1.System Description

Autonomous Robot that has two options:

1.Wireless option

1. Controlling robot direction through pc using WI-FI.

2.Robot Measures environment’s temperature and send it every 5 Seconds

To the PC through WI-FI.

3.Robot should display its moving direction on the LCD.

2.Normal option

1.Robot moves autonomous and uses Ultra-sonic to detect objects.

2.If there are no objects to detect the Robot moves forward with 80% of its maximum speed.

3.If the Robot detected an object 50cm from the Robot then the Robot should decrease its speed to 30%.

4. If the Robot detected an object 30cm from the Robot then the Robot should stop and turn right under(following) the same logic.

5. If the Robot detected an object less than 30cm from the Robot then the Robot should stop then moves backward until the distance is 30cm and then moves right under(following) the same logic.

6.Robot should display its moving direction on the LCD.

# System architecture pattern

System architecture pattern is ***Monolithic***

# System Constrains

1. The maximum code size can’t exceed 250KB

2. don’t exceed more 35KB of RAM.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| OS | ROBOT\_CONTROL | | STEERING | | | DISPLAY | | DATA-LOGGER | | | STD  TYPES |
| Communication manger | | | | | | | | | |
| Ultra-sonic | LCD | | WI-FI | | | DC-MOTOR | | | TEMP-SENSOR |
| BIT  MATH |
| DIO | PWM | | ICU | ADC | | TIMER | | UART | |
| **Microcontroller** | | | | | | | | | |

# 2.Layerd Architecture

1.MCAL layer

1.Closed layer

2.It includes dependent microcontroller hardware peripherals drivers that

we need for the project.

2.On board layer

1.open layer

2.it includes all external hardware drivers used in the project.

3. Services layer

1.closed layer

2. It has a communication manger that handles the data sent through different communication protocols, and it has an operating system that handles the whole system APIs and handle the system Timing.

4. Application layer

1.closed layer

2. It has 3 modules that control the robot direction and handles the data.

# 3.Sw Data Types

|  |  |  |
| --- | --- | --- |
| Name | Robot\_Status\_e |  |
| Type | Enumeration |  |
| Range | ROBOT\_INIT | 0 |
|  | ROBOT\_FORW | 1 |
|  | ROBOT\_BACKW | 2 |
|  | ROBOT\_RIGHT | 3 |
|  | ROBOT\_LEFT | 4 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ErrorStatus |  |
| Type | Enumeration |  |
| Range | E\_OK | 0 |
|  | E\_NOK | 1 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | DIO\_PIN\_t |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_0: | 0 |
|  | GPIO\_PIN\_31 | 31 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | DIO\_DIRECTION\_t |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_OUTPUT | 1 |
|  | GPIO\_PIN\_INPUT | 0 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ADC\_Channel\_t |  |
| Type | Enumeration |  |
| Range | ADC\_CHANNEL\_1: | 0 |
|  | ADC\_CHANNEL\_8 | 7 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | MOTOR\_ID\_t |  |
| Type | Enumeration |  |
| Range | MOTOR\_ID\_1: | 0 |
|  | MOTOR\_ID\_4 | 3 |
| Discerption |  |  |

# 4.SW layers

1.MCAL

1.GPIO

1. Description: this Driver is used to Initialize DIO PIN or write on PORT.

2. Constrains: The maximum number of DIO\_PIN is 31.

3.Configrations Data Types: NULL

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | DIO\_PIN\_t |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_0: | 0 |
|  | GPIO\_PIN\_31 | 31 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | DIO\_DIRECTION\_t |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_OUTPUT | 1 |
|  | GPIO\_PIN\_INPUT | 0 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | DIO\_PinStatus\_t |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_HIGH | 1 |
|  | GPIO\_PIN\_LOW | 0 |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | Gpio\_initPin | | | |
| Arguments | INPUT | | Pin | DIO\_PIN\_t | |
| Specify pin no from DIO\_PIN\_0 🡪 DIO\_PIN\_31 | | |
| direction | | DIO\_DIRECTION\_t |
| Specify pin as input or output | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | E\_OK | | 0 | | |
| E\_NOK | | 1 | | |
| Re-entrant | Yes | | | | |
| Synchronous | Yes | | | | |
| Description | Initializing DIO a specified pin as Input/output | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | Gpio\_ReadPin | | | |
| Arguments | INPUT | | Pin | DIO\_PIN\_t | |
| Specify pin NUM from DIO\_PIN\_0 🡪 DIO\_PIN\_31 | | |
| Null | |  |
|  | | |
| OUTPUT | | Data | | uint8\_t \* |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | E\_OK | | 0 | | |
| E\_NOK | | 1 | | |
| Re-entrant | Yes | | | | |
| Synchronous | Yes | | | | |
| Description | Reading a Specified pin status | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | Gpio\_WritePin | | | |
| Arguments | INPUT | | Pin | DIO\_PIN\_t | |
| Specify pin NUM from DIO\_PIN\_0 🡪 DIO\_PIN\_31 | | |
| Data | | DIO\_PinStatus\_t |
| Choosing pin status to be high or low | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | E\_OK | | 0 | | |
| E\_NOK | | 1 | | |
| Re-entrant | Yes | | | | |
| Synchronous | Yes | | | | |
| Description | Writing to pin status to be high or low | | | | |

6.Sequence Diagram:

2.UART

1. Description: This Driver is used to configure UART channel to be able to send and receive over UART bus.

2. Constrains: There is only one UART channel.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | BAUD\_RATE\_t |  |
| Type | Enumeration |  |
| Range | UART\_BAUD\_4800 | 0 |
|  | UART\_BAUD\_9600 | 1 |
|  | UART\_BAUD\_19200 | 2 |
|  | UART\_BAUD\_115200 | 3 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | UART\_Parity\_Bit |  |
| Type | Enumeration |  |
| Range | UART\_PARITY\_EVEN | 0 |
|  | UART\_PARITY\_ODD | 1 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | STOP\_BITS\_t |  |
| Type | Enumeration |  |
| Range | UART\_STOP\_BIT\_1 | 0 |
|  | UART\_STOP\_BIT\_2 | 1 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | UART\_USE\_ERROR\_DETECTION |  |
| Type | Enumeration |  |
| Range | NO ERROR DETECTION | 0 |
|  | ERROR\_DETECTION\_ENABLED | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | UART\_MODE |  |
| Type | Enumeration |  |
| Range | UART\_TX | 0 |
|  | UART\_RX | 1 |
|  | UART\_TX\_RX\_MODE | 2 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | UART\_FRAME\_WIDTH |  |
| Type | Enumeration |  |
| Range | UART\_FRAME\_8\_BIT | 0 |
|  | UART\_FRAME\_9\_BIT | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Function Name | | | | UART\_Init | | | | | |
| Arguments | | INPUT | | | | NULL | | NULL | |
| NULL | | | |
| NULL | | | NULL |
| NULL | | | |
| OUTPUT | | | | NULL | | | NULL |
| INPUT/OUTPUT | | | | NULL | | | NULL |
| RETURN | | NULL | | | | NULL | | | |
| NULL | | | | NULL | | | |
| Re-entrant | | Yes | | | | | | | |
| Synchronous | | Yes | | | | | | | |
| Description | | Initialize UART baud rate and date frame. | | | | | | | |
| Function Name | | | UART\_Transmit | | | | | | |
| Arguments | INPUT | | | | NULL | | NULL | | |
| NULL | | | | |
| NULL | | NULL | | |
| NULL | | | | |
| OUTPUT | | | | Data | | uint8\_t \* | | |
| INPUT/OUTPUT | | | | NULL | | NULL | | |
| RETURN | E\_OK | | | | 0 | | | | |
| E\_NOK | | | | 1 | | | | |
| Re-entrant | NO | | | | | | | | |
| Synchronous | Yes | | | | | | | | |
| Description | Transmitting the data buffer over UART. | | | | | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | UART\_Receive | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | Yes | | | |
| Description | Receive data over UART and store it in Data\_Buffer. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | UART\_DeInit | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | YES | | | | |
| Synchronous | Yes | | | | |
| Description | Resting all UART ringsters and disable UART peripheral. | | | | |

6.Sequence Diagram:

3.TIMER

1. Description: This driver is used to configure a timer channel in Normal mode to count tick time.

2. Constrains: Just choose Timer channel not used in PWM or ICU.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | Timer\_Prescaler\_t |  |
| Type | Enumeration |  |
| Range | TIMER\_PSC\_0 | 0 |
|  | TIMER\_PSC\_8 | 1 |
|  | TIMER\_PSC\_16 | 2 |
|  | TIMER\_PSC\_64 | 3 |
|  | TIMER\_PSC\_128 | 4 |
|  | TIMER\_PSC\_256 | 5 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | TIMER\_CounterMode\_t |  |
| Type | Enumeration |  |
| Range | TIMER\_UP\_COUUNTING | 0 |
|  | TIMER\_DOWN\_COUNTING | 1 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | TIMER\_CHANNEL |  |
| Type | Enumeration |  |
| Range | TIMER\_CHANNEL\_1 | 0 |
|  | TIMER\_CHANNEL\_2 | 1 |
|  | TIMER\_CHANNEL\_3 | 2 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | TICK |  |
| Type | Uint8\_t |  |
| Range | 1:50 |  |
|  |  |  |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | TIMER\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | NO | | | | |
| Synchronous | Yes | | | | |
| Description | Initialize the specified timer channel and apply configurations. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | TIMER\_Start | | |
| Arguments | INPUT | | TimeInSeconds | uint32\_t |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | Yes | | | |
| Synchronous | NO | | | |
| Description | Start timer tick count. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | TIMER\_Stop | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Stop timer tick counting | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | TIMER\_DeInit | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Resting all Timer channel registers and disable Timer channel. | | | | |

6.Sequence Diagram:

4.PWM

1. Description: this driver is used to configure PWM channel and change duty cycle of PWM signal.

2. Constrains: Just choose PWM channel not used in Timer or ICU

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | PWM\_Prescaler\_t |  |
| Type | Enumeration |  |
| Range | PWM\_PSC\_0 | 0 |
|  | PWM\_PSC\_8 | 1 |
|  | PWM\_PSC\_16 | 2 |
|  | PWM\_PSC\_64 | 3 |
|  | PWM\_PSC\_128 | 4 |
|  | PWM\_PSC\_256 | 5 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | PWM\_Polarity\_t |  |
| Type | Enumeration |  |
| Range | PWM\_ACTIVE\_HIGH | 0 |
|  | PWM\_ACTIVE\_LOW | 1 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | PWM\_CHANNEL |  |
| Type | Enumeration |  |
| Range | PWM\_CHANNEL\_1 | 0 |
|  | PWM\_CHANNEL\_2 | 1 |
|  | PWM\_CHANNEL\_3 | 2 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | PWM\_FAST\_MODE |  |
| Type | Enumeration |  |
| Range | PWM\_FAST\_MODE\_ENABLE | 0 |
|  | PWM\_FAST\_MODE\_DISABLE | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | PWM\_DutyCycle |  |
| Type | Uint8\_t |  |
| Range | 1:100 |  |
|  |  |  |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | PWM\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing PWM channel and apply configuration. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | PWM\_Start | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | Yes | | | |
| Synchronous | NO | | | |
| Description | enable PWM start Generating a square wave | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | PWM\_Stop | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Stop PWM. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | PWM\_ChnageDutyCycle | | |
| Arguments | INPUT | | Duty\_cycle | PWM\_DutyCycle |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | enable change PWM duty cycle | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| unction Name | | PWM\_Deinit | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Resting all PWM channel registers and disable PWM channel. | | | | |

6.Sequence Diagram:

5.ICU

1. Description: This driver is used to configure ICU Channel to be able to capture external signals.

2. Constrains: just don’t use a channel used in Timer or PWM.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | ICU\_Prescaler\_t |  |
| Type | Enumeration |  |
| Range | ICU\_PSC\_0 | 0 |
|  | ICU\_PSC\_8 | 1 |
|  | ICU\_PSC\_16 | 2 |
|  | ICU\_PSC\_64 | 3 |
|  | ICU\_PSC\_128 | 4 |
|  | ICU\_PSC\_256 | 5 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ICU\_Edge\_Type\_t |  |
| Type | Enumeration |  |
| Range | FALLING\_EDGE | 0 |
|  | RISING\_EDGE | 1 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ICU\_CHANNEL |  |
| Type | Enumeration |  |
| Range | ICU\_CHANNEL\_1 | 0 |
|  | ICU\_CHANNEL\_2 | 1 |
|  | ICU\_CHANNEL\_3 | 2 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ICU\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing ICU channel and apply configuration. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | ICU\_StartCpture | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | Yes | | | |
| Synchronous | no | | | |
| Description | enable ICU Preph. and start capturing. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ICU\_StopCpture | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Disable ICU start capturing | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ICU\_GetCptureValue | | | |
| Arguments | INPUT | | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | | NULL | NULL |
| INPUT/OUTPUT | | | NULL | NULL |
| RETURN | ICU register value. | | Uint32\_t | | |
| Re-entrant | NO | | | | |
| Synchronous | yes | | | | |
| Description | return Input capture timing value. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ICU\_Deinit | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Resting all ICU channel registers and disable ICU channel. | | | | |

6.Sequence Diagram:

. 6.ADC

1. Description: this driver is used to configure ADC channel by setting up pins as analog and apply the configurations.

2. Constrains: there is only 9 ADC channels with maximum 12-bit resolution.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | ADC\_Prescaler\_t |  |
| Type | Enumeration |  |
| Range | ADC\_PSC\_2 | 0 |
|  | ADC\_PSC\_4 | 1 |
|  | ADC\_PSC\_6 | 2 |
|  | ADC\_PSC\_8 | 3 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ADC\_Resolution\_t |  |
| Type | Enumeration |  |
| Range | ADC\_12\_BIT\_RESOLUTION | 0 |
|  | ADC\_10\_BIT\_RESOLUTION | 1 |
|  | ADC\_8\_BIT\_RESOLUTION | 2 |
|  | ADC\_6\_BIT\_RESOLUTION | 3 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ADC\_CONTINUOUS\_MODE |  |
| Type | Enumeration |  |
| Range | ADC\_CONTINUOUS\_MODE\_ENABLE | 0 |
|  | ADC\_CONTINUOUS\_MODE\_DISABLE | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ADC\_DATA\_ALIGNMENT |  |
| Type | Enumeration |  |
| Range | ADC\_DATA\_RIGHT\_ALIGNMENT | 0 |
|  | ADC\_DATA\_LEFT\_ALIGNMENT | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | ADC\_Channel\_t |  |
| Type | Enumeration |  |
| Range | ADC\_CHANNEL\_1: | 0: |
|  | ADC\_CHANNEL\_8 | 7 |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ADC\_Init | | | |
| Arguments | INPUT | | ADC\_Channel | ADC\_Channel\_t | |
| ADC\_CHANNEL\_1 🡪 ADC\_CHANNEL\_8 | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing ADC channel and apply configuration. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | ADC\_StartConversion | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | Yes | | | |
| Synchronous | No | | | |
| Description | Triggering the ADC conversion using software and wait until it finishes the conversion. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ADC\_StopConversion | | | |
| Arguments | INPUT | | ADC\_Channel | ADC\_Channel\_t | |
| ADC\_CHANNEL\_1 🡪 ADC\_CHANNEL\_8 | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | stop ADC channel just from conversion. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ADC\_GetADCRead | | | |
| Arguments | INPUT | | | ADC\_Channel | ADC\_Channel\_t |
| ADC\_CHANNEL\_1 🡪 ADC\_CHANNEL\_8 | |
| NULL | NULL |
| NULL | |
| OUTPUT | | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | | NULL | NULL |
| RETURN | ICU register value. | | Uint32\_t | | |
| Re-entrant | NO | | | | |
| Synchronous | yes | | | | |
| Description | store the ADC\_Channel Read into the Data\_Buffer. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | ADC\_Deinit | | | |
| Arguments | INPUT | | ADC\_Channel | ADC\_Channel\_t | |
| ADC\_CHANNEL\_1 🡪 ADC\_CHANNEL\_8 | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Resting all ADC channel registers and disable ADC channel. | | | | |

6.Sequence Diagram:

2.ON BOARD

1.Ultrasonic

1. Description: This Driver used for Initialize Ultra-sonic module to calculate Distance.

2. Constrains: NULL.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | ULTRASONIC\_PORT |  |
| Type | Enumeration |  |
| Range | ULTRASONIC\_PORTA | 0 |
|  | ULTRASONIC\_PORTB | 1 |
|  | ULTRASONIC\_PORTC | 2 |
|  | ULTRASONIC\_PORTD | 3 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ULTRASONIC\_TRIGGER\_PIN |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_0: | 0 |
|  | GPIO\_PIN\_31 | 31 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | ULTRASONIC\_ECHO\_PIN |  |
| Type | Enumeration |  |
| Range | GPIO\_PIN\_0: | 0 |
|  | GPIO\_PIN\_31 | 31 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | UltraSonic\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing Ultra-Sonic Trigger && Echo Pins. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | UltraSonic\_Read | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Start Ultra-Sonic to work && measure distance store the calculated distance in the data buffer. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | UltraSonic\_DeInit | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | De-init Ultra-sonic Trigger && Echo Pins. | | | | |

6.Sequence Diagram:

2.LCD

1. Description: This Driver is used to initialize LCD as a hardware and commands.

2. Constrains: Choose the upper data port or lower data port just if you used 4 bits data mode.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | LCD\_DATA\_PORT |  |
| Type | Enumeration |  |
| Range | LCD\_DATA\_PORTA | 0 |
|  | LCD\_DATA\_PORTB | 1 |
|  | LCD\_DATA\_PORTC | 2 |
|  | LCD\_DATA\_PORTD | 3 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | LCD\_CONTROL\_PORT |  |
| Type | Enumeration |  |
| Range | LCD\_ CONTROL\_PORTA | 0 |
|  | LCD\_ CONTROL\_PORTB | 1 |
|  | LCD\_ CONTROL\_PORTC | 2 |
|  | LCD\_ CONTROL\_PORTD | 3 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | LCD\_DATA\_MODE |  |
| Type | Enumeration |  |
| Range | LCD\_DATA\_BITS\_MODE\_4 | 0 |
|  | LCD\_DATA\_BITS\_MODE\_8 | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | LCD\_DATA\_PORT\_PART |  |
| Type | Enumeration |  |
| Range | UPPER\_PORT\_PINS | 0 |
|  | LOWER\_PORT\_PINS | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LCD\_Ctrl\_Cfg\_s |  | |
| Type | Structure |  | |
| E1 | LCD\_Enable\_Pin | Type | DIO\_PIN\_t |
| E2 | LCD\_ReadWrite\_Pin | DIO\_PIN\_t |
| E3 | LCD\_RS\_Pin | DIO\_PIN\_t |
| Configuration Time | Linking-Configuration |  |  |
| Discerption |  |  | |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | LCD\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing LCD DATA && CONTROL Ports && Pins. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | LCD\_Display | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Transmitting the following data buffer to be displayed on LCD. | | | |

6.Sequence Diagram:

3.DC MOTOR

1. Description: This driver is used for configuring Dc-motor and controlling its speed.

2. Constrains: Don’t use more than four motors.

3.Configrations Data Types:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DC\_MotorsPinCfg\_s |  | |
| Type | Structure |  | |
| E1 | DC\_MOTOR1\_PIN | Type | DIO\_PIN\_t |
| E2 | DC\_MOTOR2\_PIN | DIO\_PIN\_t |
| E3 | DC\_MOTOR3\_PIN | DIO\_PIN\_t |
| E4 | DC\_MOTOR4\_PIN | DIO\_PIN\_t |
| Configuration Time | Linking-Configuration |  |  |
| Discerption |  |  | |

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | MOTOR\_ID\_t |  |
| Type | Enumeration |  |
| Range | MOTOR\_ID\_1 | 0 |
|  | MOTOR\_ID\_2 | 1 |
|  | MOTOR\_ID\_3 | 2 |
|  | MOTOR\_ID\_4 | 3 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | PWM\_DutyCycle |  |
| Type | Uint8\_t |  |
| Range | 1:100 |  |
|  |  |  |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | DcMotor\_Init | | | |
| Arguments | INPUT | | MotorID | MOTOR\_ID\_t | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Initialize the specified Motor Pins to be ready to start moving. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | DcMotor\_Start | | |
| Arguments | INPUT | | MotorID | MOTOR\_ID\_t |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Start specified Dc Motor to move. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | DcMotor\_Stop | | | |
| Arguments | INPUT | | MotorID | MOTOR\_ID\_t | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | NO | | | | |
| Synchronous | Yes | | | | |
| Description | Stop the specified Dc Motor. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | DcMotor\_ChangeSpeed | | |
| Arguments | INPUT | | Duty\_cycle | PWM\_DutyCycle |
| NULL | |
| MotorID | MOTOR\_ID\_t |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | Yes | | | |
| Description | change the motor speed using PWM by changing its duty cycle. | | | |

6.Sequence Diagram:

4.WI-FI

1. Description: This Driver used for Initialize WIFI module (ESP)

To be able to send and receive data over UART wirelessly.

2. Constrains: it’s a must to choose WIFI Mode.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | WIFI\_MODE |  |
| Type | Enumeration |  |
| Range | WIFI\_CLIENT\_MODE | 0 |
|  | WIFI\_SERVER\_MODE | 1 |
| Configuration Time | Pre-compile |  |
| Discerption |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Esp\_NetworkCfg\_s |  | |
| Type | Structure |  | |
| E1 | WIFI\_SSID | Type | Char\* |
| E2 | WIFI\_PASSWORD | Char\* |
| E3 | WIFI\_PORT | Char |
| Configuration Time | Linking-Configuration |  |  |
| Discerption |  |  | |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | WiFI\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Initialize WIFI by configure UART baud rate and date frame. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | WiFi\_Transmit | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | No | | | |
| Synchronous | yes | | | |
| Description | Transmitting the following data buffer over Wifi. | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | WiFi\_Receive | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Receive data over UART wirelessly over WIFI and store in Data\_Buffer. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | WiFi\_DeInit | | | |
| Arguments | INPUT | | NULL | NULL | |
|  | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Resting all UART ringsters and disable UART peripheral and Disable hardware. | | | | |

6.Sequence Diagram:

5.TEMPREURE SENSOR

1. Description: this driver is used to configure an ADC Channel for temperature sensor (LM35).

2. Constrains: Maximum Temperature is 75.

3.Configrations Data Types:

|  |  |  |
| --- | --- | --- |
| Name | ADC\_Channel\_t |  |
| Type | Enumeration |  |
| Range | ADC\_CHANNEL\_1: | 0 |
|  | ADC\_CHANNEL\_8 | 7 |
| Configuration Time | Linking-Configuration |  |
| Discerption |  |  |

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | TmepSensor\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Initializing temprature sesnsor ADC Channel. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | TmepSensor\_Read | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Start TmepSensor to work&& measure Temperature store the calculated Temperature in the data buffer. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | TmepSensor\_DeInit | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | Disable temp sensor ADC Channel. | | | | |

6.Sequence Diagram:

3.SERIVIES

1.OS

2.Communication manger

1. Description: A communication manger that handles the data sent

through different communication protocols.

2. Constrains: don’t use both SPI\_ID to send data.

3.Configrations Data Types: NULL.

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | Data\_ID\_t |  |
| Type | Enumeration |  |
| Range | TIMESTAMP\_ID | 0 |
|  | DIRECTION\_ID | 1 |
|  | TEMPRATURE\_ID | 2 |
| Discerption |  |  |

|  |  |  |
| --- | --- | --- |
| Name | Bus\_ID\_t |  |
| Type | Enumeration |  |
| Range | SPI\_ID | 0 |
|  | UART\_ID | 1 |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | CommManger\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | yes | | | | |
| Synchronous | yes | | | | |
| Description | initialize the communication protocols Buses. | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | CommManger\_Send | | |
| Arguments | INPUT | | ID | Data\_ID\_t |
| Specify what kind of data you want send. | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Transmitting the following data buffer over Wifi. | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | CommManger\_Receive | | |
| Arguments | INPUT | | ID | Data\_ID\_t |
| Specify what kind of data you want send. | |
| NULL | NULL |
| NULL | |
| OUTPUT | | Data\_Buffer | uint8\_t \* |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | yes | | | |
| Description | Receive the data over the bus chosen by ID.  Store the data received into the Data buffer. | | | |

6.Sequence Diagram:

4.APPLICATION

1.ROBOT\_CONTROL

1. Description: This is the main module that is the brain of the application that sends date to other app modules.

2. Constrains: Other application module can’t control the Robot.

3.Configrations Data Types: NULL.

4. APIs Data Types:

|  |  |  |
| --- | --- | --- |
| Name | Robot\_Status\_e |  |
| Type | Enumeration |  |
| Range | ROBOT\_INIT | 0 |
|  | ROBOT\_FORW | 1 |
|  | ROBOT\_BACKW | 2 |
|  | ROBOT\_RIGHT | 3 |
|  | ROBOT\_LEFT | 4 |
| Discerption |  |  |

5.APIs Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | RobotControl\_MainFunction | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | NO | | | |
| Description | this main function is responsible to Update robot movement status | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | RobotControl\_GetDirection | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| Return | ROBOT\_INIT | | 0 | |
| ROBOT\_FORW | | 1 | |
| ROBOT\_BACKW | | 2 | |
| ROBOT\_RIGHT | | 3 | |
| ROBOT\_LEFT | | 4 | |
| Re-entrant | Yes | | | |
| Synchronous | yes | | | |
| Description | return robot status. | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | RobotControl\_GetTemprature | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | Temperature of Range [25,75]. | | | |
| Re-entrant | Yes | | | |
| Synchronous | yes | | | |
| Description | return robot environment's temperature | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | RobotControl\_ChangeStatus | | | |
| Arguments | INPUT | | Status | Robot\_Status\_e | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Change Robot Status. | | | | |

6.Sequence Diagram:

2.DATA LOGGER

1. Description: This module deals with data that have been sent by robot control module.

2. Constrains: NULL.

3.Configrations Data Types: NULL.

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | DataLogger\_MainFunction | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | NO | | | |
| Description | Get the temperature and send it through WIFI to the PC With the help of UART | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | DataLogger\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | initialize WIFI Hardware and communication manger | | | | |

6.Sequence Diagram:

3.STEERING

1. Description: This module controls the direction of the robot according to its status sent by the robot control module.

2. Constrains: NULL.

3.Configrations Data Types: NULL.

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | Steering\_MainFunction | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | NO | | | |
| Description | steer the Robot to the needed direction based on the calculated distance and change The Robot Status. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | Steering\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | Initialize The motors and The Ultrasonic. | | | | |

6.Sequence Diagram:

4.DISPLAY

1. Description: This module Displays the data sent by robot control module on LCD.

2. Constrains: NULL.

3.Configrations Data Types: NULL.

4. APIs Data Types: NULL.

5.APIs Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function Name | | Display\_MainFunction | | |
| Arguments | INPUT | | NULL | NULL |
| NULL | |
| NULL | NULL |
| NULL | |
| OUTPUT | | NULL | NULL |
| INPUT/OUTPUT | | NULL | NULL |
| RETURN | E\_OK | | 0 | |
| E\_NOK | | 1 | |
| Re-entrant | NO | | | |
| Synchronous | NO | | | |
| Description | get the data you want (Robot Direction) to display and display it on LCD. | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | | Display\_Init | | | |
| Arguments | INPUT | | NULL | NULL | |
| NULL | | |
| NULL | | NULL |
| NULL | | |
| OUTPUT | | NULL | | NULL |
| INPUT/OUTPUT | | NULL | | NULL |
| RETURN | NULL | | NULL | | |
| NULL | | NULL | | |
| Re-entrant | Yes | | | | |
| Synchronous | yes | | | | |
| Description | initialize LCD or any other displays. | | | | |

6.Sequence Diagram: