

- ICU

Description : this module should control the MCU hardware timer should be initialized as input capture unit , provide the high state duration of the pin assigned to it

Module Constrains :

- ICU can only be assigned to 1 pin

- APIs Data Types :

name	ICU_periode_t
type	Uint16
range	0-65535
description	Duration of the high state of a pin

name	ICU_trigger_t
type	enum
range	0-1
description	0 : faling_edge 1 : raising_edge

- Configuration data types

name	ICU_Timer_config_t
type	structure
Elements	Precaler : uint8 Trigger_state : ICU_trigger_t
configuration	Linking configuration
discription	This structure holds the configuration for the ICU timer

- APIs table

Function name	ICU_init
Argument	Inputs : void output : void
Return	void
synchronous	yes
Reentrant	no
description	Used to initialize the ICU module

Function name	ICU_Get_duration
Argument	Inputs : void output : periode :ICU_periode_t*
Return	E_OK : valid result E_NOK : no valid result
synchronous	yes
Reentrant	yes
description	Used to enable or disable the pwm channel

- DIO

Description : this module should control the MCU DIO pins , should control the initialization of the pins and state of each pins

- APIs Data Types :

name	Port_id_t
type	enum
range	0-3
description	Represent the port id 0 : PORT_A 1 : PORT_B 2 : PORT_C 3 : PORT_D

name	Pin_id_t
type	enum
range	0-7
description	Represent the pin id 0 : PIN_0 1 : PIN_1 2 : PIN_2 3 : PIN_3 4 : PIN_4 5 : PIN_5 6 : PIN_6 7 : PIN_7

name	Pin_state_t
type	enum
range	0-1
description	Represent the pin state 0 : PIN_HIGH 1 : PIN_LOW

name	Pin_DIR_t
type	enum
range	0-1
description	Represent the pin direction 0 : PIN_OUTPUT 1 : PIN_INPUT

- Configuration data types

name	pin_config_t
type	structure
Elements	Port_id : Port_id_t pin_id : Pin_id_t direction : Pin_DIR_t pull_up_down : Pin_state_t
configuration	Linking configuration
discription	This structure holds the configuration for the ICU timer

- APIs table

Function name	DIO_PIN_CONFIG
Argument	Inputs : PIN_CONFIG : pin_config_t output : void
Return	void
synchronous	yes
Reentrant	no

description	Used to initialize for a specified pin

Function name	DIO_get_state
Argument	Inputs : pin : Pin_id_t output : void
Return	State : Pin_state_t
synchronous	yes
Reentrant	yes
description	Used to get pin state

Function name	DIO_set_state
Argument	Inputs : pin : Pin_id_t State : Pin_state_t output : void
Return	void
synchronous	yes
Reentrant	yes
description	Used to set pin state

- on board layer

- ultrasonic

Description : this module is responsible to provide measure distance using ICU module and convert the ICU reading to distance

Module Constrains :
 - can measure distance up to 4 meters

 - APIs Data Types :

name	Ultrasonic_distance_t
type	Uint16
range	0-65535
description	The ultrasonic measures distance

name	Ultrasonic_return_t
type	eum
range	0-2
description	Represent the output state 0 : Ultra_OK 1 : ultra_waiting 2 :ultra_error

- APIs table

Function name	**Ultrasonic_init**
Argument	**Inputs : void** **output : void**
Return	**void**
synchronous	**yes**
Reentrant	**no**
description	**Used to initialize the ultrasonic module by initialize ICU and DIO PINS**

Function name	**Ultrasonic_get_read**
Argument	**Input : void** **Output : distance :** **Ultrasonic_distance_t ***
Return	**Ultra_OK** **ultra_waiting** **ultra_error**
synchronous	**no**
Reentrant	**no**
description	**Used to get the measured distance**

- dc_motor

Description : this module is responsible to control the motors throw H bridge using DIO , PWM

Module Constrains :

- control only 4 motors

- APIs Data Types :

name	Motor_id_t
type	Enum
range	0-3
description	Represent the motor ID

name	Motor_dir_t
type	Enum
range	0-1
description	Represent the motor direction 0 : backward 1 : forward

name	Motor_speed_t
type	Uint8
range	0-100
description	Represent the motor speed precentage

name	Motor_state_t
type	enum
range	0-1
description	0 : motor_start 1 : motor_stop

- Configuration data types

name	Motor_cfg_t
type	structure
Elements	Motor_1_pwm_ch : Pwm_channel_t Motor_2_pwm_ch : Pwm_channel_t Motor_3_pwm_ch : Pwm_channel_t Motor_4_pwm_ch : Pwm_channel_t
configuration	Linking configuration
discription	Structure that map every motor to PWM channel

- APIs table

Function name	**Motors_init**
Argument	**void**
Return	**void**
synchronous	**yes**
Reentrant	**yes**
description	**Used to initialize dc motors module by initialize pwm and DIO pins**

Function name	**Motor_set_state**
Argument	Input : state : Motor_state_t ID : Motor_id_t
Return	void
synchronous	yes
Reentrant	no
description	Sets the motor state to stop or start

Function name	**Motor_get_state**
Argument	Input : ID : Motor_id_t
Return	state : Motor_state_t
synchronous	yes
Reentrant	yes
description	get the motor state stop or start

Function name	**Motor_configure**
Argument	**Input : ID : Motor_id_t** **: dir :Motor_dir_t** **: speed : Motor_speed_t**
Return	**void**
synchronous	**yes**
Reentrant	**no**
description	**Configure the motor speed and direction**

- Application layer

- robot move

Description : this module responsible for reading the ultrasonic sensor and make the robot movement decision

- APIs table

Function name	ROBOT _INIT
Argument	**void**
Return	**void**
synchronous	**yes**
Reentrant	**no**
description	**Init the robot movement by initialize the ultrasonic sensor**

Function name	Robot_movement_update
Argument	**void**
Return	**void**
synchronous	**no**
Reentrant	**no**
description	**Read the ultrasonic sensor and decide the direction of the robot move**

- steering

Description : this module is responsible for converting the robot movement decisions from the robot move module to motors directions and speed

- APIs Data Types :

name	Steering_direction_t
type	enum
range	0-3
description	Represent the robot directions 0 : forward 1 : right 2 : left 3 : backward

name	Steering_speed_t
type	Uint8
range	0-100
description	Represent the robot speed

name	Steering_state_t
type	enum
range	0-1
description	0 : stop 1 f: start

- APIs table

Function name	Steering_init
Argument	void
Return	void
synchronous	yes
Reentrant	no
description	Initialize the robot_movement by initialize the ultrasonic sensor

Function name	Steering_set_direction
Argument	Input : dir : Steering_direction_t
Return	void
synchronous	yes
Reentrant	no
description	Set the robot direction

Function name	Steering_set_state
Argument	Input : state :Steering_state_t
Return	void
synchronous	yes
Reentrant	no
description	Set the robot state to stop or start